Die approbierte Originalversion dieser Dissertation ist an der Hauptbibliothek der Technischen Universität Wien aufgestellt (http://www.ub.tuwien.ac.at).

The approved original version of this thesis is available at the main library of the Vienna University of Technology (http://www.ub.tuwien.ac.at/englweb/).

DISSERTATION

Interval-based Clock State and Rate Synchronization

ausgeführt zum Zwecke der Erlangung des akademischen Grades
eines Doktors der technischen Wissenschaften
unter der Leitung von

AO.UNIV.PROF. DR.TECHN. ULRICH SCHMID
Inst.-Nr. E183/1
Institut für Automation

eingereicht an der Technischen Universität Wien
Technisch-Naturwissenschaftliche Fakultät

von

DIPL.-ING. M.SC. KLAUS SCHOSSMAIER
Mat.-Nr. 8625231
Finsterergasse 6/2/28
A-1220 Wien

Wien, im September 1998 ______________________

Kurzfassung

Die lokale Zeit kann an den Knoten eines verteilten Systems von deren Uhren abgelesen werden, wobei die Uhrenstände und die Ganggeschwindigkeiten sowohl zueinaneinder synchronisiert sein müssen (interne Uhrensynchronisation), als auch ein Bezug zur Weltzeit gegeben sein muß (externe Uhrensynchronisation). Die Synchronität wird durch die Exekution von verteilten Algorithmen erreicht, welche mit Unsicherheiten herrührend von variierenden Übertragungsverzögerungen von Paketen, Uhrendriften, nicht vernachlässigbaren Granularitäten und vor allem mit einem ganzen Spektrum von Systemfehlern fertig werden müssen.

Unsere Forschung im Rahmen des Projekts SynUTC ist gekenntzeichnet durch das hochgesteckte Ziel, mit kommerzieller Technologie eine Synchronisation im 1 μs-Bereich zu erzielen. Das bedeutet, daß abgesehen von speziellen algorithmischen Betrachtungen eine gewisse Unterstützung seitens der Hardware notwendig ist, im konkreten das exakte Zeitstempeln von Paketen, die Zeitinformationen beinhalten, in Verbindung mit einer Uhrenkonstruktion, die sich durch eine hohe Auflösung und einer feinen Korrekturmöglichkeit bezüglich Uhrenstand und Ganggeschwindigkeit auszeichnet. Zusätzlich benötigt man eine Schnittstelle, um von GPS Empfängern externe Zeitinformationen zu bekommen. All diese Funktionalitäten werden von einem M-Modul (genannt NTI) mit einem selbstentwickelten VLSI Chip (genannt UTCSU) bereitgestellt, welche ebenfalls in dieser Dissertation tangiert werden.

Im Gegensatz zu herkömmlichen Methoden bedienen wir uns einem wohldefinierten Intervallansatz, um sowohl Systemparameter als auch algorithmischen Größen zu repräsentieren. Das erweist sich deswegen als vorteilhaft, weil fast alle Aspekte der Uhrensynchronisation einheitlich betrachtet werden können, wodurch man gute Einsichten erhält, wie derartige Algorithmen funktioniern. Im speziellen entwicklen wir *Accuracy/Precision*-Intervalle bzw. *Rate/Consonance*-Intervalle zusammen mit nützlichen Operationen, um den Stand bzw. die Ganggeschwindigkeit der Uhren im verteilten System zu erfassen. Weiters führen wir den Begriff der *global Time/Rate* als Mitteln zur *worst case* Analyse von unseren Algorithmen ein. Von besonderer Bedeutung sind intervallbasierende *Convergence Functions*, die für die Berechnung von entsprechenden Korrekturwerten der Uhren herangezogen werden; trotz fehlerhafter Eingabeintervalle von anderen Knoten.

Zusammenfassend die wichtigsten Resultate dieser Dissertation: Unser Algorithmus zur Synchronisation der Uhrenstände OP-STATE erhebt den Anspruch optimal bezüglich der *worst case precision* und der maximalen Uhrenkorrektur zu sein, und liefert geringfügig suboptimale dynamische Schranken der *Accuracy*. Darüberhinaus offenbart unsere ausführliche Analyse, daß die Granularitäten und diskreten Korrekturtechniken der Ganggeschwindigkeit von Uhren einen entscheidenden Einfluß auf den erzielbaren Grad der Synchronität haben. Als besondere Neuheit demonstriert unser Algorithmus zur Synchronisation der Ganggeschwindigkeiten der Uhren OP-RATE, daß im wesentlichen die Stabilität der Oszillatoren für die gegenseitge Abweichung der Ganggeschwindigkeit von Uhren verantwortlich ist, welche signifikant geringer ausfallen kann, als die üblicherweise spezifiziere maximale Uhrendrift.

Abstract

Local time can be observed at nodes in a distributed system by using their clocks, whose state and rate are required to be in sync with each other (internal clock synchronization) and related to real-time (external clock synchronization) as well. The synchrony is accomplished by running distributed algorithms, which have to cope with uncertainties arising from varying packet transmission delays, clock drifts, non-zero granularities, and above all a whole range of system faults.

Our research within the scope of project SynUTC is driven by the challenging goal to achieve a worst case synchronization tightness in the 1 μs-range with commercial-off-the-shelf technology. This means that apart from advanced algorithmic matters, some hardware support for exact timestamping of packets containing time information is required in conjuction with a high-resolution clock device with fine-grained state and rate adjustment capabilities. Additionally, an interface for GPS receivers is needed to obtain external time information. All those features are provided by an M-Module (called NTI) built around a custom VLSI chip (called UTCSU), which are also touched here.

Unlike traditional approaches, we employ a well-founded interval paradigm to represent both system parameters and algorithmic quantities. This entails the advantage of a uniform view of almost all aspects of clock synchronization, which in turn allows to gain precious insights how such algorithms work. In particular, we establish accuracy/precision resp. rate/consonance intervals along with suitable operations to capture the state resp. rate of clocks in the distributed system. Moreover, we introduce the notion of internal global time/rate as a vehicle for the worst case analysis of our algorithms. Of particular interest are interval-based convergence functions that are in charge of computing proper clocks adjustments despite faulty input intervals from remote nodes.

To highlight the major results, our clock state algorithm OP-STATE exhibits optimality in terms of worst case precision and maximum clock adjustment, and maintains slightly suboptimal on-line accuracy bounds. Moreover, our thorough analysis reveals that clock granularities and discrete rate adjustment techniques have a considerable impact upon the achievable degree of synchronization. As a novelty, our clock rate algorithm OP-RATE demonstrates that the oscillator stability is essentially responsible for the achievable mutual clock drift, which may be significantly smaller than the commonly specified maximum drift.

Acknowledgments

I would like to thank all people that helped me to finish this thesis, in particular, my advisor Prof. Ulrich Schmid, the head of project SynUTC, who initially invited me to join the Department of Automation and work in the area of distributed real-time systems. His knowledge, thoroughness, and work methods gave me an idea how a computer scientist should be. I especially appreciate his way of guiding me through all the worries when doing research and writing papers.

I gratefully acknowledge Prof. Mehdi Jazayeri, the head of the Distributed Systems Group of the Information Systems Institute, for his valuable comments and suggestions when grading my work.

Moreover, I would like to acknowledge the excellent cooperation with all members of the SynUTC team. My special thanks go to Martin Horauer, Thomas Mandl, Dieter Höchtl, and Bettina Weiss for discussing details of this work. I am grateful to many people at the department, in particular, Prof. Gerhard H. Schildt, Johann Klasek, Wolfgang Kastner, and Johann Blieberger. I also want to thank my friends Hans-Peter Berger, Jörg Bachmayer, Bernhard Nikodem, Christoph Sonderegger, and Stefan Stöckler for their durable help, as well as my US friends Sue Moon, Lisa Rudnick, Maria Brunel, and Thomas Hahn for their joyous e-mails.

Last but not least, I would like to say thanks to my parents, Alfred and Brunhilde Schossmaier, and sister Helga, for providing me with all the possible support through the years. Thank you for believing in me!

This research has been supported by Austrian Science Foundation (FWF) under grant P10244-ÖMA, the Austrian National Bank (OeNB) under grant 6454, the Austrian Federal Ministry of Science and Transport (BMWV) under contract Zl.601.577/2-IV/B/9/96 as well as the START programme Y41-MAT.

Each life converges to some centre
Expressed or still;
Exists in every human nature
A goal,

Admitted scarcely to itself, it may be,
Too fair
For credibility's temerity
To dare.

Adored with caution, as a brittle heaven,
To reach
Were hopeless as the rainbow's raiment
To touch,

Yet persevered toward, surer for the distance;
How high
Unto the saints' slow diligence
The sky!

Ungained, it may be, by a life's low venture,
But then,
Eternity enables the endeavoring
Again.

EMILY DICKINSON

Contents

Chapter 1 Preface **1**
1.1 Project SynUTC 1
1.2 Interval-based Clock Synchronization 3
1.3 Outline 5

Chapter 2 Hardware Support for Clock Synchronization **8**
2.1 Introduction 8
2.2 System Architecture 10
2.3 Network Time Interface (NTI) 12
  2.3.1 Requirements 12
  2.3.2 Design 15
  2.3.3 Hardware/Software Interface 17
  2.3.4 Integration 19
2.4 Universal Time Coordinated Synchronization Unit (UTCSU) 21
  2.4.1 Functional Specification 22
  2.4.2 Unit Implementation 34
  2.4.3 Programming Model 42
  2.4.4 Design Methodology 49
2.5 Relation to other Approaches 50
2.6 Summary and Future Work 53

Chapter 3 Interval-based Clock State Synchronization **54**
3.1 Introduction 54
3.2 The Interval-Based Paradigm 57
  3.2.1 Accuracy Intervals 58
  3.2.2 Precision Intervals 59
3.3 System Modelling 62
  3.3.1 Discrete Local Interval Clocks 63
  3.3.2 Network Communications 72
  3.3.3 Fault Model 75

3.4 Generic Algorithm 78
3.5 Analysis 82
    3.5.1 Internal Global Time 84
    3.5.2 Instantaneous Correction 86
    3.5.3 Continuous Amortization Technique 109
3.6 Summary and Future Research 111

Chapter 4 Interval-based Clock Rate Synchronization 113

4.1 Introduction 113
4.2 System Modelling 115
    4.2.1 Oscillator 116
    4.2.2 Local Clock 119
    4.2.3 Processor 120
    4.2.4 Communication Subsystem 120
    4.2.5 Faults 121
4.3 Building Blocks 121
    4.3.1 Interval Paradigm 122
    4.3.2 Local Rate Intervals 123
    4.3.3 Relative Rate Measurement 127
    4.3.4 Remote Rate Intervals 132
    4.3.5 Consonance Intervals 134
4.4 Clock Rate Algorithm 137
    4.4.1 Round Structure 137
    4.4.2 Generic Algorithm 139
    4.4.3 Example 141
    4.4.4 Convergence Function 143
    4.4.5 Validation Function 148
    4.4.6 Abstract Fault Model 150
4.5 Analysis 151
    4.5.1 Round Beginnings 152
    4.5.2 Interval Dissemination 153
    4.5.3 Rate Adjustments 162
    4.5.4 Consonance 164
    4.5.5 Drift 168
4.6 Summary and Future Research 170

Chapter 5 Interval-based Convergence Functions 171
5.1 Introduction 171
5.2 Marzullo's Function 172
    5.2.1 Accuracy Properties 174
    5.2.2 Precision Properties 177
    5.2.3 Reference Point Setting 182
5.3 Optimal Precision Convergence Function 186
    5.3.1 Characteristic Functions 188
    5.3.2 Clock State Algorithm 202
    5.3.3 Clock Rate Algorithm 211
5.4 Summary and Future Research 215

Chapter 6 Complete Synchronization Algorithm 216
6.1 Introduction 216
6.2 Transmission Delay Measurement 217
6.3 Interlocking Clock State and Rate Algorithm 219
    6.3.1 Accuracy/Precision Interval Deterioration 220
    6.3.2 Improved Clock State Algorithm 224
    6.3.3 Closing Analysis 227
    6.3.4 Numerical Example 237
6.4 Extensions 239

List of Symbols 242

Bibliography 246

Curriculum Vitae 253

Chapter 1

PREFACE

Designing distributed fault-tolerant real-time applications is usually considerably simplified when synchronized clocks are available. Temporally ordered events are in fact beneficial for a wide variety of tasks, ranging from relating sensor data gathered at different nodes up to fully-fledged distributed algorithms, see [31] for examples. Providing mutually synchronized local clocks is known as the *fault-tolerant internal clock synchronization* problem, and numerous solutions have been worked out, see [46] or [65] for an overview and [75] for a bibliography.

If system time provided by synchronized clocks must also have a well-defined relation to *Universal Time Coordinated* (UTC), the only official and legal standard time, then the *fault-tolerant external clock synchronization* problem needs to be addressed. Unlike internal synchronization, it did not receive much attention until recently, when highly accurate and cheap receivers for the *Global Positioning System* (GPS) became widespread, see [5]. A representative overview of the current research on external clock synchronization may be found in [53].

1.1 Project SynUTC

In January 1995 the research project SynUTC (which stands for Synchronized Universal Time Coordinated) was launched at the Department of Automation, TU Vienna. It is devoted to the problem of how to establish a highly accurate, common notion of time among the nodes of a distributed fault-tolerant real-time system. More specifically, assuming that nodes are solely interconnected by some data network (e.g., Ethernet), our system will provide each node with a local clock device that is kept within a few μs of each other, despite of faulty ones in the system. Moreover, incorporating an external time source like a GPS receiver, any local clock will be kept within a few μs of UTC as well.

Figure 1.1 provides a good overview of the various research tasks of this project. Basically, the mainstreams are:

1. Design and worst case analysis of clock synchronization algorithms, which embraces ones for clock state and rate synchronization along with ones for obtaining various system parameters on-line.

2. Development of an "engineered" implementation of our algorithms including appropriate hardware support, which end up in building an ASIC and an M-Module.

3. Evaluation of our algorithms on an experimental testbed, which requires to incorporate our hard/software in a state-of-the-art real-time system kernel.

Further information and plenty of other documents concerning project SynUTC can be found at our homepage `http://www.auto.tuwien.ac.at/Projects/SynUTC/`

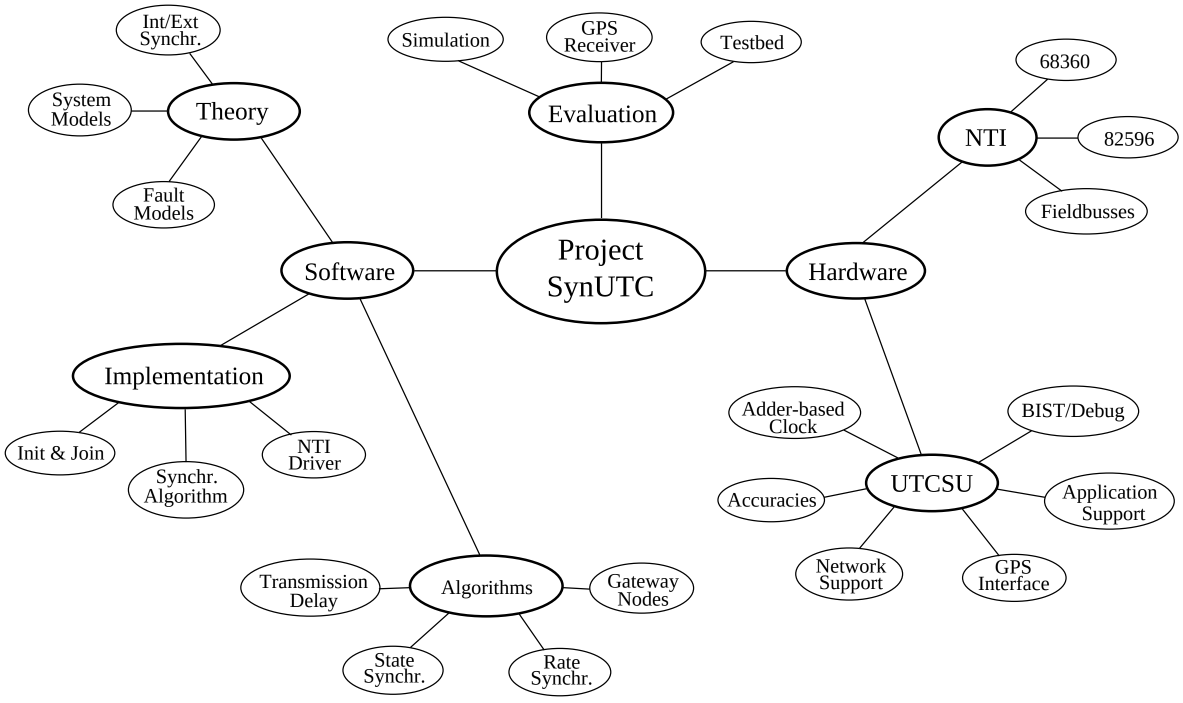


Figure 1.1: *Overview of Project SynUTC*

This thesis focuses on the algorithmic concepts and underlying system and fault models for internal/external clock synchronization in the μs-range. For that purpose, we make use of intervals to describe all quantities (e.g., clocks' state/rate) that are relevant for clock synchronization. They are suitable to capture the inevitable uncertainties affecting these quantities, and they allow to gain precious insights how our algorithms work.

1.2 Interval-based Clock Synchronization

Interval-based clock synchronization —introduced in [50] and further developed in a number of papers [55], [60], [52], [51], etc.— relies on the interval-based paradigm originally introduced in [35] and [25]. Real-time t (usually UTC) is not just represented by a single time-dependent clock value $C(t)$ here, but rather by an *accuracy interval* $\boldsymbol{A}(t)$ that must satisfy $t \in \boldsymbol{A}(t)$. As illustrated in Figure 1.2, accuracy intervals are usually provided by combining an ordinary clock $C(t)$ with a time-dependent *interval of accuracies* $\boldsymbol{\alpha}(t) = [-\alpha^-(t), \alpha^+(t)]$ taken relatively to the clock's value, leading to $\boldsymbol{A}(t) = [C(t) - \alpha^-(t), C(t) + \alpha^+(t)]$.

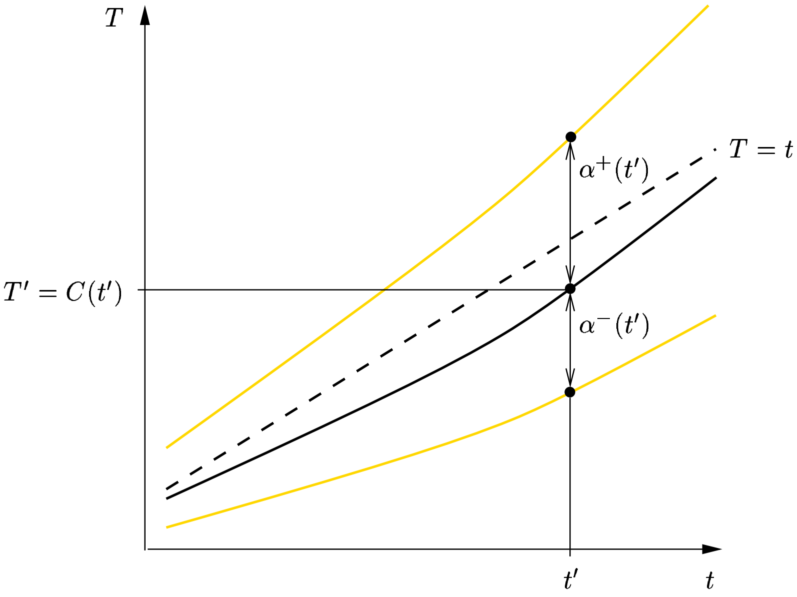


Figure 1.2: *Accuracy Interval*

For interval-based clock synchronization, we hence assume that each node p in the system is equipped with an *interval clock* that continuously displays its local accuracy interval $\boldsymbol{A}_p(t)$. An *interval-based (external) clock synchronization algorithm* is in charge of maintaining $\boldsymbol{A}_p(t)$ so that the following is guaranteed:

(R1) *Precision requirement*: There is some *precision* $\pi_{\max} \geq 0$ such that $|C_p(t) - C_q(t)| \leq \pi_{\max}$ for all nodes p, q that are non-faulty up to real-time t.

(R2) *Accuracy requirement*: The interval of accuracies $\boldsymbol{\alpha}_p(t)$ is such that $-\alpha_p^+(t) \leq C_p(t) - t \leq \alpha_p^-(t)$ for all nodes p that are non-faulty up to real-time t.

Note that (R2) can be used to specify both external and internal synchronization, simply by requesting $\alpha_p^+(t)$, $\alpha_p^-(t)$ to be less than a fixed accuracy $\alpha_{\max}$. Furthermore, we restrict our attention to $\alpha_p^-(t), \alpha_p^+(t) = \mathcal{O}(t)$, where the implied constant[†] is (much) smaller than 1, which excludes "degenerated" cases because $C_p(t)$ must be within a linear envelope of real-time here, cf. [6].

All interval-based clock synchronization algorithms developed so far make use of the basic structure of the generic algorithm introduced and analyzed in [55], see Chapter 3. At each node p, the following steps are periodically executed leading to a round-based execution well-known from traditional clock synchronization algorithms:

1. When $C_p(t) = kP$, $k \geq 1$ denoting the current round, a *clock synchronization packet* (CSP) containing $\boldsymbol{A}_p(t)$ is broadcast to each node within the system.
2. Upon reception of a CSP, the received accuracy interval is preprocessed to make it compatible with the accuracy intervals of the other nodes received during the same round.
3. When $C_p(t) = kP + \Lambda$ for some suitable $\Lambda > 0$, an interval-valued *convergence function* is applied to the set of preprocessed intervals to compute and subsequently enforce an improved accuracy interval.

Two basic operations are required in step 2, where the exchanged intervals are made compatible with each other while preserving the inclusion of real-time: First, *delay compensation* is applied to the interval received in a CSP to account for the effects of transmitting an accuracy interval over a network. To account for the maximum transmission delay uncertainty, the received interval must be enlarged appropriately. Second, *drift compensation* is used to shift the resulting interval to some common point in real-time by means of the local clock $C_p(t)$. Since clocks may have a non-zero drift, a sufficient enlargement ("deterioration") of the interval is required here. Note that drift compensation must also be performed continuously by the local interval clock during the remainder of each round. All these issues are the essence of Chapter 3.

In step 3, a suitable *convergence function* is applied to the set of preprocessed accuracy intervals. It is in charge of providing a new (smaller) accuracy interval for the local interval clock that guarantees (R1) and (R2), despite of some possibly faulty input intervals. In fact, it is solely the convergence function that determines both performance and fault-tolerance degree of our interval-based clock synchronization algorithm, see Chapter 5.

[†] Throughout this thesis, we frequently use the $\mathcal{O}(\cdot)$-notation to characterize the order of magnitude of less important terms. Recall that $f(x) = \mathcal{O}(g(x))$ for $x \to x_0$ if there is some (reasonably small) positive constant M such that $|f(x)| \leq M|g(x)|$ for $x \to x_0$. In addition, we make use of asymptotic approximations like $(1 \pm x)^{-1} = 1 \mp x + \mathcal{O}(x^2)$ valid for $x \to 0$.

Generally speaking, the major advantage of interval-based clock synchronization is its ability to provide each node with a local on-line bound on the own clock's deviation from real-time. Since accuracy intervals are maintained dynamically, they are quite small on the average, which compares favorably to the "static" worst case accuracy bounds known for traditional clock synchronization algorithms. The price to be paid for this additional information, however, is the need of explicit bounds on certain system parameters like transmission delays. These bounds can either be compiled statically into the algorithm from a-priori information or, preferably, determined on-line through an accurate *round-trip-based transmission delay measurement*, see Chapter 6.

In fact, our ambitious goal of a precision/accuracy in the 1 μs-range makes it inevitable to utilize on-line bounds on the maximum clock drift provided by a suitable *clock rate synchronization algorithm*. More specifically, the instantaneous clock rates $v_p(t) = dC_p(t)/dt$ of all non-faulty clocks have to be synchronized in such a way that the following is ensured:

(R3) *Consonance requirement*: There is some *consonance* γ such that $|v_p(t) - v_q(t)| \leq \gamma$ for all nodes p, q that are non-faulty up to real-time t.

(R4) *Drift requirement*: There is some *drift* δ_p such that $|v_p(t) - 1| \leq \delta_p$ for all nodes p that are non-faulty up to real-time t.

Obviously, requirement (R3) calls for an internal and (R4) for an external synchronization of the clock rates. The interval-based algorithm introduced and analyzed in [60] effectively reduces the maximum consonance/drift without necessitating highly accurate and stable oscillators at each node, see Chapter 4.

1.3 Outline

The thesis is structured in following chapters:

Chapter 2 provides a description of our *Network Time Interface* (NTI) M-Module supporting high-accuracy external clock synchronization by hardware, see also [20] and [21]. Designed for maximum network controller and CPU independence, the NTI provides a turn-key solution for adding high-resolution synchronized clocks to distributed real-time systems built upon hardware with M-Module interfaces. The NTI is built around our custom VLSI chip *Universal Time Coordinated Synchronization Unit* (UTCSU), which contains most of the hardware support required for interval-based clock synchronization, see [61] and [63]. The centerpiece of the UTCSU is a local clock based on an adder and driven by a fixed-frequency oscillator. This novel clock design allows a fine grained rate adjustability apt for maintaining both local time with linear continuous amortization and accuracy information as needed in interval-based clock synchronization. Additional features incorporated in our UTCSU are facilities to timestamp clock synchronization data packets, interfaces to couple GPS receivers, some application support as well as sophisticated self-test machinery. Apart from addressing design and engineering issues of the chip, we also provide a basic programming model.

Chapter 3 develops and analyzes a simple interval-based algorithm suitable for *fault-tolerant clock state synchronization*, see also [55] and [54]. Unlike usual internal synchronization approaches, our convergence function-based algorithm provides approximately synchronized clocks maintaining both precision and accuracy w.r.t. external time. This is accomplished by means of a time representation relying on intervals that capture external time, providing accuracy information encoded in interval lengths. The algorithm, which is generic w.r.t. the convergence function and relies on either instantaneous correction or continuous amortization for clock adjustment, is analyzed by utilizing a novel, interval-based framework for establishing worst case precision and accuracy bounds subject to a fairly detailed system model. Apart from individual clock rate and transmission delay bounds, our system model incorporates non-standard features like clock granularity and broadcast latencies as well. Relying on a suitable notion of internal global time, our analysis unifies treatment of precision and accuracy, ending up in striking conceptual beauty and expressive power.

Chapter 4 addresses the problem of synchronizing the rate of clocks in a fault-tolerant distributed system, see also [59] and [60]. Contrived to bring the rate (i.e., speed) of all non-faulty clocks in accordance, *clock rate synchronization algorithms* are very similar to usual state synchronization ones. Major differences, however, arise from the fact that the quantities to be synchronized are not directly accessible and that they do not proceed linearly with time. Relying on an interval-based paradigm, we introduce a basic system model and suitable building blocks for a generic convergence function-based rate synchronization algorithm. Our rigorous analysis of the achievable consonance (i.e., mutual rate deviation) and drift (i.e., deviation towards the ideal rate of 1 Sec/sec) reveals that it is the clocks' rate stability (i.e., maximum rate change per unit of time) that takes over the role of maximum hardware drift rate in traditional clock synchronization approaches.

Chapter 5 is devoted to an interval-based convergence function based on Marzullo's function leading to our *optimal precision* algorithms OP-STATE and OP-RATE. For the first one we provide worst case bounds for precision and accuracy subject to a realistic system model including an elaborate hybrid fault model covering send and receive omissions up to arbitrary faults. Apart from revealing that clock granularity and discrete rate adjustment techniques seriously affect worst case precision and accuracy, we show that OP-STATE achieves optimal worst case precision and optimal maximum clock adjustment, see also [51]. Based on these results, we give worst case bounds for consonance and drift of algorithm OP-RATE that is in charge of synchronizing the clock's rate.

Chapter 6 combines the clock state and rate algorithm to a complete synchronization algorithm. We explore this generically given algorithm by making improvements to the analysis of the clock state algorithm, taking into account that rate synchronization depends only loosely on state synchronization. When joining both frameworks the conceptional exquisiteness reaches its crest in the invention of an *improved internal global time* that re-starts at state resynchronization instances and progresses as internal global rate does. In addition, we motivate and devise methods to measure other system parameters on-line, most importantly, data about the *transmission delay* of packets. A summary of extensions to our approach can be found here as well.

The following table summarizes the keywords of the various chapters of this thesis.

chapter	*heading*	*keywords*
1	preface	SynUTC, round-based algorithms
2	hardware support	distributed systems, MA-Module, ASIC, GPS
3	state synchronization	clock granularities, accuracy/precision intervals
4	rate synchronization	oscillator stability, rate/consonance intervals
5	convergence functions	fault models, Marzullo, worst case analysis
6	complete algorithm	internal global time/rate, on-line measurements

Table 1.1: *Keywords of the Chapters*

◇

Chapter 2

HARDWARE SUPPORT FOR CLOCK SYNCHRONIZATION

2.1 Introduction

Dealing with time is inherent to the real-time computing domain, since applications need to interact both correctly and timely with the environment. In order to meet specified time constraints, activities like timestamping external events, scheduling resources, and initiating actions require an advanced time service. A clock is the physical basis of such a service, usually composed of a quartz oscillator that drives a hardware counter. Although small-scale systems are content with a central clock, modern large-scale real-time systems become necessarily distributed due to their spatial outspread and fault-tolerance requirements, see [67]. Exemplary applications that exhibit such characteristics include transportation (e.g., avionics), manufacturing (e.g., rolling-mill), energy-providing (e.g., nuclear power plant) or scientific (e.g., radio-astronomy) systems. The distributed nature aggravates the installation of a time service considerably, because autonomous running clocks have a tendency to drift apart or might fail grossly in their rate or state. Hence clocks need to be synchronized by virtue of a suitable algorithm executed on each node. Usually, both performance and correctness of the system is vitally affected by the degree of synchrony, see [27].

Most synchronization schemes are purely software-based, i.e., they run on off-the-shelf processing and networking hardware, providing a precision in the 10 ms-range only. A considerably better precision can be achieved with dedicated hardware support. In the fieldbus area, for instance, there are some more recent research activities, like the CAN-Bus project by [13], that target a few ms. A time service with precision in the 10 μs-range can be built on top of the pioneering *Clock Synchronization Unit* (CSU) described in [22]. Similar ideas are exploited in the hardware assisted clock synchronization scheme of [45]. Even smaller precisions can be attained by means of clock voting with phase locked loops, see [46], but we do not consider such solutions because of their extra clocking network.

The most widely used external clock synchronization scheme is undoubtly the *Network Time Protocol* (NTP) designed for disseminating UTC among workstations throughout the Internet, see [38]. Under realistic conditions, worst case accuracies of approximately 20 ms were observed by [71]. There are also some recent solutions of the external synchronization problem in the LAN domain, see [11], [55] or [72]. The latter describes a software-based approach that "sprays" external time obtained from GPS satellites into broadcast-type LANs with accuracies in the 10 μs-range.

It is well-known that the precision achieved by any clock synchronization approach depends heavily on the uncertainty (i.e., variability) of the end-to-end delay arising in the exchange of time information. Hence, the quality of clock synchronization is primarily determined by the properties of the communications subsystem, which can be classified on the bridged physical distance as follows:

(I) If the interconnected nodes are only a few 10 meters apart, a dedicated and usually fully connected clocking "network" exhibiting small and constant propagation delays is sometimes affordable. This setting allows the construction of phase-locked-loop clocks with clock voting for increased fault-tolerance, which can provide a precision down to the ns-range, see [46] for an overview.

(II) Nodes within a few 100 meters of each other are usually interconnected by a packet-oriented communications subsystem, where sending data packets is the only means for exchanging (time) information. Almost any work on clock synchronization addresses this type of systems, preferably for fully connected point-to-point networks. Typical distributed real-time systems employ (redundant) LANs like fieldbusses based on shared broadcast channels, which provide almost deterministic transmission delays but incur a considerable medium access uncertainty. Purely software-based solutions achieve a precision in the ms-range, which can be brought down to the μs-range with moderate hardware support, see Section 2.5.

(III) World-wide distributed systems connected via long haul networks constitute an entirely different class of systems. In fact, they have to cope with end-to-end transmission delays that are potentially unbounded and highly variable due to the inevitable queueing delays at intermediate gateway nodes (e.g. in case of congestion and/or failures). The most prominent external clock synchronization scheme for such settings is undoubtly the NTP.

In our SynUTC project, outlined in [50] and Section 1.1, we focus on external clock synchronization for large-scale distributed real-time systems and aim 1 μs as both precision and accuracy, hence type (II) of the above classification. Note that our approach can also be adopted to more general topologies commonly known as WANs-of-LANs, provided that all gateway nodes are also equipped with the NTI, see [50] and [62]. Of course, such ambitious goals can only be achieved with proper hardware support. The appropriate features are presented in the remaining sections of this chapter, which is organized as follows:

After a brief introduction of the overall system architecture in Section 2.2, we present our Network Time Interface (NTI) M-Module in Section 2.3, which hosts the Universal Time Coordinated Synchronization Unit (UTCSU) described in Section 2.4. A reasonably detailed discussion of related work including "pure-GPS" solutions can be found in Section 2.5, and finally Section 2.6 concludes with short summary of our accomplishments as well as some directions of further work.

2.2 System Architecture

From an abstract point of view, modern real-time systems are physically distributed networks of nodes hosting hard- and software resources for providing application-specific services that exhibit predictable behavior. Our focus rests on the time service, a basic subsystem of any real-time system, that offers its pertinent timing features to a number of higher-level services.

In order to cope with the complexity of a typical real-time system, we decompose it into so called *synchronization subnets* (SSN), similar to [38]. Each SSN comprises a collection of nodes interconnected by a packet-oriented data network. Regarding the contribution to the time service and considering cost-performance tradeoffs, we can distinguish between four types of nodes:

- *Client-nodes* execute the synchronization algorithm in a passive way. More precisely, they merely glean synchronization data from the attached SSN and adjust their local clocks accordingly, see [7].

- *Secondary-nodes* execute the synchronization algorithm in an active way by periodically exchanging synchronization data among other non-Client nodes within the SSN. Their purpose is to provide fault-tolerance, most importantly, guaranteed precision even in case of total loss of external time (aka. *flywheeling*).

- *Primary-nodes* behave like Secondary-nodes but provide access to external time, e.g., via GPS receivers, as well. Multiple Primary-nodes are useful for increasing the fault-tolerance level w.r.t. faults of external time sources.

- *Gateway-nodes* act as Secondary-nodes in two or more SSNs, making time dissemination between SSNs feasible. Such nodes have additional functionalities to control the system-wide flow of synchronization data.

An example of a simple system architecture is given in Figure 2.1, showing three SSNs linked by a single and a double Gateway-node connection. Note that, irregardless of this example, a tree-like hierarchy is not mandatory for our approach.

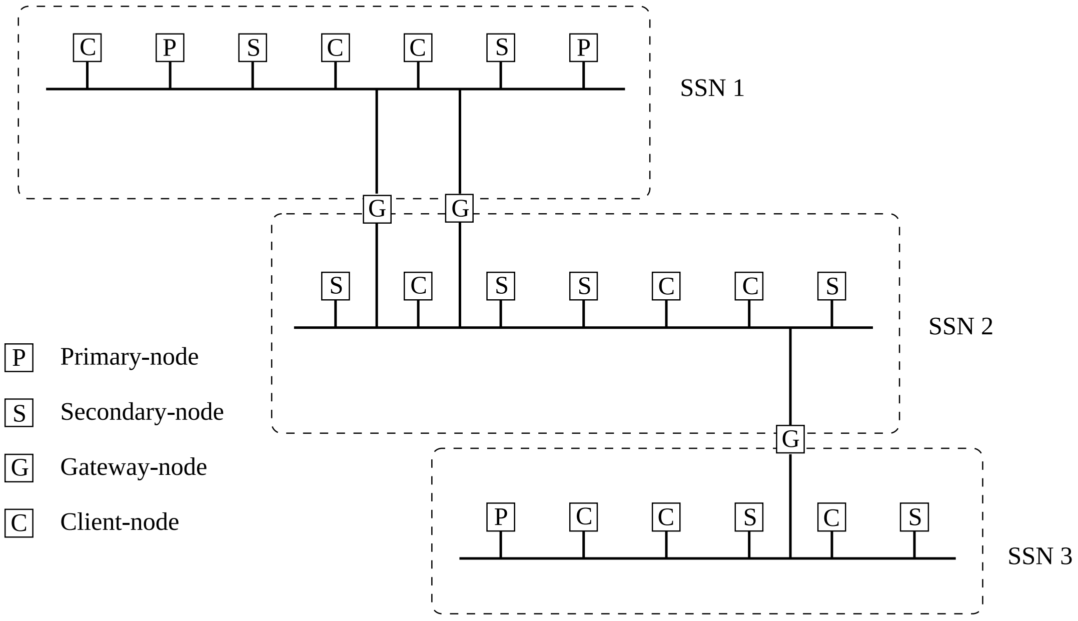


Figure 2.1: *System Architecture*

Although nodes may have different functionalities, their hardware architecture remains to be uniform. Figure 2.2 outlines the basic hardware components required for clock synchronization. Each node must be equipped with a hardware clock (our UTCSU-ASIC, see Section 2.4), a general purpose CPU (the node's central processor or, preferably, a dedicated microprocessor or microcontroller) responsible for executing the software part of the clock synchronization algorithm, and a *Communications Coprocessor* (COMCO), which provides access to the network (e.g., Ethernet) by reading/writing data packets from/to (shared) memory independently of CPU operation (e.g., via DMA). For external synchronization purposes, some nodes must also be provided with external time sources like GPS receivers.

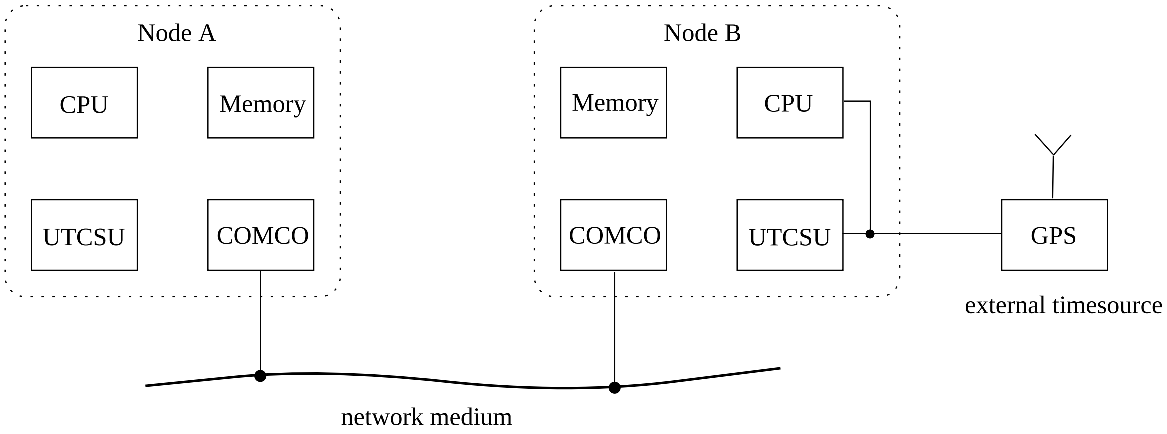


Figure 2.2: *Basic Clock Synchronization Hardware Architecture*

2.3 Network Time Interface (NTI)

As already pointed out in Section 2.1, our approach targets distributed systems consisting of computing nodes interconnected by an ordinary packet-oriented data network, so we did not bother ourselves with developing a fully-fledged node hardware, but rather to extend existing CPU boards with adequate support. Mezzanine busses are certainly the easiest way to accomplish this, and given the M-Modules simplicity, robustness, size, and low cost, we eventually decided to implement the NTI as an MA-Module. Starting with a requirements analysis in Subsection 2.3.1, an overview of the resulting features of the NTI M-Module is provided in Subsection 2.3.2. A short description of the NTI's hardware/software interface is provided in Subsection 2.3.3, and Subsection 2.3.4 outlines how we incorporated the NTI into a start-of-the-art real-time kernel.

2.3.1 Requirements

Providing hardware support for highly accurate/precise clock synchronization is primarily driven by the requirement of exact timestamping of CSPs at both sending and receiving side. In fact, the work of [29] revealed that even n ideal clocks cannot be synchronized with a worst case precision less than $\varepsilon\,(1 - 1/n)$ in presence of a *transmission/reception time uncertainty* ε, which is defined as the variability of the difference between the real times of CSP timestamping at the peer nodes. Unfortunately, there are several steps involved in packet transmission/reception that could contribute to ε, cf. [24]:

1. Sender-CPU assembles the CSP
2. Sender-CPU signals sender-COMCO to take over for transmission

3. Sender-COMCO tries to acquire the network medium
4. Sender-COMCO reads CSP data from memory and pushes the resulting bit stream onto the medium
5. Receiver-COMCO pulls the bit stream from the medium and writes CSP data into memory
6. Receiver-COMCO notifies receiver-CPU of packet reception via interrupt
7. Receiver-CPU processes CSP

Purely software-based clock synchronization approaches perform CSP timestamping upon transmission resp. reception in steps 1 resp. 7, which means that ε incorporates the medium access uncertainty ($3 \to 4$), any variable network delay ($4 \to 5$), and the reception interrupt latency ($6 \to 7$). The first one can be quite large for any network utilizing a shared medium, and the last one is seriously impaired by code segments with interrupts disabled. Fortunately, in our LAN-based setting, we can safely neglect the contribution from $4 \to 5$ since there are no (load- and hop-dependent) queueing delays from intermediate gateway nodes. Therefore, the resulting transmission/reception uncertainty emerges primarily from $1 \to 4$ resp. $5 \to 7$ at the sending resp. receiving node itself.

In an effort to reduce ε, clock synchronization hardware should thence be placed as close as possible to the network facilities. Ideally, a CSP should be timestamped at the sender resp. receiver exactly when, say, its first byte is pushed on resp. pulled from the medium. However, this needs support from the interior of the COMCO, which is usually not available. In order to support existing network controller technology, a less tight method of coupling has to be considered.

For this purpose, our NTI uses a refinement of the widely applicable DMA-based coupling method proposed in [24]. The key idea is to insert a timestamp on-the-fly into the memory holding a CSP in a way that minimizes the transmission/reception uncertainty. More specifically, a modified address decoding logic for the memory is used, which

1. generates trigger signals that sample a timestamp into dedicated UTCSU registers when a certain byte within the transmit resp. receive buffer for a CSP is read resp. written,
2. transparently maps the sampled transmit timestamp into some portion of the transmit buffer.

Note that this special functionality is only present when a transmit/receive buffer is accessed by the COMCO, whereas CPU-accesses are just plain memory accesses.

To illustrate the entire process of CSP timestamping, we briefly discuss one possible scenario depicted in Figure 2.3; alternative scenarios may be found in [62].

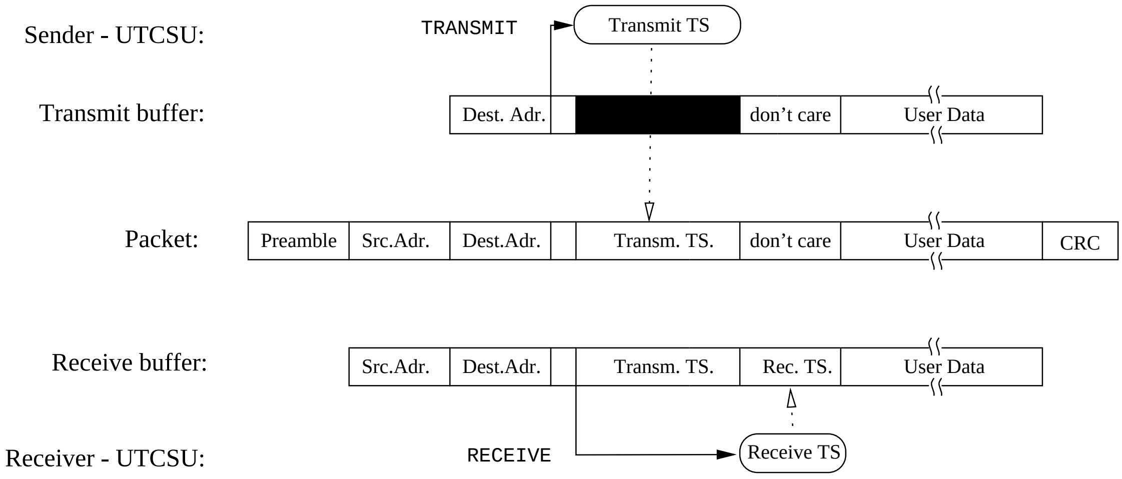


Figure 2.3: *Packet Timestamping*

Whenever the COMCO fetches data from the transmit buffer holding the CSP for transmission, it has to read across the particular address that causes the decoding logic to generate the trigger signal `TRANSMIT`. Upon occurrence of this signal, the UTCSU puts a *transmit timestamp* into a dedicated sample register, which is transparently mapped into a certain succeeding portion of the transmit buffer and hence automatically inserted into the outgoing packet. Note that the trigger address and the mapping address may be different. By the same token, when the COMCO at the receiving side writes a certain portion of the receive buffer in memory, the trigger signal `RECEIVE` is generated by the decoding logic, which causes the UTCSU to sample the *receive timestamp* into a dedicated register. Subsequently, the timestamp can be saved in an unused portion of the receive buffer by the CPU upon reception notification or by a similar transparent mapping technique, see [62] for more details.

The proposed approach works for any COMCO that directly accesses CSP data in memory. Suitable chipsets are available for a wide variety of networks, ranging from fieldbusses like Profibus over Ethernet up to advanced high-speed FDDI or ATM networks. COMCOs that provide on-chip storage for entire packets, as is the case for most CAN controllers, however, cannot be used, unless clock synchronization is explicitly supported by exporting the required trigger signals, see [23]. In this (ideal) case, the UTCSU can be even more tightly coupled to the COMCO, which leads to a further reduction of ε.

It is worth mentioning that assessing the transmission/reception uncertainty of a particular COMCO usually requires some experimental evaluation, see Section 2.3.4. In fact, since CSP timestamping occurs in step 4 resp. 5 of the data transmission/reception sequence introduced earlier, the only activities that still contribute to ε are the "ongoing" data transmission and the bus arbitration necessary for COMCO memory write cycles upon CSP reception. The former uncertainty, i.e., the time between fetching a byte from the transmit buffer and trying to deposit it in the receive buffer, obviously depends on the internal architecture (FIFOs etc.) of the COMCOs at both ends. Whereas adjusting the trigger position of the transmit/receive timestamp may help in reducing/circumventing certain impairments, it is nevertheless not easy to find and justify a suitable choice without actual measurements. In general, we should point out that numerous technical hurdles have to be surmounted to make this approach working properly, see [17] for the twisted details of a particular prototype implementation based on a FORCE CPU-30 board.

2.3.2 Design

M-Modules [34] are an open, simple, and robust mezzanine bus interface primarily designed for VME carrier boards, which are commonly used in Europe. *MA-Modules* are enhanced M-Modules, providing a 32 bit data bus instead of the 16 bit one of the original M-Modules. The address space consists of 256 bytes *I/O-space* accessible via the standard M-Module interface and up to 16 MB of *memory-space* addressed by multiplexing the MA-Module data bus. The asynchronous bus interface requires the module to generate an acknowledge signal for termination of a bus cycle only, thus minimizing the control logic on-board the M-Module. Further signals in the M-Module interface comprise a single vectorized interrupt line and two additional DMA control lines. The unit construction design of the 146 × 53 mm (single-height) M-Modules provides a peripheral I/O D-sub connector on the front panel, two plug connectors to the carrier board for peripheral I/O and MA-interface, and an intermodule port connector for interconnecting several M-Modules.

We found MA-Modules well-suited for crafting the evaluation prototype of our clock synchronization hardware, mostly driven by the major requirement about CPU and COMCO independence, recall Section 2.3.1. In this and the following subsections, we will provide a short overview of the NTI design; consult [19] for all technical details. Figure 2.4 shows the major components.

The UTCSU-ASIC contains most of the dedicated hardware support for interval-based clock synchronization, see Section 2.4 for its features. It is driven by an on-board temperature-compensated (TCXO) or ovenized (OCXO) quartz oscillator; alternatively,

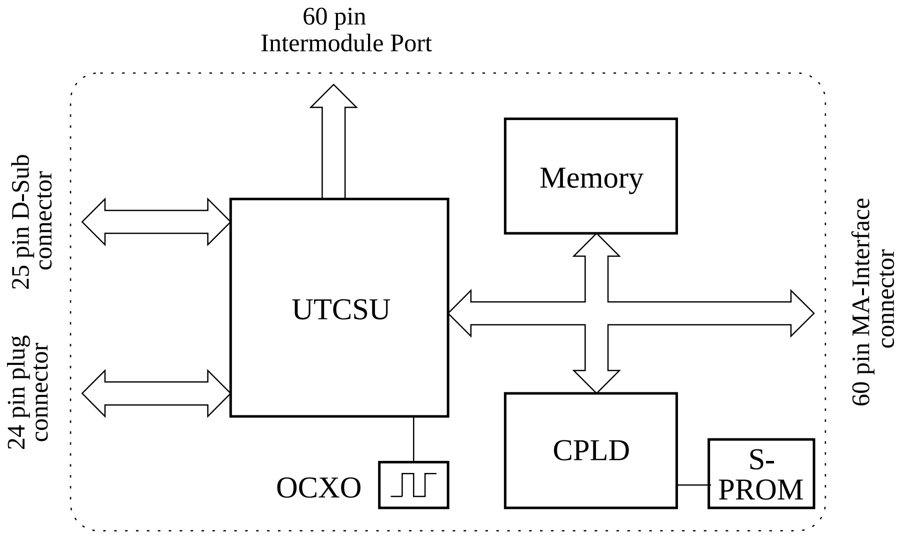


Figure 2.4: *NTI Block Diagram*

an external frequency source like the 10 MHz output of an GPS receiver can be used.

All UTCSU application time/accuracy-stamp inputs and duty timer outputs as well as all interfaces to GPS receivers are available via the M-Module's front-panel 25 pin D-sub connector. In addition, all receive and transmit time/accuracy-stamp signals are fed to the carrier board via the 24 bit pin plug connector. The M-Modules' intermodule port is eventually used to export the NTPA-bus required for future extension modules, and to facilitate internal connection of modularized GPS receivers. Note that high-speed opto-couplers or transceivers are provided for all inputs to ensure a decoupled and reliable interface.

The memory serves as control and data interface between the CPU and the COMCO, providing the special functionality for COMCO accesses as outlined in Section 2.3.1. It consists of four 64KB × 16 bit SRAM chips and supports byte, word, and longword read/write accesses. In Section 2.3.3, the memory map of the current version of the NTI will be explained in some detail.

All required decoding and glue logic of the NTI is incorporated in a single, in-circuit programmable *complex programmable logic device* (CPLD), which has been programmed in VHDL. It adapts the UTCSU and the memory to the MA-Module interface, forwards interrupt requests from the UTCSU to the carrier-board, generates the acknowledgment signal terminating a bus cycle, and gives access to the serial PROM; this read only memory stores identification and revision information according to the M-Module specification, see [34]. Note carefully that it is just a matter of re-programming the VHDL code of the CPLD to support a different COMCO with our NTI.

2.3.3 Hardware/Software Interface

All accesses to UTCSU registers and NTI memory are performed by addressing the M-Modules memory-space. As explained in Section 2.3.1, read/writes of the COMCO require additional logic to provide timestamping functionalities. To distinguish between CPU and COMCO accesses, the CPLD maps two address regions to the same physical memory as illustrated in Figure 2.5.

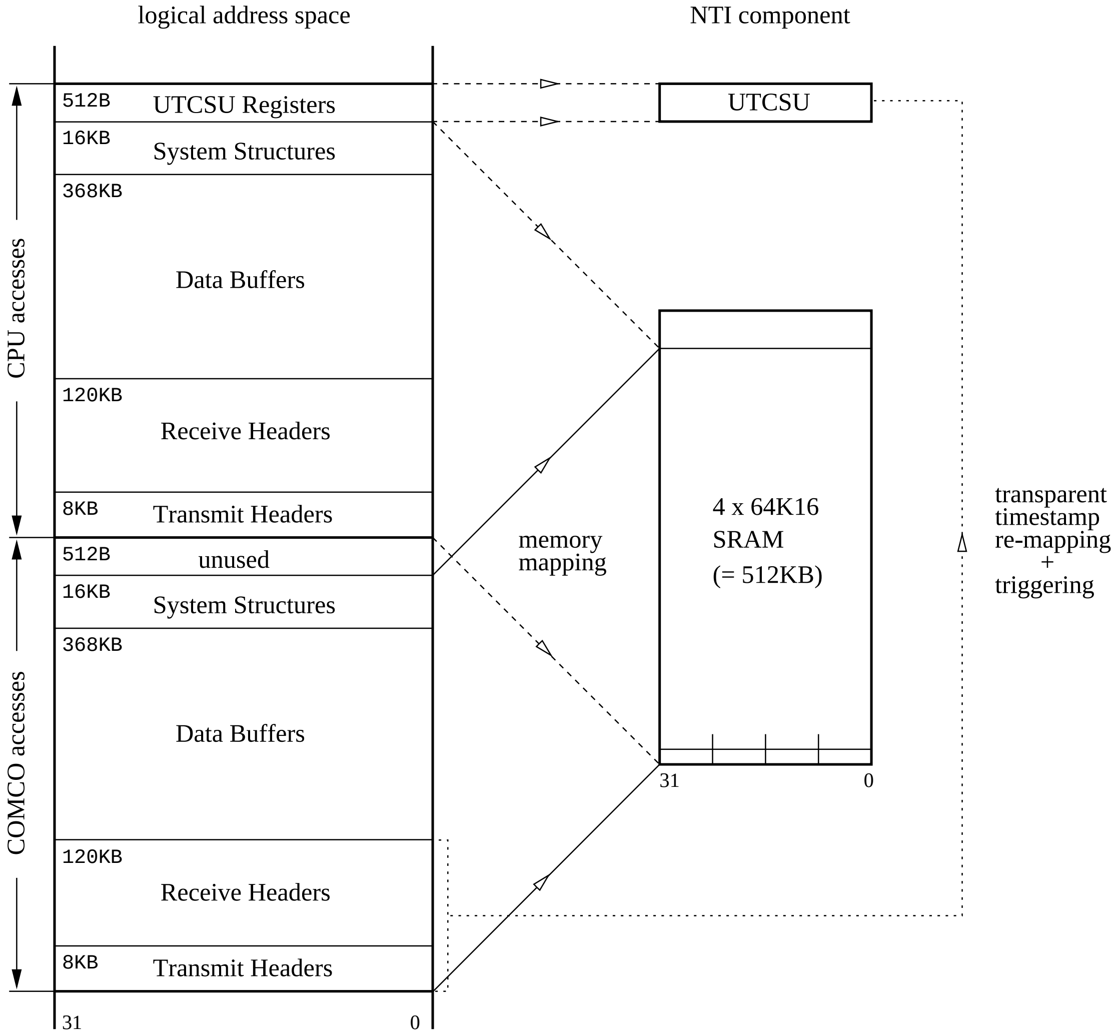


Figure 2.5: *Memory Map of the NTI*

On top of the memory map is the NTI memory's 512 KB address region for CPU-accesses, which is decoded without special functionality, beginning with a 512 byte segment containing the UTCSU registers. The 512 KB region for COMCO-accesses to NTI memory starts at address 0 and is divided into four sections: The *System Structures* section holds the command interface and system data structures required by the COMCO, the *Data Buffers* are available for ordinary packet data. Special functionalities apply only to accesses in the *Receive Headers* resp. *Transmit Headers* sections, which hold packet-specific control and routing information (e.g., source and destination addresses) for received resp. transmitted CSPs.

The CPLD is currently programmed to support the 82596CA Ethernet coprocessor from INTEL using 64 byte receive and transmit headers. Figure 2.6 outlines the offsets within each header that need to be supervised here.

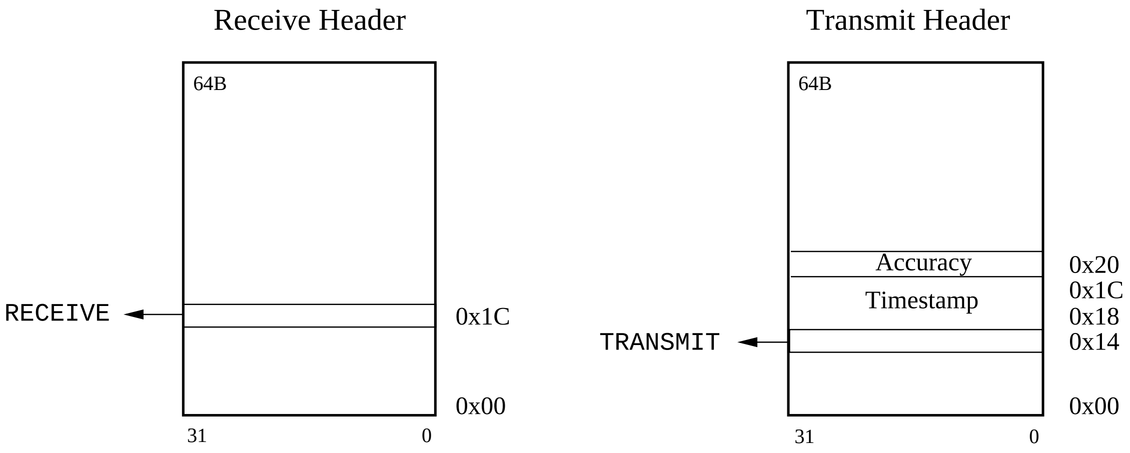


Figure 2.6: *Receive and Transmit Header*

When the 82596CA writes offset 0x1C within a receive header upon reception of a CSP, the timestamp trigger signal `RECEIVE` for the UTCSU is generated. In addition, the base address of the accessed receive header is stored into a dedicated NTI-register to facilitate further interrupt processing, as explained below. Similarly, when the 82596CA reads offset 0x14 within a transmit header upon transmission of a CSP, a timestamp trigger signal `TRANSMIT` is issued to the UTCSU. Since the UTCSU registers holding the sampled time/accuracystamp are mapped to the memory addresses with offsets 0x18–0x20 in the transmit header, they are transparently inserted into the outgoing data packet.

In addition to the UTCSU registers and the shared memory, there are also a few registers provided by the NTI itself, primarily for the purpose of interrupt handling. They are accessible via the M-Modules I/O-space according to the memory map in Figure 2.7.

The *Receive Header Base* register at address 0x00 is required for correctly assigning receive timestamps to data packets: After the UTCSU has sampled a receive time/accuracystamp, it must be moved to an unused portion of the appropriate CSP before the next CSP drops in. This can be done in an ISR activated by a `RECEIVE` transition interrupt,

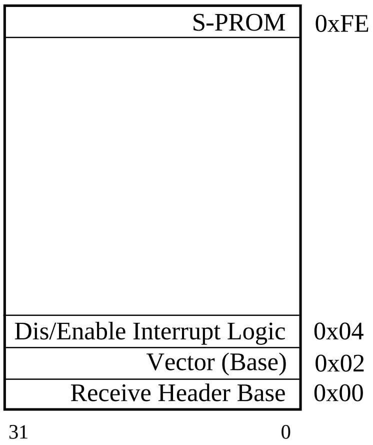


Figure 2.7: *Memory Map I/O-space of the NTI*

but which cannot reliably determine the address of the receive header associated with the sampled timestamp, thus the NTI latches this address into the *Receive Header Base* register upon the occurrence of the `RECEIVE`-signal. There are of course alternatives, which, however, do not work in general: For example, one might try to move the timestamp in the packet reception ISR, where the base address of the receive buffer is of course known. Unfortunately, this might be too late for avoiding a timestamp loss in case of back-to-back CSPs. Also inappropriate are schemes that try to exploit a sequential order of received packets, since there might be CSPs that trigger a timestamp but are eventually discarded, e.g., due to an incorrect CRC.

Apart from the access-byte to the M-Module's serial PROM at address 0xFE, there are two additional NTI registers controlling interrupt generation: The *Vector (Base)* register at address 0x02 can be used to program the interrupt vector generated upon an UTCSU interrupt. Note that the final vector also includes the state of the three UTCSU interrupt pins `INT-T`, `INT-N`, and `INT-A`. Accessing register *Dis/Enable Interrupt Logic* at address 0x04 eventually enables (further) NTI interrupts; it is usually written immediately prior to leaving the interrupt service routine.

2.3.4 Integration

This section briefly surveys how we incorporated the NTI and our clock synchronization software into the state-of-the-art[†] industrial multiprocessing/multitasking real-time kernel

---

[†] Note that we are of course aware of the fact that customary kernels are quite inadequate for building well-founded distributed real-time systems. We think, however, that one should try to gradually introduce sound concepts into existing systems, rather than persuade industry to discard familiar (ready-to-use but insufficient) technology in favor of a novel (immature but sufficient) one.

pSOS^{+m} from Integrated Systems, Inc. Figure 2.8 outlines the complete software structure of a node, including the underlying hardware.

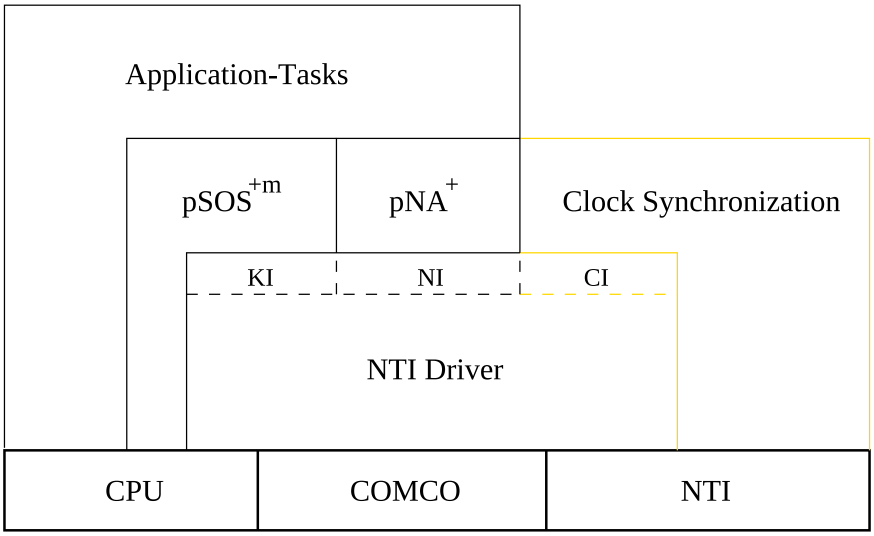


Figure 2.8: *Software Structure of a Node*

The entire NTI software is embedded in a pSOS^{+m} add-on that effectively hides clock synchronization from the application tasks. The heart of this add-on is the *NTI driver* [47], which actually multiplexes three different interfaces to the COMCO:

1. *Kernel Interface* (KI): pSOS^{+m} supports multiprocessing by providing remote objects (tasks, queues, semaphores, etc.) that are internally managed via RPCs. To keep the kernel reasonably independent of the particular communications network, a user-supplied KI is required that maps a simple message-passing interface to the particular COMCO.

2. *Network Interface* (NI): In addition to kernel services, application tasks can communicate with remote sites via TCP/IP sockets if the additional software component pNA^{+} is present. Like the pSOS^{+m} kernel, pNA^{+} is kept hardware-independent by means of a user-supplied NI, which is similar to the KI but plugs into a different message-passing interface.

3. *Clock Interface* (CI): The third component that requires network services is our clock synchronization algorithm. Again, a simple message-passing interface CI is sufficient here. Recall that CSPs sent/received via the CI are timestamped, as described in Section 2.3.1.

Viewed at the level of application tasks, a usual pSOS^{+m}/pNA^{+}-environment is at hand that provides synchronized clocks by means of the application support of the NTI. In fact, apart from the created computing and networking load, clock synchronization is performed totally transparent to the application.

The current version of the NTI driver has been developed for MOTOROLA's MVME-162 CPU (M68040 CPU + INTEL 82596CA Ethernet coprocessor) in conjunction with a passive VME carrier-board hosting the NTI M-Module, see [33] for a comprehensive documentation. A simple pSOS^{+m}-application has been written, which allowed us to assess the resulting transmission/reception time uncertainty ε. It turned out that ε is well below 500 ns for most CSP transmissions, except for a few instances where we encountered values up to 1.8 μs. Those large values are primarily caused by the suboptimal hardware architecture of our evaluation system, since any access has to pass several asynchronous bus interfaces and can hence occasionally experience large synchronizing delays. Moreover, since the passive carrier-board used is actually a CPU-module (ACQ i6040), it may happen that COMCO accesses have to wait for the completion of the on-board execution required for forwarding NTI-interrupts to the VME-bus. Consequently, we think that the actual ε of the 82596CA in conjunction with our NTI is below 200 ns.

As a final remark, we note that a transition to a different hardware only requires redevelopment of the network controller's part of the NTI driver (written in C) and perhaps some reprogramming of the VHDL code for the CPLD. In fact, we are considering a new version of the driver and the CPLD that supports ACQ's i6040 multiprocessor VME-CPU; this module utilizes an M68040 in conjunction with a M68EN360 QUICC network coprocessor and has 2 MA-Slots on board. Note that the i6040 is particularly suitable for our purpose, since it allows the CPU-32 on-chip the M68EN360 to operate concurrently with the M68040 (avoiding *companion mode*). Due to the fact that the MA-Slots can be accessed from the M68EN360 without disturbing the execution of the M68040, we could hence execute the clock synchronization software completely on-chip the M68EN360.

2.4 Universal Time Coordinated Synchronization Unit (UTCSU)

We packed most of the hardware support into the Universal Time Coordinated Synchronization Unit (UTCSU). A functional specification of it is given in Subsection 2.4.1, which encompasses features like our novel adder-based clock. The UTCSU is decomposed into physical units in Subsection 2.4.2, along with an explanation of implementation details. A programming model of the chip is given in Subsection 2.4.3, and Section 2.4.4 rounds off by discussing design methodology concerns.

2.4.1 Functional Specification

Modularity, flexibility, and performance issues influenced us to realize the UTCSU as an ASIC. The subsequent sections will shed more light on its interior, first by means of a functional specification, see [62] for a comprehensive version, and later on by documenting the development process, see [28] for meticulous details.

Interfacing

During requirement analysis of the chip we took much care to keep all interfaces as straightforward and general as possible. Our first step to specify the UTCSU is done by outlining its interfaces.

Coupling the UTCSU to the *System-Bus* enables communication with the other components. A wide range of existing bus architectures, employing data bus widths of 8, 16 or 32 bits, big/little endian byte ordering, different bus access times, interrupt schemes, etc. should be supported without additional glue logic. Note that the connection to the System-Bus is also a prerequisite for transparently mapping UTCSU-registers containing sampled timestamps into transmit/receive buffers in local memory, as required for exact timestamping of CSPs.

Meeting accuracy requirement of 1 μs, we decided to use GPS technology for injection of external time, hence the UTCSU must be able to interface various GPS timing-receivers. For applications with only moderate accuracy requirements or to increase fault tolerance, other sources of external time, like receivers for DCF77 or WWV, see [26], can be connected to the UTCSU via the same interface.

To support applications, there should be at least means to trigger actions at programmable points in time and capabilities for event timestamping. Moreover, to ease the implementation of more elaborate timing features, like the proposed timing processor of [15], without changing the UTCSU, we export time/accuracy information via a local unidirectional *NTPA-Bus* for external processing.

Depending on the type of node, these interfaces will have different significance. For instance, only Primary-nodes make use of the GPS interface, whereas Gateway-nodes utilize the interface to several COMCOs extensively. In summary, we give the first rather loose specification rule

Specification 2.1 (Interfaces) *The UTCSU must provide a versatile interface to commercially available system-busses and GPS timing-receivers, and additional interfacing facilities to timestamp and generate external pulses. Furthermore, it should export local time/accuracy information on a dedicated NTPA-Bus.*

Maintaining Local Time

The maintenance of local time at a node will be the core function of the UTCSU. In the first place, a format has to be established to encode local time. We use a straightforward binary coding scheme, closely following the NTP-time format from [38], where the upper 32 bits are interpreted as standard seconds relative to UTC and the lower 32 bits give the associated fractional part. For specification purposes we will use the following notation to denote a specific part of this format: (u, v) covers bits ranging from the most significant bit position u to the least significant one v, thus affecting a quantity given by $\sum_{i=v}^{u} b_i \, 2^i$, where $b_i \in \{0, 1\}$. An optional prefix "±" denotes a signed value. Table 2.1 defines NTP-bit numbers and their time equivalents for the ease of reference. It is divided in columns for the integer part $(+31, 0)$, the fractional part $(-1, -32)$ and the ultrafractional part $(-33, -64)$. Note that "n" stands for nano (10^{-9}), "p" pico (10^{-12}), "f" for femto (10^{-15}), "a" for atto (10^{-18}) and "z" for zepto (10^{-21}). For example, NTP-range $(-1, -8)$ pinpoints values up to slightly less then a second in multiples of 3.81 ms.

The clock synchronization algorithm is responsible for computing adjustments for both state and rate in order to meet accuracy and precision requirements. *State adjustment* means to add/subtract/set instantaneously a particular amount to local time, and *rate adjustment* means slowing down/speeding up its progression. Thus, we need to incorporate a clock in the UTCSU that represents local time in the NTP-based format and is adjustable in both rate and state.

Specification 2.2 (Local Time) *The UTCSU needs to host a digital clock representing local time over the NTP-range* $(+31, -24)$*, which is arbitrarily state adjustable for initialization purposes and rate adjustable not coarser than* 10^{-8} s/s*.*

As mentioned earlier, UTC is going to be our reference timescale, because of its worldwide availability through GPS technology and its legal character. However, UTC should be carefully considered in conjunction with real-time systems due to its non-chronoscopic nature, see [24]. The *Bureau International de l'Heure* (BIH) inserts or deletes leap seconds at predefined points in time in order to harmonize it with astronomically derived timescales, e.g., *Universal Time* (UT) versions. Strictly speaking, chronoscopic time standards, such as *Temps Atomique International* (TAI) or *GPS-Time*, should be given preference to establish a time service for real-time systems. Notwithstanding that, we proclaim

Specification 2.3 (Leap Seconds) *The clock for local time inside the UTCSU has to be equipped with facilities to handle leap seconds by either inserting or deleting pending ones at programmable points in time.*

integer part		*fractional part*		*ultrafractional part*	
bit#	*time equivalence*	*bit#*	*time equivalence*	*bit#*	*time equivalence*
+31	68 years	-01	500 ms	-33	116.42 ps
+30	34 years	-02	250 ms	-34	58.21 ps
+29	16 years	-03	125 ms	-35	29.10 ps
+28	8.5 years	-04	62.5 ms	-36	14.56 ps
+27	4.25 years	-05	31.25 ms	-37	7.28 ps
+26	2.13 years	-06	15.62 ms	-38	3.64 ps
+25	1.08 years	-07	7.81 ms	-39	1.82 ps
+24	194.18 days	-08	3.81 ms	-40	909.50 fs
+23	97.09 days	-09	1.90 ms	-41	454.75 fs
+22	48.55 days	-10	976.56 μs	-42	227.38 fs
+21	24.27 days	-11	488.28 μs	-43	113.69 fs
+20	12.17 days	-12	244.14 μs	-44	56.84 fs
+19	6.07 days	-13	122.07 μs	-45	28.42 fs
+18	3.03 days	-14	61.03 μs	-46	14.21 fs
+17	1.52 days	-15	30.51 μs	-47	7.11 fs
+16	18.02 hours	-16	15.25 μs	-48	3.55 fs
+15	9.10 hours	-17	7.62 μs	-49	1.77 fs
+14	4.55 hours	-18	3.81 μs	-50	888.18 as
+13	2.27 hours	-19	1.90 μs	-51	444.09 as
+12	1.13 hours	-20	953.67 ns	-52	222.04 as
+11	34.13 min	-21	476.83 ns	-53	111.02 as
+10	17.07 min	-22	238.41 ns	-54	55.51 as
+09	8.53 min	-23	119.21 ns	-55	27.76 as
+08	4.27 min	-24	59.60 ns	-56	13.88 as
+07	2.13 min	-25	29.80 ns	-57	6.94 as
+06	1.07 min	-26	14.90 ns	-58	3.47 as
+05	32 s	-27	7.45 ns	-59	1.73 as
+04	16 s	-28	3.72 ns	-60	867.36 zs
+03	8 s	-29	1.86 ns	-61	433.68 zs
+02	4 s	-30	931.32 ps	-62	216.84 zs
+01	2 s	-31	465.66 ps	-63	108.42 zs
+00	1 s	-32	232.83 ps	-64	54.21 zs

Table 2.1: *NTP-bit Numbers and their Time Equivalence*

Maintaining Accuracy Intervals

Dealing with the accuracy requirement means that local time has to follow UTC as close as possible. Unfortunately, UTC is neither directly observable nor permanently accessible, so only a range can be determined where UTC currently lies. More specifically, in our setting we use an *accuracy interval* $\boldsymbol{A}(t)$ capturing UTC in the sense that $t \in \boldsymbol{A}(t)\ \forall t \geq t_0$. This interval-based approach was introduced in [35], [37] and continued in [25]. The OSF/DCE time model [43] and newer versions of NTP [40] make use of this notion of time as well.

Our UTCSU is tailored to support external clock synchronization using the interval-based *clock validation technique* proposed in [50]. The algorithms employed herein rely on accuracy intervals that are maintained locally at a node by an *upper accuracy* $\alpha^+(t)$ and a *lower accuracy* $\alpha^-(t)$ meant relative to the local clock $C(t)$, such that $\boldsymbol{A}(t) = [C(t) - \alpha^-(t), C(t) + \alpha^+(t)]$. Observe that accuracies are always understood as maximum UTC deviations, otherwise we would be clairvoyant. The reason to make accuracies time-dependent is to enlarge them properly in order to account for maximum oscillator drifts, hence sustaining UTC inclusion. This process of linear *deterioration* would go on perpetually, however, periodic resynchronizations executed by a clock synchronization algorithm aim to shrink $\boldsymbol{A}(t)$, by exploiting knowledge of UTC provided by GPS receivers.

Specification 2.4 (Accuracy Interval) *The UTCSU needs to maintain an asymmetrical accuracy interval over the NTP-range* $(-8, -23)$*, which is arbitrarily adjustable and performs deterioration in the range of* $10^{-8} \ldots 10^{-4}$ *s/s.*

A few supplements to maintain accuracy intervals need to be brought to attention. The accuracy values should be permanently compared against bounding registers, generating an interrupt whenever the former exceeds the latter. Furthermore, a wrap-around during deterioration is undesirable; actually, in such situations we require that accuracy registers stay at their maximum value and trigger an interrupt.

Specification 2.5 (Accuracy Interval Overruns) *The upper and lower accuracy values inside the UTCSU need not wrap-around, and in case of exceeding the corresponding* $(-8, -23)$*-register* `BOUND`± *a dedicated interrupt should be raised.*

Adder-Based Clocks

Meeting the specified rate adjustability for local time or the deterioration dynamics for accuracies with ordinary clocks consisting of an oscillator pacing a hardware counter turns out to be a challenging task. As a matter of fact, conventional discrete rate adjustment techniques, like tick advancing/delaying employed in the CSU of [24], would require very high oscillator frequencies (around 100 MHz) for a smooth rate adjustment of 10^{-8} s/s. Moreover, our NTP-based time representation would enforce a binary oscillator frequency. One alternative pursued in [39] replaces the fixed-frequency oscillator by a voltage controlled one (VCO) that is put into a carefully engineered phase-locked loop (PLL).

We propose an *adder-based clock* (ABC) in order to maintain local time, where each oscillator tick entails an addition of a particular amount *clock-step* to a register holding local time. Any rate change can easily be achieved by varying clock-step, which goes into effect instantaneously and retains linearity, see Figure 2.9. To make this approach working properly, however, we have to expand the registers holding local time and clock-step on the less significant side to accumulate minute time amounts. Targeting a nominal frequency of 2^{24} Hz, it becomes necessary to extend the NTP-range of local time to $(+31, -51)$ in order to reach the demanded rate adjustability, since $2^{-51}/2^{-24} \approx 7.45 \cdot 10^{-9}$ s/s. Of course, timestamps derived for local time are only predicative in the NTP-range $(+31, -24)$, so that the smallest meaningful unit of time becomes to 59.6 ns, called *time granularity*. Bits $(-25, -51)$ are concealed from the outside and solely used for rate correction purposes, whereby the smallest perceivable unit of time becomes to 444.1 as, called *clock granularity*.

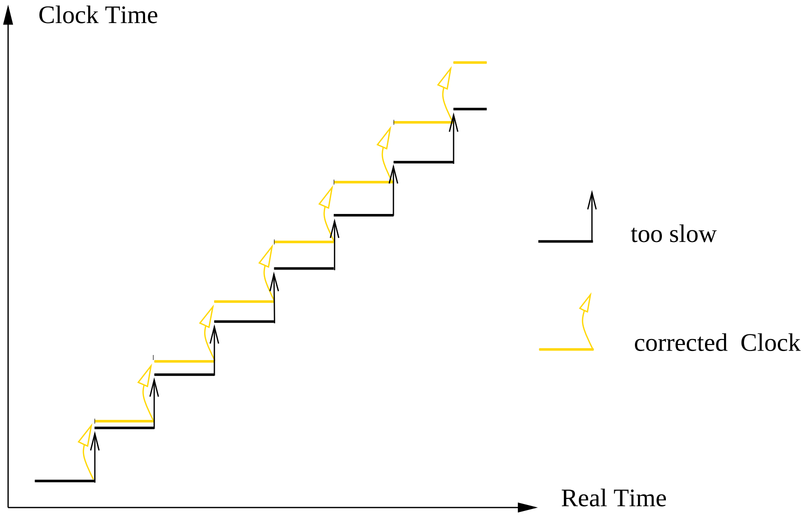


Figure 2.9: *Adder-based Clock Principle*

Beyond that, the adder-based approach allows us to use a non-binary oscillator frequency, e.g., 10 MHz from GPS timing-receivers. However, the NTP-range of local time has to be further extended to preserve the usually excellent quality of such frequency sources. We found it sufficient to append 8 bits on the less significant side, thus yielding a range of $(+31, -59)$, since truncation errors are limited to 2^{-59} s per oscillator tick. In particular, for a 10 MHz input frequency, the truncation errors impair the frequency by approximately $0.6 \cdot 10^{-11}$.

Specification 2.6 (ABC for Local Time Maintenance) *The UTCSU needs to implement an adder-based clock, composed of a* $(+31, -59)$*-register* `NTPTIME` *representing local time, an external oscillator input designed for frequencies in the range of* $2^{20} \dots 2^{24}$ *Hz and a* $(-20, -59)$*-register* `STEP` *to hold the increment for each oscillator tick.*

For a better understanding of adder-based clocks, let us briefly examine their operation in more detail. As mentioned above, a particular clock-step is added at each oscillator tick, which can be separated into the reciprocal of the nominal oscillator frequency and a fractional correction value. Since our clock exhibits a much coarser time granularity than clock granularity, an accumulation of correction values may cause discontinuities in the sequence of clock states. More specifically, when the correction values run up to the time granularity at a particular tick, the adder-based clock makes either no advancement or advances by twice the time granularity, depending on the accumulation nature. When correcting an oscillator drift of say 1 ppm with our clock specified in Specification 2.6, such effects occur roughly every 60 ms for using a nominal input frequency of 2^{24} Hz. For an in-depth treatment of granularities entangled in clock synchronization consult [55].

The adder-based approach is also well suited for maintaining the upper accuracy $\alpha^+(t)$ and the lower one $\alpha^-(t)$. In fact, each accuracy quantity can be tracked by an ABC similar to the one for local time, where one register is playing the accumulating role and another one is holding the deterioration for each oscillator tick. In the sequel we denote ABC components for the lower resp. upper accuracy by adding the suffix "-" resp. "+", or "±" to capture both.

The registers for an ABC representing accuracy span 45 bits internally, including a sign bit, thus ranging over $\pm(-8, -51)$. Note that only the upper 16 bits are considered by the clock synchronization algorithm. Targeting accuracies smaller than $2^{-23}\ s \approx 120$ ns appears unrealistic with our approach. Nonetheless, it is facile to scale accuracy values externally as much as desired.

The deterioration registers are 16 bit wide, including a sign bit, covering the NTP-range $\pm(-37, -51)$. This arrangement allows a minimum deterioration of about 10^{-8} s/s, by the same line of justification as for the local clock, and a maximum deterioration of 2^{-36} s/tick or approximately 244 μs/s for a nominal oscillator frequency of 2^{24} Hz, which should be sufficient even for worst quartz oscillators. The necessity of sign bits arises from continuous amortization.

Specification 2.7 (ABC for Accuracy Interval Maintenance) *The UTCSU needs to implement two adder-based clocks for the maintenance of an asymmetrical accuracy interval, composed of* $\pm(-8,-51)$*-registers* `ALPHA±`*, an external oscillator input designed for frequencies in the range of* $2^{20} \ldots 2^{24}$ *Hz (the same as for local time), and* $\pm(-37,-51)$*-registers* `LAMDBA±` *to hold the deterioration for each oscillator tick.*

The UTCSU is designed for a maximum operating frequency of 2^{24} Hz. Trading ASIC production costs against time service qualities and considering downsizing the chip for specific applications, we allow to run the UTSCU with lower frequencies as well. Changing the frequency impacts clock characteristics as shown in Table 2.2. A lower operating frequency increases the clock rate adjustability (expressed as how long a 1 μs correction takes by applying the smallest non-zero clock-step change), renders the maximum accuracy deterioration smaller, and reduces the meaningful timestamp range.

nominal oscillator frequency [Hz]	*rate adjustability* [max s for 1 μs]	*maximum accuracy deterioration* [μs/s]	*meaningful time-stamp range*
$2^{20} = 1,048.576$	2148	15	$(+31,-20)$
$2^{21} = 2,097.152$	1073	31	$(+31,-21)$
$2^{22} = 4,194.304$	537	61	$(+31,-22)$
$2^{23} = 8,388.608$	268	122	$(+31,-23)$
$10,000.000$	255	146	$(+31,-23)$
$2^{24} = 16,777.216$	134	244	$(+31,-24)$

Table 2.2: *UTCSU Characteristics for Customary Operating Frequencies*

Continuous Amortization

Periodically, the clock synchronization algorithm becomes active and computes adjustments for both rate and state in order to achieve internal/external synchronization. Enforcing these adjustments deserves special attention, since real-time applications dictate specific properties. In particular, timestamps obtained from local clocks need to be monotonic and free of discontinuities of predefined extent. A technique called *linear continuous amortization* is used to carry out state adjustments, see [56]. As a novelty, the UTCSU provides hardware support for this clock setting method.

The basic principle is rather simple: Instead of adjusting the clock state instantaneously, we achieve the same effect by modifying the clock rate for a specific amount of time, called *amortization period* t_{amort}. During this time we say that the local clock is in its *amortization phase*, whereas the remaining period is known as *pure phase*. As an obvious consequence, these two phases alternate perpetually, controlled by the clock synchronization algorithm.

On the other hand, accuracies are allowed to change instantaneously, since they are kept relative to the local clock. However, deteriorations must be modified during the amortization phase due to the clock rate set forth by state adjustment.

Figure 2.10 gives an example to illustrate both phases. The period before t_1 shows clock $C(t)$ in its pure phase, advancing with pure rate $\dot{C}(t) = v$, and accuracies $\alpha^{\pm}(t)$ growing with pure deterioration $\dot{\alpha}^{\pm}(t) = \lambda^{\pm}$. Before kicking off continuous amortization, the parameters for both amortization phase and successive pure phase need to be computed. The amortization phase commences at t_1, which entails a reduction of $\alpha^{\pm}(t)$ by the accuracy adjustments $a^{\pm}$, clock $C(t)$ advances from now on with the amortized rate $\hat{v}$, and finally $\alpha^{\pm}(t)$ deteriorates with $\hat{\lambda}^{\pm}$. A counter times out t_{amort} at t_2, causing the transition to the new pure phase.

In our example, $\alpha^{+}(t)$ shrinks during the amortization phase, and $\alpha^{-}(t)$ happens to be negative at the beginning, which justifies the sign bits introduced in Specification 2.7. To remedy the passage with negative accuracy, the externally accessible accuracy is set to zero during this time, which obviously implies a valid accuracy interval. Recall that we have to maintain the invariant that UTC is permanently enclosed by envelopes $e^{-}(t)$ and $e^{+}(t)$ defined by $C(t) - \alpha^{-}(t)$ and $C(t) + \alpha^{+}(t)$, respectively.

Having explained the mechanism for continuous amortization, we specify additional UTCSU elements separated in issues concerning local time and accuracy.

Specification 2.8 (Continuous Amortization ref. Local Time) *The adder-based clock for local time has to carry out state adjustments by continuous amortization. To that end, features to switch between pure/amortized clock rates need to be implemented, to commence the amortization phase at a programmable point in time, and a* $(+8, -24)$-*counter* `AMORTTIMER` *to earmark the end.*

Specification 2.9 (Continuous Amortization ref. Accuracy) *The two adder-based clocks for the accuracy interval must switch in lock-step with the clock for local time between pure/amortized deterioration. Additionally, registers* `ALPHA±` *need to be relative*

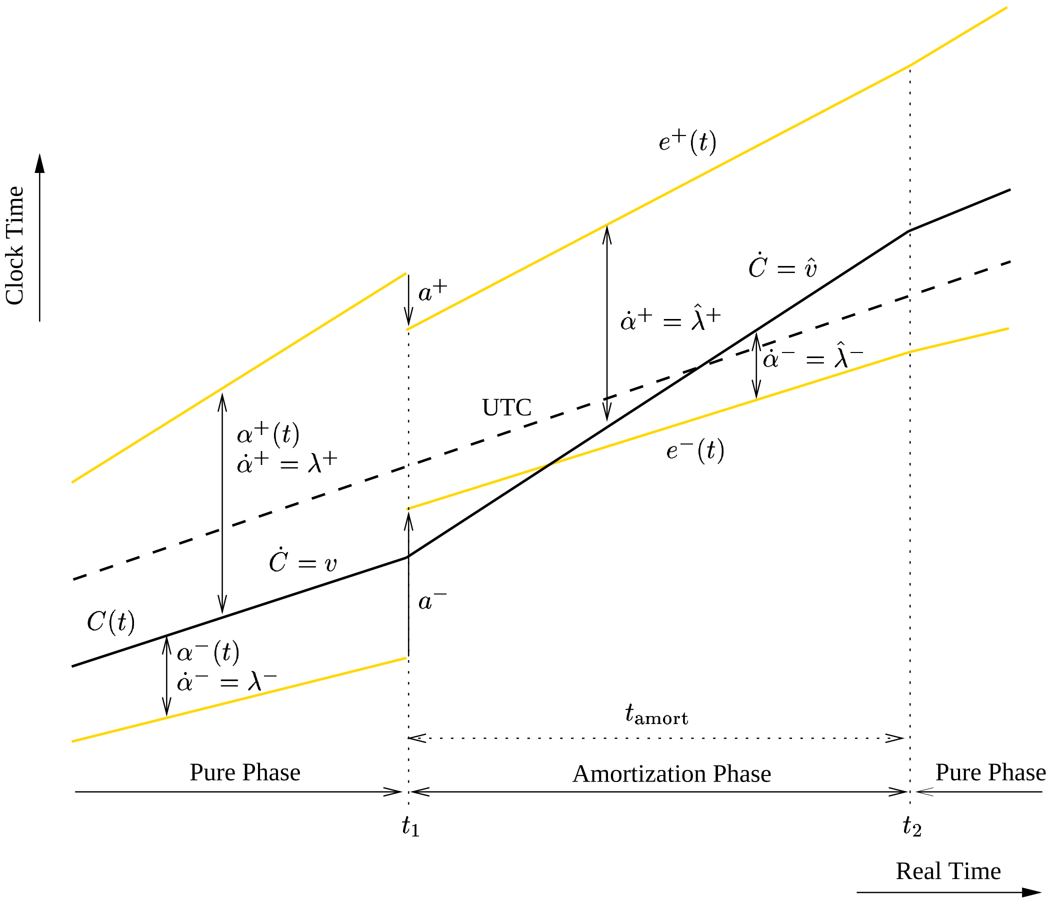


Figure 2.10: *Pure and Amortization Phase*

adjustable by $\pm(-8,-38)$*-registers* `STATESET±`*, and negative accuracies during the amortization phase have to be suppressed by forcing the externally accessible part* $(-8,-23)$ *to zero.*

If Υ denotes the amount that clock $C(t)$ gains during continuous amortization w.r.t. the non-amortized clock, the previous pure rate v, the amortized rate $\hat{v}$, and the start value T_{amort} of counter `AMORTTIMER` are related by

$$\frac{\Upsilon}{T_{\text{amort}}} = \hat{v} - v = \psi.$$

Apparently, there is one degree of freedom involved. Making T_{amort} large entails a smooth transition but might jeopardize the precision/accuracy of the local clock. However, we showed in [55] that there exists an upper bound on T_{amort} given by $\psi \geq \rho_{\max}$, where $\rho_{\max}$ denotes the sum of the maximum positive and negative drift of any clock in the system, such that results obtained for an instantaneous clock adjustment can be carried over to the continuous amortization case, see Chapter 3.

Event Timestamping and Generating

Hitherto we have specified elements to maintain local time and accuracies, which will eventually be used for event time/accuracy-stamping and event generating purposes. These two features play an important role in several domains.

The first one assists in exact CSP timestamping at both sending and receiving side, which is crucial for tight clock synchronizations. Extending the pioneering work of [24], CSPs are stamped with local time and accuracy just at the moment when they are actually leaving the node, and with local time when arriving at the peer node. This necessitates coordinated support from the UTCSU and the embedding hardware. The UTCSU obtains the CSP transmission/reception events via dedicated input lines for latching time/accuracy in dedicated registers, and further the embedding hardware is in charge of mapping them transparently into the CSP transmit/receive buffer, see Section 2.3.1

Considering Gateway-nodes and the need to cope with fault-tolerant communication architectures (i.e., triple redundant), we have to accommodate multiple units supporting independent CSP stamping.

Specification 2.10 (CSP Stamping) *The UTCSU needs to be equipped with six units each capable of sampling the* $(+31, -24)$*-range of register* `NTPTIME` *and the* $(-8, -23)$*-range of registers* `ALPHA±` *on a CSP transmission, and the* $(+31, -24)$*-range of register* `NTPTIME` *on a CSP arrival. Both types of events are announced by special polarity programmable input lines* `TRANSMIT[1..6]` *and* `RECEIVE[1..6]`.

In a similar way, an external time reference is linked to the UTCSU. We decided to use the GPS system to inject UTC due to our high accuracy and availability requirements. GPS is an earth orbiting satellite based navigation system operated by the US Air Force under the direction of the Department of Defense, see [5] for an overview. It includes a *Standard Positioning Service* (SPS) for a worldwide civilian use. When *Selective Availability* (SA) is enabled, the horizontal position can be obtained within 100 m (95 percent) and GPS-Time with a maximum error of 340 ns (95 percent). Note that GPS-Time is steered to be within 1 μs of UTC, without taking leap seconds into account. It is derived from atomic clocks both at ground stations and on board the satellites.

There is a large number of different GPS timing-receivers available. Nearly all of them output an on-time pulse at 1 pulse per second (aka. 1PPS), and more expensive models also provide a 10 MHz reference frequency disciplined to GPS-Time. The associated information identifying the pulse along with other data is provided some considerable time after the 1PPS, usually via a serial interface. We exempt the UTCSU from processing this information and defer it to the CPU. Finally, a few GPS timing-receivers even tell their status on-line (e.g., data valid, satellites out of view) by means of special output lines.

For redundancy purpose and for the sake of testing several receivers against each other, it appears appropriate to equip the UTCSU with multiple units for coupling GPS receivers.

Specification 2.11 (GPS Coupling) *The UTCSU needs to be endowed with three independent units capable of sampling the* $(+31,-24)$*-range of register* `NTPTIME` *and the status of the connected GPS timing-receiver whenever the corresponding 1PPS becomes active. Polarity programmable input lines* `1PPS[1..3]` *resp.* `STATUS[1..3]` *mediate these events resp. report the receiver status.*

Most clock synchronization algorithms periodically invoke a routine that broadcasts local time/accuracy via CSPs to the other nodes within the SSN. Hence, the UTCSU has to provide one *duty-timer* (DT) for starting a CSP transmission and one for terminating the reception period. A duty-timer consists of a writable register and a comparator to check whether local time is equal or greater, which will be reported by a dedicated interrupt. Special care is required to arm/disarm such DTs to avoid programming pitfalls, see Section 2.4.3.

Specification 2.12 (Duty Timers) *The UTCSU needs to provide two freely programmable duty-timers for each SSN attachment to support the protocol for CSP exchange, in total* `DUTYA[1..6]` *and* `DUTYB[1..6]` *all ranging over* $(+31,-16)$. *If they equal or exceed the current local time then a dedicated interrupt should be raised.*

Indeed, application requirements on a time service can vary considerably. The conceivable spectrum ranges from basic features, e.g., reading the local clock or providing a programmable frequency output, over more advanced ones, like a timestamp FIFO as required by the distributed event-based monitoring system VTA of [49], up to elaborate timing support, such as the high-precision timer developed by [15].

We decided to provide only basic on-chip application support. It includes atomic readings of local time and accuracy, time/accuracy-stamping of external application-related events, and generation of interrupts with the help of a duty timer. Future advanced application features can be realized externally by tapping the `NTPA-Bus`, which exports local time and accuracy. The following specification rule reveals more details about all those UTCSU features.

Specification 2.13 (Application Support) *First, register* `NTPTIME` *in the* $(+31,-24)$*-range and registers* `ALPHA`$\pm$ *in the* $(-8,-23)$*-range must be atomically readable by simple read operations. Second, the UTCSU needs to be furnished with nine independent units dedicated to application-support for stamping events that occur on polarity programmable input lines* `APP[1..9]`*, with local time/accuracy in the same ranges as above. Third, a dedicated* `NTPA-Bus` *has to export local time/accuracy, again in the same ranges. Finally, a* $(+31,-16)$*-duty-timer* `APPL` *is to be provided for generating external pulses and application-specific interrupts.*

Self-Test and Debugging Features

To ease system development and implementation of applications depending on a time service, most test- and debugging-features should be supported by hardware. A classical method is to guard register `NTPTIME` with additional control bits, providing a means to detect/correct flipped bits to some extent.

Specification 2.14 (Checksums) *The UTCSU must protect the* $(+31,-24)$*-range of register* `NTPTIME` *by computing an appropriate 8 bit checksum to enforce a Hamming distance of 4.*

Another well known self-checking mechanism is based on compression functions over a sequence of time/accuracy values. More specifically, the UTCSU should compute two functions over a certain time frame: *blocksums*, which are summations over a specified range, and *signatures*, which employ a generator polynomial akin to CRC-checksums. These calculations can be done in parallel by a redundant UTCSU or by a general purpose CPU, occasionally verified against the values derived inside the UTCSU. Note that one has to trade (low) checking frequency for (high) detection latency.

Specification 2.15 (Blocksums and Signatures) *The UTCSU needs to compute blocksums and signatures of the* $(+31,-24)$*-range of register* `NTPTIME` *and the* $(+8,-23)$*-range of registers* `ALPHA`$\pm$ *over a start/stop prearranged period of time.*

A *snapshot* denotes a mechanism that samples relevant internal registers to get a glance of the current UTCSU state. Either triggered by hardware or software, registers `NTPTIME` and `ALPHA`$\pm$ need to be saved in their full extent to certain shadow registers for postprocessing. Besides checking for local errors, this mechanism allows experimental evaluation of the time service precision, by triggering simultaneously snapshots on all nodes with a common line. To be complete, another special input line allows the UTCSU to (re)start its operation from a well defined state when an external pulse occurs. This features multiple redundant UTCSUs starting simultaneously at each node and allows to bypass elegantly initial clock synchronization for the sake of testing purposes.

Specification 2.16 (Snapshots) *The UTCSU needs to include a soft- and hardware snapshot mechanism to make atomic full-length shots of registers* `NTPTIME` *and* `ALPHA`±. *Furthermore we require a feature to start the UTCSU from a well defined state by activating the special input line* `SYNCRUN`.

2.4.2 Unit Implementation

Functional issues, ensuing from our desire to capture all hardware requirements for a highly accurate/precise time service, guided the process of developing the UTCSU specification. In the course of designing this ASIC, we crafted a layout of physical units fostering a clean top-down implementation. More specifically, starting from the specification [62] given in Section 2.4.1, the UTCSU internal units portrayed in Figure 2.11 were eventually identified and designed in [28]. A first look at the boundary reveals interfaces for the System-Bus, GPS receivers, CSP stamping, and applications. The interior consists of 10 generic units connected by a 32 bit broad `I-Bus`, resulting in a homogeneous architecture. The following subsections introduce these units in some detail, depending on the level of interest and originality.

Bus Interface Unit (BIU)

The BIU contains logic for embedding the UTCSU in a wide variety of system architectures as demanded in Specification 2.1. In particular, it makes the ASIC applicable to System-Bus widths of 8, 16 or 32 bits, little or big endian byte ordering, and different access times. A technique called *dynamic bus sizing* is used to solve this common interfacing problem and we recommend [41] for technical details.

Interrupt Unit (ITU)

Throughout the UTCSU specification we encountered several dedicated interrupts to announce certain conditions asynchronously. We grouped all possible interrupt sources in three categories, resulting in the following interrupt lines:

- Interrupt `INT-T` indicates events relevant to the clock synchronization algorithm as specified in Specification 2.12 (duty-timers) and Specification 2.5 (accuracy overruns).

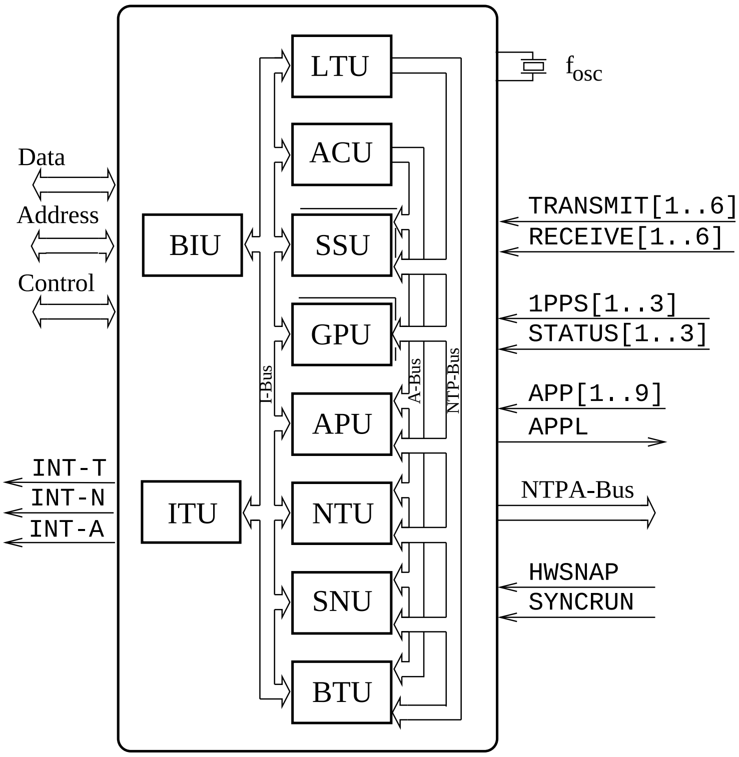


Figure 2.11: *Interior of the UTCSU*

- Interrupt `INT-N` indicates external events as specified in Specification 2.10 (stamping leaving/arriving CSPs) and Specification 2.11 (1PPS pulses from GPS-receivers).
- Interrupt `INT-A` indicates application-related events as demanded in Specification 2.13 as well as events for chip testing purposes.

The ITU contains the logic for generating these interrupts. Any interrupt source can be individually enabled/disabled via configuration registers `UTCINTEN1/2`, and pending interrupts are reflected by registers `UTCSTAT1/2`. For further details about handling these registers turn to Section 2.4.3.

Local Time Unit (LTU)

According to Specification 2.6 and 2.8 local time has to be maintained by an adder-based clock. This boils down to set up a 91 bit adder capable of carrying out an addition at each oscillator tick. Due to the advanced space and time requirements, much effort has been invested to devise an optimal adder architecture, see [18].

Figure 2.12 shows schematically the architecture of the LTU with the 91 bit adder `NTPADDER` right in the center. The addend is realized as a positive feedback of the previous adder output held by the $(+31,-59)$-register **NTPTIME**, which in turn provides local time. Initialization takes place via multiplexer **NTPMUX** and preload registers **MSSET**, **TSSET** and **USSET**.

The augend is supplied by the $(-20,-51)$-register **STEPPUREACT** resp. **STEPAMORT** during pure resp. amortization phase. Their granularity is extended by register **STEPLOW**, which holds a fixed value in the $(-52,-59)$-range to account for non-binary oscillator frequencies. Muliplexer **AMORTMUX** is in charge of switching between pure and amortizing clock-step values as demanded by Specification 2.8. As long as the programmable 32 bit counter **AMORTTIMER** is running, the augend originates from **STEPAMORT**, otherwise from **STEPPUREACT**. The counter can be activated by a duty timer or immediately by writing a dedicated register address. Preload register **STEPPURE** is necessary to hold the clock-step value for the following pure phase, which starts when the counter expires. See Figure 2.10 for a better understanding of the transitions between the two phases.

According to Specification 2.3, the adder-based clock has to cope with leap seconds as well. Basically, the leap second correction hardware consists of multiplexer **LEAPMUX**, which affects the upper 32 bits of the augend in such a way that either one standard second is added or subtracted just as required. An additional duty-timer can be programmed to initiate these time corrections.

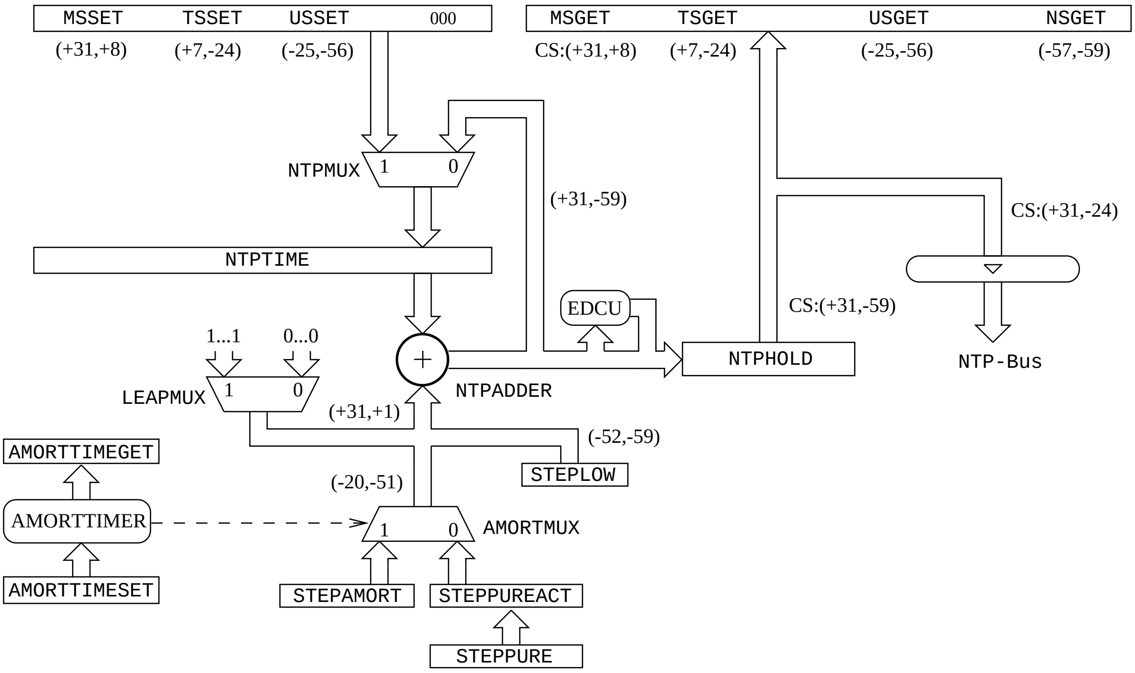


Figure 2.12: *Local Time Unit (LTU)*

An 8 bit *Checksum* (CS) is computed for the $(+31, -24)$-range of register `NTPTIME` by the *Error Detection and Correction Unit* (EDCU) as stipulated in Specification 2.14, using a modified Hamming Code of distance 4. The `CS` bits together with the `NTPADDER` output are buffered on each rising edge of the oscillator pulse in an intermediate register `NTPHOLD`. This introduces a delay of one oscillator tick between computation and sampling, but the clock synchronization algorithm is not affected, since adjustments are only applied in a relative fashion. Therefore, we can operate the local clock one tick ahead of actual time internally, so that correct time is perceived after buffering.

Adhering to byte-orientation, the output of register `NTPHOLD` is decomposed into four portions. The 24 bit *Marcostamp*-range $(+31, +8)$ together with the 32 bit *Timestamp*-range $(+7, -24)$ including the 8 `CS` bits constitute the internal 64 bit `NTP-Bus` (see Figure 2.11) from where all timestamps are obtained. The remaining 32 bit *Microstamp*-range $(-25, -56)$ and the 3 bit *Nanostamp*-range $(-57, -59)$ are internally used for rate corrections. Nevertheless, all portions are atomically accessible through holding registers `CS:MSGET, TSGET, USGET` and `NSGET` for testing purposes.

Accuracy Unit (ACU)

The ACU maintains $\alpha^+(t)$ and $\alpha^-(t)$ as demonstrated in Figure 2.10, forming the local accuracy interval $A(t)$ around local clock $C(t)$. Specification 2.7 and 2.9 demand a dedicated ABC for each $\alpha^+(t)$ and $\alpha^-(t)$, but due to symmetry, it suffices to discuss only one of them.

Apart from a few modifications and supplements, the basic structure of this adder-based clock is similar to the one inside the LTU, see Figure 2.13. In the center we find a 45 bit adder `ACCADDER`, which is able to deal with negative numbers. Unlike the LTU-adder, however, it stays at its upper limit and triggers an interrupt `INT-T` instead of wrapping around in case of overrun.

Register `ALPHA` is situated in the feedback loop of the addend, so that it actually provides the current accuracy value $\alpha^+(t)$ or $\alpha^-(t)$, respectively. Initialization with the preloaded value in the $\pm(-8, -38)$-register `ALPHASET` happens via multiplexer `ALPHAMUX`, whereby the $(-39, -51)$-part is fixed to all ones upon initialization.

In analogy to the LTU, multiplexer `LAMBDAMUX` selects the augend from $\pm(-37, -51)$-register `LAMBDAPUREACT` resp. `LAMBDAAMORT` during the pure resp. amortization phase as specified in Specification 2.9. Both registers hold 16 bit signed deterioration values that are internally sign-extended to 45 bits via multiplexer `LAMBDASMUX`. Executing continuous amortization for local time controlled by counter `AMORTTIMER`, switching between deterioration values takes place simultaneously with the clock-step registers inside the LTU.

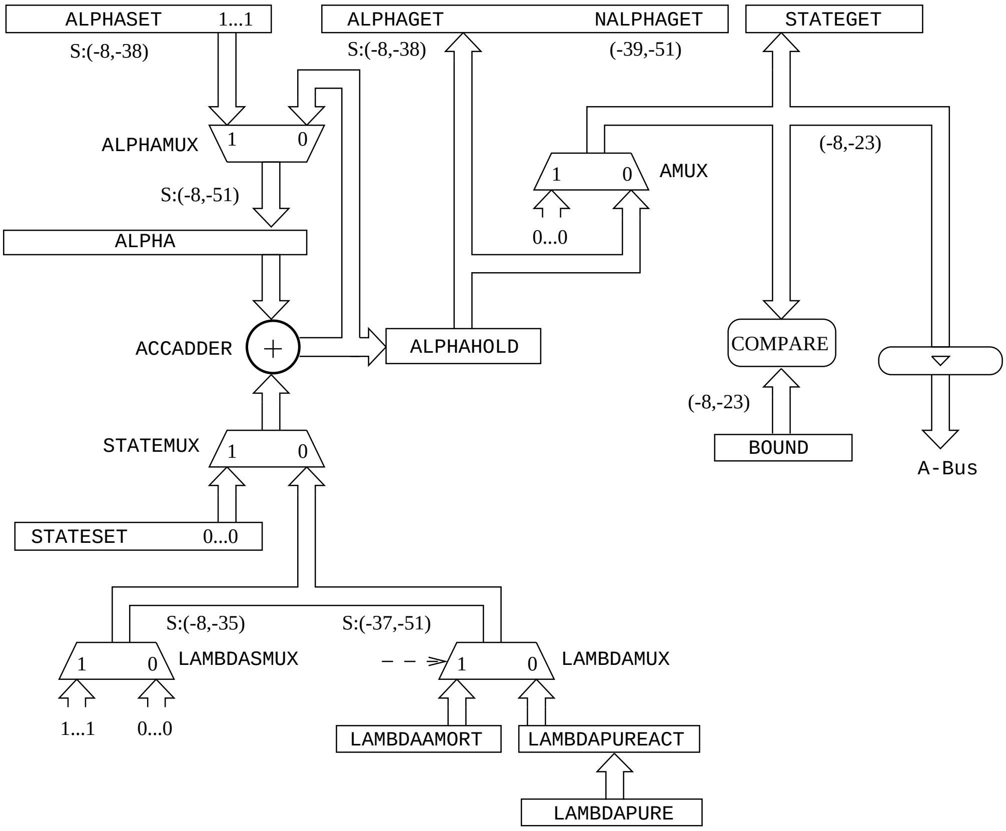


Figure 2.13: *Generic Accuracy Unit (ACU)*

No deterioration takes place right at the transition from pure to amortization phase, but rather register `ALPHA` is relatively adjusted via multiplexer `STATEMUX` by an offset value previously written to register `STATESET`. Again, Figure 2.10 helps to clarify the subtleties at this pivotal transition point.

During the amortization phase, the content of accuracy register `ALPHA` can become negative. Following Specification 2.9, multiplexer `AMUX` converts such negative values into zero controlled by the associated sign bit. The raw accuracy values are nevertheless accessible via registers `ALPHAGET` and `NALPHAGET`, which are sampled simultaneously when LTU register `TSGET` is read.

Both negative and positive accuracy are exported in the $(-8, -23)$-range, constituting the UTCSU internal `A-Bus+` resp. `A-Bus-`. Together they are referred as the 32 bit broad `A-Bus` from where all accuracystamps are taken, see Figure 2.11). Note that the output of adder `ACCADDER` is buffered at each rising edge of the oscillator pulse into register `ALPHAHOLD`, thus the accuracies on the `A-Bus` suffer from the same latency of one tick as local time on the `NTP-Bus`; recall our comments in Section 2.4.2.

Implementing Specification 2.5 results in a 16 bit comparator block `COMPARE` that permanently checks the `A-Bus` against a $(-8, -23)$-register `BOUND`. If the current value on the `A-Bus` happens to be greater or equal to `BOUND`, an interrupt `INT-T` will be raised.

Synchronization Subnet Unit (SSU)

Specification 2.10 demands one SSU for each SSN attachment, where each unit comprises a set of registers to sample the `NTP-` and `A-Bus` on the occurrence of packet reception/transmission events mediated by the embedding hardware as explained in Chapter 2.3.1. The latter is also responsible to map transparently the corresponding SSU registers, see Section 2.4.3 for an enumeration, into a certain portion of the transmit/receive buffer of the adjoined COMCO.

Moreover, Specification 2.12 introduces two duty timers for each SSU, that perform a comparison of the `NTP-Bus` against a 48 bit preload value. An interrupt `INT-T` is raised whenever the `NTP-Bus` becomes greater or equal to the preloaded value, provided that this interrupt source is enabled.

GPS Unit (GPU)

External clock synchronization requires access to an external time source. As imposed by Specification 2.11, we use GPS receivers to inject UTC or GPS-Time. Figure 2.14 shows an exemplary interface structure to couple one receiver to a Primary-node. For redundancy purposes, up to three receivers can be attached to a single UTCSU.

Both lines `1PPS[1]` and `STATUS[1]` of the receiver are directly connected to the GPU[1] in order to trigger a timestamp and sample the receiver status on the occurrence of an active 1PPS. The interior of a GPU is made up of our usual sample registers, see Section 2.4.3 for a listing, whereas the most significant CS bit is replaced by the status bit of the receiver. To adapt the GPU for different GPS receivers, the active 1PPS edge can be programmed to be either the rising or the falling one. Advanced GPS timing-receivers offer an additional frequency output with high stability characteristics, e.g. line `10Mhz` in our example, which is most suitable for being used as the oscillator input frequency pacing the ABCs.

Events signalled by the 1PPS are conceptually analogous to the arrival of CSPs. However, information identifying the pulse (full seconds relative to UTC) is delivered after the occurrence of the 1PPS, usually via a RS232 interface. The task of preprocessing this data, sending commands to the receiver for configuration purposes, and interpreting other data concerning status, navigation, etc. are not handled by the UTCSU, but rather by the CPU of the embedding hardware.

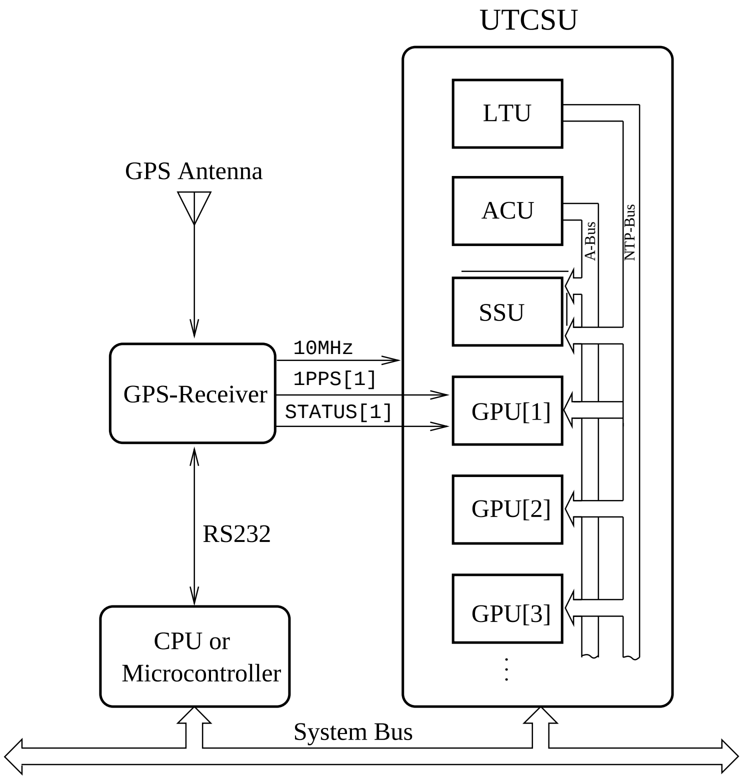


Figure 2.14: *Coupling of GPS Receivers*

Application Unit (APU)

According to Specification 2.13, the APU provides features to generate interrupts at specific points in time and to record the occurrence time of application-related events. The former is implemented by a duty-timer `APPL` to generate an `INT-A` interrupt, similar to one in the SSU as explained in Section 2.4.2. The second feature is more expensive, since tracing events requires atomic sampling of both `NTP-` and `A-Bus`. The APU accommodates nine register sets, see Section 2.4.3, to stamp pulses on the polarity programmable input lines `APP[1..9]` with time and accuracy simultaneously. Moreover, to ease higher-level recovery of applications in case of a clock fault, CPU read accesses of these registers are memorized by a certain reference status bit.

Network Time Interface Unit (NTU)

Additional application timing support can be provided externally by means of the `NTPA-Bus` as required by Specification 2.13. The UTCSU produces 64 bit time information onto the `NTP-Bus` and 32 bit accuracy information onto the `A-Bus` at each oscillator tick; given an operating frequency of 2^{24} Hz, this amounts to 192 MByte/s. Indeed, exporting such a stream of data turns out to be a challenging matter. To reduce the pin count of our chip, the NTU achieves this performance by pushing out data via the 48 bit wide multiplexed `NTPA-Bus` driven by both edges of the oscillator pulse.

Snapshot Unit (SNU)

The SNU comprises three debugging/test services called software snapshot, hardware snapshot and synchronous operation. As motivated by Specification 2.16, snapshots are means to sample the current internal state of the UTCSU.

A software snapshot can be triggered by writing a dedicated SNU register address or on expiration of a duty-timer. This entails a simultaneous sampling of the complete LTU register `NTPHOLD` into the already introduced registers `MSGET`, `TSGET`, `USGET`, and `NSGET`, as well as sampling ACU registers `ALPHAHOLD`± into registers `ALPHAGET`±, `NALPHAGET`± and `STATEGET`±. Further reads without latching semantics deliver the before sampled data.

A hardware snapshot is triggered by an external pulse on the polarity programmable line `HWSNAP`. Due to the asynchronous nature of such an event, it merely samples both `NTP-` and `A-Bus` into appropriate SNU registers, enumerated in Section 2.4.3.

Finally, input line `SYNCRUN` gives the UTCSU a "go"-functionality. In other words, all three ABCs can be started simultaneously at a specific point in time triggered by the test environment.

Built-In Test Unit (BTU)

In principle, the sequence of all time/accuracy values generated inside the UTCSU could be computed externally as well. Unfortunately, a continuous check is prohibited by the tremendous throughput, as already pointed out above. However, Specification 2.15 specifies two compression methods over a certain set of time/accuracy-stamps, that allow less frequent data exchanges between the UTCSU and external verification devices. As a result, the UTCSU is endowed with a BTU that computes signatures and blocksums for both `NTP-` and `A-Bus`. Note that these two methods can be used independently of each other.

The upper half of Figure 2.15 shows the logic to compute a signature for the `A-Bus`, where the accuracy values are fed from above into a *linear feedback shift register* (LFSR), see [76]. These registers are implementing the evaluation of a polynomial $P(x)$ with coefficients out of $\{0,1\}$. Each power x^k is reflected by a D-Flip-Flop (D-FF) fed from the x^{k-1} stage exored with the corresponding bit from the input data. Powers with non-zero coefficient obtain an additional feedback from the D-FF$_0$ output. In particular, a polynomial of degree 56, namely $P_{\mathrm{NTP-Bus}}(x) = x^{56} + x^{22} + x^{21} + x^1 + 1$, compresses timestamps and a polynomial of degree 32, namely $P_{\mathrm{A-Bus}}(x) = x^{32} + x^{28} + x^{27} + x^1 + 1$, takes care of accuracystamps. The lower half of Figure 2.15 shows an adder feedbacked by the 32 bit register **ACCSUM** to compute blocksums over the **A-Bus**.

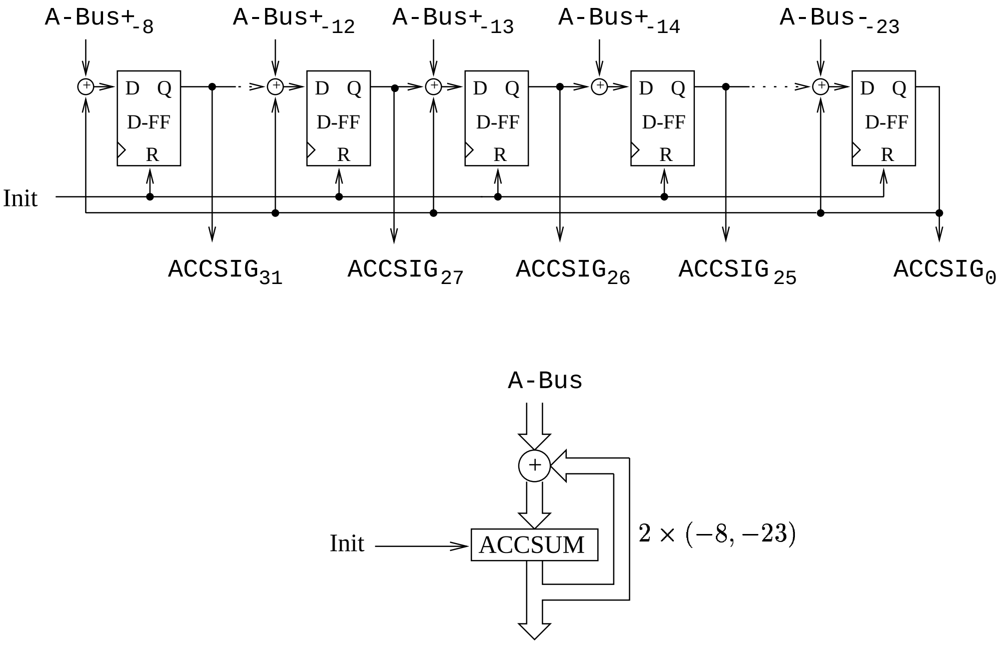


Figure 2.15: *Compressing Accuracystamps with Signatures and Blocksums*

Only a complete sequence of time/accuracy-stamps is useful to be processed in the LFSR or adder block, hence specific start/stop-points have to be defined. Transitions between pure and amortization phases are particularly suitable for that purpose. Section 2.4.3 remarks on programming issues about these features.

2.4.3 Programming Model

We wrap up our UTCSU picture by presenting a synoptic view from the programming standpoint. Organized in subjects concerning clocks, interrupts, sampling and testing, we inspect essential operations tagged with programming guidelines. Tables will summarize related UTCSU elements by showing their type and providing a brief description. Three types can be distinguished: An internal register indicated by the corresponding NTP-range or width, a pseudo register to trigger certain actions, or a pin for external events. A forthcoming datasheet will provide a complete programming description including register maps, timing diagrams and electrical characteristics.

Clock Management

The management of ABCs can be separated in issues concerning initialization and adjustments. Initializing the UTCSU turns out to be a delicate matter, since several stages have to be passed through, namely

1. set all registers to a default value via a hardware reset,
2. set augend (clock rate and accuracy deterioration) of ABCs, and
3. set addend (clock and accuracy state) of ABCs.

All UTCSU registers are set to a default value whenever the chip is powered up or when an external hardware reset occurs. In particular, LTU-register `NTPTIME` becomes zero, ACU-registers `ALPHA`± all ones, and the various registers holding the corresponding augend are cleared to prevent a progression of the adder-based clocks.

The second stage targets the active augends of the adder-based clocks belonging to the pure phase, i.e. LTU-register `STEPPUREACT` and ACU-registers `LAMBDAPUREACT`± holding the pure clock-step resp. pure accuracy deterioration values. As shown in Figures 2.12 and 2.13, these registers cannot be initialized directly. Still, we get a handle on them by first writing their preload registers `STEPPURE` resp. `LAMBDAPURE`± and by subsequently commencing a short period of continuous amortization. Only register `STEPLOW` can be set directly to account for a non-binary oscillator frequency. Table 2.3 summarizes programming elements relevant at this stage.

element	*type*	*description*
`STEPPURE`	$(-20, -51)$	clock-step for pure phase
`STEPLOW`	$(-52, -59)$	clock-step for non-binary frequency adaption
`LAMBDAPURE+`	$\pm(-37, -51)$	upper accuracy deterioration for pure phase
`LAMBDAPURE-`	$\pm(-37, -51)$	lower accuracy deterioration for pure phase

Table 2.3: *UTCSU Elements for Rate/Deterioration Initialization*

In the third stage, the addends of the adder-based clocks, i.e., LTU-register `NTPTIME` and ACU-registers `ALPHA`±, are initialized. Since they are reflecting the state of local time/accuracy, their value has to be set appropriately to correspond to the time of setting them. As a consequence, preload registers must be provided to hold the initialization values in advance that are transferred atomically to `NTPTIME` and `ALPHA`± upon a suitable event. Local time is initialized with the aid of three 32 bit registers `MSSET`, `TSSET` and

`USSET`. When no accuracy information is at hand for initialization, we can stick to the maximum introduced during hardware reset, which serves literally as infinity. Otherwise, preload register `ALPHASET` supplies both `ALPHA+` and `ALPHA-`, launching a symmetrical accuracy interval. The event of actually transferring preloaded values is generated by writing pseudo-registers `NTPSET` and `ALPHASET`, see Table 2.4. To ease simultaneous starting of distributed UTCSUs during the testing phase or in redundant configurations, the transfer event can also be a pulse on the external line `SYNCRUN`.

element	*type*	*description*
`MSSET`	$(+31, +8)$	macrostamp portion of clock state
`TSSET`	$(+7, -24)$	timestamp portion of clock state
`USSET`	$(-25, -56)$	mircostamp portion of clock state
`ALPHASET`	$\pm(-8, -38)$	initial upper/lower accuracy
`NTPSET`	pseudo	sets clock state and upper/lower accuracy on write
`ALPHAPNSET`	pseudo	sets only upper/lower accuracy on write
`SYNCRUN`	pin	sets clock state and upper/lower accuracy on ext. pulse

Table 2.4: *UTCSU Elements for Time/Accuracy Initialization*

Once initialized, clocks need to be adjusted periodically in order to maintain internal/external synchronization. Adjustments values for local time and accuracies are handed over at each pure to amortization phase transition as already explained in Section 2.4.1. The UTCSU enforces these adjustments autonomously, which renders programming rather simple. In fact, the elements of Table 2.5 need to be computed before the next amortization phase, encompassing values for clock-step register `STEPAMORT`, for deterioration registers `LAMBDAAMORT`±, and for accuracy adjustment registers `STATESET`±. Additionally, registers `STEPPURE` and `LAMBDAPURE`± need to be preloaded for the subsequent pure phase. The duration in oscillator ticks of the amortization phase can be set via counter `AMORTTIMESET`, whereas the start can be either triggered by writing pseudo register `STARTAMORT` or by activating duty-timer `DUTYB[1]`.

Interrupt Management

There are as much as 64 interrupt sources within the UTCSU, which are statically mapped to three dedicated interrupt lines introduced in Section 2.4.2. Interrupt processing in general works as follows, see Table 2.6. To initialize an interrupt source for further interrupts, we have to clear it by setting the appropriate bit in register `UTCINTCLEAR1`

element	*type*	*description*
`STEPAMORT`	$(-20, -51)$	clock-step for amortization phase
`LAMBDAAMORT+`	$\pm(-37, -51)$	upper accuracy deterioration for amortization phase
`LAMBDAAMORT-`	$\pm(-37, -51)$	lower accuracy deterioration for amortization phase
`STATESET+`	$\pm(-8, -38)$	upper accuracy adjustment in two's complement
`STATESET-`	$\pm(-8, -38)$	lower accuracy adjustment in two's complement
`AMORTTIMESET`	32 bit counter	duration of amortization phase in oscillator ticks
`STARTAMORT`	pseudo	starts amortization phase on write
`DUTYB[1]`	$(+31, -16)$	duty-timer to start amortization phase

Table 2.5: *UTCSU Elements for Time/Accuracy Adjustments*

or `UTCINTCLEAR2`, and to enable it by setting the appropriate bit in register `UTCINTEN1` or `UTCINTEN2`. A pending interrupt is mirrored by a status bit in registers `UTCSTAT1` or `UTCSTAT2`, which can be polled by the *interrupt service routine* (ISR) to identify the originating source of the interrupt. Before leaving the ISR, the interrupt source must be cleared as described above.

element	*type*	*description*
`UTCINTCLEAR1/2`	32 bit each	clears pending interrupts
`UTCINTEN1/2`	32 bit each	enables/disables interrupts
`UTCSTAT1/2`	32 bit each	reflects the status of the interrupt condition

Table 2.6: *UTCSU Elements for Interrupt Handling*

Many interrupts are associated with external events, like leaving/arriving CSPs, 1PPS pulses from GPS timing-receivers, or occurrence of application-oriented events. Their handling will be described in Section 2.4.3. Here we proceed with interrupts that are internally caused by the UTCSU, in particular by DTs and accuracy overruns. Of course, many other exceptional UTCSU conditions are also announced by `INT-T` interrupts, like an overrun of `NTPTIME` somewhere in year 2036 or clocking failures.

Various duty-timers are used for generating interrupts or for triggering certain actions, see Table 2.7 for a complete list. Each DT spans 48 bits, organized in a 32 bit high-part $(+31, 0)$ and in a 16 bit low-part $(-1, -16)$ including an enable bit `E` to arm/disarm its operation. If armed, the content of each duty-timer is permanently compared against the current time on the `NTP-Bus` and corresponding status bits in registers `UTCSTAT1` and `UTCSTAT2` indicate the condition “less” (duty-timer < `NTP-Bus`) or “greater-or-equal” (duty-timer ≥ `NTP-Bus`). In case of a transition from “less” to “greater-or-equal”, a dedicated interrupt `INT-T` will be generated. Note that activating a DT with an old value w.r.t. the current time given on the `NTP-Bus` also entails an interrupt.

element	*type*	*description*
`DUTYA[1..6]-HIGH`	$(+31, 0)$	SSU duty-timers `A[1..6]`
`DUTYA[1..6]-LOW`	`E:`$(-1, -16)$	– ” –
`DUTYB[1]-HIGH`	$(+31, 0)$	duty-timer to start amortization
`DUTYB[1]-LOW`	`E:`$(-1, -16)$	– ” –
`DUTYB[2]-HIGH`	$(+31, 0)$	duty-timer to terminate CSP receptions
`DUTYB[2]-LOW`	`E:`$(-1, -16)$	– ” –
`DUTYB[3]-HIGH`	$(+31, 0)$	duty-timer to insert/delete leap seconds
`DUTYB[3]-LOW`	`E:`$(-1, -16)$	– ” –
`DUTYB[4..6]-HIGH`	$(+31, 0)$	auxiliary SSU duty-timers `B[4..6]`
`DUTYB[4..6]-LOW`	`E:`$(-1, -16)$	– ” –
`SW-HIGH`	$(+31, 0)$	SNU duty-timer for software snapshot
`SW-LOW`	`E:`$(-1, -16)$	– ” –
`APPL-HIGH`	$(+31, 0)$	APU duty-timer for applications
`APPL-LOW`	`E:`$(-1, -16)$	– ” –

Table 2.7: *UTCSU Elements for Duty-Timers*

Without external scaling, the capacity to hold accuracies in registers `ALPHA`± is limited to 7.81 ms each, while overflows cause an interrupt `INT-T`. A more selective supervision of maximum accuracy can be programmed with the help of registers `BOUND`±, also displayed in Table 2.8. Similar to duty-timers, whenever the condition “below” (`A-Bus`± < `BOUND`±) tips to “above-or-equal” (`A-Bus`± ≥ `BOUND`±), an interrupt `INT-T` will be generated. Setting registers `BOUND`± to the maximum disables this feature.

element	*type*	*description*
`BOUND+`	$(-8, -23)$	bound for upper accuracy
`BOUND-`	$(-8, -23)$	bound for lower accuracy

Table 2.8: *UTCSU Elements to Bound Accuracies*

Sampling Management

Sampling activities can stem from software primitives or from external events. The first ones are also known as *atomic clock readings*, providing allied portions of local time/accuracies only. Whenever register `TSGET` is read by software, it entails a simultaneous latching of full-scale local time and accuracy in certain UTCSU-registers listed in Tables 2.9. Later on the sampled values can be read externally at register addresses without latching semantics.

element	*type*	*description*
`CS:MSGET`	`CS:`$(+31, +8)$	macrostamp portion of clock state with checksum
`TSGET`	$(+7, -24)$	timestamp portion of clock state (triggers sampling)
`USGET`	$(-25, -56)$	microstamp portion of clock state
`NSGET`	$(-57, -59)$	nanostamp portion of clock state
`ALPHAGET+`	$\pm(-8, -38)$	high portion of signed upper accuracy
`ALPHAGET-`	$\pm(-8, -38)$	high portion of signed lower accuracy
`NALPHAGET+`	$(-39, -51)$	low portion of upper accuracy
`NALPHAGET-`	$(-39, -51)$	low portion of lower accuracy
`STATEGET+`	$(-8, -23)$	upper accuracy on `A-Bus+`
`STATEGET-`	$(-8, -23)$	lower accuracy on `A-Bus-`

Table 2.9: *UTCSU Elements for Sampling Local Time and Accuracy*

Crucial for UTCSU operation are features to stamp external events with time/accuracy values. Apart from sampling, a `INT-N` interrupt is raised upon occurrence if enabled at all. Table 2.10 recapitulates all elements relevant for this functionality. Registers `UTCCONF1` and `UTCCONF2` determine the polarity on which input pulses cause sampling. Besides that, they contain bits to control leap-second insertion/deletion and to configure self-test features, see Section 2.4.3. In a straightforward way, pulses on pins `TRANSMIT[1..6]` resp. `RECEIVE[1..6]` inform the UTCSU about leaving resp. arriving CSPs. The various sample registers can be found in the second and third block of Table 2.10. Similarly, pulses on pins `1PPS[1..3]` resp. `STATUS[1..3]` inform the UTCSU about an active on-time pulse resp. status of a connected GPS timing-receiver. The fourth block of Table 2.10 indicates the corresponding sample registers. Finally, pins `APP[1..9]` are available for time/accuracy-stamping of application events. The bottom block of Table 2.10 shows the associated sample registers including `APPCLEAR`. Every read access of registers `MSAPP[1..9]`, `TSAPP[1..9]` or `ACCAPP[1..9]` sets a status bit in register `UTCSTAT1`, which can be cleared by writing `APPCLEAR`.

element	*type*	*description*
`UTCCONF1/2`	32 bit each	configuration bits
`TRANSMIT[1..6]`	pins	indicates a CSP transmission
`MSXMT[1..6]`	`CS`:$(+31,+8)$	sample of upper NTP-Bus portion
`TSXMT[1..6]`	$(+7,-24)$	sample of lower NTP-Bus portion
`ACCXMT[1..6]`	$(-8,-23)$:$(-8,-23)$	sample of `A-Bus`
`RECEIVE[1..6]`	pins	indicates a CSP arrival
`MSRCV[1..6]`	`CS`:$(+31,+8)$	sample of upper NTP-Bus portion
`TSRCV[1..6]`	$(+7,-24)$	sample of lower NTP-Bus portion
`1PPS[1..3]`	pins	indicates a 1PPS pulse from GPS receivers
`STATUS[1..3]`	pins	indicates the status of GPS receivers
`MSGPS[1..3]`	`status`:`CS`:$(+31,+8)$	sample of upper `NTP-Bus` portion with receiver status
`TSGPS[1..3]`	$(+7,-24)$	sample of lower `NTP-Bus` portion
`APP[1..9]`	pins	indicates an external application event
`MSAPP[1..9]`	`CS`:$(+31,+8)$	sample of upper `NTP-Bus` portion
`TSAPP[1..9]`	$(+7,-24)$	sample of lower `NTP-Bus` portion
`ACCAPP[1..9]`	$(-8,-23)$:$(-8,-23)$	sample of `A-Bus`
`APPCLEAR`	pseudo	clears the application reference status bit

Table 2.10: *UTCSU Elements for Stamping External Events*

Testing Management

Snapshot mechanisms are implemented for test and verification purposes as motivated in Section 2.4.1. Before using them, certain configuration bits in `UTCCONF1` must be set appropriately. If enabled, a pulse on pin `HWSNAP` triggers a hardware snapshot, i.e., the `NTP-Bus` is latched into registers `MSSNU` and `TSSNU`, and the `A-Bus` into `ACCSNU`. The top block of Table 2.11 summarizes these elements. A software snapshot is triggered *directly*, when pseudo register `SWSNAP` is written, or *programmed* when duty-timer `SW` fires. All registers listed in Tables 2.9 are affected by activating this mechanism, which provides a complete view of the current state of the whole chip.

The computation of signatures and blocksums for external verification can be started and stopped by a `SYNCRUN` event, a software snapshot, or at the beginning of an amortiza-

element	*type*	*description*
`HWSNAP`	pin	triggers a hardware snapshot
`MSSNU`	`CS`:$(+31, +8)$	sample of upper `NTP-Bus` portion
`TSSNU`	$(+7, -24)$	sample of lower `NTP-Bus` portion
`ACCSNU`	$(-8, -23)$:$(-8, -23)$	sample of `A-Bus`
`SWSNAP`	pseudo	triggers a software snapshot
`SW`	$(+31, -16)$	duty-timer to trigger a software snapshot

Table 2.11: *UTCSU Elements for Snapshots*

tion phase. Configuration bits in register `UTCCONF1` must be set accordingly to enable the desired operation. Furthermore, these events (except of `SYNCRUN`) latch the results into appropriate registers as given in Table 2.12. Usually, the period over which these compression functions are calculated include the pure phase, the length of which is provided by counter `PUREPHASE` for external verification purposes.

element	*type*	*description*
`MSSIG`	32 bit	signature of macrostamp portion of time
`TSSIG`	32 bit	signature of timestamp portion of time
`ACCSIG`	32 bit	signature of upper and lower accuracy
`MSSUM`	32 bit	blocksum of macrostamp portion of time
`TSSUM`	32 bit	blocksum of timestamp portion of time
`ACCSUM`	32 bit	blocksum of upper and lower accuracy
`PUREPHASE`	32 bit counter	duration of pure phase in oscillator ticks

Table 2.12: *UTCSU Elements for External Verification*

2.4.4 Design Methodology

The chip design process evolved in several stages. In the following we document them briefly by pointing out their particular objective and the tools used.

Starting out from the functional specification of [62], an elaborate process of successive refinement was conducted that eventually ended up in coding and synthesizing all required units, see [28]. Based on this knowledge, we worked out a complete behavioral VHDL description of the UTCSU. At the same time, we coded a simulation model of the node consisting of a basic VHDL model of both CPU and embedding hardware including the COMCO. At the next stage, algorithmic operations were verified to ensure that all required functionalities are present and behave as required. Again, this model was refined and recoded to meet our specification and to make the code ready for synthesis.

To obtain a *gate-level netlist*, we used SYNOPSYS Design Compiler for synthesis into the ES2 0.7 μm standard cell CMOS process foundry. A postprocessing tool was used to verify the resulting code against the behavioral simulation. Table 2.13 summarizes some technical chip data derived at this stage. Afterwards, SYNOPSYS Test Compiler inserted *full scan path logic* with a multiplexed flip-flop style and *boundary scan logic* according to IEEE 1149.1. CADENCE DFWII back-end tools finalized the design with *place&route* for ES2. Timing requirements and technology rules were cross-checked by parasitic extraction.

item	*value*
chip area	105 mm^2
max. operating frequency	25 MHz
equivalent gates	66,500
pins	208

Table 2.13: *UTCSU Chip Data*

2.5 Relation to other Approaches

Despite of the large body of related research work on clock synchronization, there are only a few papers that deal with hardware support. In fact, most published papers restrict their attention to purely software-based approaches, targeting a precision in the ms-range only. A remarkable exception is the software-based synchronization scheme of [72], which “sprays” external time obtained via GPS into broadcast-type LANs with a precision/accuracy in the 10 μs-range. The utilized *a posteriori agreement technique*, however, rests on the strong assumption that at least one broadcast among $f+1$ attempted ones is completely fault-free.

In [24], it was shown that considerably better results can be achieved with any clock synchronization algorithm if some dedicated hardware support is present. In fact, the pioneering CSU briefly discussed therein, consult [42] for details, allows to construct synchronized clocks with a precision in the 10 μs-range for broadcast networks. Even better results are claimed for the CSU’s successor described in [23], which is tailored to fieldbusses for automotive applications. The stated worst case precision of a few μs, however, is quite optimistic.[‡] Another hardware-assisted clock synchronization scheme for (not necessarily fully connected) point-to-point networks is briefly outlined in [45]. It is also inspired by [24] and targets a precision in the 100 μs-range. A full assessment, however, is impossible due to lacking details.

Our NTI also leans on the general hardware architecture proposed in [24]. In fact, we utilize the same (DMA-based) method of packet timestamping, although extended by several "uncertainty-saving" engineering improvements. For example, instead of triggering a receive timestamp upon occurrence of the packet reception interrupt (after the whole packet has been received), we generate this trigger when a specific address within the receive buffer is written upon CSP reception. In addition, our NTI supports different offsets for timestamp triggering and transparent mapping, which can help in reducing the effects of COMCO-peculiarities upon ε, see Section 2.3.1.

The apparent similarities between the CSU and our UTCSU-ASIC, however, are only implied by the general requirements put on any hardware support for clock synchronization. In fact, our UTCSU differs from the CSU in many important ways:

- The UTCSU supports an interval-based clock synchronization paradigm, which requires maintenance of local accuracy intervals, see Section 1.2.
- Our approach aims at a worst case precision/accuracy in the 1 μs-range, which demands a considerably improved granularity as well as excellent rate and state adjustment capabilities.
- We employ fundamentally different implementations of the vital functional units on-chip the UTCSU. The strikingly elegant and simple adder-based clock design, for example, surpasses any existing approach we are aware of. This is also true for the unwieldy clock device of [23], which may be viewed as a concatenation of an adder-based clock and a counter, see [62].
- The UTCSU provides features like hardware support for continuous amortization and leap second insertion/deletion, which are not found in alternative approaches.
- Last but not least, the tremendous advances in VLSI technology allowed us to overcome the CSU's obvious design limitations. Our UTCSU provides dozens of wide internal registers (64 bit NTP-time format + 32 bit accuracy) and many additional units like application timestamping features and interfaces to GPS receivers, which would have been impossible to accommodate in the late 80ies.

[‡] In fact, those figures have been obtained by completely ignoring the large granularity $G = 1$ μs of the utilized clock, which has a tremendous impact on the achievable worst case precision, see Chapter 3 and 5.

Relating our NTI to purely hardware-based approaches, in particular, phase-locked-loop clocks and dedicated GPS receivers, is more difficult due to different system assumptions. For example, PLL-clocks provide a superior accuracy/precision down to the ns-range, but require a dedicated and fully connected clocking network, recall our classification in Section 2.1. Hence, solutions like [46] cannot be used in distributed systems like ours, where nodes are interconnected by a standard data network only.

On the other hand, in view of the negligible costs of GPS receivers, it is tempting to solve the clock synchronization problem simply by equipping each node of a distributed system with a modular GPS timing receiver. Although this solution provides an excellent accuracy/precision in the 100 ns-range, it is not feasible when stringent fault-tolerance requirements are to be met. First of all, it is arguable for certain fault-tolerant applications to make a pivotal service like clock synchronization completely dependent upon a system with single points of failures (e.g., in the GPS control segment, see [5]). Moreover, although GPS receivers provide reliable information most of the time, it is nevertheless true that erroneous output may occur: We conducted a 2-month continuous experimental evaluation [16] of the output of six different GPS receivers, which confirmed that it is problematic to always trust the output of a GPS receiver. A real-time systems architecture like the one of [15], which simply phase-locks an internal clock to the 1PPS output of a GPS timing receiver, can therefore assign incorrect timestamps —taken arbitrarily in a one second interval— if there is just a single incorrect 1PPS pulse. In [12], it was even noted that a prototype TDMA communications system at MOTOROLA eventually broke down due to a certain GPS failure.

Apart from fault-tolerance considerations, there are also practical problems with dedicated GPS receivers that severely limit their applicability. Just consider the problem of accommodating and connecting the “forest” of antennas required for a, say, distributed factory automation system with 100 nodes. After all, any GPS antenna needs full view of the sky and is quite sensitive to multipath reception caused by buildings and other obstacles. Techniques for antenna multiplexing might be used to reduce the number of required antennas, but such techniques reduce the signal quality and can hence serve a few GPS receivers per antenna only. Another problem with dedicated GPS receivers is the large time-to-satellite-fix, which implies that it may take 30 seconds or more until correct time information is delivered after power up. This in turn implies a very large node join in case of re-integrating a failed node.

On the other hand, our NTI in conjunction with interval-based clock validation provides a way to escape from all the abovementioned problems by simultaneously increasing the fault-tolerance degree and decreasing the number of GPS receivers required in the system. This is basically done without additional costs, since our technique uses the existing data network only. The only price paid is a slightly worse precision/accuracy, which is hopefully acceptable for most applications. We should add that the 1 μs-range precision/accuracy claimed for our technique is worst case, i.e., almost never attained in practice. In fact, some preliminary conducted measurements/simulations indicate that the average case precision/accuracy is at least one order of magnitude better than the worst case one.

2.6 Summary and Future Work

The hardware implementation of highly accurate/precise synchronized clocks in the real-time systems domain faces two difficulties: First, a clock circuitry is necessary for maintaining local time/accuracy with sufficient state graininess and rate dynamics. Second, recording and generating events in the proximity of the clock has to be done with minimal uncertainty. Only profound and well designed hardware allows to meet these requirements. Hence, a major result of our research is the development of an NTI M-Module that provides a cheap way of extending state-of-the-art real-time systems technology with fault-tolerant synchronized clocks with a worst case precision/accuracy in the 1 μs-range. The NTI is built around our UTCSU-ASIC that incorporates adder-based clocks and sophisticated event handling facilities. In this chapter we gave the specification of the UTCSU in terms of 16 rules focusing on the key implementation parts, and presented a basic programming model.

We are currently negotiating a pilot industrial project in the area of on-line fault location for underground power cables. The —already patented— idea is to equip each incoming/outgoing power trunk in a power plant/transformer station with a detector that produces a digital signal when a transient indicating a cable break or short-circuit arrives. This signal is fed to the application timestamp input of the NTI of a nearby located microprocessor node, which is in turn connected to all the other nodes in the same power plant/transformer station by a fiber-optic data network. By relating the timestamps gathered at both ends of a cable in case of damage, it is possible to determine the fault location within a few 10 meters.

Although this is by now the only practical application we are aware of that really requires our high precision/accuracy, we are nevertheless convinced that others in computer science, like multimedia or mobile computing, will eventually emerge when enabling technology is available.

◇

Chapter 3

INTERVAL-BASED CLOCK STATE SYNCHRONIZATION

3.1 Introduction

One important problem that needs to be addressed when dealing with distributed real-time systems is the issue of global time. To support very large —world-wide— distributed applications, like automatic instrument landing systems (ILS), for example, each computing node in the system should have local access to a reliable system time satisfying the following two application requirements:

(A) *Accuracy* (= maximum deviation of local clock reading from real-time)

Time rules daily life and for that reason most commercial computer applications, for instance a flight reservation system. Hence, system time must have a well-defined relation to the only official and legal world-wide standard *Universal Time Coordinated* (UTC), which is a “paper clock” computed from the weighted average of approximately 200 atomic clocks located all over the world, with leap seconds occasionally inserted/deleted to keep UTC close to astronomical time (not exceeding a deviation of 0.9 seconds to UT1). It is made publicly available world-wide primarily by means of radio transmission, most notably by the NAVSTAR *Global Positioning System* (GPS), which has changed the world of accurate time and position measurement completely, see [5] for an up-to-date overview.

Providing an accurate global time in distributed systems is usually termed as the *external clock synchronization problem*, due to the fact that UTC has to be provided externally to the system. The probably most well-known solution is the *Network Time Protocol* (NTP), which was designed to establish a global time related to UTC in the Internet, see [38] or [40]. NTP provides its clients with an average accuracy below 10 ms, which amply fulfills the modest accuracy requirements of typical applications.

(P) *Precision* (= maximum difference of simultaneous local clock readings)

Algorithms for (fault-tolerant) distributed systems are usually considerably simplified and improved when approximately synchronized local clocks are available, see [27] for some examples. Distributed real-time systems depend upon precise global time even at a very low level of operation. For example, timestamping is often employed for establishing a global order of (external) events occurring at different nodes.

Providing mutually synchronized local clocks is known as the *internal clock synchronization problem*, and numerous solutions have been worked out (at least in scientific research) under the term *fault-tolerant clock synchronization*, see [65] for an overview. In fact, there are more than 60 papers listed in the 1993 bibliography [75] of clock synchronization in distributed systems. The actual precision requirements of typical applications are in the range below 1 ms, although increasingly demanding applications like an airborne flight control system supporting ILS will certainly push these requirements down to the μs-range. Aiming at that high precision is also advantageous w.r.t. improving the performance of most distributed algorithms.

Different components (i.e., applications) within a heterogeneous distributed system have usually different requirements concerning accuracy and/or precision. However, it should be clear that a global time that satisfies both requirements simultaneously is preferable over a solution that provides "translations" between parts of the system employing their own idea of precise/accurate time, see [25].

Unfortunately, it turns out that establishing a precise global time that also relates to some external time standard like UTC in fault-tolerant distributed systems is not a simple matter of combining techniques from (A) and (P). Informally, it is difficult to add accuracy to existing solutions for internal synchronization, since such algorithms are necessarily reluctant to obey "authoritative" information of a few UTC time sources due to fault-tolerance. On the other hand, whereas high precision is of course implied by high accuracy, it is usually not feasible to build a fault-tolerant time service that is purely based on accuracy, since highly accurate UTC is not continuously available.

Although it has been recognized early that the problem of *fault-tolerant external clock synchronization* constitutes a research topic in its own right, see [4], it did not receive much attention until recently. A short overview of existing earlier work is given in [50], whereas our research on *interval-based clock validation* aims at a solution of the external clock synchronization problem for large-scale, fault-tolerant real-time systems. Clock validation algorithms are based on the idea of verifying whether the highly accurate external time provided by non-faulty UTC time sources is consistent with some less accurate but more reliable “validation time” formed by exchanging information of all local clocks in the system. If so, the distinguished time is accepted, otherwise, it is discarded and the nodes rely on the validation time instead.

There might of course be phases of unavailability of accurate time information from the UTC time sources. For clock validation, this means that the system automatically undergoes a transition to internal synchronization, and a transition back to normal operation when UTC is available again. This *flywheel operation* implies, however, that the clock synchronization algorithm (employed for computing the validation time) must not only ensure precision but should maintain high accuracy as well.

When system time must have a defined relation to external time, there is a promising alternative to the “one-dimensional” point of view sufficient for internal synchronization, viz. the *interval-based paradigm* introduced in [35] and [37]. Interval-based algorithms represent time information relating to an external standard like UTC by intervals that are known (better to say supposed) to contain UTC. Given a set of such intervals from different sources, a usually smaller interval that actually contains UTC may be determined, even if some of the source intervals are faulty.

We consider the interval-based paradigm as being exceptionally suitable for dealing with fault-tolerant external clock synchronization. Since accuracy is maintained dynamically here, it provides an average case behavior that is much better than the worst case one. By contrast, worst-case accuracy bounds for internal synchronization algorithms are necessarily static in nature, allowing no improvement w.r.t. average case at all. Surprisingly enough, however, interval-based approaches did not receive much attention in research. To our knowledge, there is only Lamport’s technical report [25], Marzullo’s work [36] on replicated sensors, and our paper [50] introducing interval-based clock validation that further exploit ideas of [35]. However, it is worth mentioning that both DTS, the *Digital Time Synchronization Service* of OSF/DCE, and newer versions of NTP are built upon the interval-based paradigm, see [43] and [40], respectively.

This chapter provides description and analysis of a simple interval-based algorithm suitable for internal clock synchronization, which is generic w.r.t. the convergence function employed. Thus, our results are given in terms of some characteristic parameters of the convergence function, see [57]. In order to determine precision and accuracy of a particular instance of our algorithm, it only remains to determine the characteristic parameters of the particular convergence function, and to plug them into the generic results.

The outline of the rest of the chapter is as follows: Section 3.2 introduces the interval-based paradigm, focusing on both accuracy and precision intervals. The system model for processors, local clocks, and communications in conjunction with a discussion of the (generic) fault model is contained in Section 3.3. Section 3.4 eventually provides our clock synchronization algorithm and outlines some of its properties. The generic analysis of precision and accuracy, along with the definition of internal global time and the treatment of continuous amortization, is given in Section 3.5. Finally, some conclusions and directions for further research are appended in Section 3.6

3.2 The Interval-Based Paradigm

To introduce the interval-based paradigm, we have to establish some basic notation and operations on intervals first. We use bold letters like $\boldsymbol{I}$ to denote a real interval $\boldsymbol{I} = [x, y]$, $x \leq y$, with lower and upper *edge* x and y, respectively; the *empty interval* $\emptyset$ satisfies $\nexists t : t \in \emptyset$. A *set* of intervals is denoted by a calligraphic bold letter like $\boldsymbol{\mathcal{I}}$. For an interval $\boldsymbol{I} = [x, y]$, $||\boldsymbol{I}|| = y - x$ denotes its *length* and $\text{center}(\boldsymbol{I}) = (x + y)/2$ its *centerpoint*. The *sum* of two intervals is defined by $[x, y] + [u, v] = [x + u, y + v]$, the *scalar product* by $b[x, y] = [bx, by]$ for $b \geq 0$, and $\boldsymbol{I} + a = \boldsymbol{I} + [a, a] = [x + a, y + a]$ for some arbitrary scalar a. For two intervals $[x, y]$, $[u, v]$, the *intersection* reads $[x, y] \cap [u, v] = [\max\{x, u\}, \min\{y, v\}]$ if $u \leq y$ and $v \geq x$, or $\emptyset$ otherwise, and the *union* is $[x, y] \cup [u, v] = [\min\{x, u\}, \max\{y, v\}]$, even valid for $[x, y] \cap [u, v] = \emptyset$.

Most of the intervals encountered in our setting contain a distinguished *reference point* that partitions the interval into a *negative* and a *positive accuracy*. To that end, we introduce the notation

$$\boldsymbol{I} = [r \pm \boldsymbol{\alpha}] = [r - \alpha^-, r + \alpha^+] = [x, y]_r, \tag{3.1}$$

where

r	$=$	$\text{ref}(\boldsymbol{I})$	$\boldsymbol{I}$'s reference point (also called midpoint),
α^+	$=$	$\text{acc}^+(\boldsymbol{I}) \geq 0$	$\boldsymbol{I}$'s positive (upper) accuracy,
α^-	$=$	$\text{acc}^-(\boldsymbol{I}) \geq 0$	$\boldsymbol{I}$'s negative (lower) accuracy,
x	$=$	$\text{left}(\boldsymbol{I}) = r - \alpha^-$	$\boldsymbol{I}$'s lower edge (envelope),
y	$=$	$\text{right}(\boldsymbol{I}) = r + \alpha^+$	$\boldsymbol{I}$'s upper edge (envelope).

Note that we usually suppress the subscript r in intervals of the form $[r - \alpha^-, r + \alpha^+]_r$ for brevity.

When referring to an interval $[r \pm \boldsymbol{\alpha}]$, $\boldsymbol{\alpha} = [-\alpha^-, \alpha^+]$ denotes its *interval of accuracies* and $\alpha = ||\boldsymbol{\alpha}|| = \alpha^+ + \alpha^-$ this interval's length. For intervals $\boldsymbol{I} = [r \pm \boldsymbol{\alpha}]$ and $\boldsymbol{J} = [s \pm \boldsymbol{\beta}]$, we have $||\boldsymbol{I}|| = \alpha^+ + \alpha^- = \alpha$, center$(\boldsymbol{I}) = r + (\alpha^+ - \alpha^-)/2$, $\boldsymbol{I} + \boldsymbol{J} = [r + s \pm \boldsymbol{\gamma}]$ where $\boldsymbol{\gamma} = \boldsymbol{\alpha} + \boldsymbol{\beta} = [-(\alpha^- + \beta^-), \alpha^+ + \beta^+]$, $\boldsymbol{I} + a = [r + a \pm \boldsymbol{\alpha}]$ for an arbitrary scalar a, and $b\boldsymbol{I} = [br \pm \boldsymbol{\mu}]$ with $\boldsymbol{\mu} = b\boldsymbol{\alpha} = [-b\alpha^-, b\alpha^+]$ for any scalar $b \geq 0$. There is also a notation to express intervals obtained from (3.1) by *swapping* the positive and negative accuracy, namely

$$\overline{\boldsymbol{I}} = \overline{[r \pm \boldsymbol{\alpha}]} = [r \mp \boldsymbol{\alpha}] = [r - \alpha^+, r + \alpha^-] = [r \pm \overline{\boldsymbol{\alpha}}] \tag{3.2}$$

where $\overline{\boldsymbol{\alpha}} = [-\alpha^+, \alpha^-]$.

3.2.1 Accuracy Intervals

The core idea of the interval-based paradigm introduced in [35] is to represent time information by a time-dependent *accuracy interval*. More specifically, an accuracy interval $\boldsymbol{A} = \boldsymbol{A}(t)$ representing real-time t is an interval satisfying $t \in \boldsymbol{A}$. Accuracy intervals are primarily provided by *interval clocks*, which are interval-valued functions $\boldsymbol{C}(t) = [L(t), U(t)]$ with the edges $L(t)$ and $U(t)$ forming lower and upper *envelope*, respectively, of real-time t. In practice, interval clocks are implemented as an ordinary clock $C(t)$ in conjunction with a time-dependent interval $\boldsymbol{\alpha}(t) = [-\alpha^-(t), \alpha^+(t)]$ of *accuracies* taken relatively to the clock's value, hence $\boldsymbol{C}(t) = [C(t) - \alpha^-(t), C(t) + \alpha^+(t)]$.

Definition 3.1 (Interval Relations) *Accuracy intervals are categorized as follows:*

(1) Two accuracy intervals $\boldsymbol{I} = \boldsymbol{I}(t_1)$ and $\boldsymbol{J} = \boldsymbol{J}(t_2)$ are compatible iff they both represent the same real-time $t_1 = t_2 = t$.

(2) Two compatible accuracy intervals $\boldsymbol{I}$ and $\boldsymbol{J}$ are consistent iff $\boldsymbol{I} \cap \boldsymbol{J} \neq \emptyset$.

(3) An accuracy interval $\boldsymbol{I} = \boldsymbol{I}(t)$ representing real-time t is accurate iff $t \in \boldsymbol{I}$.

Note that two compatible and accurate accuracy intervals are consistent, whereas two compatible consistent accuracy intervals $\boldsymbol{I}, \boldsymbol{J}$ are not necessarily accurate since possibly $t \notin \boldsymbol{I} \cap \boldsymbol{J}$. Moreover, consistency is not transitive in general since non-empty intersections $\boldsymbol{I} \cap \boldsymbol{J}$ and $\boldsymbol{J} \cap \boldsymbol{K}$ do not imply $\boldsymbol{I} \cap \boldsymbol{K} \neq \emptyset$. However, we have the following lemma:

Lemma 3.1 (Consistency and Intersection) *If $n \geq 2$ compatible accuracy intervals $\boldsymbol{I}_1, \ldots, \boldsymbol{I}_n$ are* mutually consistent *in the sense that they are all pairwise consistent, then $\bigcap_{j=1}^{n} \boldsymbol{I}_j \neq \emptyset$.*

Proof. Using induction, we first have $\boldsymbol{I}^1 = \boldsymbol{I}_1 \neq \emptyset$, and provided that $\boldsymbol{I}^{k-1} = \bigcap_{i=1}^{k-1} \boldsymbol{I}_i \neq \emptyset$ we conclude non-emptiness for $\boldsymbol{I}^k$: Assuming the contrary, $\boldsymbol{I}_k$ would lie entirely left (or right) of $\boldsymbol{I}^{k-1}$ so that it cannot be consistent with the interval $\boldsymbol{I}_l$, $1 \leq l \leq k-1$ whose left (right) edge delimits $\boldsymbol{I}^{k-1}$. □

3.2.2 Precision Intervals

Apart from the requirement of being accurate, a property of a single accuracy interval, we also have to deal with the precision requirement that applies to (the reference points of) a set of accuracy intervals. In the traditional framework, a set of ordinary clocks is called *precise* with *precision* π during some time interval D, iff $|C_p(t) - C_q(t)| \leq \pi$ for $t \in \mathcal{T}$ (and clocks progress linearly with t). In our setting, precision can easily be added to accuracy, since none of the interval relations given in Definition 3.1 involves the reference point; it is not required for accuracy purposes and can in principle be set to any point within the accuracy interval.

Unlike traditional approaches, we utilize a definition of precision that is based on intervals, inspired by the following lemma:

Lemma 3.2 (Precision Equivalence) *Given some* $\boldsymbol{\pi} = [-\pi^-, \pi^+]$ *with* $\pi^-, \pi^+ \geq 0$ *and* $\pi = ||\boldsymbol{\pi}|| = \pi^- + \pi^+$, *and* $n \geq 2$ *non-negative real numbers* $r_1, \ldots, r_n$, *we have*

$$|r_i - r_j| \leq \pi \quad \text{for all } i,j \qquad \Longleftrightarrow \qquad \bigcap_{j=1}^{n} [r_j \pm \boldsymbol{\pi}] \neq \emptyset.$$

Proof. To show direction $\Leftarrow$, we note that there is some $t' \in \bigcap_{j=1}^{n} [r_j \pm \boldsymbol{\pi}]$ so that immediately $\pi \geq \max_{1 \leq i \leq n}\{r_i\} - \min_{1 \leq j \leq n}\{r_j\} \geq |r_i - r_j|$ for all i, j. The other direction follows from Lemma 3.1, since $|r_i - r_j| \leq \pi$ for all i, j implies that the intervals $[r_j \pm \boldsymbol{\pi}]$ are mutually consistent. □

This suggests the following definition of precision:

Definition 3.2 (Precision Intervals) *Given* $\boldsymbol{\pi} = [-\pi^-, \pi^+]$ *with* $\pi^-, \pi^+ \geq 0$ *and* $\pi = ||\boldsymbol{\pi}|| = \pi^- + \pi^+$, *and a set of* $n \geq 2$ *compatible accuracy intervals* $\boldsymbol{\mathcal{I}} = \{\boldsymbol{I}_1, \ldots, \boldsymbol{I}_n\}$ *with* $\boldsymbol{I}_j = [r_j \pm \boldsymbol{\alpha}_j]$, *the* $\boldsymbol{\pi}$-precision interval $\hat{\boldsymbol{I}}_j$ *associated with* $\boldsymbol{I}_j$ *is defined as*

$$\hat{\boldsymbol{I}}_j = [r_j \pm \boldsymbol{\pi}].$$

The set $\boldsymbol{\mathcal{I}}$ *is called* $\boldsymbol{\pi}$*-precise iff* $\bigcap_{j=1}^{n} \hat{\boldsymbol{I}}_j \neq \emptyset$.

Keep in mind that the associated $\boldsymbol{\pi}$-precision interval $\hat{\boldsymbol{I}}$ of an accuracy interval $\boldsymbol{I}$ is not separately maintained, but rather computed from the reference point of $\boldsymbol{I}$. Thus, precision and accuracy are orthogonal issues here. Note also that we cannot safely assume $\hat{\boldsymbol{I}}_p \subseteq \boldsymbol{I}_p$ in case of small accuracies.

Our definition of $\boldsymbol{\pi}$-precision has several immediately apparent consequences, most importantly, that $\boldsymbol{\pi}$-precision implies precision π:

Lemma 3.3 ($\boldsymbol{\pi}$-Precision vs. Precision) *Given a $\boldsymbol{\pi}$-precise set $\boldsymbol{\mathcal{I}} = \{\boldsymbol{I}_1, \ldots, \boldsymbol{I}_n\}$ of $n \geq 2$ compatible accuracy intervals $\boldsymbol{I}_j = [r_j \pm \boldsymbol{\alpha}_j]$ and $\pi = ||\boldsymbol{\pi}||$, then*

(1) $||\hat{\boldsymbol{I}}_i \cup \hat{\boldsymbol{I}}_j|| \leq 2\pi$ *for any* $1 \leq i, j \leq n$,

(2) $|r_i - r_j| \leq \pi$ *for any* $1 \leq i, j \leq n$.

Proof. The first assertion follows from the fact that $\hat{\boldsymbol{I}}_i$ and $\hat{\boldsymbol{I}}_j$ are consistent and that $||\hat{\boldsymbol{I}}_l|| \leq \pi$ for $1 \leq l \leq n$, according to the definition of the associated $\boldsymbol{\pi}$-precision intervals. The second assertion is an immediate consequence of Definition 3.2 and Lemma 3.2. □

Lemma 3.4 (Precision by Accurateness) *If the $n \geq 2$ compatible accuracy intervals $\boldsymbol{\mathcal{I}} = \{\boldsymbol{I}_1, \ldots, \boldsymbol{I}_n\}$ with $\boldsymbol{I}_j = [r_j \pm \boldsymbol{\alpha}_j]$ are accurate and $\alpha_j^- \leq \alpha^-$, $\alpha_j^+ \leq \alpha^+$ for all j, then $\boldsymbol{\mathcal{I}}$ is also $\boldsymbol{\pi}$-precise for any $\boldsymbol{\pi} \supseteq [-\alpha^-, \alpha^+]$.*

Proof. Using $\boldsymbol{\pi} \supseteq [-\alpha^-, \alpha^+]$ in Definition 3.2, we have $\hat{\boldsymbol{I}}_j \supseteq \boldsymbol{I}_j$ and the statement of the lemma follows from $t \in \bigcap_{j=1}^n \boldsymbol{I}_j \neq \emptyset$. □

Lemma 3.5 (Composition of Precisions) *Let two sets $\boldsymbol{\mathcal{I}} = \{\boldsymbol{I}_1, \ldots, \boldsymbol{I}_n\}$ and $\boldsymbol{\mathcal{J}} = \{\boldsymbol{J}_1, \ldots, \boldsymbol{J}_m\}$ of compatible accuracy intervals be $\boldsymbol{\pi}$-precise and $\boldsymbol{\pi}'$-precise, respectively. If $\left(\bigcap_{j=1}^n \hat{\boldsymbol{I}}_j\right) \cap \left(\bigcap_{i=1}^m \hat{\boldsymbol{J}}_i\right) \neq \emptyset$, then the set $\boldsymbol{\mathcal{I}} \cup \boldsymbol{\mathcal{J}}$ is $\boldsymbol{\pi} \cup \boldsymbol{\pi}'$-precise.*

Proof. Since the $\boldsymbol{\pi}$-precise set $\boldsymbol{\mathcal{I}}$ and the $\boldsymbol{\pi}'$-precise set $\boldsymbol{\mathcal{J}}$ are also $\boldsymbol{\pi} \cup \boldsymbol{\pi}'$-precise, the statement follows immediately from Definition 3.2. □

Our definition of $\boldsymbol{\pi}$-precision is a key issue in our novel interval-based framework for precision analysis. Nevertheless, there are only a few occasions where we actually face $\boldsymbol{\pi}$-precise intervals according to Definition 3.2. In most cases, we employ the slightly stronger predicate of *$\boldsymbol{\pi}$-correctness* as provided in Definition 3.3. It characterizes the $\boldsymbol{\pi}$-precision interval $\hat{\boldsymbol{I}}$ associated with an accuracy interval $\boldsymbol{I}$ as being accurate w.r.t. an appropriately defined *internal global time* $\tau = \tau(t)$. Actually, $\boldsymbol{I} = \boldsymbol{I}(t)$ is called $\boldsymbol{\pi}$-correct iff both $\tau \in \hat{\boldsymbol{I}}$ and $t \in \boldsymbol{I}$, in other words, iff $\boldsymbol{I}$ is accurate w.r.t. both τ and t, as shown in Figure 3.1.

At this point there is no need to elaborate on how internal global time is actually maintained; consult Section 3.5.1 for details. Intuitively, we exploit the fact that our clock synchronization algorithm maintains the set $\boldsymbol{\mathcal{C}} = \boldsymbol{\mathcal{C}}(t)$ of non-faulty interval clocks $\boldsymbol{C}_p(t)$ so that it is $\boldsymbol{\pi}_0$-precise at (periodic) resynchronization real-times $t^{R,(k)}$, characterizing the beginning of the $(k+1)$-th *round* $(k \geq 0)$. This property allows us to define round k's unique internal global time $\tau^{(k)} = \tau^{(k)}(t)$ by $\tau^{(k)}(t) = \tau^{(k)}(t^{R,(k)}) + (t - t^{R,(k)})$ for any

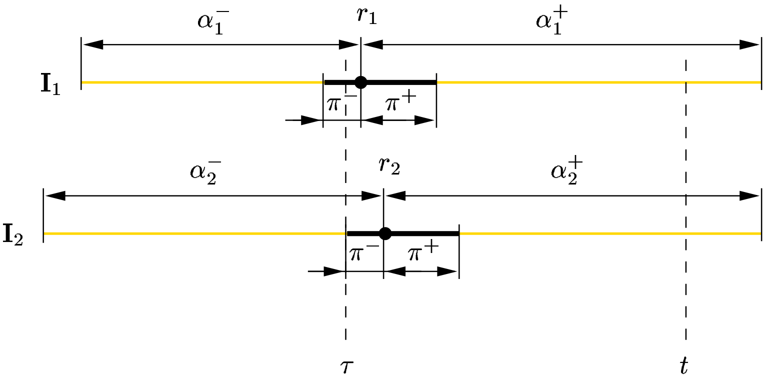


Figure 3.1: *Accuracy and Precision Intervals*

real-time t; the "fixed point" $\tau^{(k)}(t^{R,(k)})$ is some arbitrary value satisfying $\tau^{(k)}(t^{R,(k)}) \in \bigcap_{\boldsymbol{C}_j \in \boldsymbol{\mathcal{C}}} \hat{\boldsymbol{C}}_j(t^{R,(k)}) \neq \emptyset$. For large accuracies, it is likely that $\tau^{(k)}(t) \neq t$, although Lemma 3.4 justifies that choosing $\tau(t) = t$ is possible if accuracies are sufficiently small (i.e., $\pi > \alpha$). However, internal global time of any round progresses as real-time does, so that it can be used interchangeably with real-time if one is interested in measuring durations only.

Definition 3.3 ($\boldsymbol{\pi}$-correctness) *For* $\boldsymbol{\pi} = [-\pi^-, \pi^+]$ *with* $\pi^-, \pi^+ \geq 0$,

(1) an accuracy interval $\boldsymbol{I}$ *is* $\boldsymbol{\pi}$*-accurate (w.r.t. internal global time of round k) iff the* $\boldsymbol{\pi}$*-precision interval* $\hat{\boldsymbol{I}} = \hat{\boldsymbol{I}}(\tau^{(k)}) = \hat{\boldsymbol{I}}\big(\tau^{(k)}(t)\big)$ *associated with* $\boldsymbol{I}$ *satisfies* $\tau^{(k)} \in \hat{\boldsymbol{I}}$,

(2) an accuracy interval $\boldsymbol{I}$ *is* $\boldsymbol{\pi}$*-correct (w.r.t. internal global time of round k) iff* $\boldsymbol{I}$ *is both* $\boldsymbol{\pi}$*-accurate and accurate,*

(3) a set $\boldsymbol{\mathcal{I}}$ *of compatible intervals is* $\boldsymbol{\pi}$*-correct if all* $\boldsymbol{I} \in \boldsymbol{\mathcal{I}}$ *are* $\boldsymbol{\pi}$*-correct.*

Lemma 3.6 (Relation T and τ) *If the intervals* $\boldsymbol{I}_1(t_1) = [T_1 \pm \boldsymbol{\alpha}_1]$ *and* $\boldsymbol{I}_2(t_2) = [T_2 \pm \boldsymbol{\alpha}_2]$ *are* $\boldsymbol{\pi}_1$*-accurate and* $\boldsymbol{\pi}_2$*-accurate (w.r.t. internal global time of the same round), respectively, and* $\tau_1 = \tau(t_1)$, $\tau_2 = \tau(t_2)$, *then* $\tau_1 - \tau_2 = t_1 - t_2 \in T_1 - T_2 + \boldsymbol{\pi}_1 + \overline{\boldsymbol{\pi}}_2$.

Proof. Since, for $i = 1, 2$, $\tau_i \in \hat{\boldsymbol{I}}_i(\tau_i)$ according to the asserted $\boldsymbol{\pi}_i$-correctness of $\boldsymbol{I}_i$, we have $T_i - \pi_i^- \leq \tau_i \leq T_i + \pi_i^+$. Subtracting those inequalities and recalling the notion of swapped intervals easily provides the statement of the lemma; note that $\tau_1 - \tau_2 = t_1 - t_2$ since internal global time (of the same round) progresses as real-time does. □

In the following we will frequently employ the abbreviation $\tau = \tau(t) = \tau^{(k)}(t)$ as above when the particular round k is clear from the context. Moreover, we can usually unambiguously write $\hat{\boldsymbol{I}} = \hat{\boldsymbol{I}}(t) = \hat{\boldsymbol{I}}(\tau^{(k)})$ for $\tau^{(k)} = \tau^{(k)}(t)$, meaning that $\hat{\boldsymbol{I}}$ represents $\tau^{(k)}(t)$ iff $\boldsymbol{I}$ represents t.

The following lemma is a simple corollary of Lemma 3.6 for $t_1 = t_2 = t$:

Lemma 3.7 (Distance Reference Points) *If the compatible intervals* $\boldsymbol{I}_1(t) = [T_1 \pm \boldsymbol{\alpha}_1]$ *and* $\boldsymbol{I}_2(t) = [T_2 \pm \boldsymbol{\alpha}_2]$ *are* $\boldsymbol{\pi}_1$*-accurate and* $\boldsymbol{\pi}_2$*-accurate (w.r.t. internal global time of the same round), respectively, then* $T_2 - T_1 \in \boldsymbol{\pi}_1 + \overline{\boldsymbol{\pi}}_2$.

Note that this result, which is of central importance for precision analysis, can be viewed as a consequence of the fact that both $\hat{\boldsymbol{I}}_1(\tau)$ and $\hat{\boldsymbol{I}}_2(\tau)$ must contain τ, so that $T_2 - T_1 \in \left[-(\pi_2^+ + \pi_1^-), \pi_2^- + \pi_1^+\right] = \boldsymbol{\pi}_1 + \overline{\boldsymbol{\pi}}_2$.

It should be obvious that a set of compatible $\boldsymbol{\pi}$-correct intervals $\boldsymbol{\mathcal{I}} = \{\boldsymbol{I}_1, \ldots, \boldsymbol{I}_n\}$ is $\boldsymbol{\pi}$-precise due to $\tau \in \bigcap_{j=1}^{n} \hat{\boldsymbol{I}}_j \neq \emptyset$. Therefore, Lemmas 3.3–3.5 are valid for $\boldsymbol{\pi}$-correct sets as well. Of course, since our precision analysis is primarily based upon those lemmas, it would in principle be sufficient to deal with $\boldsymbol{\pi}$-precise intervals. Introducing internal global time and $\boldsymbol{\pi}$-accurateness, however, allows to reason about precision by considering each local interval clock separately, i.e., without explicitly relating it to the other clocks in the system, which greatly simplifies the analysis.

As a consequence, most of the intervals encountered in our analysis are $\boldsymbol{\pi}$-correct ones, so that it does not make much sense to adhere to a strict separation of accuracy/precision terminology. In the remaining sections, will use *interval* as a standard term of generic meaning, while accuracy interval or precision interval is only used if we want to stress the particular “instance” of the interval in question.

3.3 System Modelling

In this section, we will provide the system model and its parameters. Similar models are well-known from the analysis of internal clock synchronization and other distributed algorithms for fault-tolerant systems. However, non-standard features are incorporated in our discrete clock model, which is interval-based and deals with non-zero clock granularity, and in the model of the communication subsystem, which contains parameters for broadcast latencies and limited transmission bandwidth.

We consider a distributed system consisting of n *nodes*, which may communicate with each other by message passing over a suitable communication network. Each node is equipped with a *processor* that executes the clock synchronization algorithm, an adjustable *local clock*, and a *network interface*.

As we will see later on, all computations required for clock synchronization purposes are essentially periodic and require integer arithmetic only. As far as the execution speed of the processor executing the algorithm is concerned, we assume the following:

Assumption 3.1 (Execution Times) *A single computation required for clock synchronization purposes at a non-faulty node p is completed within η_p seconds. Let $\eta_{\max} \geq \eta_p$, $\eta_{\min} \leq \eta_p$ be suitable uniform bounds on the execution time of all non-faulty nodes p.*

Note that we do not make further assumptions about application tasks the processor might perform concurrently with clock synchronization; a fault-free node p must solely guarantee the bound η_p.

In the following two subsections, we will develop the system model for local clocks and for the communication subsystem without considering faults. The fault model is discussed in the last subsection.

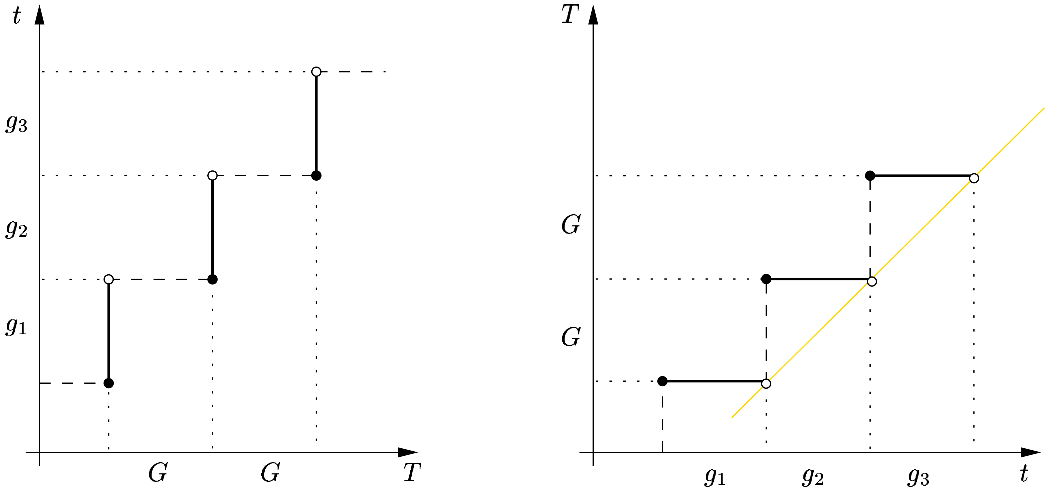
3.3.1 Discrete Local Interval Clocks

The local clock of a node is assumed to be built upon a physical clock (usually driven by a quartz oscillator) of non-zero granularity G (micro-)seconds, which allows adjustment of rate and state. Non-zero granularity implies that the clock is incremented by G at discrete points in real-time (called *clock ticks*) only, posing a particular challenge to system modelling.

A clock is usually modelled as a monotonic function $C : t \to T$ mapping *real-time* t to *logical time* $T = C(t)$. Note that we use the convention of writing upper-case names (like T) for logical time and lower-case ones (like t) for real-time values throughout the paper. Most often, $C(t)$ is assumed to be a continuous (differentiable) function, although existing clocks are modelled appropriately by discrete step-functions only. Up to our knowledge, however, discretization and the adverse effect of non-zero clock granularity has been investigated only in [64].

In our analysis, we employ an alternative discretization that is more suitable for the interval-based paradigm. Instead of considering clocks $T = C(t)$ in the first place, we start with *inverse clocks* $t = c(T)$ mapping logical time to real-time. More specifically, "inverting" the approach of [58], we assume that real-time t advances instantaneously a (varying) real value g at each logical tick of the clock, and remains constant everywhere else. Clock ticks take place every (fixed) $G > 0$ logical time seconds, modelling non-zero clock granularity.

Of course $c(T)$ is only meaningful for $T = kG$ being a multiple of the clock granularity G. Unfortunately, when $c(T)$ is actually defined as the inverse of the step-function modelling clock $C(t)$, it is multi-valued (infinitely many values, representing the progress of real-time) at $T = kG$. We enforce a proper function, however, by defining $c(kG)$ to be the value before advancing the clock by g, i.e., $\lim_{T' \to T-} c(T') = c(T) = \lim_{T' \to T+} c(T') - g$ for $T = kG$. The following Figure 3.2 illustrates this.



inverse clock $c(T)$ clock $C(t)$

Figure 3.2: *Discrete Clocks*

Note that this definition of $c(T)$ calls for setting the clock $C(\theta)$'s value at real-time θ, where a tick takes place, to the value after advancing C by G, i.e., $\lim_{t' \to \theta-} C(t') + G = C(\theta) = \lim_{t' \to \theta+} C(t')$. This is due to the fact that $C(t)$ and $c(T)$ must be inverse at real-time θ where $C(t)$ ticks, thus $\theta = c(\Theta)$ for $\Theta = C(\theta)$. In fact, we assume that discrete local time approximates continuous local time by a leading (majorizing) step-function.

In practice, a local clock $C_p(t)$ is primarily characterized by its *intrinsic rate* $r_p = G/g$, which gives the amount of logical time seconds the clock advances per real-time second. More specifically, one usually assumes that there is some $\rho_p^+, \rho_p^- \ll 1$, that account for the clock's rate deviations due to oscillator frequency offset, aging, temperature dependency, noise, etc., such that

$$1 - \rho_p^+ + \mathcal{O}(\rho_p^{+2}) = \frac{1}{1 + \rho_p^+} \leq r_p \leq \frac{1}{1 - \rho_p^-} = 1 + \rho_p^- + \mathcal{O}(\rho_p^{-2}). \tag{3.3}$$

In our clock model, we will employ the equivalent condition $1 - \rho_p^- \leq r_p^{-1} \leq 1 + \rho_p^+$ on the *intrinsic inverse rate* r_p^{-1}, i.e., the rate of the inverse clock, which gives the amount of real-time seconds the inverse clock $c_p(T)$ advances per logical time second.

Unfortunately, granularity G and intrinsic inverse rate r_p^{-1} are not sufficient to describe completely the behavior of discrete clocks. First, *rate adjustment uncertainties* are introduced when local clocks utilize an "artificial" rate generated by discrete rate adjustment techniques. Since it is difficult to fine-tune the frequency of an ordinary quartz oscillator (as opposed to a voltage controlled oscillator, where this is easy), techniques have been developed that allow rate adjustment of a clock by occasionally tampering with raw oscillator ticks. This type of clocks tick at the intrinsic rate most of the time, whether they are adjusted or not. However, when the accumulated deviation between intended and observed local time is about to exceed some bound u, the next tick is modified: If the clock is to be slowed down resp. speeded up, the next regular tick is delayed resp. advanced. Of course, this causes an additional uncertainty in the relation between logical time and real-time not explained by the intrinsic clock rate, which must be taken into account explicitly.

In addition, we have to account for the fact that practical clocks cannot be state adjusted with infinite resolution, but only with a certain *clock setting granularity* $G_S \leq G$. Whereas $G_S = G$ is easily provided by making the clock register writable, it is considerably more expensive to implement fine-grained clock setting capabilities ($G_S < G$). Apart from employing state adjustment controlled by a continuous amortization algorithm, see Section 3.5.3, instantaneous state correction could be implemented directly by utilizing a clock register with higher (internal) resolution G_S or, alternatively, by delaying/advancing the time of setting the clock. In any case, G_S should be considered as the "internal" granularity of the clock, as opposed to the (coarser) granularity G available for external clock reading.

Note that fine-grained clock setting and discrete rate adjustment should not be considered independent of each other, as it might be the case in a sub-optimal clock design. More specifically, we assume that the error between intended and observed local time caused by rate adjustment is added to the initial clock setting error prior to deciding when oscillator ticks are to be modified. For example, in case of $u = G$, if a clock driven by a slow oscillator is set to $G + G/2$, an additional clock tick should be introduced as soon as the accumulated error of the subsequent clock ticks becomes $G/2$, otherwise the total error will exceed $u = G$ by then.

With these preparations, we are ready for stating our basic model of local clocks. It does not incorporate explicit rate adjustment capabilities required for continuous amortization, which are added in Section 3.5.3. However, rate adjustments may already be incorporated here for fine-tuning of the intrinsic rate.

Assumption 3.2 (Local Clocks) *Each node p is endowed with a discrete* local clock *$C_p(t)$, which increments by $G > 0$ seconds at each clock tick and allows state adjustment with clock setting granularity of at least $G_S = G/K$ seconds, for some integer $K \geq 1$. In* *the absence of resynchronizations, intrinsic inverse rate and rate adjustment uncertainty of the clock of a non-faulty node p are such that*

$$(1-\rho_p^-)(\Theta_i-\Theta_1)-u_p^- \leq \theta_i-\theta_1 \leq (1+\rho_p^+)(\Theta_i-\Theta_1)+u_p^+ \tag{3.4}$$

with $\Theta_j = C_p(\theta_j)$ *is guaranteed for any sequence of* $i \geq 2$ *successive clock ticks* θ_j, $1 \leq j \leq i$. *For* $\boldsymbol{\rho}_p = [-\rho_p^-, \rho_p^+]$ *denoting the clock's intrinsic inverse rate deviation bounds and* $\boldsymbol{u}_p = [-u_p^-, u_p^+]$ *its maximum rate adjustment uncertainty, let* $\boldsymbol{\rho}_{\max} = [-\rho_{\max}^-, \rho_{\max}^+] \supseteq \bigcup_p \boldsymbol{\rho}_p$ *with* $\rho_{\max} = \rho_{\max}^- + \rho_{\max}^+$ *and* $\boldsymbol{u}_{\max} = [-u_{\max}^-, u_{\max}^+] \supseteq \bigcup_p \boldsymbol{u}_p$ *with* $u_{\max} = u_{\max}^- + u_{\max}^+ = \mathcal{O}(G)$ *be suitable uniform bounds for all (non-faulty) nodes p.*

Of course, inequality (3.4) is also valid for $i = 1$, although the bounds are not particularly meaningful, besides from the fact that $\boldsymbol{u}_p$ can be used to account for the fractional clock setting value ($< G$) in case of fine-grained clock setting.

It seems appropriate here to sketch how conceivable clock implementations map to the above model, i.e., how G, $\boldsymbol{\rho}_p$, G_S and $\boldsymbol{u}_p$ are to be chosen for a certain implementation. First of all, we note that $u_p^- = u_p^+$ for all discrete rate adjustment techniques we are aware of. More specifically, although u_p^+ bounds the logical vs. real-time deviation Δ caused by a fast clock (say, $\Delta > 0$), whereas u_p^- is meaningful for a slow clock ($\Delta < 0$), it is apparent that both $\Delta > 0$ and $\Delta < 0$ occurs when slowing down or speeding up the clock via rate adjustment. In fact, the actual sign of the deviation Δ depends on whether a modified tick is at the beginning ($j = 1$) or beyond the end ($j > i$) of the sequence of ticks under consideration. Consequently, it is the maximum value of any scenario that determines $u_p^- = u_p^+$.

The particular clock models considered are as follows:

- *Counter Clock:* Implemented by means of a fixed-frequency oscillator that increments a counter by $G = 1/f_{osc}$ (usually, $G = 2^{-k}$ for binary counters or $G = 10^{-k}$ for decimal ones) at each tick, it follows that $\boldsymbol{\rho}_p$ is the intrinsic inverse rate deviation bound $\boldsymbol{\rho}_{osc}$ of the oscillator, $G_S = G$, and $\boldsymbol{u}_p = [0, 0]$ since there are no rate adjustment capabilities.

- *Voltage Controlled Oscillator (VCO):* Replacing the fixed-frequency oscillator above by a VCO adds (continuous) rate adjustment capabilities, so that $G = 1/f_{osc}^0$ for the intrinsic (non-adjusted) oscillator frequency f_{osc}^0, $\boldsymbol{\rho}_p = \boldsymbol{\rho}_{osc}^0$, $\boldsymbol{u}_p = [0, 0]$, and $G_S \ll G$ provided state correction is done by continuous amortization.

- *Tick Advancing/Delaying:* One technique for discrete rate adjustment, employed in the CSU of [24], is based on a fixed-frequency oscillator running at a multiple

$f_{osc} = m f_{clock}$, $m \geq 2$, of the desired clock frequency $f_{clock} = 1/G$. Without rate adjustment, every m-th oscillator tick is used to increment the counter, so that $G = 1/f_{clock} = m/f_{osc}$. To speed up the clock, the $m-1$-th oscillator tick is used occasionally, that is, when the accumulated deviation between intended and observed logical time is about to exceed $G/m = 1/f_{osc}$. Similarly, the $m+1$-th oscillator tick is used occasionally if the clock is to be slowed down. Therefore, it is immediately apparent that $\boldsymbol{\rho}_p = \boldsymbol{\rho}_{osc}$, $G_S = G/m$ because clock setting may be delayed in multiples of G/m (both for instantaneous state correction and continuous amortization), and $\boldsymbol{u}_p = [-G/m + \mathcal{O}(G\rho_{osc}), G/m + \mathcal{O}(G\rho_{osc})]$, where the remainder terms account for the deviation between real-time and (observed) logical time.

- *Tick Insertion/Deletion:* This approach is very similar to the above one and is most efficient for $f_{osc} = 2f_{clock}$, so that $G = 1/f_{clock} = 2/f_{osc}$. However, instead of shifting all (future) oscillator ticks as a consequence of any single correction instant, it just inserts an additional oscillator tick between two regular ones in case of speeding up, and suppresses a regular tick if slowing down the clock. Therefore, future ticks occur at the same instants as they would have occurred if no rate correction took place. It follows immediately that $\boldsymbol{\rho}_p = \boldsymbol{\rho}_{osc}$, $G_S = G$ (both for instantaneous state correction and continuous amortization) since slowing down only works in multiples of G (although speeding up would allow $G/2$), and $\boldsymbol{u}_p = [-G + \mathcal{O}(G\rho_{osc}), G + \mathcal{O}(G\rho_{osc})]$.

- *Adder-Based Clock:* This novel clock architecture utilized in our UTCSU-ASIC (see [62] or Chapter 2) uses a fixed-frequency oscillator that drives an adder instead of a counter. At each oscillator tick, a high-resolution ($G_S \ll G^0_{osc} = 1/f_{osc}$) clock register is incremented by a programmable register `STEP`, which usually contains the intrinsic value G^0_{osc}. It can be modified to any value $G^0_{osc} + \delta G_S$ for rate adjustment purposes, with $\delta G_S > 0$ speeding up and $\delta G_S < 0$ slowing down the clock. Hence, an adder-based clock has in fact variable internal granularity, equal to the content of `STEP`. State correction (with clock setting granularity G_S) can be carried out by continuous amortization if `STEP` holds appropriate increments.

  The clock's actual granularity G, which should satisfy $G \geq G^0_{osc}$ to be meaningful, is imposed by the fact that the clock register is externally accessible with resolution G only, resulting in a reading error of up to G. That is, the "fractional part" with resolution $G_S \ll G$ is not visible externally, but nevertheless (continuously) maintained internally. It follows that $\boldsymbol{\rho}_p = \boldsymbol{\rho}_{osc}$ and $\boldsymbol{u}_p = [-G^0_{osc} + \mathcal{O}(G\rho_{osc}), G^0_{osc} + \mathcal{O}(G\rho_{osc})]$ for reasonable clock rate corrections $\delta G_S = \mathcal{O}(\rho_{osc})$, since the clock register is occasionally incremented by $2G$ in case of speeding up the clock ($\texttt{STEP} > G$), and not incremented at all in case of slowing it down ($\texttt{STEP} < G$).

Now we will establish a relation between real-time and logical time intervals as measured on a non-faulty local clock. Recalling the definition of $c(T)$ from the beginning, we have for any t and $T = C(t)$

$$c(T) \leq t < c(T+G). \tag{3.5}$$

Therefore, we easily obtain

$$c(T) - c(T_0 + G) < t - t_0 < c(T+G) - c(T_0) \tag{3.6}$$

for any t, t_0 and $T = C(t), T_0 = C(t_0)$. If $t = \theta$ and/or $t_0 = \theta_0$ denotes some real-time where clock $C(t)$ ticks, we can use the stronger relation $\theta = c(\Theta)$ instead of (3.5), so that the corresponding G in (3.6) may be dropped (however, $<$ has to be replaced by $\leq$). The following definition helps in unifying this situation.

Definition 3.4 (Synchrony) *Real-time t is in synchrony with a node's local clock $C(t)$ iff $t = \theta$ for some real-time θ where $C(t)$ ticks. Let the indicator function of non-synchrony be defined as*

$$I_{t \neq \theta} = I_\theta(t) = \begin{cases} 0 & \textit{if } t \textit{ is in synchrony with } C(t), \\ 1 & \textit{otherwise.} \end{cases} \tag{3.7}$$

With the help of this definition, we can generalize (3.6) to

$$c(T) - c(T_0 + I_{t_0 \neq \theta_0} G) \leq t - t_0 \leq c(T + I_{t \neq \theta} G) - c(T_0). \tag{3.8}$$

For a non-faulty node p, (3.4) immediately provides

$$(1 - \rho_p^-)\Delta T - u_p^- \leq c_p(T + \Delta T) - c_p(T) \leq (1 + \rho_p^+)\Delta T + u_p^+, \tag{3.9}$$

for any T, ΔT being integer multiples of G. Combining this with (3.8), we easily obtain the following inequalities estimating the real-time interval $t - t_0$ in terms of the corresponding logical time interval $T - T_0 = C_p(t) - C_p(t_0)$ for a non-faulty clock in the absence of resynchronizations:

Lemma 3.8 (Duration Estimations) *Let t_0 and $t \geq t_0$ be two arbitrary points in real-time and $T = C_p(t)$, $T_0 = C_p(t_0)$ the corresponding points in logical time at node p. If clock $C_p(t)$ is non-faulty and if there are no adjustments, we have*

$$\begin{aligned} t - t_0 &\geq (T - T_0)(1 - \rho_p^-) - u_p^- - I_{t_0 \neq \theta_0} G(1 - \rho_p^-) \\ t - t_0 &\leq (T - T_0)(1 + \rho_p^+) + u_p^+ + I_{t \neq \theta} G(1 + \rho_p^+) \end{aligned} \tag{3.10}$$

and the converse

$$\frac{t - t_0 - u_p^+}{1 + \rho_p^+} - I_{t \neq \theta} G \leq T - T_0 \leq \frac{t - t_0 + u_p^-}{1 - \rho_p^-} + I_{t_0 \neq \theta_0} G. \tag{3.11}$$

Proof. Due to monotonicity of $C_p(t)$, we always have $T \geq T_0$ since we assumed $t \geq t_0$. If $T > T_0 + G$, (3.10) follows immediately from plugging in (3.9) at both sides of (3.8). Moreover, it is immediately apparent that (3.10) is also valid for $T = T_0$ and $T = T_0 + G$, although the lower bound is not particularly meaningful. Finally, the converse relation follows by trivial algebraic manipulations. □

Neglecting terms of order $\mathcal{O}(\rho_p^2)$ and $\mathcal{O}(G\rho_p)$ in (3.11) we easily obtain the common formula

$$(t - t_0)(1 - \rho_p^+) - u_p^+ - I_{t \neq \theta} G \leq T - T_0 \leq (t - t_0)(1 + \rho_p^-) + u_p^- + I_{t_0 \neq \theta_0} G;$$

note that ρ_p^-, ρ_p^+ are swapped here. Apart from u_p^-, u_p^+, this estimate is the one of [64] improved w.r.t. the quite usual case where t or/and t_0 is in synchrony with the ticks of $C_p(t)$. In that case, there is no need to spoil the appropriate upper/lower bound by G, actually halving (or even ruling out completely) the adverse effects of non-zero clock granularity.

We link the above clock model with the interval-based paradigm introduced in Section 3.2 by means of the most important *drift compensation operation*, cf. [25]: Consider an accurate interval $\boldsymbol{I} = \boldsymbol{I}(t_0) = [T_0 \pm \boldsymbol{\alpha}]$ that somehow appears at a node at some arbitrary real-time t_0. In order to provide an interval $\boldsymbol{I}' = \boldsymbol{I}'(t_1)$ representing some arbitrary real-time $t_1 \geq t_0$ based on $\boldsymbol{I}$ locally at the node, one should move (the reference point of) $\boldsymbol{I}$ to the right by $t_1 - t_0$, thus providing the obviously accurate interval $\boldsymbol{I}' = \boldsymbol{I} + (t_1 - t_0)$. Unfortunately, $t_1 - t_0$ is not available, but can be approximated via local clock $C(t)$. Shifting an interval $\boldsymbol{I}$ from t_0 to t_1 using $C(t)$ is called *dragging*; it provides the interval $\boldsymbol{I}' = \boldsymbol{I} + C(t_1) - C(t_0) = [T_1 \pm \boldsymbol{\alpha}]$. That interval's accurateness, however, might be violated due to the error in approximating $t_1 - t_0$ by $C(t_1) - C(t_0)$.

Deterioration is required to maintain accurateness of the dragged interval. This is done by “blowing up” the dragged interval’s reference point to an interval accounting for the maximum possible approximation error given by (3.10), which amounts to enlarge the positive and negative accuracy of interval $\boldsymbol{I}$ according to (3.12) in Definition 3.5.

Definition 3.5 (Drift Compensation) *The result of drift compensation of an accurate interval* $\boldsymbol{I} = \boldsymbol{I}(t_0)$ *representing an arbitrary real-time* t_0 *(where* $C(t_0) = T_0$*) to some arbitrary real-time* $t_1 \geq t_0$ *(where* $C(t_1) = T_1$*) by means of a local clock with intrinsic inverse rate deviation bound* $\boldsymbol{\rho} \subseteq \boldsymbol{\rho}_{\max}$ *and rate adjustment uncertainty* $\boldsymbol{u} \subseteq \boldsymbol{u}_{\max}$ *is the accurate interval* $\boldsymbol{I}' = \boldsymbol{I}'(t_1)$ *defined by*

$$\boldsymbol{I}' = \boldsymbol{I} + T_1 - T_0 + (T_1 - T_0)\boldsymbol{\rho} + \boldsymbol{u} + I_{t_0 \neq \theta_0}\overline{\boldsymbol{G}} + I_{t_1 \neq \theta_1}\boldsymbol{G}_{\boldsymbol{\rho}}, \tag{3.12}$$

where

$$\boldsymbol{G}_{\boldsymbol{\rho}} = [0, G(1 + \rho^{+}_{\max})],\ \overline{\boldsymbol{G}} = [-G, 0],\ \textit{and}\ \boldsymbol{G} = [0, G]. \tag{3.13}$$

Figure 3.3 shows an example of drift compensated intervals based upon some initial interval $[T_0 \pm \boldsymbol{\alpha}]$, at (equidistant) local times $T_i = T_0 + i\Delta T$, $i \geq 1$, in case of $\rho^- > \rho^+$, a fast (but deaccelerating clock), and $u, G \ll \Delta T$. For accurateness, deterioration must ensure that the resulting interval $\boldsymbol{I}'$ intersects with the line $T = t$.

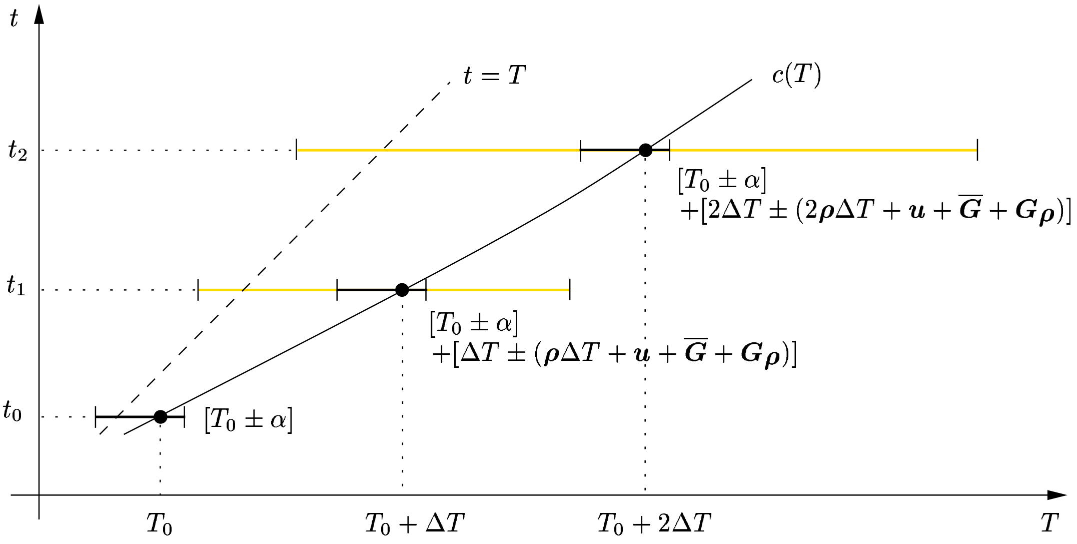

t
$t = T$
$c(T)$
t_2
$[T_0 \pm \alpha]$
$+[2\Delta T \pm (2\boldsymbol{\rho}\Delta T + \boldsymbol{u} + \overline{\boldsymbol{G}} + \boldsymbol{G}_{\boldsymbol{\rho}})]$
t_1
$[T_0 \pm \alpha]$
$+[\Delta T \pm (\boldsymbol{\rho}\Delta T + \boldsymbol{u} + \overline{\boldsymbol{G}} + \boldsymbol{G}_{\boldsymbol{\rho}})]$
t_0
$[T_0 \pm \alpha]$
T_0
$T_0 + \Delta T$
$T_0 + 2\Delta T$
T

Figure 3.3: *Drift Compensation*

A few remarks on drift compensation are appropriate here:

It is important to realize that the errors due to rate adjustment uncertainties (accounted for via $\boldsymbol{u}$) do not add up *in a single (uninterrupted)* drift compensation operation. Unfortunately, they can add up in subsequent drift compensations that are separated by some other operation, like network transmission, drift compensation at another node, or even computation of the convergence function. Therefore, it will turn out that $\boldsymbol{u}_{\max}$ spoils achievable worst case accuracy and precision even more than granularity does(!), see Theorem 3.1. However, we should note that it is very unlikely in practice to have executions where the worst case behavior is actually attained.

Almost any meaningful t_1 is in synchrony with the local clock, since activities of the clock synchronization algorithm are usually initiated when the local clock reaches some predefined value. Therefore, the term involving $\boldsymbol{G\rho}$ in (3.12) is encountered in our analysis only a few times, for example, when determining maximum precision, see Theorem 3.1.

Inequality (3.12) is valid for $T_1 \geq T_0$, although the negative accuracy of $\boldsymbol{I}'$ is not particularly meaningful (unnecessarily large) when $T_1 = T_0$ or $T_1 = T_0 + G$, recall the proof of Lemma 3.8. Hence, periods of drift compensation lasting at least $2G$ are preferable w.r.t. tightness of the bounds.

Equipped with those prerequisites, we can eventually introduce the clock model suitable for the interval based paradigm: Each node p has to provide a *local interval clock* $\boldsymbol{C}_p(t)$, which is implemented by an interval of accuracies $\boldsymbol{\alpha}_p$ relative to the nodes instantaneous local clock value $C_p(t)$. In the absence of adjustments (i.e., when running at its intrinsic rate), $\boldsymbol{C}_p(t)$ must be accurate despite of the fact that $C_p(t)$ may drift away from real-time. Hence, $\boldsymbol{\alpha}_p$ must be maintained according to (3.12):

Assumption 3.3 (Local Interval Clocks) *Each node p provides a* local interval clock

$$\boldsymbol{C}_p(t) = \begin{cases} [C_p(t) - \alpha_p^-(t), C_p(t) + \alpha_p^+(t)] & \text{if } t = \theta \text{ is in synchrony with } C_p(t), \\ [C_p(\theta_i) - \alpha_p^-(\theta_i), C_p(\theta_i) + \alpha_p^+(\theta_i)] & \text{for } \theta_i \leq t < \theta_{i+1} \text{ otherwise} \end{cases}$$

via a local interval of accuracies $\boldsymbol{\alpha}_p(t) = [-\alpha_p^-(t), \alpha_p^+(t)]$ *of granularity* G_S *taken relatively to the node's local clock* $C_p(t)$*, which is maintained by means of the following operations:*

- (Re-)Initializing $\boldsymbol{C}_p(t^R)$: *C_p along with $\boldsymbol{\alpha}_p$ can be set atomically to a new interval (all values, including reference point, being integer multiples of G_S) at any synchronous real-time t^R.*
- Reading $\boldsymbol{C}_p(t)$: *C_p along with $\boldsymbol{\alpha}_p$ can be read consistently at any real-time t.*
- Deteriorating $\boldsymbol{C}_p(\theta)$: *$\boldsymbol{\alpha}_p$ is enlarged by $G\boldsymbol{\rho}_p$ at each clock tick θ of C_p. Moreover, at the first tick after (re-)initializing $\boldsymbol{C}_p$ (if not already incorporated due to a reference*

point not being an integer multiple of G), an additional $\boldsymbol{u}_p$ is added to incorporate the rate adjustment uncertainty.

Note that it does not make much sense to maintain accuracies with a resolution below G_S, since the clock setting error spoils accuracy by the same amount it spoils the clock value. Choosing G_S as the resolution for accuracies is also advantageous due to the fact that all computations of the clock synchronization algorithm can be performed by using integer arithmetic, provided that all non-integer parameters compiled into the algorithm are integer multiples of G_S.

Of course, the clock synchronization algorithm is responsible for periodically re-initializing $\boldsymbol{C}_p(t^R)$ in an accurate way, so that $t \in \boldsymbol{C}_p(t)$ for all $t \geq t^R$ is guaranteed for a non-faulty node p by accurateness of deterioration, cf. (3.12). The correctness of this statement involves a subtle issue, though, if applications are allowed to read $\boldsymbol{C}_p(t)$ at arbitrary (non-synchronous) real-times t. More specifically, in implementations where any reading of the local clock is synchronized to clock ticks, as is the case when using our UTCSU-ASIC, it is of course sufficient to guarantee accurateness for synchronous real-times (so that only $\boldsymbol{u}_p$ must be incorporated). However, in settings that allow reading of the local clock at arbitrary real-times t, reading $\boldsymbol{\alpha}_p(t)$ must (explicitly or implicitly) incorporate $G\boldsymbol{\rho}$ as well, since $\boldsymbol{C}_p(t)$ has to be accurate for any $\theta \leq t < \theta + g$ here.

Deteriorating $\boldsymbol{C}_p$ can either be performed by adding $G\boldsymbol{\rho}_p$ at each clock tick or, in an accumulated fashion, by adding $\big(C(t_a) - C(t_{a-1})\big)\boldsymbol{\rho}_p$ at the a-th clock reading access at real-time t_a. For the adder-based clock in our UTCSU-ASIC we implemented an approximation of the former technique in hardware: Instead of adding $G\boldsymbol{\rho}_p$ at any clock tick, we add $G^0_{osc}\boldsymbol{\rho}_p$ at any oscillator tick. This approximation introduces an error of at most $\mathcal{O}(G\rho_p)$ in $\boldsymbol{\alpha}_p$, which vanishes in the already present remainder terms, see Theorem 3.1.

3.3.2 Network Communications

To model the communication subsystem, we first recall that our clock state synchronization algorithm operates in periodic rounds, taking place every P_S (logical time) seconds. At the end of each round, *clock synchronization messages* (CSM) are exchanged among the n nodes of the distributed system in a *full message exchange* (FME). More specifically, in a single FME, each node p transmits a CSM consisting of the accuracy interval $\boldsymbol{A}_p(t^A_{pq}) = [C_p(t^A_{pq}) - \alpha^-_p(t^A_{pq}), C_p(t^A_{pq}) + \alpha^+_p(t^A_{pq})]$ to node q at some real-time t^A_{pq}. Applying a suitable convergence function to the set of received and preprocessed accuracy intervals, each node eventually computes a local clock correction value that enforces precision and accuracy.

The required communication primitive for this setting is a basic (unreliable) broadcast operation, something that is easily implemented by means of n send operations in a fully connected point-to-point network, or even provided in hardware by modern broadcast-type networks. The most important requirements are upper and lower bounded transmission delays, i.e., synchronous behavior.

Assumption 3.4 (Transmission Characteristics) *We assume a synchronous communication network exhibiting the following properties:*

(1) *If a non-faulty node p of the distributed system initiates its broadcast at some arbitrary time t_p^I, there is a uniform bound $\lambda_{\max} \geq 0$ on the possible delay up to time t_p^S when it actually starts the broadcast transmission; $\lambda_{\max}$ is called the* maximum broadcast latency.

(2) *If the broadcast of a non-faulty node p starts at time t_p^S, there is a bound $\omega_p \geq t_{pl}^A - t_p^S \geq 0$ on the delay up to time t_{pl}^A when the transmission to the last node l required for broadcasting is activated. Let $\omega_{\max} \geq \omega_p$ be a suitable uniform bound called the* maximum broadcast operation delay. *Moreover, the "indicator function" of making use of a pure broadcast network is*

$$H = \begin{cases} 1 & \textit{if } \omega_{\max} = 0 \textit{ (pure broadcast network)}, \\ 2 & \textit{otherwise.} \end{cases} \tag{3.14}$$

(3) *If some node p activates its transmission to some node $q \neq p$ at time t_{pq}^A and no transmission fault occurs, node q receives the message at time t_q^p, with the* transmission delay $\delta'_{pq} = t_q^p - t_{pq}^A$ *satisfying*

$$\delta'_{pq} \in [\delta_{pq} - \varepsilon_{pq}^-, \delta_{pq} + \varepsilon_{pq}^+], \tag{3.15}$$

where δ_{pq} represents the deterministic part and $\boldsymbol{\varepsilon}_{pq} = [-\varepsilon_{pq}^-, \varepsilon_{pq}^+]$ the maximum uncertainty of δ'_{pq} (of course, $\delta_{pq} \geq \varepsilon_{pq}^-$). Let $\boldsymbol{\varepsilon}_{\max} = [-\varepsilon_{\max}^-, \varepsilon_{\max}^+] \supseteq \bigcup_{p,q \neq p} \boldsymbol{\varepsilon}_{pq}$ with $\varepsilon_{\max} = \varepsilon_{\max}^- + \varepsilon_{\max}^+$ and $\delta_{\max} \geq \delta_{pq}$, $\delta_{\min} \leq \delta_{pq}$ be suitable uniform bounds for all (non-faulty) pairs of nodes p, $q \neq p$, with the additional technical condition

$$\delta_{\min} \boldsymbol{\rho}_{\max} \subseteq \boldsymbol{\varepsilon}_{\max}. \tag{3.16}$$

Note that this condition expresses the quite reasonable assumption that time-keeping during transmission by any non-faulty clock is more accurate than exploiting the synchronous network behavior.

(4) The accuracy interval in the CSM is transmitted with limited resolution. More specifically, we assume that the local clock value $C_p(t_{pq}^A)$ is transmitted in a way that preserves its granularity G. Both lower and upper accuracy $\alpha_p^-(t_{pq}^A)$ and $\alpha_p^+(t_{pq}^A)$ are transmitted with finite resolution $R_A = LG_S$ for some integer $L \geq 1$; let

$$G_A = R_A - G_S \tag{3.17}$$

be the loss in resolution.

We assume that a CSM is timestamped with $\boldsymbol{C}_p(t_{pq}^A)$ at the moment of actual transmission t_{pq}^A, not at the moment of initiation t_p^I of the broadcast or at the moment t_p^S of actually starting it. This ensures that the relatively large maximum broadcast latency $\lambda_{\max}$ and/or the maximum broadcast operation delay $\omega_{\max}$ does not impair δ'_{pq} and hence achievable precision and accuracy. Therefore, we can cope with both the extended capabilities provided by our UTCSU-ASIC and with traditional settings ($\omega_{\max} = \lambda_{\max} = 0$ and including any uncertainty in ε_{pq}).

Our model can be adopted to a wide variety of different networks: $\lambda_{\max} > 0$ and $\omega_{\max} = 0$ models broadcast-type networks, whereas $\lambda_{\max} = 0$ and $\omega_{\max} > 0$ is appropriate for point-to-point networks without hardware broadcast capabilities. Note that we can even deal with approaches that stagger CSM transmissions in time to avoid peak network load, simply by making $\omega_{\max}$ large enough to cover the whole period of transmissions.

In the interval-based paradigm, a *delay compensation operation* is responsible for coping with transmission delays, see [25]. Basically, delay compensation maintains accurateness of intervals that are transmitted over a network experiencing variable transmission delays according to Assumption 3.4. If an accurate interval $\boldsymbol{I} = [T \pm \boldsymbol{\alpha}]$ is sent from node p to $q \neq p$, experiencing some transmission delay $\delta' \in [\delta \pm \boldsymbol{\varepsilon}]$, an accurate interval $\boldsymbol{I}''$ (which covers the unobservable $\boldsymbol{I}'$ representing the real-time of reception at the sender-node p) is constructed at the receiving node q by shifting the original interval $\boldsymbol{I}$ by δ, and blowing up the shifted interval's reference point to an interval $\boldsymbol{\varepsilon}$ in order to compensate for uncertainties in the transmission delay. In addition, we have to account for the effects of finite transmission resolution R_A of accuracies. Since accuracies α^-, α^+ of the sending node p are always multiples of G_S according to Assumption 3.3, truncation to $R_A = LG_S$ for some integer $L \geq 1$ introduces an error of at most $G_A = R_A - G_S$.

Definition 3.6 (Delay Compensation) *The result of delay compensation of an accurate interval $\boldsymbol{I}$ transmitted from node p to node $q \neq p$ in the absence of faults is the accurate interval*

$$\boldsymbol{I}'' = \boldsymbol{I} + \boldsymbol{2G}_A + [\delta_{pq} \pm \boldsymbol{\varepsilon}_{pq}], \tag{3.18}$$

where the loss in accuracy resolution during transmission is covered by $\mathbf{2}\boldsymbol{G}_A = [-G_A, G_A]$.

Note that we did not consider the effect of non-zero clock granularity G at the receiving node q here, since a drift compensation operation takes place at q later on. Bear in mind, however, that the real-time of reception t' at q is usually not in synchrony with q's clock.

Figure 3.4 should make delay compensation straightforward. We assume that the experienced transmission delay is $\delta'_{pq} > \delta_{pq}$ and $G_A \ll \varepsilon_{pq}$. The middle time axis corresponds to real-time, whereas the upper and lower ones display logical time at p and q, respectively. The sender node p's rate r_p is presumed to be 1, i.e., its local clock progresses as real-time does. Interpreting Figure 3.4, one should consider the intervals $\boldsymbol{I}$ and $\boldsymbol{I}''$ as "fixed", whereas the reception time t' and hence the interval $\boldsymbol{I}'$ may vary with δ'_{pq}.

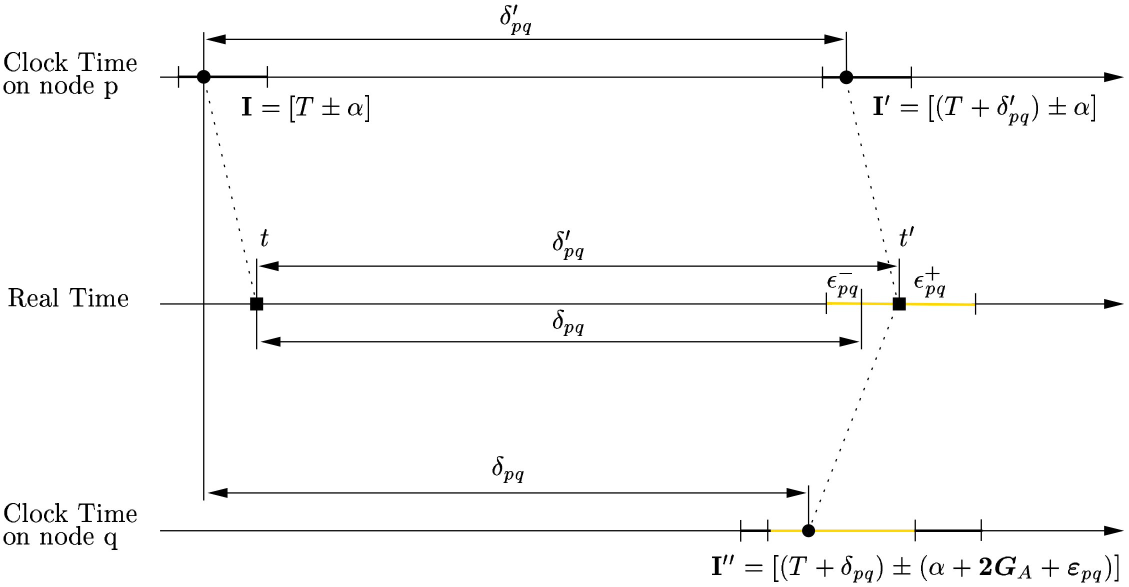

δ'_{pq}
Clock Time on node p
$\mathbf{I} = [T \pm \alpha]$
$\mathbf{I}' = [(T + \delta'_{pq}) \pm \alpha]$
t
δ'_{pq}
t'
ϵ^-_{pq}
ϵ^+_{pq}
Real Time
δ_{pq}
δ_{pq}
Clock Time on node q
$\mathbf{I}'' = [(T + \delta_{pq}) \pm (\alpha + \mathbf{2}\boldsymbol{G}_A + \varepsilon_{pq})]$

Figure 3.4: *Delay Compensation*

3.3.3 Fault Model

All assumptions in the previous subsections are meaningful for the fault-free case only. Dealing with fault-tolerant systems, a pertinent fault model is required. However, since we are considering a generic algorithm and its analysis, it does not make much sense to stipulate a particular fault model here — not even an advanced hybrid one as in [70] or [2]. After all, it is the convergence function that is primarily concerned about faults. Consequently, we will assume that an *abstract fault model* $\mathcal{F}$ is provided along with a particular convergence function. It is abstract in the sense that it gives information on faults not in terms of faulty system components, but rather by classifying the intervals $\boldsymbol{I}_q^p$ provided to the convergence function at node q as a result of reception and preprocessing of the broadcast(s) of node p, see (3.19)/(3.20).

Eventually, any $\mathcal{F}$ must incorporate a (convincing) way of tracing back abstract faults to component faults in order to be meaningful in practice. The following issues are to be considered here:

Our generic algorithm imposes only a few limitations on the severity of faults. More specifically, faulty nodes or network components may in principle perform arbitrarily, including transmission/reception of any number of arbitrary messages without, however, being capable of causing (serious) "global" disturbance of system operation, e.g. by

1. *impersonating* other nodes,
2. *flooding/jamming* the network or non-faulty receiving nodes (violating $\lambda_{\max}$ and/or $\omega_{\max}$, or causing excessive transmission delays of other broadcasts).

Note that this is easily guaranteed in a fully-connected point-to-point network, but is difficult to ensure for a (non-redundant) broadcast channel.

Viewed from a single (non-faulty) receiving node q's perspective, an interval $\boldsymbol{I}_q^p$ resulting from reception and preprocessing of node p's broadcast(s) during an FME can be faulty in various ways:

1. *Omission faults*, caused by an omissive faulty node p or transient errors during message reception, resulting in $\boldsymbol{I}_q^p = \emptyset$.
2. *Timing faults*, due to a faulty node/clock p or excessive transmission delays, resulting in a non-accurate and/or non-$\boldsymbol{\pi}$-accurate $\boldsymbol{I}_q^p$.
3. *Clock (value) faults*, caused by a faulty node p or a damaged message, resulting in a non-accurate and/or non-$\boldsymbol{\pi}$-accurate $\boldsymbol{I}_q^p$.

Apart from those faults, which arise in traditional clock synchronization as well, we face additional *accuracy faults* that are unique in the interval-based paradigm. Adopting the terminology of [36], we distinguish three different types:

1. *Truncated accuracy faults*, caused by accuracies being too small, resulting in a non-accurate $\boldsymbol{I}_q^p$.
2. *Bounded accuracy faults*, due to accuracies that are too large but bounded (usually in a way that makes them indistinguishable from accuracies provided in a non-faulty broadcast), resulting in an accurate but not particularly meaningful $\boldsymbol{I}_q^p$.

3. *Unbounded accuracy faults*, resulting in an accurate but meaningless $\boldsymbol{I}_q^p$.

Obviously, accuracy faults do not affect $\boldsymbol{\pi}$-precision intervals, since $\boldsymbol{\pi}$ is not transmitted but rather compiled into the algorithm. However, one has to account for the possibility that $\hat{\boldsymbol{I}}_q^p$ is $\boldsymbol{\pi}$-accurate despite of the fact that $\boldsymbol{I}_q^p$ suffers from a truncated accuracy fault, and also the opposite situation where $\hat{\boldsymbol{I}}_q^p$ is not $\boldsymbol{\pi}$-accurate although it is accurate because of a bounded/unbounded accuracy fault.

Viewed from the perspective of corresponding intervals $\boldsymbol{I}_p^s$, $\boldsymbol{I}_q^s$ at two (non-faulty) nodes p and q (for the same sending node s) in a single FME, we encounter the following faults:

1. *Arbitrary faults* covering (almost, see first item) any kind of faulty behavior, including a Byzantine (two-faced, "asymmetric", cf. [2]) one. Arbitrary faults may be caused by nodes sending different messages to different receivers or by excessive transmission delays at the receiving ends. Note that both faults can also occur in broadcast-type networks, since the elementary broadcast operation is not assumed to be reliable, see [50]. Of course, some receiving nodes may experience an omission or deliver a non-faulty interval in an arbitrarily faulty broadcast as well.
2. Depending on the convergence function, there are usually one or more classes of faults that may be considered as *restricted faults*, in the sense that they can be tolerated "easier" than arbitrary ones. For example, tolerating f consistently perceived timing faults ("symmetric" faults as in [2]), usually requires $n \geq 2f+1$ nodes (instead of $n \geq 3f+1$ in case of arbitrary faults). Again, some receiving nodes may experience an omission fault instead of providing a faulty interval; delivery of a non-faulty interval, however, usually turns the fault into an arbitrary one.
3. *Omission faults* are usually perceived differently at different receiving nodes p and q. Traditionally, they are attributed to sending nodes ("strictly omissive asymmetric faults", cf. [2]), although most receive omissions occur (independently!) at the receiving nodes. Hence, viewed globally, they cannot be traced back to (a reasonably small number of) omissive sending nodes.
4. *Crash faults* (and other "benign" faults according to the terminology of [2]) are consistently detectable at all nodes. However, a node that crashes during a broadcast operation produces (at least) restricted faults due to inconsistent reception.

Note finally that the four types of faults above, which are well-known from traditional clock synchronization, are only meaningful for $\boldsymbol{\pi}$-precision intervals, not for accuracy intervals. For a more formal explanation consult Chapter 5.

We close this section with Table 3.1 that gives a flavor of the order of magnitude of the various parameters introduced so far. They are valid for the class of distributed systems portrayed in Chapter 2.

parameter	*magnitude*	*description*
n	100	number of nodes
η_p	10 ms	overall execution time
G	100 ns	clock granularity
G_S	10^{-15} s	clock setting granularity
ρ_p^+, ρ_p^-	10^{-6}	inverse rate deviation bound
u_p^+, u_p^-	100 ns	rate adjustment uncertainty
$\lambda_{\max}$	1 s	maximum broadcast latency
$\omega_{\max}$	0 s	maximum broadcast operation delay
δ_{pq}	10 μs	deterministic transmission delay
ε_{pq}^-, ε_{pq}^+	< 1 μs	transmission delay uncertainty
G_A	100 ns	accuracy transmission loss

Table 3.1: *System Parameters with typical Values*

3.4 Generic Algorithm

This section contains the description of our generic clock state synchronization algorithm. It employs the common round-based structure from other internal synchronization algorithms. Periodically, every P_S (logical time) seconds, the algorithm executes the following steps:

1. Initiation of a *full message exchange* (FME) to provide each node with the accuracy intervals of all other nodes (involving delay compensation operations, recall Definition 3.6).

2. Preprocessing of the set of received accuracy intervals to make them all compatible.

3. Application of a suitable interval-valued convergence function to the set of preprocessed intervals to compute and subsequently apply a correction value for the local interval clock.

4. Keeping track of real-time by means of the local interval clocks of the nodes (involving drift compensation operations, recall Definition 3.5) up to the next round.

The abovementioned basic operations, i.e., delay compensation and drift compensation, are required to make the exchanged intervals compatible with each other while maintaining $\boldsymbol{\pi}$-correctness. More specifically, all intervals gathered at node q in an FME are preprocessed to represent a common point in time t_q^R as follows: For an accuracy interval $\boldsymbol{A}_p$ sent by node $p \neq q$, delay compensation (3.18) is applied to provide the receiver q with an initial interval $\boldsymbol{A}_p''$ that estimates $\boldsymbol{A}_p$ at the (non-synchronous) real-time of reception t_q^p, when $C_q(t_q^p) = T_q^p$. That interval $\boldsymbol{A}_p''$ is then dragged locally by means of the receiver's clock, utilizing drift compensation (3.12), to some common, synchronous point in real-time t_q^R defined by $C_q(t_q^R) = T_q^R$. Therefore, we arrive at the compatible intervals

$$\begin{aligned} \boldsymbol{I}_q^p(t_q^R) &= \boldsymbol{A}_p + [T_q^R - T_q^p + \delta_{pq} \pm (\boldsymbol{2G}_A + \boldsymbol{\varepsilon}_{pq})] \\ &\quad + (T_q^R - T_q^p)\boldsymbol{\rho}_q + \boldsymbol{u}_q + \overline{\boldsymbol{G}} \end{aligned} \tag{3.19}$$

for $p \neq q$. Provided that T_q^R is chosen large enough to ensure that the intervals of all non-faulty nodes can be received and processed, it is immediately apparent that $\boldsymbol{I}_q^p$ is accurate —and, as we will justify in Section 3.5, $\boldsymbol{\pi}$-accurate for some suitable $\boldsymbol{\pi}$ as well— if (1) $\boldsymbol{A}_p$ was accurate, (2) transmission delay was not excessive, and (3) the receiver q is not faulty; recall our discussion of the abstract fault model $\mathcal{F}$ in Section 3.3.3.

In addition to the intervals obtained from remote nodes $p \neq q$, there is also the accuracy interval of the own node q that needs to be considered. Of course, no actual transmission is required here, so we just have $\boldsymbol{I}_q^q = \boldsymbol{C}_q(t_q^R) = [T_q^R \pm \boldsymbol{\alpha}_q^R]$. Observe carefully that it is possible to compute $\boldsymbol{C}_q(t_q^R)$ in advance by exploiting knowledge of some $\boldsymbol{C}_q(\theta) = [T \pm \boldsymbol{\alpha}]$ with $T < T_q^R$ from the same round and without continuous amortization being active: Since exactly $T_q^R - T$ clock ticks must occur between t and t_q^R, we obviously have $\boldsymbol{C}_q(t_q^R) = \boldsymbol{C}_q(t) + T_q^R - T + (T_q^R - T)\boldsymbol{\rho}_q$ due to intrinsic deterioration; recall Assumption 3.3. Therefore, we can imagine a zero-delay "loop-back transmission" of the accuracy interval $\boldsymbol{A}_q = \boldsymbol{C}_q(t_{qq}^A) = [T_{qq}^A \pm \boldsymbol{\alpha}_{qq}^A]$ at the FME initiation, which "arrives" instantaneously at node q, thus $T_q^q = T_{qq}^A$. Incorporating this we finally arrive at

$$\boldsymbol{I}_q^q(t_q^R) = \boldsymbol{A}_q + T_q^R - T_q^q + (T_q^R - T_q^q)\boldsymbol{\rho}_q. \tag{3.20}$$

Now, given the set of node q's intervals $\boldsymbol{I}_q^p$ above, a function that provides a (small) interval that both contains real-time t_q^R and enhances precision —despite of some possibly faulty ones— is required. Adhering to the terminology introduced in [58], we call such a function a convergence function. The clock synchronization algorithm defined below is stated on top of a generic convergence function $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$, which can be any interval-valued function that satisfies certain properties stated in Definition 3.11.

Definition 3.7 (Clock State Algorithm) *The generic algorithm is defined in terms of its parameters, the initial and periodic synchronization actions.*

Parameters: *All parameters are integer multiples of G_S (cf. Assumption 3.3) unless otherwise specified:*

- *total number n of nodes,*
- *node p's intrinsic inverse rate deviation bound $\boldsymbol{\rho}_p$ and uniform bound $\boldsymbol{\rho}_{\max} \supseteq \bigcup_p \boldsymbol{\rho}_p$ with $\rho_{\max} = ||\boldsymbol{\rho}_{\max}|| = \rho^-_{\max} + \rho^+_{\max}$ (defined in Assumption 3.2),*
- *clock granularity G, clock setting granularity G_S, node p's maximum rate adjustment uncertainty $\boldsymbol{u}_p = [-u^-_p, u^+_p]$, and uniform maximum rate adjustment uncertainty $\boldsymbol{u}_{\max} \supseteq \bigcup_p \boldsymbol{u}_p$ with $u_{\max} = u^-_{\max} + u^+_{\max}$ (defined in Assumption 3.2),*
- *transmission delay characteristics δ_{sp}, $\boldsymbol{\varepsilon}_{sp}$ for all nodes s, uniform bounds $0 \leq \delta_{\min} \leq \min_{p,q}\{\delta_{pq}\}$, $\delta_{\max} \geq \max_{p,q}\{\delta_{pq}\}$ and $\boldsymbol{\varepsilon}_{\max} \supseteq \bigcup_{p,q} \boldsymbol{\varepsilon}_{pq}$ with $\varepsilon_{\max} = ||\boldsymbol{\varepsilon}_{\max}|| = \varepsilon^-_{\max} + \varepsilon^+_{\max}$, and accuracy transmission loss G_A (defined in Assumption 3.4),*
- *computation delay compensation (integer multiple of G) guaranteeing node p's maximum computation time η_p (defined in Assumption 3.1), chosen according to*

  $$E_p \geq \frac{\eta_p + u^-_p}{1 - \rho^-_p},$$

  and uniform bounds $E_{\max} \geq \max_p\{E_p\}$ and $0 \leq E_{\min} \leq \min_p\{E_p\}$; usually $E_p = E_{\max} = E_{\min}$ is chosen to be the same at all nodes p,
- *broadcast delay compensation $\Lambda + \Omega$ (integer multiple of G), chosen to satisfy*

  $$\Lambda + \Omega \geq \frac{\lambda_{\max} + \omega_{\max} + u^-_{\max}}{1 - \rho^-_{\max}},$$

  in conjunction with Δ below ensures that resynchronization starts only after all CSMs broadcast by non-faulty nodes during an FME have arrived (see Assumption 3.4),
- *transmission delay compensation Δ (integer multiple of G) chosen according to*

  $$\Delta \geq \frac{\pi_0 + u_{\max} + \delta_{\max} + (P_S - E_{\min} + \pi^-)\rho_{\max} + \varepsilon^+_{\max}}{1 + \rho^+_{\max}} \tag{3.21}$$

  (defined in Lemma 3.11), where $\pi_0 = ||\boldsymbol{\pi}_0||$ and π^- (defined in Theorem 3.1) depend on the convergence function employed,
- *round period $P_S \geq \Lambda + \Omega + \Delta + E_{\max}$ (integer multiple of G),*

Initial synchronization: *At each node q, the local interval clock* $\boldsymbol{C}_q$ *must be initialized to the accuracy interval* $\boldsymbol{A}_q^0 = [T_q^0 - \alpha_q^{0-}, T_q^0 + \alpha_q^{0+}]$ *at some synchronous real-time* t_q^0 *by some external means. This initialization must ensure*

- $t_q^0 \in \boldsymbol{A}_q^0$,
- $T_q^0 \in [\Lambda + \Omega + \Delta + E_q \pm \boldsymbol{\pi}]$,
- $\boldsymbol{\alpha}_q^0 \subseteq \boldsymbol{\pi}_0$,

where $\boldsymbol{\pi}$ *and* $\boldsymbol{\pi}_0$ *depend on the convergence function employed, see Theorem 3.1.*

Periodic Synchronization: *Near the end of each round* $k \geq 0$*, every node q in the system performs the following operations:*

(S) *CSM Send: Periodically at times* $C_q(t_q^I) = T^I = (k+1)P_S$*, node q initiates a broadcast. The message* M_{qp} *sent to node p at some real-time* t_{qp}^A *during that broadcast operation contains the accuracy interval* $\boldsymbol{A}_{qp} = [T_{qp}^A \pm \boldsymbol{\alpha}_{qp}^A] = \boldsymbol{C}_q(t_{qp}^A)$*. For the zero-delay "loop-back transmission" to the own node q,* $t_{qq}^A = t_q^I$ *so that* $T_{qq}^A = T^I = (k+1)P_S$*.*

(R) *CSM Reception: When a message* M_{pq} *from node p arrives at node q at real-time* t_q^p*, when* $C_q(t_q^p) = T_q^p$*, the interval* $\boldsymbol{I}_q^p$ *given in (3.19)/(3.20) is computed and stored in an ordered set* $\boldsymbol{\mathcal{I}}_q$*. For the definition of the resynchronization time* T_q^R *turn to item (T).*

(C) *Computation: At* $C_q(t_q^{\Lambda+\Omega+\Delta}) = T^{\Lambda+\Omega+\Delta} = (k+1)P_S + \Lambda + \Omega + \Delta$ *in round k, a convergence function* $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ *(see Definition 3.11) is applied to the compatible intervals stored in* $\boldsymbol{\mathcal{I}}_q$*, yielding the result* $\boldsymbol{S}_q$*. Finally,* $\boldsymbol{\mathcal{I}}_q$ *is re-initialized to the empty set for the next round* $k+1$*.*

(T) *Termination and Resynchronization: At* $C_q(t_q^R) = T_q^R = (k+1)P_S + \Lambda + \Omega + \Delta + E_q$ *in round k, q's interval clock* $\boldsymbol{C}_q$ *is set to* $\boldsymbol{S}_q$ *(instantaneously or by continuous amortization).*

A few remarks are certainly appropriate here:

A number of parameters defined in our system model (Assumptions 3.1–3.4) must be provided (statically or dynamically) to the instance of the algorithm running at a node p, e.g., rate deviation bound $\boldsymbol{\rho}_p$, transmission delay parameters δ_{pq}, $\boldsymbol{\varepsilon}_{pq}$, and quantities Δ, Λ, Ω, E_p related to λ, ω, and η_p, respectively. The particular convergence function might also require some parameters. Therefore, our algorithm depends on those parameters not implicitly as traditional synchronization algorithms do, but rather explicitly.

The needed initial synchronization is automatically provided when the algorithm is used in our clock validation framework, see Section 3.1. Clock validation assumes that there are some *primary nodes* having their physical clock disciplined by an UTC time source of high accuracy, which may, however, fail arbitrarily. In normal operation, a primary node p's local clock C_p provides UTC with some *a priori* accuracy α^0, such that $t \in [C_p(t) - \alpha^0, C_q(t) + \alpha^0]$, for all real-times $t \geq 0$. Temporarily, we may assume that local clocks are initially synchronized when flywheeling begins. Furthermore we assume w.l.o.g. that real-time and logical time start at $t = 0$ and $T = 0$, respectively, at the beginning of round 0.

Steps (S), (C), and (T) of the algorithm are triggered when the local clock reaches some specific point in (logical) time, so that they are effectively sequenced. Step (R), however, takes place asynchronously when a CSM drops in. Note also that the execution time required for computing the convergence function is usually smaller when the latter is performed piecewise in steps (R). Of course, if η_p is chosen appropriately, our results apply to this situation as well.

Resetting the local interval clock instantaneously in step (T) of the algorithm would cause non-monotonicity and non-continuity of local time. This is circumvented in practice by means of the continuous amortization algorithm of [56]: To perform state correction of the local clock, its rate is modified by a fixed amount $\pm\psi$ for a programmable period until the clock has changed its value as desired. This algorithm, which is supported in hardware by our UTCSU-ASIC, is particularly attractive as it does not impair the worst-case precision and accuracy obtained for instantaneous correction if ψ is chosen suitably, see Theorem 3.2 for details.

In the remaining sections, we will analyze worst case accuracy and precision of the algorithm given in Definition 3.7, stating expressions in terms of characteristic parameters of the convergence function. The major results obtained herein will be summarized in Theorem 3.1 and 3.2.

3.5 Analysis

From the description of the algorithm in the preceding section, it should be reasonably obvious that accurateness of the local interval clocks $\boldsymbol{C}_p(t)$ of all non-faulty nodes is maintained during all rounds. In fact, all operations involved (drift compensation, delay compensation, and application of the convergence function) are explicitly designed to preserve accurateness. This is not that obvious for precision, however, so we find it appropriate to give an informal overview how precision is actually maintained.

First of all, recall that any local interval clock $\boldsymbol{C}_p(t)$ is defined by three values, namely accuracies $\alpha_p^-(t)$, $\alpha_p^+(t)$ and local clock value $C_p(t)$ as reference point. Dealing solely with accuracy, the lower and the upper edge would be sufficient. Hence, the reference point can be set appropriately to achieve the orthogonal goal of maintaining the precision condition $|C_p(t) - C_q(t)| \leq \pi$ for all non-faulty nodes p and q. From Lemma 3.3 we know that this is achieved when the interval clocks of all non-faulty nodes are kept $\boldsymbol{\pi}$-precise.

The actual approach taken is particularly attractive due to the fact that precision during a round is automatically maintained when accuracy is. To understand how this works, assume that all members of the set $\boldsymbol{\mathcal{C}}(t)$ of non-faulty interval clocks are $\boldsymbol{\pi}_0$-correct at some real-time t^0, i.e., that their associated $\boldsymbol{\pi}_0$-precision intervals contain $\tau(t^0)$, and recall that internal global time τ progresses as real-time does. When trying to capture real-time $t' > t^0$ by $\boldsymbol{C}_p(t')$, we must deteriorate (enlarge) the accuracy interval in order to compensate for the drift of the local clock. However, if this is done properly to capture real-time t', it is clear that the associated $\boldsymbol{\pi}$-precision interval $\hat{\boldsymbol{C}}_p(t')$ captures internal global time $\tau(t') > \tau(t^0)$ as well, provided that $\boldsymbol{\pi}$ is the result of enlarging $\boldsymbol{\pi}_0$ by the maximum amount any $\boldsymbol{C}_j$ has been enlarged. It is important to understand, though, that enlargement of precision intervals is just a matter of analysis, since the algorithm does not deal with precision intervals at all. Anyway, it follows that $\boldsymbol{\mathcal{C}}(t')$ must be $\boldsymbol{\pi}$-correct.

Whereas enlarging $\boldsymbol{\pi}_0$ to $\boldsymbol{\pi}$ guarantees that $\tau(t)$ lies in the intersection of the $\boldsymbol{\pi}$-precision intervals of all non-faulty nodes, this cannot ensure a bounded precision for $t \to \infty$. Periodic resynchronizations are required for that purpose, giving rise to our round-based algorithm. More specifically, at the end of the k-th round, the nodes' current $\boldsymbol{\pi}$-correct local interval clocks are re-initialized to newly computed accuracy intervals that are $\boldsymbol{\pi}_0$-precise for $\boldsymbol{\pi}_0 \subset \boldsymbol{\pi}$ (= precision enhancement). Note that we cannot safely assume $\boldsymbol{\pi}_0$-correctness here, since it may be the case that internal global time $\tau^{(k)}$ for round k does not lie in the intersection of the new $\boldsymbol{\pi}_0$-precision intervals! However, if we define a new internal global time $\tau^{(k+1)}$ for round $k+1$, independently of its predecessor $\tau^{(k)}$, we have of course $\boldsymbol{\pi}_0$-correctness w.r.t. $\tau^{(k+1)}$. Consequently, the resynchronization launches the next round $k+1$ during which initial precision $\boldsymbol{\pi}_0$ again deteriorates to $\boldsymbol{\pi}$.

Keep in mind that only $\boldsymbol{\pi}$-precision intervals $\hat{\boldsymbol{C}}_p$ are affected by precision enhancement. The local interval clocks $\boldsymbol{C}_p$ itself must continuously track real-time t, so that the accuracies could grow. Actually, accuracy in round k can be viewed as an accumulation of the $\boldsymbol{\pi}$-precision intervals during round $0, \ldots, k$. This eventually explains why t and τ will usually be apart, as mentioned in Section 3.2.2.

3.5.1 Internal Global Time

When trying to formalize the concept of internal global time, a number of technical difficulties arise. First of all, we have to establish a notion that allows us to deal with multivalued local time. Near a resynchronization event at node p, occurring at real-time t_p^R, one could be interested in local time $T_p^R = C_p(t_p^R)$ taken from the clock before resynchronization, and/or in $T_p^{R'} = C_p'(t_p^R)$ read from the already resynchronized clock. To express this situation unambiguously, we employ the well-thumbed technical device of *virtual clocks*: When round k at node p commences at real-time $t_p^{R,(k-1)}$, a virtual clock $\boldsymbol{C}_p^{(k)}$ is instantiated that progresses according to the physical clock C_p up to time $t_p^{R,(k)}$, when the $(k+1)$-th resynchronization event takes place. At this instant, a new virtual clock $\boldsymbol{C}_p^{(k+1)}$ (initialized with a value based on the convergence function applied to the intervals taken from the FME) is instantiated, which then proceeds concurrently with $\boldsymbol{C}_p^{(k)}$ in the same way. Needless to say, it is the set of virtual clocks of round k that defines the instance $\tau^{(k)} = \tau^{(k)}(t)$ of internal global time.

The probably most awkward problem when trying to define internal global time, however, arises from the fact that the contributing intervals reside at different nodes. After all, resynchronizations at different nodes do not occur simultaneously. Albeit two nodes are within the same round k most of the time, there are short periods where one has already resynchronized (and thus started its round $k+1$) while the other one has not. Nevertheless, accuracy intervals from different nodes must be made compatible to form a $\boldsymbol{\pi}$-precise set. For practical purposes, this requires dragging by means of the local clock and utilizing drift compensation. given by Definition 3.5). For the purpose of analysis, however, there is no need to make intervals residing at different nodes compatible in a "practicable" way. Rather than using dragging and drift compensation, it is sufficient to employ a simple, ideal shift-operation:

Definition 3.8 (Shifting) *The result of* shifting *an interval* $\boldsymbol{I} = \boldsymbol{I}(t_1)$ *to some point in time* $t' \geq t_1$ *is the interval* $\boldsymbol{J} = \boldsymbol{J}(t') = \mathrm{shift}_{t'}(\boldsymbol{I}) = \boldsymbol{I} + t' - t_1$. *For a set of* $n \geq 1$ *intervals* $\boldsymbol{\mathcal{I}} = \{\boldsymbol{I}_1(t_1), \ldots, \boldsymbol{I}_n(t_n)\}$, *the shifted set* $\boldsymbol{\mathcal{J}}$ *of compatible intervals all representing some arbitrary* $t' \geq \max_{1 \leq i \leq n}\{t_i\}$ *is defined by* $\boldsymbol{\mathcal{J}} = \mathrm{shift}_{t'}(\boldsymbol{\mathcal{I}}) = \{\boldsymbol{I}_1(t_1) + t' - t_1, \ldots, \boldsymbol{I}_n(t_n) + t' - t_n\}$.

Of course, any interval in $\boldsymbol{\mathcal{J}}$ above represents t'. However, keep in mind that they are artificial constructions, i.e., that they could only be provided by dragging $\boldsymbol{I}$ with an ideal, continuous real-time clock. Note that restricting t' to a point in time greater than any t_j of the intervals in $\boldsymbol{\mathcal{I}}$ is not really necessary.

Lemma 3.9 (Precision Shifted Intervals) *Let* $\boldsymbol{\mathcal{I}} = \{\boldsymbol{I}_1(t_1), \ldots, \boldsymbol{I}_n(t_n)\}$ *be a set of intervals with* $\boldsymbol{I}_i(t_i) = [T_i \pm \boldsymbol{\alpha}_i]$ *representing real-time* t_i *for* $1 \leq i \leq n$, *and* $\boldsymbol{\mathcal{J}} = \{\boldsymbol{J}_1(t'), \ldots, \boldsymbol{J}_n(t')\} = \mathrm{shift}_{t'}(\boldsymbol{\mathcal{I}}) = \{\boldsymbol{I}_1(t_1) + t' - t_1, \ldots, \boldsymbol{I}_n(t_n) + t' - t_n\}$ *for some arbitrary* $t' \geq \max_{1 \leq i \leq n}\{t_i\}$.

(1) If, for any i, *the interval* $\boldsymbol{I}_i$ *in* $\boldsymbol{\mathcal{I}}$ *is* $\boldsymbol{\pi}_i$*-correct w.r.t. internal global time of the same round* k, *then* $\boldsymbol{J}_i = \boldsymbol{I}_i + t' - t_i$ *is also* $\boldsymbol{\pi}_i$*-correct, and the whole shifted set* $\boldsymbol{\mathcal{J}}$ *is* $\boldsymbol{\pi}'$*-correct for* $\boldsymbol{\pi}' = \bigcup_i \boldsymbol{\pi}_i$,

(2) If $\boldsymbol{\mathcal{J}}$ *is* $\boldsymbol{\pi}$*-correct w.r.t. the internal global time of round* k, *then* $\tau_i = \tau^{(k)}(t_i) \in \hat{\boldsymbol{I}}_i(\tau_i)$ *for any* i.

Proof. To prove the first statement of the lemma, we note that $\boldsymbol{\pi}_i$-correctness of $\boldsymbol{I}_i$ implies $\boldsymbol{\pi}_i$-correctness of $\boldsymbol{J}_i = \boldsymbol{I}_i + t' - t_i$, since $\hat{\boldsymbol{J}}_i = \hat{\boldsymbol{I}}_i + \tau' - \tau_i$ due to the fact that internal global time (of the same round) progresses as real-time does, i.e., $\tau' - \tau_i = t' - t_i$. The asserted $\bigcup_i \boldsymbol{\pi}_i$-correctness follows from $n-1$ applications of Lemma 3.5 to the union of the singletons $\{\boldsymbol{I}_j + t' - t_j\}$ forming $\boldsymbol{\mathcal{J}}$.

To prove the second statement of the lemma, we use the same argument as before to derive $\hat{\boldsymbol{J}}_i(\tau_i) = \hat{\boldsymbol{I}}_i(\tau_i) + \tau' - \tau_i$ for $\tau' = \tau(t')$, so that the set of $\boldsymbol{\pi}$-precision intervals associated with $\boldsymbol{\mathcal{J}}$ reads $\hat{\boldsymbol{\mathcal{J}}} = \{\hat{\boldsymbol{I}}_1(\tau_1) + \tau' - \tau_1, \ldots, \hat{\boldsymbol{I}}_n(\tau_n) + \tau' - \tau_n\}$. The asserted $\boldsymbol{\pi}$-correctness of $\boldsymbol{J}_i$ implies $\tau' \in \hat{\boldsymbol{I}}_i(\tau_i) + \tau' - \tau_i$, hence $\tau_i \in \hat{\boldsymbol{I}}_i(\tau_i)$. □

It is helpful to view the precision of shifted intervals as an *ideal* one, in contrast to the *observable* precision obtained by applying drift compensation. We have the following relation between ideal and observable precision:

Lemma 3.10 (Shifting vs. Drift Compensation) *Let* $\boldsymbol{\mathcal{I}} = \{\boldsymbol{I}_1(t_1), \ldots, \boldsymbol{I}_n(t_n)\}$ *be a set of intervals* $\boldsymbol{I}_i(t_i) = [T_i \pm \boldsymbol{\alpha}_i]$ *residing at node* p_i *and representing real-time* t_i *being in synchrony with clock* C_{p_i}. *For some arbitrary* $t' \geq \max_{1 \leq i \leq n}\{t_i\}$, *define* $\boldsymbol{\mathcal{I}}' = \{\boldsymbol{I}'_1(t'), \ldots, \boldsymbol{I}'_n(t')\}$ *to be the set of compatible intervals* $\boldsymbol{I}'_i(t')$ *obtained from* $\boldsymbol{I}_i(t_i)$ *by applying drift compensation at node* p_i.

(1) If the shifted set $\boldsymbol{\mathcal{J}} = \mathrm{shift}_{t'}(\boldsymbol{\mathcal{I}}) = \{\boldsymbol{I}_1(t_1) + t' - t_1, \ldots, \boldsymbol{I}_n(t_n) + t' - t_n\}$ *of compatible intervals all representing* t' *is* $\boldsymbol{\pi}$*-precise, then* $\boldsymbol{\mathcal{I}}'$ *is* $\boldsymbol{\pi}'$*-precise for*

$$\boldsymbol{\pi}' = \boldsymbol{\pi} + \bigcup_i (T'_i - T_i)\boldsymbol{\rho}_{p_i} + \boldsymbol{u}_{p_i} + \boldsymbol{G}\boldsymbol{\rho}, \tag{3.22}$$

where $T'_i - T_i = C_{p_i}(t') - C_{p_i}(t_i)$ *satisfies*

$$\frac{t' - t_i - u^+_{p_i}}{1 + \rho^+_{p_i}} - G \leq T'_i - T_i \leq \frac{t' - t_i + u^-_{p_i}}{1 - \rho^-_{p_i}}. \tag{3.23}$$

(2) If, for all i, the interval $\boldsymbol{I}_i$ is $\boldsymbol{\pi}_i$-correct, then $\boldsymbol{\mathcal{I}}'$ is $\boldsymbol{\pi}'$-correct for

$$\boldsymbol{\pi}' = \bigcup_i \boldsymbol{\pi}_i + (T_i' - T_i)\boldsymbol{\rho}_{p_i} + \boldsymbol{u}_{p_i} + \boldsymbol{G}_{\boldsymbol{\rho}}. \tag{3.24}$$

Proof. Recalling Definition 3.5 of drift compensation, (3.22) and also (3.24) follow immediately, since (ideal) $\boldsymbol{\pi}$ must be enlarged to capture the maximum deviation from real-time that can arise when dragging the intervals $\boldsymbol{I}_i$ from t_i to t'; recall that t_i is in synchrony with node p_i's clock. The bounds on $T_i' - T_i = C_{p_i}(t') - C_{p_i}(t_i)$ given by (3.23) are obtained directly from (3.11). □

The above lemma allows us to carry over any result involving shifted intervals to the situation where those intervals are actually compatible in the real system, i.e., when they are read at some common point in real-time t'. Apart from the inevitable effect of granularity, there is an additional enlargement essentially proportional to $(t' - t_i)\boldsymbol{\rho}_{p_i}$.

With these preparations, we are ready for the precise definition of internal global time:

Definition 3.9 (Internal Global Time) *Let $\boldsymbol{\mathcal{C}}^{(k+1)}$ for $k \geq 0$ be the set of the non-faulty nodes' virtual interval clocks $\boldsymbol{C}_j^{(k+1)}(t_j^{R,(k)})$ of round $k+1$ at their $(k+1)$-th resynchronization instants $t_j^{R,(k)}$, when switching from round k to $k+1$ takes place at node j. If* $\mathrm{shift}_{t^{R,(k)}}(\boldsymbol{\mathcal{C}}^{(k+1)})$ *for some $t^{R,(k)} \geq \max_q\{t_q^{R,(k)}\}$ is $\boldsymbol{\pi}_0$-precise, we define* internal global time $\tau^{(k+1)}(t)$ *for round $k+1$ by*

$$\begin{aligned} \tau^{(k+1)}(t) &= \tau^{(k+1)}(t^{R,(k)}) + (t - t^{R,(k)}) \qquad \text{for } k \geq 0, \\ \tau^{(0)}(t) &= t \end{aligned}$$

valid for all t, where $\tau^{(k+1)}(t^{R,(k)}) \in \bigcap_{\boldsymbol{J} \in \mathrm{shift}_{t^{R,(k)}}(\boldsymbol{\mathcal{C}}^{(k+1)})} \hat{\boldsymbol{J}} \neq \emptyset$.

Note that defining $\tau^{(0)}(t) = t$ is justified by applying Lemma 3.4 to the initial synchronization assumption given in Definition 3.7.

3.5.2 Instantaneous Correction

In this subsection, we will provide our interval-based framework for analyzing worst case precision and accuracy of the generic algorithm given in Definition 3.7 for instantaneous state corrections. As in [58], we will describe convergence functions by a few characteristic parameters (functions) and derive expressions in terms of those. To obtain the final results for a particular instance of the algorithm, it is only necessary to determine the characteristic functions of the particular convergence function and to plug them into Theorems 3.1 and 3.2.

The general outline of our generic analysis is quite straightforward: We provide a sequence of lemmas that characterize how accuracy/precision intervals evolve in a single round. Starting from Lemma 3.11 describing the set of intervals fed into the convergence function, Lemma 3.12 guarantees that the precision provided at the beginning of round k is re-established at the beginning of round $k+1$. On top of that, a simple induction proof can be conducted to establish our major Theorem 3.1, which provides results for instantaneous correction. Finally, adopting the achievements of [56], it follows that those results are also valid in case of continuous amortization.

Our interval-based framework surpasses traditional approaches to precision analysis due to its conceptual beauty and high flexibility w.r.t. incorporating features like clock granularity, broadcast latencies, etc. This is primarily a consequence of our notion of internal global time, which allows us to reason about precision by considering each local interval clock separately without explicitly relating it to the other clocks in the ensemble. Even more, in our analysis, there is no need to consider the "position" of intervals, i.e., local clock values, at all. In fact, any information required is encoded in the interval of accuracies $\boldsymbol{\alpha}$ resp. in the associated $\boldsymbol{\pi}$-precision interval of an interval $\boldsymbol{I}(t) = [T \pm \boldsymbol{\alpha}]$, since all non-faulty accuracy intervals must contain real-time t resp. internal global time τ by construction, which thus serves as a "common reference" for relating them. Of course, the particular reference point may lie anywhere in $[t - \alpha^+, t + \alpha^-]$ resp. $[t - \pi^+, t + \pi^-]$, according to the actually experienced drift, transmission delay, and initial accuracy, but we do not have to deal with it explicitly.

The following first major lemma describes how precision evolves during a round, including local drift compensation and interval dissemination in the FME. Keep in mind that the resulting intervals $\boldsymbol{I}_q^p$ are fed into the convergence function.

Lemma 3.11 (FME Dissemination) *Let* $\boldsymbol{A}_p = [T_p \pm \boldsymbol{\alpha}_p] = \boldsymbol{C}_p^{(k)}(t_p^{R,(k-1)})$ *be the accuracy interval of node p's interval clock at real-time* $t_p = t_p^{R,(k-1)}$*, when round k (for some fixed* $k \geq 0$*) starts, and denote by* $\boldsymbol{\mathcal{A}}$ *the subset of the* $\boldsymbol{A}_p$*'s of those nodes p that remain non-faulty during round k. Let* $T_q^R = (k+1)P_S + \Lambda + \Omega + \Delta + E_q$ *be the logical time when the* $(k+1)$*-th resynchronization instant —happening at real-time* $t_q^R = t_q^{R,(k)}$ *and terminating round k— is scheduled at node q, and let* $\boldsymbol{I}_q^p = \boldsymbol{I}_q^p(t_q^R)$ *be the interval (3.19)/(3.20) that is obtained at node q as the result of delay and drift compensation of a node p's accuracy interval transmitted during the FME in round k.*

If, at the beginning of the round,

[1] the accuracies of any $\boldsymbol{A}_p = [T_p \pm \boldsymbol{\alpha}_p] \in \boldsymbol{\mathcal{A}}$ *are integer multiples of* G_S*,*

[2] $\mathrm{shift}_{t'}(\boldsymbol{\mathcal{A}})$, *$t' \geq \max_p\{t_p\}$ arbitrary, is $\boldsymbol{\pi}_0$-correct for some $\boldsymbol{\pi}_0 = [-\pi_0^-, \pi_0^+]$ with $\pi_0 = \pi_0^- + \pi_0^+$,*

[3] there is some $\boldsymbol{\pi} = [-\pi^-, \pi^+]$ with $\pi = \pi^- + \pi^+ \geq \pi_0$ such that

$$T_p \in [kP_S + \Lambda + \Omega + \Delta + E_p \pm \boldsymbol{\pi}] \tag{3.25}$$

for any $\boldsymbol{A}_p \in \boldsymbol{\mathcal{A}}$,

[4] transmission delay, broadcast delay, computation time compensation, and round period are integer multiples of G satisfying

$$\begin{aligned}
\Delta &\geq \frac{\pi_0 + u_{\max} + \delta_{\max} + (P_S - E_{\min} + \pi^-)\rho_{\max} + \varepsilon_{\max}^+}{1 + \rho_{\max}^+}, && (3.26)\\
\Lambda + \Omega &\geq \frac{\lambda_{\max} + \omega_{\max} + u_{\max}^-}{1 - \rho_{\max}^-}, && (3.27)\\
E_p &\geq \frac{\eta_p + u_p^-}{1 - \rho_p^-}, && (3.28)\\
E_{\max} &\geq \max_p\{E_p\}, \\
0 \leq E_{\min} &\leq \min_p\{E_p\}, \\
P_S &\geq \Lambda + \Omega + \Delta + E_{\max},
\end{aligned}$$

we have the following results:

(1) Any non-faulty receiving node q is able to form its set $\boldsymbol{\mathcal{I}}_q$ of intervals $\boldsymbol{I}_q^p = \boldsymbol{I}_q^p(t_q^R) = [T_q^{p,R} \pm \boldsymbol{\alpha}_q^{p,R}]$ —given by (3.19)/(3.20)— at least η_q real-time seconds before resynchronization takes place at time t_q^R. By default, $\boldsymbol{I}_q^p = \emptyset$ if node p's CSM did not arrive in time. Any $\boldsymbol{I}_q^p \in \boldsymbol{\mathcal{N}}_q \subseteq \boldsymbol{\mathcal{I}}_q$ (contained in the subset $\boldsymbol{\mathcal{N}}_q$ of non-faulty intervals) is accurate with accuracies being integer multiples of G_S, which satisfy

$$\begin{aligned}
\boldsymbol{\alpha}_q^{p,R} \subseteq\ & \boldsymbol{\alpha}_p + \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \mathbf{2}\boldsymbol{G}_A + \boldsymbol{\varepsilon}_{pq} \\
& + (P_S - \Delta - E_p)\boldsymbol{\rho}_p + (E_q + \Delta - \delta_{pq})\boldsymbol{\rho}_q \\
& + (\Lambda + \Omega)\Big[-\max\{\rho_q^- - \rho_p^-, 0\}, \max\{\rho_q^+ - \rho_p^+, 0\}\Big] \\
& + \mathcal{O}(\pi + P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}
\end{aligned} \tag{3.29}$$

and

$$\begin{aligned}
\boldsymbol{\alpha}_q^{p,R} \supseteq\ & \boldsymbol{\alpha}_p + \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \mathbf{2}\boldsymbol{G}_A + \boldsymbol{\varepsilon}_{pq} \\
& + (P_S - \Delta - E_p)\boldsymbol{\rho}_p + (E_q + \Delta - \delta_{pq})\boldsymbol{\rho}_q \\
& - (\Lambda + \Omega)\Big[-\max\{\rho_q^- - \rho_p^-, 0\}, \max\{\rho_q^+ - \rho_p^+, 0\}\Big] \\
& + \mathcal{O}(\pi + P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}
\end{aligned} \tag{3.30}$$

for $p \neq q$, and

$$\boldsymbol{\alpha}_q^{q,R} = \boldsymbol{\alpha}_q + \boldsymbol{u}_q + P_S\boldsymbol{\rho}_q + \mathcal{O}(\pi)\boldsymbol{\rho}_q. \tag{3.31}$$

Moreover, any $\boldsymbol{I}_q^p \in \boldsymbol{\mathcal{N}}_q$ is $\boldsymbol{\pi}_q^p$-correct with

$$\begin{aligned}
\boldsymbol{\pi}_q^p \;\subseteq\; & \boldsymbol{\pi}_0 + \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{pq} + (P_S - \Delta - E_p)\boldsymbol{\rho}_p + (E_q + \Delta - \delta_{pq})\boldsymbol{\rho}_q \\
& + (\Lambda + \Omega)\Big[-\max\{\rho_q^- - \rho_p^-, 0\}, \max\{\rho_q^+ - \rho_p^+, 0\}\Big] \\
& + \mathcal{O}(\pi + P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}
\end{aligned} \tag{3.32}$$

for $p \neq q$, and

$$\boldsymbol{\pi}_q^q = \boldsymbol{\pi}_0 + \boldsymbol{u}_q + P_S\boldsymbol{\rho}_q + \mathcal{O}(\pi)\boldsymbol{\rho}_q. \tag{3.33}$$

The whole set $\boldsymbol{\mathcal{N}}_q$ is $\boldsymbol{\pi}_q^H$-correct for

$$\begin{aligned}
\boldsymbol{\pi}_q^H \;\subseteq\; & \boldsymbol{\pi}_0 + \boldsymbol{u}_{\max} + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{\max} \\
& + (P_S - \Delta - E_{\min})\boldsymbol{\rho}_{\max} + (E_q + \Delta - \delta_{\min})\boldsymbol{\rho}_q \\
& + \mathcal{O}(\pi + P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}.
\end{aligned} \tag{3.34}$$

(2) *The set $\boldsymbol{\mathcal{N}} = \bigcup_{q \text{ non-faulty}} \boldsymbol{\mathcal{N}}_q$ containing all non-faulty intervals $\boldsymbol{I}_q^p$ at all non-faulty receiving nodes q has the property that $\operatorname{shift}_{t''}(\boldsymbol{\mathcal{N}})$ for some arbitrary t'' satisfying $t'' \geq \max_q\{t_q^R\}$ is $\boldsymbol{\pi}^H$-correct for*

$$\begin{aligned}
\boldsymbol{\pi}^H \;\subseteq\; & \boldsymbol{\pi}_0 + 2\boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{\max} + (P_S + E_{\max} - E_{\min} - \delta_{\min})\boldsymbol{\rho}_{\max} \\
& + \mathcal{O}(\pi + P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}.
\end{aligned} \tag{3.35}$$

(3) *Finally, the set $\boldsymbol{\mathcal{N}}^p$ formed by the non-faulty nodes' "perceptions" $\boldsymbol{I}_q^p$ of the accuracy interval transmitted by a single non-faulty node p has the property that $\operatorname{shift}_{t''}(\boldsymbol{\mathcal{N}}^p)$ for $t'' \geq \max_q\{t_q^R\}$ is both $\boldsymbol{\pi}^p$-correct and $\boldsymbol{\pi}_I$-precise with*

$$\begin{aligned}
\boldsymbol{\pi}^p \;\subseteq\; & \boldsymbol{\pi}_0 + \boldsymbol{u}_p + \boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{\max} \\
& + (P_S - \Delta - E_p)\boldsymbol{\rho}_p + (E_{\max} + \Delta - \delta_{\min})\boldsymbol{\rho}_{\max} \\
& + (\Lambda + \Omega)\Big[-(\rho_{\max}^- - \rho_p^-), \rho_{\max}^+ - \rho_p^+\Big] \\
& + \mathcal{O}(\pi + P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max},
\end{aligned} \tag{3.36}$$

$$\begin{aligned}
\boldsymbol{\pi}_I \;\subseteq\; & \boldsymbol{\varepsilon}_{\max} + H\boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + (\Lambda + \Omega + \Delta + E_{\max} - \delta_{\min})\boldsymbol{\rho}_{\max} \\
& + \mathcal{O}(\pi_0 + P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}.
\end{aligned} \tag{3.37}$$

Proof. From the description of the clock synchronization algorithm in Definition 3.7, we know that any node p initiates its broadcast in round k when its local clock reaches time $(k+1)P_S$. Due to broadcast latency $\lambda_{\max}$ and broadcast operation delay $\omega_p \leq \omega_{\max}$ according to Assumption 3.4, the CSM to node q is actually transmitted when the local clock of the transmitting node p reads time $T_{pq}^A = C_p(t_{pq}^A)$ satisfying

$$0 \leq T_{pq}^A - (k+1)P_S \leq \frac{\lambda_{\max} + \omega_{\max} + u_{\max}^-}{1 - \rho_{\max}^-} \leq \Lambda + \Omega, \tag{3.38}$$

recall (3.11) in conjunction with (3.27). Of course, t_{pq}^A is the real-time when the CSM (containing the accuracy interval $\boldsymbol{A}_{pq}^A = [T_{pq}^A \pm \boldsymbol{\alpha}_{pq}^A]$) is actually transmitted. Combining (3.38) with (3.25), we easily obtain

$$\begin{aligned} T_{pq}^A - T_p &\leq (k+1)P_S + \Lambda + \Omega - (kP_S + \Lambda + \Omega + \Delta + E_p - \pi^-) \\ &= P_S - \Delta - E_p + \pi^- \end{aligned} \tag{3.39}$$

and similarly

$$T_{pq}^A - T_q \leq P_S - \Delta - E_q + \pi^-. \tag{3.40}$$

The CSM from node p is received (we consider non-faulty intervals here) at node q delayed by δ'_{pq}, hence at real-time $t_q^p = t_{pq}^A + \delta'_{pq}$, when the local clock of node q reads T_q^p. We will now establish bounds on $T_q^p - T_{pq}^A$.

Relating the points in real-time involved in the evolution of a round (from the beginning to reception of the FME) yields

$$t_q^p - t_q = t_q^p - t_p - (t_q - t_p) = t_{pq}^A + \delta'_{pq} - t_p - (t_q - t_p). \tag{3.41}$$

Since t_q, t_p, and t_{pq}^A (but not t_q^p) are in synchrony with their respective clocks, applying (3.10) to $t_q^p - t_q$ and $t_{pq}^A - t_p$ provides

$$\begin{aligned} (T_q^p - T_q)(1 - \rho_q^-) - u_q^- &\leq (T_{pq}^A - T_p)(1 + \rho_p^+) + u_p^+ + \delta'_{pq} - (t_q - t_p) \\ (T_q^p - T_q)(1 + \rho_q^+) + G(1 + \rho_q^+) + u_q^+ &\geq (T_{pq}^A - T_p)(1 - \rho_p^-) - u_p^- + \delta'_{pq} - (t_q - t_p). \end{aligned}$$

Some algebraic manipulations produce

$$\begin{aligned} (T_q^p - T_{pq}^A)(1 - \rho_q^-) &\leq (T_q - T_{pq}^A)(1 - \rho_q^-) + (T_{pq}^A - T_p)(1 + \rho_p^+) \\ &\quad + \delta'_{pq} - (t_q - t_p) + u_q^- + u_p^+ \\ &\leq T_q - t_q - (T_p - t_p) + (T_{pq}^A - T_q)\rho_q^- + (T_{pq}^A - T_p)\rho_p^+ \\ &\quad + \delta_{pq} + \varepsilon_{pq}^+ + u_q^- + u_p^+ \end{aligned}$$

and

$$\begin{aligned} G(1+\rho_q^+) + (T_q^p - T_{pq}^A)(1+\rho_q^+) &\geq (T_q - T_{pq}^A)(1+\rho_q^+) + (T_{pq}^A - T_p)(1-\rho_p^-) + \delta'_{pq} \\ &\quad - (t_q - t_p) - u_q^+ - u_p^- \\ &\geq T_q - t_q - (T_p - t_p) - (T_{pq}^A - T_q)\rho_q^+ \\ &\quad - (T_{pq}^A - T_p)\rho_p^- + \delta_{pq} - \varepsilon_{pq}^- - u_q^+ - u_p^- \end{aligned}$$

Abbreviating $\mu_{pq} = T_q - t_q - (T_p - t_p)$ and recalling (3.39) and (3.40), we eventually obtain

$$\begin{aligned} T_q^p - T_{pq}^A &\leq \frac{1}{1-\rho_q^-}\Big[\mu_{pq} + \delta_{pq} + (P_S - \Delta - E_q + \pi^-)\rho_q^- \\ &\quad + (P_S - \Delta - E_p + \pi^-)\rho_p^+ + \varepsilon_{pq}^+ + u_q^- + u_p^+\Big], && (3.42) \\ T_q^p - T_{pq}^A &\geq \frac{1}{1+\rho_q^+}\Big[\mu_{pq} + \delta_{pq} - (P_S - \Delta - E_q + \pi^-)\rho_q^+ \\ &\quad - (P_S - \Delta - E_p + \pi^-)\rho_p^- - \varepsilon_{pq}^- - u_q^+ - u_p^-\Big] - G. && (3.43) \end{aligned}$$

Since all $\boldsymbol{A}_i = [T_i \pm \boldsymbol{\alpha}_i] \in \boldsymbol{\mathcal{A}}$ are $\boldsymbol{\pi}_0$-correct according to precondition [2] of our lemma, Lemma 3.6 provides $-\pi_0 \leq \mu_{pq} \leq \pi_0$ since $\boldsymbol{\pi}_0 + \overline{\boldsymbol{\pi}}_0 = [-\pi_0, \pi_0]$, so that it follows from (3.42) that

$$\begin{aligned} T_q^p &\leq T_{pq}^A + \frac{\pi_0 + \delta_{pq} + (P_S - \Delta - E_{\min} + \pi^-)(\rho_q^- + \rho_p^+) + \varepsilon_{pq}^+ + u_q^- + u_p^+}{1-\rho_q^-} \\ &\leq (k+1)P_S + \Lambda + \Omega + \Delta - \Delta \\ &\quad + \frac{\pi_0 + \delta_{\max} + (P_S - \Delta - E_{\min} + \pi^-)\rho_{\max} + \varepsilon_{\max}^+ + u_{\max}}{1-\rho_{\max}^-} \\ &= (k+1)P_S + \Lambda + \Omega + \Delta \\ &\quad + \frac{\pi_0 + \delta_{\max} + (P_S - E_{\min} + \pi^-)\rho_{\max} + \varepsilon_{\max}^+ + u_{\max}}{1-\rho_{\max}^-} - \Delta\left(\frac{\rho_{\max}}{1-\rho_{\max}^-} + 1\right) \\ &\leq (k+1)P_S + \Lambda + \Omega + \Delta; \end{aligned}$$

the last step is confirmed by plugging in the definition of Δ according to (3.26). This result eventually assures that $\boldsymbol{I}_q^p$ from any non-faulty node $p \neq q$ is available for computing the convergence function at node q at latest when q's clock reads $(k+1)P_S + \Lambda + \Omega + \Delta$, leaving a logical time duration E_q up to time T_q^R when resynchronization will take place. This is also true for the interval $\boldsymbol{I}_q^q$ from the own node, which can in fact be (pre-)computed at any time, recall the remark following (3.19). Remembering (3.28), this means that at least η_q real-time seconds are available for computing the convergence function, as asserted in item (1) of our lemma.

Apart from that, inequalities (3.42) and (3.43) may be condensed into the more convenient form

$$T_q^p - T_{pq}^A = \delta_{pq} + \mathcal{O}(\pi_0 + P_S \rho_{\max} + G + \varepsilon_{\max}), \tag{3.44}$$

where we used $u_{\max} = \mathcal{O}(G)$ according to Assumption 3.2.

With this preparatory work, we can attack the results given in item (1) of the lemma. According to our expositions in Section 3.4, each receiving node q relies on (3.19)/(3.20) to compute the interval

$$\begin{aligned} \boldsymbol{I}_q^p = \boldsymbol{I}_q^p(t_q^R) &= \boldsymbol{A}_{pq}^A + [T_q^R - T_q^p + \delta_{pq} \pm \boldsymbol{\varepsilon}_{pq} + \boldsymbol{2G}_A] \\ &\quad + (T_q^R - T_q^p)\boldsymbol{\rho}_q + \boldsymbol{u}_q + \overline{\boldsymbol{G}} \end{aligned} \tag{3.45}$$

from the accuracy interval $\boldsymbol{A}_{pq}^A$ received from node $p \neq q$. Incorporating the term that accounts for local deterioration (see Assumption 3.3) at node p from local time T_p where the round started (with the accuracy interval $\boldsymbol{A}_p = [T_p \pm \boldsymbol{\alpha}_p]$) up to transmission of the CSM to node q at time T_{pq}^A, we easily obtain an expression for the interval of accuracies $\boldsymbol{\alpha}_q^{p,R}$ of $\boldsymbol{I}_q^p = [T_q^{p,R} \pm \boldsymbol{\alpha}_q^{p,R}]$ from (3.45):

$$\begin{aligned} \boldsymbol{\alpha}_q^{p,R} &= \boldsymbol{\alpha}_p + (T_{pq}^A - T_p)\boldsymbol{\rho}_p + \boldsymbol{u}_p + \boldsymbol{\varepsilon}_{pq} + \boldsymbol{2G}_A + (T_q^R - T_q^p)\boldsymbol{\rho}_q + \boldsymbol{u}_q + \overline{\boldsymbol{G}} \\ &= \boldsymbol{\alpha}_p + \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \boldsymbol{2G}_A + \boldsymbol{\varepsilon}_{pq} \\ &\quad + (T_{pq}^A - T_p)\boldsymbol{\rho}_p + (T_q^R - T_{pq}^A)\boldsymbol{\rho}_q - (T_q^p - T_{pq}^A)\boldsymbol{\rho}_q. \end{aligned} \tag{3.46}$$

Since we assumed in Definition 3.7 that all parameter values appearing in (3.46) are integer multiples of G_S, which is true for $\boldsymbol{\alpha}_p$ according to precondition [1] of our lemma as well, $\boldsymbol{\alpha}_q^{p,R}$ is also an integer multiple of G_S as asserted in item (1). Considering

$$f(c) = (c-a)r_p + (b-c)r_q \leq \begin{cases} (c_{\max} - a)r_p + (b - c_{\max})r_q & \text{if } r_p \geq r_q, \\ (c_{\min} - a)r_p + (b - c_{\min})r_q & \text{if } r_p \leq r_q \end{cases}$$

for $r_p, r_q \geq 0$ (and similarly with $c_{\max}$ and $c_{\min}$ exchanged for the lower bound), which follows immediately from $f(c) = c(r_p - r_q) - ar_p + br_q$, we obtain

$$\begin{aligned} f(c) &\leq (c_{\max} - a)r_p + (b - c_{\max})r_q + \max\{r_q - r_p, 0\}(c_{\max} - c_{\min}) \\ f(c) &\geq (c_{\max} - a)r_p + (b - c_{\max})r_q - \max\{0, r_p - r_q\}(c_{\max} - c_{\min}) \end{aligned} \tag{3.47}$$

Using the above upper bound in equation (3.46) for $\boldsymbol{\alpha}_q^{p,R}$ and recalling (3.39), $T_q^R = (k+1)P_S + \Lambda + \Omega + \Delta + E_q$ in conjunction with (3.38), and relation (3.44), we obtain the upper bound

$$\begin{aligned}\boldsymbol{\alpha}_q^{p,R} \;\subseteq\;& \boldsymbol{\alpha}_p + \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \boldsymbol{2G}_A + \boldsymbol{\varepsilon}_{pq} \\ &+ (P_S - \Delta - E_p + \pi^-)\boldsymbol{\rho}_p + (E_q + \Delta)\boldsymbol{\rho}_q \\ &+ (\Lambda + \Omega)\big[-\max\{\rho_q^- - \rho_p^-, 0\}, \max\{\rho_q^+ - \rho_p^+, 0\}\big] \\ &- \delta_{pq}\boldsymbol{\rho}_q + \mathcal{O}(\pi_0 + P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_q; \end{aligned} \tag{3.48}$$

remembering that $\pi_0 \leq \pi$ according to precondition [3] of our lemma, the result stated in (3.29) follows. For (3.30), we just have to employ in the lower bound (3.47) instead of the upper bound in the derivation above, which amounts to replacing the term $(\Lambda + \Omega)\big[-\max\{\rho_q^- - \rho_p^-, 0\}, \max\{\rho_q^+ - \rho_p^+, 0\}\big]$ in (3.48) by $-(\Lambda + \Omega)\big[-\max\{0, \rho_p^- - \rho_q^-\}, \max\{0, \rho_p^+ - \rho_q^+\}\big]$.

We still have to investigate $\boldsymbol{\alpha}_q^{p,R}$ in case of $p = q$. Incorporating the term $(T_{qq}^A - T_q)\boldsymbol{\rho}_q + \boldsymbol{u}_q$, which accounts for deterioration at node q from local time T_q up to $T_{qq}^A = (k+1)P_S$, the instant of the virtual "loop-back transmission", in (3.20) provides

$$\boldsymbol{\alpha}_q^{q,R} = \boldsymbol{\alpha}_q + (T_q^R - T_q)\boldsymbol{\rho}_q + \boldsymbol{u}_q,$$

from which (3.31) follows immediately.

The above derivations for accuracy immediately carry over to $\boldsymbol{\pi}_q^p$-correctness of $\boldsymbol{I}_q^p$ for a suitably chosen $\boldsymbol{\pi}_q^p$: Given that $\boldsymbol{A}_p$ was $\boldsymbol{\pi}_0$-correct at the beginning of round k (at real-time t_p), we only have to add the maximum uncertainties caused by the drift and delay compensation operations. Recall that internal global time for any fixed round progresses as real-time does, so that maintaining accuracy w.r.t. real-time by enlarging the interval automatically maintains "accuracy" w.r.t. internal global time as well. Note that the term $\boldsymbol{2G}_A$ accounting for limited accuracy transmission resolution (see Assumption 3.4) can be ignored here, since precision intervals like $\boldsymbol{\pi}_0$ are not handled by the algorithm but rather computed in our analysis. Adopting (3.46) appropriately, literally the same derivation that led to (3.29) resp. (3.31) provides (3.32) resp. (3.33).

Moreover, by virtue of Lemma 3.5, we find that the set $\boldsymbol{\mathcal{N}}_q$ of all non-faulty $\boldsymbol{I}_q^p$ at node q is $\boldsymbol{\pi}_q^H$-correct for $\boldsymbol{\pi}_q^H = \bigcup_{p \neq q} \boldsymbol{\pi}_q^p$. Straightforward majorizations of (3.32) easily confirm the value given in (3.34); note that setting $\boldsymbol{\rho}_p = \boldsymbol{\rho}_{\max}$ causes the term involving $[-\max\{\rho_q^- - \rho_p^-, 0\}, \max\{\rho_q^+ - \rho_p^+, 0\}]$ to vanish. Finally, $\boldsymbol{\pi}_q^q \subseteq \boldsymbol{\pi}_q^H$ is also true by virtue of the technical condition $\delta_{\min}\boldsymbol{\rho}_{\max} \subseteq \boldsymbol{\varepsilon}_{\max}$ according to (3.16), so that genuinely $\boldsymbol{\pi}_q^H = \bigcup_p \boldsymbol{\pi}_q^p$ as required.

To prove item (2), we note that the asserted $\boldsymbol{\pi}^H$-correctness of the shifted set $\boldsymbol{\mathcal{N}}$ is a straightforward consequence of Lemma 3.5 applied to $\bigcup_{q \text{ non-faulty}} \mathrm{shift}_{t''}(\boldsymbol{\mathcal{N}}_q)$. The bound stated in (3.35) follows from

$$\begin{aligned}\boldsymbol{\pi}^H &= \bigcup_q \boldsymbol{\pi}_q^H \\ &\subseteq \boldsymbol{\pi}_0 + 2\boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{\max} \\ &\quad + (P_S - \Delta - E_{\min})\boldsymbol{\rho}_{\max} + (E_{\max} + \Delta - \delta_{\min})\boldsymbol{\rho}_{\max} \\ &\quad + \mathcal{O}(\pi + P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}.\end{aligned}$$

Finally, turning our attention to the set $\boldsymbol{\mathcal{N}}^p$ of intervals obtained at different (non-faulty) receivers for the same broadcast (of non-faulty node p), Lemma 3.5 applied to $\bigcup_{q\neq p} \mathrm{shift}_{t''}(\boldsymbol{I}_q^p)$ yields $\boldsymbol{\pi}^p = \bigcup_{q\neq p} \boldsymbol{\pi}_q^p$, and trivial majorizations easily provide (3.36). Again, $\boldsymbol{\pi}_q^q \subseteq \boldsymbol{\pi}^p$ by virtue of (3.16), so that $\boldsymbol{\pi}^p = \bigcup_q \boldsymbol{\pi}_q^p$ as required.

In addition, it is clear that almost the same accuracy interval $\boldsymbol{A}_{pq}^A$ appears in $\boldsymbol{I}_q^p$ of any receiver q. More specifically, the only difference is the deterioration that occurs at the sender p between $T_{\min}^p = \min_q\{T_{pq}^A\}$ and the particular T_{pq}^A under consideration. Conceptually, this may be viewed as if node p has commenced with an interval $[T_{\min}^p \pm \mathbf{0}]$ of length 0. Starting from an equation adopted from (3.45) in a similar way, we obtain

$$\begin{aligned}\boldsymbol{\pi}_I &\subseteq \bigcup_{q,p\neq q}\Big[(T_{pq}^A - \min_q\{T_{pq}^A\})\boldsymbol{\rho}_p + (H-1)\boldsymbol{u}_p + \boldsymbol{\varepsilon}_{pq} \\ &\qquad + (T_q^R - T_{pq}^A)\boldsymbol{\rho}_q - (T_q^p - T_{pq}^A)\boldsymbol{\rho}_q + \boldsymbol{u}_q\Big] + \overline{\boldsymbol{G}} \qquad (3.49) \\ &\subseteq \boldsymbol{\varepsilon}_{\max} + B\boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + \bigcup_{q,p\neq q} (T_q^R - \min_q\{T_{pq}^A\})\boldsymbol{\rho}_{\max} - \delta_{\min}\boldsymbol{\rho}_{\max} \\ &\quad + \mathcal{O}(\pi_0 + P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}\end{aligned}$$

by majorizing $\boldsymbol{\rho}_p$, $\boldsymbol{\rho}_q$ by $\boldsymbol{\rho}_{\max}$ and using (3.44). Note that $T_{pq}^A - \min_q\{T_{pq}^A\} = 0$ in case of a broadcast network ($H = 1$, see Assumption 3.4), so that no deterioration occurs here. The result stated in (3.37) follows from recalling that $T_q^R - \min_q\{T_{pq}^A\} \leq \Lambda + \Omega + \Delta + E_{\max}$ by virtue of (3.38) and the definition of the logical resynchronization time T_q^R. Note that the contribution $\big(T_q^R - (k+1)P_S\big)\boldsymbol{\rho}_q = (\Lambda + \Omega + \Delta + E_q)\boldsymbol{\rho}_q$ of the own node q to $\boldsymbol{\pi}^I$ is again covered by expression (3.37) due to the technical condition (3.16). This eventually completes the proof of Lemma 3.11. □

It is important to understand that the set $\boldsymbol{\mathcal{N}}^p$ is $\boldsymbol{\pi}^H$-correct and $\boldsymbol{\pi}_I$-precise, but not necessarily $\boldsymbol{\pi}_I$-correct. After all, we cannot assume that internal global time $\tau^{(k)}(t)$ of the current round k lies in the intersection of the (quite small) $\boldsymbol{\pi}_I$-precision intervals associated with the elements of $\boldsymbol{\mathcal{N}}^p$.

Next we will provide the properties of generic convergence functions in terms of their characteristic functions. We start with the following auxiliary definition:

Definition 3.10 (Translation Invariance, Weak Monotonicity) *Given two sets* $\boldsymbol{\mathcal{I}} = \{\boldsymbol{I}_1, \dots, \boldsymbol{I}_n\}$ *and* $\boldsymbol{\mathcal{J}} = \{\boldsymbol{J}_1, \dots, \boldsymbol{J}_n\}$ *of* $n \geq 1$ *accuracy intervals, an interval-valued function* $\boldsymbol{f}(\cdot)$ *of* $n \geq 1$ *interval arguments is called*

(1) weakly monotonic *iff* $\boldsymbol{I}_j \subseteq \boldsymbol{J}_j$ *with* $\mathrm{ref}(\boldsymbol{I}_j) = \mathrm{ref}(\boldsymbol{J}_j)$ *for all* $1 \leq j \leq n$ *implies* $\boldsymbol{f}(\boldsymbol{\mathcal{I}}) \subseteq \boldsymbol{f}(\boldsymbol{\mathcal{J}})$,

(2) translation invariant *iff* $\boldsymbol{f}(\boldsymbol{I}_1 + \Delta, \dots, \boldsymbol{I}_n + \Delta) = \boldsymbol{f}(\boldsymbol{I}_1, \dots, \boldsymbol{I}_n) + \Delta$ *for any real* Δ.

Note that a weakly monotonic function satisfies this property for both accuracy intervals and associated $\boldsymbol{\pi}$-precision intervals, hence $\hat{\boldsymbol{I}}_j \subseteq \hat{\boldsymbol{J}}_j$ with $\mathrm{ref}(\hat{\boldsymbol{I}}_j) = \mathrm{ref}(\hat{\boldsymbol{J}}_j)$ for all j implies $\boldsymbol{f}(\hat{\boldsymbol{\mathcal{I}}}) \subseteq \boldsymbol{f}(\hat{\boldsymbol{\mathcal{J}}})$.

Definition 3.11 (Generic Convergence Function) *Let* $\boldsymbol{\mathcal{I}}_p = \{\boldsymbol{I}_p^1, \dots, \boldsymbol{I}_p^n\}$ *resp.* $\boldsymbol{\mathcal{I}}_q = \{\boldsymbol{I}_q^1, \dots, \boldsymbol{I}_q^n\}$, $q \neq p$, *be two ordered sets of* n *compatible intervals obtained at nodes* p *resp.* q *at the end of a round, which are in accordance with a given fault model* $\mathcal{F}$. *Assuming that*

[1] the accuracies of any non-faulty $\boldsymbol{I}_p^i = [T_p^i \pm \boldsymbol{\alpha}_p^i]$ *are integer multiples of* G_S *satisfying* $\boldsymbol{\alpha}_p^i \subseteq \boldsymbol{\beta}_p^i \in \boldsymbol{\mathcal{B}}_p$ *for a given set of accuracy bounds* $\boldsymbol{\mathcal{B}}_p = \{\boldsymbol{\beta}_p^1, \dots, \boldsymbol{\beta}_p^n\}$, *and analogous for* $\boldsymbol{I}_q^i$ *with the set of accuracy bounds* $\boldsymbol{\mathcal{B}}_q$,

[2] any non-faulty $\boldsymbol{I}_p^i$ *is* $\boldsymbol{\pi}_p^i$*-correct for* $\boldsymbol{\pi}_p^i \in \boldsymbol{\mathcal{P}}_p = \{\boldsymbol{\pi}_p^1, \dots, \boldsymbol{\pi}_p^n\}$ *denoting a given set of precision bounds, and analogous for* $\boldsymbol{I}_q^i$ *with set of precision bounds* $\boldsymbol{\mathcal{P}}_q = \{\boldsymbol{\pi}_q^1, \dots, \boldsymbol{\pi}_q^n\}$,

[3] $\boldsymbol{\mathcal{P}} = \{\boldsymbol{\pi}^1, \dots, \boldsymbol{\pi}^n\}$ *with* $\boldsymbol{\pi}_p^i \cup \boldsymbol{\pi}_q^i \subseteq \boldsymbol{\pi}^i \subseteq \boldsymbol{\pi}^H$, *for some suitable* $\boldsymbol{\pi}^H$, *denotes a set of uniform precision bounds ensuring* $\boldsymbol{\pi}^i$*-correctness of both* $\boldsymbol{I}_p^i$ *and* $\boldsymbol{I}_q^i$ *(if non-faulty),*

[4] any pair of non-faulty intervals $\{\boldsymbol{I}_p^i, \boldsymbol{I}_q^i\}$ *is* $\boldsymbol{\pi}_I$*-precise for some* $\boldsymbol{\pi}_I \subseteq \boldsymbol{\pi}^H$,

[5] for any $s \in \{1, \dots, n\}$ *with both* $\boldsymbol{I}_p^s$ *and* $\boldsymbol{I}_q^s$ *being non-faulty, the common intersection of the associated precision intervals* $\hat{\boldsymbol{I}}_p^s \cap \hat{\boldsymbol{I}}_q^s \cap \hat{\boldsymbol{I}}_p^{\min_p} \cap \hat{\boldsymbol{I}}_q^{\min_q}$ *resp.* $\hat{\boldsymbol{I}}_p^s \cap \hat{\boldsymbol{I}}_q^s \cap \hat{\boldsymbol{I}}_p^{\max_p} \cap \hat{\boldsymbol{I}}_q^{\max_q}$ *has length at least* $\iota_s^+ \geq 0$ *resp.* $\iota_s^- \geq 0$ *(all integer multiples of* G_S*), where* $\min_x$ *resp.* $\max_x$ *represents that non-faulty node that leads to the leftmost* $\mathrm{right}(\hat{\boldsymbol{I}}_x^{\min_x})$ *resp. the rightmost* $\mathrm{left}(\hat{\boldsymbol{I}}_x^{\max_x})$ *for* $x \in \{p, q\}$,

let

$$\begin{aligned} \boldsymbol{S}_p &= \boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\boldsymbol{\mathcal{I}}_p) = [T'_p \pm \boldsymbol{\alpha}'_p], \\ \boldsymbol{S}_q &= \boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\boldsymbol{\mathcal{I}}_q) = [T'_q \pm \boldsymbol{\alpha}'_q]. \end{aligned}$$

The generic convergence function $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ *must be translation invariant, weakly monotonic, and has to provide accurate intervals with reference point and accuracies being integer multiples of* G_S*. Its properties are characterized by the following functions, which must be weakly monotonic w.r.t. any interval argument:*

(1) Conditional accuracy preservation functions $\Phi_\alpha^-(\cdot)$, $\Phi_\alpha^+(\cdot)$, *so that*

$$\begin{aligned} \boldsymbol{\alpha}_p' &\subseteq \Big[-\Phi_\alpha^-(\boldsymbol{\mathcal{B}}_p, \boldsymbol{\mathcal{P}}_p, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I, \forall s: \iota_s^-), \Phi_\alpha^+(\boldsymbol{\mathcal{B}}_p, \boldsymbol{\mathcal{P}}_p, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I, \forall s: \iota_s^+)\Big] \\ \boldsymbol{\alpha}_q' &\subseteq \Big[-\Phi_\alpha^-(\boldsymbol{\mathcal{B}}_q, \boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I, \forall s: \iota_s^-), \Phi_\alpha^+(\boldsymbol{\mathcal{B}}_q, \boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I, \forall s: \iota_s^+)\Big]. \end{aligned}$$

(2) Precision preservation function $\boldsymbol{\Phi}_\pi(\cdot)$, *so that* $\boldsymbol{S}_p$ *is* $\boldsymbol{\Phi}_\pi(\boldsymbol{\mathcal{P}}_p, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I)$*-correct and* $\boldsymbol{S}_q$ *is* $\boldsymbol{\Phi}_\pi(\boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I)$*-correct, with* $||\boldsymbol{\Phi}_\pi(\boldsymbol{\mathcal{P}}, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I)|| = \mathcal{O}(\pi^H)$ *for* $\pi^H = ||\boldsymbol{\pi}^H||$.

(3) Precision enhancement function $\Psi_\pi(\cdot)$, *so that the set* $\{\boldsymbol{S}_p, \boldsymbol{S}_q\}$ *is* $\boldsymbol{\pi}_0$*-precise for any* $\boldsymbol{\pi}_0$ *satisfying* $||\boldsymbol{\pi}_0|| = \pi_0 = \Psi_\pi(\boldsymbol{\mathcal{P}}, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I) < \pi^H = ||\boldsymbol{\pi}^H||$.

(4) Conditional intersection enhancement functions $\Psi_\iota^-(\cdot)$ *resp.* $\Psi_\iota^+(\cdot)$, *so that the set* $\{\boldsymbol{S}_p, \boldsymbol{S}_q\}$ *is* $\boldsymbol{\pi}_0^{\iota_{pq}^-}$*-precise resp.* $\boldsymbol{\pi}_0^{\iota_{pq}^+}$*-precise for worst case settings w.r.t.* $\mathrm{acc}^-(\boldsymbol{S}_p)$ *resp.* $\mathrm{acc}^+(\boldsymbol{S}_p)$*, with*

$$\begin{aligned} \left|\left|\boldsymbol{\pi}_0^{\iota_{pq}^-}\right|\right| = \pi_0^{\iota_{pq}^-} &= \Psi_\iota^-(\boldsymbol{\mathcal{B}}_p, \boldsymbol{\mathcal{B}}_q, \boldsymbol{\mathcal{P}}_p, \boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I, \forall s: \iota_s^-) \\ \left|\left|\boldsymbol{\pi}_0^{\iota_{pq}^+}\right|\right| = \pi_0^{\iota_{pq}^+} &= \Psi_\iota^+(\boldsymbol{\mathcal{B}}_p, \boldsymbol{\mathcal{B}}_q, \boldsymbol{\mathcal{P}}_p, \boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I, \forall s: \iota_s^+), \end{aligned}$$

and analogous $\boldsymbol{\pi}_0^{\iota_{qp}^\pm}$ *for determining the worst case settings w.r.t.* $\mathrm{acc}^\pm(\boldsymbol{S}_q)$.

Informally, the accuracy preservation functions $\Phi_\alpha^\pm(\cdot)$ give bounds on the new interval of accuracies, the precision preservation function $\boldsymbol{\Phi}_\pi(\cdot)$ gives the precision of the new accuracy intervals w.r.t. old internal global time, and the precision enhancement function $\Psi_\pi(\cdot)$ gives the precision of the new accuracy intervals w.r.t. new internal global time.

The conditional intersection enhancement functions $\Psi_\iota^\pm(\cdot)$ are in charge of keeping track how the convergence function affects the common intersection $\boldsymbol{\iota}$ of the associated precision intervals[†], which eventually determines the worst case setting for $\boldsymbol{\alpha}_p'$ and $\boldsymbol{\alpha}_q'$. Just taking $\boldsymbol{\iota} = \emptyset$ leads to overly conservative accuracy bounds, since the worst case enlargement of the accuracies cannot occur in consecutive rounds. This striking observation is owing to the fact that the worst case settings are usually adjoined with an initial precision (i.e., after resynchronization) that is better than the worst case one.

[†] To enable some degree of freedom, we do not specify which one in our generic framework.

Hence, by feeding the length ι of the common intersection as an additional parameter to the accuracy preservation functions $\Phi_\alpha^\pm(\cdot)$, the worst case enlargement of $\boldsymbol{\alpha}'_p$ and $\boldsymbol{\alpha}'_q$ can be "conditioned" on the common intersection actually present. The (conditional) intersection enhancement functions $\Psi_\iota^\pm(\cdot)$ effectively determines ι for the next round, i.e., propagates the required information over multiple rounds. Note that a lower bound is sufficient for this purpose, since excessive adjustments happen for small intersections only.

Making this idea working in practice, however, is tricky for several reasons: First of all, it is the particular convergence function that determines how many/which input intervals are involved in the worst case accuracy setting. Moreover, different ι^- resp. ι^+ are usually required for computing the worst case bound for the negative resp. positive accuracy. Also a dependence on a particular node p might be the case. To cope with those problems, we actually utilize individual lower bounds ι_s^+ resp. ι_s^-, $1 \leq s \leq n$, on the common intersection of $(\hat{\boldsymbol{I}}_p^s \cap \hat{\boldsymbol{I}}_p^{\min_p}) \cap (\hat{\boldsymbol{I}}_q^s \cap \hat{\boldsymbol{I}}_q^{\min_q})$ resp. $(\hat{\boldsymbol{I}}_p^s \cap \hat{\boldsymbol{I}}_p^{\max_p}) \cap (\hat{\boldsymbol{I}}_q^s \cap \hat{\boldsymbol{I}}_q^{\max_q})$, see Figure 3.5. To be complete, all lengths $\iota_1^\pm, \ldots, \iota_n^\pm$ are supplied as parameters in functions $\Phi_\alpha^\pm(\cdot)$ and $\Psi_\iota^\pm(\cdot)$. In order to be able to derive an expression for, say ι_p^+, the conditional intersection enhancement function $\Psi_\iota^+(\cdot)$ provides an upper bound $\pi_0^{\iota_{pq}^+}$ on the mutual precision upon $\{\boldsymbol{S}_p, \boldsymbol{S}_q\}$ computed at node p, with q under the worst case accuracy setting for α'^+_p. Since $\boldsymbol{S}_p$ and $\boldsymbol{S}_q$ are both $\boldsymbol{\pi}_0$-correct, this implies that the mutual intersection $\hat{\boldsymbol{S}}_p \cap \hat{\boldsymbol{S}}_q$ for any q has length at least $\pi_0 - \max_q\{\pi_0^{\iota_{pq}^+}\}$.

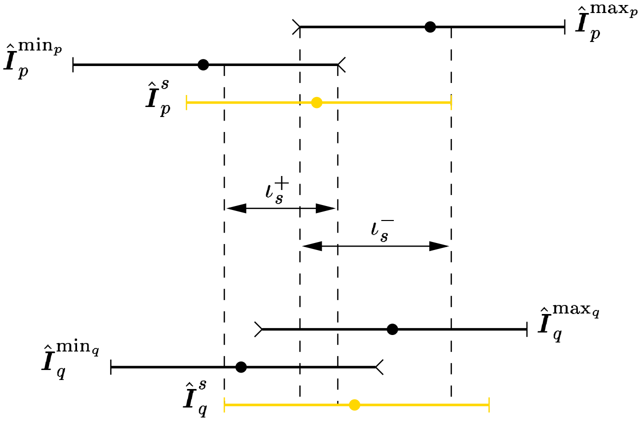

$\hat{\boldsymbol{I}}_p^{\max_p}$
$\hat{\boldsymbol{I}}_p^{\min_p}$
$\hat{\boldsymbol{I}}_p^s$
ι_s^+
ι_s^-
$\hat{\boldsymbol{I}}_q^{\max_q}$
$\hat{\boldsymbol{I}}_q^{\min_q}$
$\hat{\boldsymbol{I}}_q^s$

Figure 3.5: *Common Intersection*

Note that it is possible to provide additional arguments to any of the characteristic functions above, but we tried to keep them small for clarity. In particular, no global parameters are used, so that the behavior of the convergence function $\mathcal{CV}_\mathcal{F}(\cdot)$ can be studied for any non-faulty pair of nodes.

The following lemma describes the result of applying the generic convergence function to the intervals resulting from an FME as set forth by Lemma 3.11.

Lemma 3.12 (Application of Convergence Function) *Let* $\boldsymbol{A}_p = [T_p \pm \boldsymbol{\alpha}_p] = \boldsymbol{C}_p^{(k)}(t_p^{R,(k-1)})$ *be the accuracy interval of node* p*'s interval clock at real-time* $t_p = t_p^{R,(k-1)}$*, when round* k *(for some fixed* $k \geq 0$*) starts, and denote by* $\boldsymbol{\mathcal{A}}$ *the subset of the* $\boldsymbol{A}_p$*'s of those nodes* p *that remain non-faulty during round* k *w.r.t. a given fault model* $\mathcal{F}$*. Let* $T_q^R = (k+1)P_S + \Lambda + \Omega + \Delta + E_q$ *be the logical time when the* $(k+1)$*-th resynchronization instant —happening at real-time* $t_q^R = t_q^{R,(k)}$*— is scheduled at node* q*, and let* $\boldsymbol{I}_q^p = \boldsymbol{I}_q^p(t_q^R)$ *be the interval (3.19)/(3.20) that is obtained at node* q *as the result of delay and drift compensation of a node* p*'s accuracy interval transmitted during the FME in round* k*.*

Assume that the set $\boldsymbol{\mathcal{I}}_q$ *of intervals* $\boldsymbol{I}_q^p$ *available at node* q *is subsequently fed into a convergence function* $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ *characterized by the accuracy preservation functions* $\Phi_\alpha^\pm(\cdot)$*, precision preservation function* $\boldsymbol{\Phi}_\pi(\cdot)$*, precision enhancement function* $\Psi_\pi(\cdot)$*, and intersection enhancement functions* $\Psi_\iota^\pm(\cdot)$ *subject to fault model* $\mathcal{F}$*.*

If, at the beginning of round k*,*

[1] *the accuracies of any* $\boldsymbol{A}_p = [T_p \pm \boldsymbol{\alpha}_p] \in \boldsymbol{\mathcal{A}}$ *are integer multiples of* G_S *bounded according to*

$$\boldsymbol{\alpha}_p \subseteq \boldsymbol{\beta}_p \in \boldsymbol{\mathcal{B}} \tag{3.50}$$

for a given set of accuracy bounds $\boldsymbol{\mathcal{B}} = \{\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_n\}$*,*

[2] $\mathrm{shift}_{t'}(\boldsymbol{\mathcal{A}})$*,* $t' \geq \max_p\{t_p\}$ *arbitrary, is* $\boldsymbol{\pi}_0'$*-correct w.r.t. internal global time of round* k *for some* $\boldsymbol{\pi}_0' \subseteq \boldsymbol{\pi}_0$*, where* $\boldsymbol{\pi}_0$ *is a solution of the equation*

$$||\boldsymbol{\pi}_0|| = \Psi_\pi(\boldsymbol{\mathcal{P}}, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I) \tag{3.51}$$

for the set $\boldsymbol{\mathcal{P}} = \{\boldsymbol{\pi}^1, \ldots, \boldsymbol{\pi}^n\}$ *of uniform precision bounds* $\boldsymbol{\pi}^p \subseteq \boldsymbol{\pi}^H$ *defined by*

$$\begin{aligned}
\boldsymbol{\pi}^p &= \boldsymbol{\pi}_0 + \boldsymbol{u}_p + \boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{\max} \\
&\quad + (P_S - \Delta - E_p)\boldsymbol{\rho}_p + (E_{\max} + \Delta - \delta_{\min})\boldsymbol{\rho}_{\max} \\
&\quad + (\Lambda + \Omega)\big[-(\rho_{\max}^- - \rho_p^-), \rho_{\max}^+ - \rho_p^+\big] \\
&\quad + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}, &(3.52)\\
\boldsymbol{\pi}^{\overline{q}} &= \boldsymbol{\pi}_0 + \boldsymbol{u}_q + P_S\boldsymbol{\rho}_q + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_q, &(3.53)\\
\boldsymbol{\pi}^H &= \boldsymbol{\pi}_0 + 2\boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{\max} + (P_S + E_{\max} - E_{\min} - \delta_{\min})\boldsymbol{\rho}_{\max} \\
&\quad + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}, &(3.54)\\
\boldsymbol{\pi}_I &= \boldsymbol{\varepsilon}_{\max} + H\boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + (\Lambda + \Omega + \Delta + E_{\max} - \delta_{\min})\boldsymbol{\rho}_{\max} \\
&\quad + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}, &(3.55)
\end{aligned}$$

[3] for any $s \in \{1, \dots, n\}$ with both $\boldsymbol{A}_p^s$ and $\boldsymbol{A}_q^s$ being non-faulty, the associated precision intervals satisfy

$$\begin{aligned} ||\hat{\boldsymbol{A}}_p^s \cap \hat{\boldsymbol{A}}_q^s \cap \hat{\boldsymbol{A}}_p^{\min_p} \cap \hat{\boldsymbol{A}}_q^{\min_q}|| &\geq \iota_s^+ \geq 0, & (3.56)\\ ||\hat{\boldsymbol{A}}_p^s \cap \hat{\boldsymbol{A}}_q^s \cap \hat{\boldsymbol{A}}_p^{\max_p} \cap \hat{\boldsymbol{A}}_q^{\max_q}|| &\geq \iota_s^- \geq 0, & (3.57)\end{aligned}$$

where $\iota_s^\pm$ are integer multiples of G_S, and $\min_x$ resp. $\max_x$ represents that non-faulty node that leads to the leftmost $\mathrm{right}(\hat{\boldsymbol{A}}_x^{\min_x})$ resp. the rightmost $\mathrm{left}(\hat{\boldsymbol{A}}_x^{\max_x})$ for $x \in \{p, q\}$,

[4] any $\boldsymbol{A}_p \in \boldsymbol{\mathcal{A}}$ satisfies $T_p \in [kP_S + \Lambda + \Omega + \Delta + E_p \pm \boldsymbol{\pi}']$ for $\boldsymbol{\pi}' \subseteq \boldsymbol{\pi}$ defined by

$$\begin{aligned} \boldsymbol{\pi} &= \overline{\boldsymbol{\Phi}_\pi}(\boldsymbol{\mathcal{P}}, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I) + \boldsymbol{\pi}_0 + \boldsymbol{u}_{\max} + P_S \boldsymbol{\rho}_{\max} \\ &\quad + \mathcal{O}(P_S \rho_{\max} + G + \varepsilon_{\max}) \boldsymbol{\rho}_{\max}, & (3.58)\end{aligned}$$

[5] broadcast delay $\Lambda + \Omega$, transmission delay Δ, computation time compensation E_p, and round period P_S are as defined in item [4] of Lemma 3.11,

then the set $\boldsymbol{\mathcal{S}}$ of intervals

$$\boldsymbol{S}_q = \boldsymbol{S}_q(t_q^R) = [T_q' \pm \boldsymbol{\alpha}_q'] = \boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\boldsymbol{\mathcal{I}}_q)$$

provided by the application of the convergence function $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ to the set $\boldsymbol{\mathcal{I}}_q$ of compatible intervals (3.19)/(3.20) resulting from the FME at a non-faulty node q satisfies:

(1) $\boldsymbol{S}_q$ is accurate with accuracies being integer multiples of G_S bounded according to

$$\boldsymbol{\alpha}_q' \subseteq \Big[-\Phi_\alpha^-(\boldsymbol{\mathcal{B}}_q, \boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I, \forall s : \iota_s^-), \Phi_\alpha^+(\boldsymbol{\mathcal{B}}_q, \boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I, \forall s : \iota_s^+)\Big],$$

where the set $\boldsymbol{\mathcal{B}}_q = \{\boldsymbol{\beta}_q^1, \dots, \boldsymbol{\beta}_q^n\}$ of node q's accuracy bounds is defined by

$$\begin{aligned} \boldsymbol{\beta}_q^p &= \boldsymbol{\beta}_p + \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \boldsymbol{2G}_A + \boldsymbol{\varepsilon}_{pq} \\ &\quad + (P_S - \Delta - E_p)\boldsymbol{\rho}_p + (E_q + \Delta - \delta_{pq})\boldsymbol{\rho}_q \\ &\quad + (\Lambda + \Omega)\Big[-\max\{\rho_q^- - \rho_p^-, 0\}, \max\{\rho_q^+ - \rho_p^+, 0\}\Big] \\ &\quad + \mathcal{O}(P_S \rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max} & (3.59)\end{aligned}$$

for $p \neq q$ and

$$\boldsymbol{\beta}_q^q = \boldsymbol{\beta}_q + \boldsymbol{u}_q + P_S \boldsymbol{\rho}_q + \mathcal{O}(P_S \rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_q \qquad (3.60)$$

with $\boldsymbol{\beta}_p \in \boldsymbol{\mathcal{B}}$*; the set* $\boldsymbol{\mathcal{P}}_q = \{\boldsymbol{\pi}_q^1, \ldots, \boldsymbol{\pi}_q^n\}$ *of node* q*'s precision bounds* $\boldsymbol{\pi}_q^p \subseteq \boldsymbol{\pi}^p \subseteq \boldsymbol{\pi}^H$ *is defined by*

$$\begin{aligned}\boldsymbol{\pi}_q^p &= \boldsymbol{\pi}_0 + \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{pq} + (P_S - \Delta - E_p)\boldsymbol{\rho}_p + (E_q + \Delta - \delta_{pq})\boldsymbol{\rho}_q \\ &\quad + (\Lambda + \Omega)\big[-\max\{\rho_q^- - \rho_p^-, 0\}, \max\{\rho_q^+ - \rho_p^+, 0\}\big] \\ &\quad + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max} \qquad (3.61)\end{aligned}$$

for $p \neq q$ *and*

$$\boldsymbol{\pi}_q^q = \boldsymbol{\pi}_0 + \boldsymbol{u}_q + P_S\boldsymbol{\rho}_q + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_q. \qquad (3.62)$$

(2) $\boldsymbol{S}_q$ *is* $\boldsymbol{\Phi}_\pi(\boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I)$*-correct w.r.t. internal global time of round* k*.*

(3) $\mathrm{shift}_{t'}(\boldsymbol{\mathcal{S}})$*,* $t' \geq \max_q\{t_q^R\}$ *arbitrary, is* $\boldsymbol{\pi}_0$*-correct w.r.t. the newly defined internal global time of round* $k+1$*.*

(4) The precision interval $\hat{\boldsymbol{S}}_q$ *associated with* $\mathrm{shift}_{t'}(\boldsymbol{S}_q) \in \mathrm{shift}_{t'}(\boldsymbol{\mathcal{S}})$*,* $t' \geq \max_q\{t_q^R\}$ *arbitrary, has a common intersection with any associated precision interval from* $\mathrm{shift}_{t'}(\boldsymbol{\mathcal{S}})$ *of length at least*

$$\begin{aligned}\iota_q'^- &= \pi_0 - \max_p\{\Psi_\iota^-(\boldsymbol{\mathcal{B}}_p, \boldsymbol{\mathcal{B}}_q, \boldsymbol{\mathcal{P}}_p, \boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I, \forall s : \iota_s^-)\} \geq 0 \qquad (3.63) \\ \iota_q'^+ &= \pi_0 - \max_p\{\Psi_\iota^+(\boldsymbol{\mathcal{B}}_p, \boldsymbol{\mathcal{B}}_q, \boldsymbol{\mathcal{P}}_p, \boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I, \forall s : \iota_s^+)\} \geq 0 \qquad (3.64)\end{aligned}$$

under the worst case setting for $\mathrm{acc}^{\mp}(\boldsymbol{S}_q)$*.*

(5) Two non-faulty nodes p*,* q *resynchronize within real-time* $t_p^R - t_q^R$ *satisfying*

$$\begin{aligned}t_p^R - t_q^R &\in E_p - E_q + [-\pi_0, \pi_0] + P_S(\boldsymbol{\rho}_p + \overline{\boldsymbol{\rho}}_q) + \boldsymbol{u}_p + \overline{\boldsymbol{u}}_q \\ &\quad + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})[-\rho_{\max}, \rho_{\max}] \qquad (3.65)\end{aligned}$$

for $\pi_0 = ||\boldsymbol{\pi}_0||$*.*

(6) The maximum clock adjustment Υ_q *applied to the clock of any non-faulty node* q *satisfies* $\Upsilon_q \in \boldsymbol{\pi}_q$ *for* $\boldsymbol{\pi}_q \subseteq \boldsymbol{\pi}$ *defined by*

$$\begin{aligned}\boldsymbol{\pi}_q &= \overline{\boldsymbol{\Phi}_\pi}(\boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I) + \boldsymbol{\pi}_0 + P_S\boldsymbol{\rho}_q + \boldsymbol{u}_q \\ &\quad + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}, \qquad (3.66)\end{aligned}$$

hence $T_q' \in [(k+1)P_S + \Lambda + \Omega + \Delta + E_q \pm \boldsymbol{\pi}_q]$ *for any* $\boldsymbol{S}_q \in \boldsymbol{\mathcal{S}}$*.*

Proof. First of all, we establish some coarse "a priori" bounds on the various precision values given in precondition [2] of our lemma, which will be required for applying Lemma 3.11: Since $\boldsymbol{\pi}_0$ satisfies (3.51), it follows from $\Psi_\pi(\cdot) < \pi^H$ according to item (3) of Definition 3.11 that $\pi^H > \pi_0 = C\pi^H$ for $C < 1$. Therefore, (3.54) reveals $(1 - C)\pi^H = \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})$, so that this remainder term applies for π^H and π_0 as well. Moreover, plugging in $||\boldsymbol{\Phi}_\pi(\boldsymbol{\mathcal{P}}_p, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I)|| \leq ||\boldsymbol{\Phi}_\pi(\boldsymbol{\mathcal{P}}, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I)|| = \mathcal{O}(\pi^H)$ —as justified by weak monotonicity of $\boldsymbol{\Phi}_\pi(\cdot)$ and the bound from item (2) of Definition 3.11— into (3.58) resp. (3.66) reveals that $\pi_p \leq \pi = \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})$ for any p as well.

Now, since our preconditions are the same ones as required by Lemma 3.11, it follows from (3.29) resp. (3.31) that the accuracy of a (non-faulty) interval $\boldsymbol{I}_q^p \in \boldsymbol{\mathcal{N}}_q \subseteq \boldsymbol{\mathcal{I}}_q$ is bounded according to (3.59) resp. (3.60), which defines the set of accuracy bounds $\boldsymbol{\mathcal{B}}_q$ required for $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$. Similarly, we know from (3.32) resp. (3.33) that any $\boldsymbol{I}_q^p \in \boldsymbol{\mathcal{N}}_q$ is $\boldsymbol{\pi}_q^{p\prime}$-correct for $\boldsymbol{\pi}_q^{p\prime} \subseteq \boldsymbol{\pi}_q^p$ given by (3.61) resp. (3.62). In addition, (3.35) implies that any $\boldsymbol{I}_q^p \in \boldsymbol{\mathcal{N}}$ is $\boldsymbol{\pi}^{H\prime}$-correct with $\boldsymbol{\pi}^{H\prime} \subseteq \boldsymbol{\pi}^H$ defined in (3.54), and (3.36) establishes that any $\boldsymbol{I}_q^p \in \boldsymbol{\mathcal{N}}^p$ is $\boldsymbol{\pi}^{p\prime}$-correct for $\boldsymbol{\pi}^{p\prime} \subseteq \boldsymbol{\pi}^p$ given by (3.52); note that $\mathcal{O}(\pi') \leq \mathcal{O}(\pi) = \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})$, as shown above. Moreover, from (3.37) it follows that $\boldsymbol{\mathcal{N}}^p$ is also $\boldsymbol{\pi}_I'$-precise with $\boldsymbol{\pi}_I' \subseteq \boldsymbol{\pi}_I$ given by (3.55); again, $\mathcal{O}(\pi_0') \leq \mathcal{O}(\pi_0) = \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})$.

Therefore, we have established bounds on all the arguments of the characteristic functions of $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$. Hence, the statements asserted in item (1) and (2) of our lemma follow immediately from Definition 3.11 item (1) and (2), respectively. Note that weak monotonicity of $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ and $\Phi_\alpha^\pm(\cdot)$ is required here to carry over bounds on the source intervals to bounds on the result.

Since (3.51) in conjunction with item (3) of Definition 3.11 implies that $\text{shift}_{t'}(\boldsymbol{\mathcal{S}})$, $t' \geq \max_p\{t_p^R\}$ arbitrary, is $\boldsymbol{\pi}_0$-precise, we can define internal global time $\tau^{(k+1)} = \tau^{(k+1)}(t')$ for the new round $k + 1$ to be an arbitrary point that lies in the intersection of the intervals in $\text{shift}_{t'}(\boldsymbol{\mathcal{S}})$ (recall Definition 3.9), so that this set is actually $\boldsymbol{\pi}_0$-correct w.r.t. $\tau^{(k+1)}$, as asserted in item (3) of the lemma.

To understand item (4) of the lemma, consider the $\boldsymbol{\pi}_0'$-precision intervals drawn from set $\text{shift}_{t''}(\boldsymbol{\mathcal{A}})$, $t'' \geq \max_p\{t_p\}$. Precondition [3] states that the length of the common intersection $\hat{\boldsymbol{A}}_p^s \cap \hat{\boldsymbol{A}}_q^s \cap \hat{\boldsymbol{A}}_p^{\min_p} \cap \hat{\boldsymbol{A}}_q^{\min_q}$ is at least $\iota_s^+ \geq 0$, for all $1 \leq s \leq n$. Since the drift and delay compensation operations used to obtain the $\boldsymbol{I}_q^s$ from $\boldsymbol{A}_s$ according to (3.19)/(3.20) have been explicitly designed to preserve accurateness, the precision intervals associated with the $\boldsymbol{I}_q^s$'s fed into the convergence function $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ have a common intersection of length at least ι_s^+ as well. Applying the conditional intersection enhancement function $\Psi_\iota^+(\cdot)$ from Definition 3.11 item (4), provides an upper bound $\pi_0^{\iota_{qp}^+}$ on the precision of the set $\text{shift}_{t'}(\{\boldsymbol{S}_q, \boldsymbol{S}_p\})$, $t' \geq \max_p\{t_p^R\}$ arbitrary, for the worst case setting w.r.t. $\text{acc}^+(\boldsymbol{S}_q)$.

This implies a lower bound $\iota_{qp}^{+} = \pi_0 - \pi_0^{\iota_{qp}^{+}}$ on the length of the mutual intersection of $\hat{\boldsymbol{S}}_q \cap \hat{\boldsymbol{S}}_p$, since $\mathrm{shift}_{t'}(\boldsymbol{\mathcal{S}})$ is $\boldsymbol{\pi}_0$-correct according to item (3). To confirm (3.64), we just take the minimum over all p, which gives a lower bound $\iota_q^{+} = \pi_0 - \max_p\{\pi_0^{\iota_{qp}^{+}}\}$ on that mutual intersection for arbitrary nodes p (including $p = \min_x$). The same line of reasoning holds for the worst case setting w.r.t. $\mathrm{acc}^{-}(\boldsymbol{S}_q)$, eventually proving ι_q^{-} in (3.63).

To prove the statement of item (5), we recall that node p resynchronizes at real-time t_p^R when the virtual clock $C_p^{(k)}$ used in round k displays $T_p^R = (k+1)P_S + \Lambda + \Omega + \Delta + E_p$. Since round k started when $C_p^{(k)}$ read $T_p \geq kP_S + \Lambda + \Omega + \Delta + E_p - \pi^{-}$ according to precondition [3] of our lemma, our usual argument provides that the initial precision $\boldsymbol{\pi}_0$ has deteriorated to

$$\begin{aligned} \boldsymbol{\pi}_{o,p} &= \boldsymbol{\pi}_0 + (T_p^R - T_p)\boldsymbol{\rho}_p + \boldsymbol{u}_p \subseteq \boldsymbol{\pi}_0 + \boldsymbol{u}_p + (P_S + \pi^{-})\boldsymbol{\rho}_p \\ &\subseteq \boldsymbol{\pi}_0 + \boldsymbol{u}_p + P_S\boldsymbol{\rho}_p + \mathcal{O}(\pi)\boldsymbol{\rho}_{\max} \end{aligned} \tag{3.67}$$

when t_p^R is reached; that is, the virtual clock $\boldsymbol{C}_p^{(k)}(t_p^R) = [T_p^R \pm \boldsymbol{\alpha}_p^R]$ of a non-faulty node p is $\boldsymbol{\pi}_{o,p}$-correct. Therefore, Lemma 3.6 immediately provides

$$t_p^R - t_q^R \in T_p^R - T_q^R + \boldsymbol{\pi}_{o,p} + \overline{\boldsymbol{\pi}}_{o,q}$$

for all non-faulty nodes p and q. Plugging in the definition of the logical resynchronization times T_p^R and T_q^R and (3.67) while recalling $\boldsymbol{\pi} + \overline{\boldsymbol{\pi}} = [-\pi, \pi]$ for any $\boldsymbol{\pi}$, item (5) of our lemma follows.

Turning our attention to item (6), we recall that the new virtual clock $\boldsymbol{C}_q^{(k+1)}(t_q^R)$ at node q is initialized to $\boldsymbol{S}_q = [T_q' \pm \boldsymbol{\alpha}_q']$, so that it is $\boldsymbol{\Phi}_\pi(\boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I)$-correct w.r.t. $\tau^{(k)}$ by virtue of item (2) of our lemma. On the other hand, the virtual clock $\boldsymbol{C}_q^{(k)}(t_q^R)$ was established above to be $\boldsymbol{\pi}_{o,q}$-correct w.r.t. $\tau^{(k)}$ according to (3.67). Lemma 3.7 thus yields

$$\Upsilon_q = T_q' - T_q^R \in \boldsymbol{\pi}_{o,q} + \overline{\boldsymbol{\Phi}_\pi}(\boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I). \tag{3.68}$$

Using $\boldsymbol{\Phi}_\pi(\boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I) \subseteq \boldsymbol{\Phi}_\pi(\boldsymbol{\mathcal{P}}, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I)$ due to weak monotonicity, (3.66) and also $\boldsymbol{\pi}_q \subseteq \boldsymbol{\pi}$ follows, which in turn justifies our choice of (3.58). The bound on T_q' given in item (6) is an immediate consequence of $\Upsilon_q = T_q' - T_q^R$, eventually completing our proof. $\square$

By virtue of the Lemma 3.12, it is not difficult to present the concluding Theorem 3.1 about clock state synchronization by means of instantaneous correction. Before that, we are motivating the notion of *traditional accuracy* $\aleph$, which gives the amount local time may drift from real-time during a given time interval Δt. Although worst case bounds $\boldsymbol{\beta}$ on accuracy intervals obviously provide an upper bound on traditional accuracy because of $\aleph \in \overline{\boldsymbol{\beta}}$, it is favorable to determine the latter explicitly and in a stronger way. Taking the limit $\Delta t \to \infty$, traditional accuracy leads to the rate (see forthcoming Chapter 4) of the synchronized clocks, which is more convenient for comparison. Note that such worst case bounds are available for most existing internal synchronization algorithms, see [30], [32], [58], [66], [8], [10], or [72]. Our invention of internal global time makes it easy to deal with traditional accuracy, since we only have to bound the maximum "jump" internal global time $\tau(t)$ can experience when switching from one round to the next. This is sufficient because internal global time progresses as real-time does during a round, so that no additional deviation from real-time occurs in between synchronization instants.

Theorem 3.1 (Instantaneous Correction) *Running in a system complying to Assumptions 3.1–3.4, the clock synchronization algorithm of Definition 3.7 using the generic convergence function* $\mathcal{CV}_{\mathcal{F}}(\cdot)$ *—characterized by the accuracy preservation functions* $\Phi_\alpha^\pm(\cdot)$*, precision preservation function* $\boldsymbol{\Phi}_\pi(\cdot)$*, precision enhancement function* $\Psi_\pi(\cdot)$*, and intersection enhancement functions* $\Psi_\iota^\pm(\cdot)$ *subject to a given fault model* $\mathcal{F}$*— guarantees accuracy and precision for all rounds* $k \geq 0$ *as follows:*

(1) The accuracy interval $\boldsymbol{A}_q^{(k+1)} = \boldsymbol{A}_q^{(k+1)}(t_q^{R,(k)}) = [T_q^{(k+1)} \pm \boldsymbol{\alpha}_q^{(k+1)}]$ *provided by the local interval clock of a non-faulty node* q *at the beginning of round* $k+1$, $k \geq 0$, *satisfies* $\boldsymbol{\alpha}_q^{(k+1)} \subseteq \boldsymbol{\beta}_q^{(k+1)}$ *with*

$$\begin{aligned}
\boldsymbol{\beta}_q^{(k+1)} &= \Big[-\Phi_\alpha^-\left(\boldsymbol{\mathcal{B}}_q^{(k+1)}, \boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I, \forall s : \iota_s^{-,(k)}\right), \\
&\qquad\qquad \Phi_\alpha^+\left(\boldsymbol{\mathcal{B}}_q^{(k+1)}, \boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I, \forall s : \iota_s^{+,(k)}\right)\Big], && (3.69)\\
\boldsymbol{\beta}_q^{(0)} &= \boldsymbol{\alpha}_q^0, && (3.70)
\end{aligned}$$

where the set $\boldsymbol{\mathcal{B}}_q^{(k+1)} = \{\boldsymbol{\beta}_q^{1,(k+1)}, \ldots, \boldsymbol{\beta}_q^{n,(k+1)}\}$ *of node* q*'s accuracy bounds is defined by* $\boldsymbol{\beta}_q^{p,(k+1)} = \boldsymbol{\beta}_p^{(k)} + \boldsymbol{\zeta}_q^p$ *with*

$$\begin{aligned}
\boldsymbol{\zeta}_q^p &= \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \boldsymbol{2G}_A + \boldsymbol{\varepsilon}_{pq} \\
&\quad + (P_S - \Delta - E_p)\boldsymbol{\rho}_p + (E_q + \Delta - \delta_{pq})\boldsymbol{\rho}_q \\
&\quad + (\Lambda + \Omega)\Big[-\max\{\rho_q^- - \rho_p^-, 0\}, \max\{\rho_q^+ - \rho_p^+, 0\}\Big] \\
&\quad + \mathcal{O}(P_S \rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max} \qquad \text{for } p \neq q, && (3.71)\\
\boldsymbol{\zeta}_q^q &= \boldsymbol{u}_q + P_S \boldsymbol{\rho}_q + \mathcal{O}(P_S \rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_q, && (3.72)
\end{aligned}$$

the set $\boldsymbol{\mathcal{P}}_q = \{\boldsymbol{\pi}_q^1, \ldots, \boldsymbol{\pi}_q^n\}$ *of node* q*'s precision bounds* $\boldsymbol{\pi}_q^p \subseteq \boldsymbol{\pi}^p \subseteq \boldsymbol{\pi}^H$ *—see* *item (2)— is defined by*

$$\begin{aligned}\boldsymbol{\pi}_q^p &= \boldsymbol{\pi}_0 + \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{pq} + (P_S - \Delta - E_p)\boldsymbol{\rho}_p + (E_q + \Delta - \delta_{pq})\boldsymbol{\rho}_q \\ &\quad + (\Lambda + \Omega)\big[-\max\{\rho_q^- - \rho_p^-, 0\}, \max\{\rho_q^+ - \rho_p^+, 0\}\big] \\ &\quad + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max} \qquad \textit{for } p \neq q, && (3.73) \\ \boldsymbol{\pi}_q^q &= \boldsymbol{\pi}_0 + \boldsymbol{u}_q + P_S\boldsymbol{\rho}_q + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_q, && (3.74)\end{aligned}$$

and the minimum lengths on the common intersections are

$$\begin{aligned}\iota_q^{-,(k+1)} &= \pi_0 - \max_p\{\Psi_\iota^-(\boldsymbol{\mathcal{B}}_p^{(k+1)}, \boldsymbol{\mathcal{B}}_q^{(k+1)}, \boldsymbol{\mathcal{P}}_p, \boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I, \forall s : \iota_s^{-,(k)})\} && (3.75) \\ &\geq 0, \\ \iota_q^{+,(k+1)} &= \pi_0 - \max_p\{\Psi_\iota^+(\boldsymbol{\mathcal{B}}_p^{(k+1)}, \boldsymbol{\mathcal{B}}_q^{(k+1)}, \boldsymbol{\mathcal{P}}_p, \boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I, \forall s : \iota_s^{+,(k)})\} && (3.76) \\ &\geq 0, \\ \iota_q^{\pm,(0)} &= \pi_0 - \max_p\{\alpha_p^0\} \geq 0. && (3.77)\end{aligned}$$

(2) *The interval clocks of non-faulty nodes are synchronized to the (observable) initial worst case precision, i.e., the precision at the beginning of each round of the last non-faulty clock,*

$$\begin{aligned}\pi_{0,\max} &= \pi_0 + u_{\max} + G + (E_{\max} - E_{\min})\rho_{\max} \\ &\quad + \mathcal{O}(P_S\rho_{\max}^2 + G\rho_{\max} + \varepsilon_{\max}\rho_{\max}) && (3.78)\end{aligned}$$

with $\pi_0 = ||\boldsymbol{\pi}_0||$*, where* $\boldsymbol{\pi}_0$ *is a solution of the equation*

$$||\boldsymbol{\pi}_0|| = \Psi_\pi(\boldsymbol{\mathcal{P}}, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I) \tag{3.79}$$

for the set $\boldsymbol{\mathcal{P}} = \{\boldsymbol{\pi}^1, \ldots, \boldsymbol{\pi}^n\}$ *of uniform precision bounds* $\boldsymbol{\pi}^p \subseteq \boldsymbol{\pi}^H$ *defined by*

$$\begin{aligned}\boldsymbol{\pi}^p &= \boldsymbol{\pi}_0 + \boldsymbol{u}_p + \boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{\max} \\ &\quad + (P_S - \Delta - E_p)\boldsymbol{\rho}_p + (E_{\max} + \Delta - \delta_{\min})\boldsymbol{\rho}_{\max} \\ &\quad + (\Lambda + \Omega)\big[-(\rho_{\max}^- - \rho_p^-), \rho_{\max}^+ - \rho_p^+\big] \\ &\quad + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}, && (3.80) \\ \boldsymbol{\pi}^{\overline{q}} &= \boldsymbol{\pi}_0 + \boldsymbol{u}_q + P_S\boldsymbol{\rho}_q + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_q && (3.81) \\ \boldsymbol{\pi}^H &= \boldsymbol{\pi}_0 + 2\boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{\max} + (P_S + E_{\max} - E_{\min} - \delta_{\min})\boldsymbol{\rho}_{\max} \\ &\quad + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}, && (3.82) \\ \boldsymbol{\pi}_I &= \boldsymbol{\varepsilon}_{\max} + H\boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + (\Lambda + \Omega + \Delta + E_{\max} - \delta_{\min})\boldsymbol{\rho}_{\max} \\ &\quad + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}, && (3.83)\end{aligned}$$

where $\boldsymbol{\pi}^{\overline{q}} \subseteq \boldsymbol{\pi}^{q}$ *denotes node* q*'s own precision bound.*

(3) *The (observable) worst case precision* $\pi_{\max}$ *satisfies*

$$\begin{aligned}\pi_{\max} \;=\; & \max\Big\{\pi^- + u^+_{\max} + (E_{\max} - E_{\min})\rho^+_{\max},\\ & \qquad \pi^+ + u^-_{\max} + (E_{\max} - E_{\min})\rho^-_{\max},\\ & \qquad \pi_0 + u_{\max} + P_S\rho_{\max}\Big\}\\ & + G + \mathcal{O}(P_S\rho^2_{\max} + G\rho_{\max} + \varepsilon_{\max}\rho_{\max})\end{aligned} \tag{3.84}$$

with

$$\begin{aligned}\boldsymbol{\pi} \;=\; & \overline{\boldsymbol{\Phi}_\pi}(\boldsymbol{\mathcal{P}}, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I) + \boldsymbol{\pi}_0 + \boldsymbol{u}_{\max} + P_S\boldsymbol{\rho}_{\max}\\ & + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}.\end{aligned} \tag{3.85}$$

(4) *Resynchronization of any two non-faulty nodes* p*,* q *occurs within real-time* $t^R_p - t^R_q$ *satisfying*

$$\begin{aligned}t^R_p - t^R_q \;\in\; & E_p - E_q + [-\pi_0, \pi_0] + \boldsymbol{u}_p + \overline{\boldsymbol{u}}_q + P_S(\boldsymbol{\rho}_p + \overline{\boldsymbol{\rho}}_q)\\ & + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})[-\rho_{\max}, \rho_{\max}],\end{aligned} \tag{3.86}$$

where clock adjustments Υ_q *of at most* $\Upsilon_q \in \boldsymbol{\pi}_q \subseteq \boldsymbol{\pi}$ *defined by*

$$\begin{aligned}\boldsymbol{\pi}_q \;=\; & \overline{\boldsymbol{\Phi}_\pi}(\boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I) + \boldsymbol{\pi}_0 + \boldsymbol{u}_q + P_S\boldsymbol{\rho}_q\\ & + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}\end{aligned} \tag{3.87}$$

are applied to the clock of a non-faulty node q*.*

(5) *Let* $\boldsymbol{\Phi} = \bigcup_q \boldsymbol{\Phi}_\pi(\boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I) \subseteq \boldsymbol{\Phi}_\pi(\boldsymbol{\mathcal{P}}, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I)$*. For any round* $k \geq 0$*, the traditional accuracy* $\aleph^{(k+1)}$ *at the beginning of round* $k+1$ *satisfies*

$$\aleph^{(k+1)} = T^{(k+1)}_q - t^{R,(k)}_q \in \overline{\boldsymbol{\pi}}_0 + (k+1)(\overline{\boldsymbol{\Phi}} - \overline{\boldsymbol{\pi}}_0). \tag{3.88}$$

The inverse rate $r^{-1}_{q,syn}$ *of the synchronized clock at any node* q *evaluates to*

$$r^{-1}_{q,syn} = \lim_{k\to\infty} \frac{t^{R,(k)}_q - t^0_q}{T^{(k+1)}_q - T^0_q} \in \left[1 \pm \frac{\boldsymbol{\Phi} - \boldsymbol{\pi}_0}{P_S}\right], \tag{3.89}$$

where $T^0_q = C_q(t^0_q)$ *is node* q*'s local time at the beginning of round* $k = 0$*.*

Proof. The above results are established by carrying out an induction proof on the round k: Assuming that the accuracy intervals $\boldsymbol{A}_p = [T_p \pm \boldsymbol{\alpha}_p] = \boldsymbol{C}_p^{(k)}(t_p^{R,(k-1)})$ of all non-faulty nodes p are $\boldsymbol{\pi}_0$-correct at the beginning of round k, in the sense that

- $\mathrm{shift}_{t'}(\boldsymbol{\mathcal{A}})$, $t' \geq \max_p\{t_p^{R,(k-1)}\}$ arbitrary, is $\boldsymbol{\pi}_0$-correct,
- $T_p \in [kP_S + \Lambda + \Omega + \Delta + E_p \pm \boldsymbol{\pi}]$, and
- $\boldsymbol{\alpha}_p \subseteq \boldsymbol{\beta}_p^{(k)}$,

we show that the accuracy intervals $\boldsymbol{A}_p' = [T_p' \pm \boldsymbol{\alpha}_p'] = \boldsymbol{C}_p^{(k+1)}(t_p^{R,(k)})$ at the beginning of the succeeding round $k+1$ satisfy these precision properties and accuracy bounds as well. As a matter of fact, most results stated in our theorem follow directly from Lemma 3.12.

As far as item (1) is concerned, the recursively defined bound (3.69) follows directly from item (1) of Lemma 3.12, where we introduced the additional abbreviation $\boldsymbol{\zeta}_q^p$ in (3.71)/(3.72) since this expression is independent on round k. The lower bounds on the appropriate common intersections are carried over according to item (4) of Lemma 3.12, so that (3.75)/(3.76) follow as well. Similarly, item (4) of our theorem just combines item (5) and (6) of Lemma 3.12.

The initial case $k = 0$ is immediately implied by the initial synchronization assumption in the algorithm's Definition 3.7. More specifically, the initial accuracy $\boldsymbol{\alpha}_q^0$ of node q at the beginning of round 0 readily confirms (3.70) and (3.77).

It only remains to derive the expressions for the observable precisions $\pi_{0,\max}$ and $\pi_{\max}$ given in item (2) and (3) of our theorem. Let $\boldsymbol{\mathcal{C}}$ be a certain set of just resynchronized non-faulty virtual interval clocks $\boldsymbol{C}_p^{(k+1)}(t_p^R)$ at their respective resynchronization times $t_p^R = t_p^{R,(k)}$. From item (5) of Lemma 3.12 in conjunction with $\pi_0 = \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})$ established in its proof, it follows that $t_p^R - t_q^R \leq E_{\max} - E_{\min} + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})$ for any two non-faulty nodes p and q, so that item (2) of Lemma 3.10 provides us with the precision $\pi_{x,\max}$ of the clocks in $\boldsymbol{\mathcal{C}}$ at real-time $t_l^R = \max_p\{t_p^R\}$, when the last non-faulty node in the system resynchronizes: Provided that any $\boldsymbol{C}_p^{(k+1)}(t_p^R) \in \boldsymbol{\mathcal{C}}$ is $\boldsymbol{\pi}_x$-correct, we obtain

$$\begin{aligned}
\boldsymbol{\pi}_{x,\max} &\subseteq \boldsymbol{\pi}_x + \bigcup_p \left(\frac{t_l^R - t_p^R + u_p^-}{1 - \rho_p^-} \boldsymbol{\rho}_p + \boldsymbol{u}_p \right) + \boldsymbol{G}_{\boldsymbol{\rho}} \\
&\subseteq \boldsymbol{\pi}_x + (E_{\max} - E_{\min})\boldsymbol{\rho}_{\max} + \boldsymbol{u}_{\max} + \boldsymbol{G} \\
&\quad + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max} \qquad (3.90)
\end{aligned}$$

by recalling $u_{\max} = \mathcal{O}(G)$ from Assumption 3.2 and the definition of $\boldsymbol{G}_{\boldsymbol{\rho}}$ in (3.13).

To establish $\pi_{0,\max}$, we have to consider the set $\boldsymbol{\mathcal{C}}$ of all non-faulty virtual clocks $\boldsymbol{C}_p^{(k+1)}(t_p^R)$. Since $\mathrm{shift}_{t_l^R}(\boldsymbol{\mathcal{C}})$ is $\boldsymbol{\pi}_0$-correct w.r.t. internal global time $\tau^{(k+1)}$ due to item (3) of Lemma 3.12, we have to plug in $\boldsymbol{\pi}_x = \boldsymbol{\pi}_0$ in (3.90), which provides (3.78).

Before we can attack overall precision $\pi_{\max}$, we need some technical preparations. Consider some t_0, $t_1 \geq t_0$ being in synchrony with a node's clock $C(t)$, and denote by $\boldsymbol{I}'(t)$ the interval obtained by drift compensation of a $\boldsymbol{\pi}_0$-correct (initial) interval $\boldsymbol{I}_0 = \boldsymbol{I}(t_0)$ dragged from t_0 to t at that node. Then, it is not difficult to see that $\boldsymbol{I}'(t)$ for any t within $t_0 \leq t \leq t_1$ is $\boldsymbol{\pi}$-correct for

$$\boldsymbol{\pi} = \boldsymbol{\pi}_0 + (T_1 - T_0)\boldsymbol{\rho} + \boldsymbol{u} + \boldsymbol{G}_{\boldsymbol{\rho}}, \tag{3.91}$$

where $T_0 = C(t_0)$ and $T_1 = C(t_1)$. In fact, monotonicity of $C(t)$ implies $T_1 \geq T$ so that (3.91) follows immediately from (3.12) for $t_1 = t$. However, note the subtle fact that we cannot always infer $\boldsymbol{\pi}$-correctness of $\boldsymbol{I}'(t)$ for any $t \leq t_1$ from $\boldsymbol{\pi}$-correctness of $\boldsymbol{I}'(t_1)$. Whereas this is justified when t_1 is not in synchrony with $C(t)$, we must explicitly account for clock granularity (which amounts to adding $\boldsymbol{G}_{\boldsymbol{\rho}}$) in case that t_1 is synchronous.

Returning to our problem of determining the maximum observable overall precision, we know from (3.67) that the virtual interval clock $\boldsymbol{C}_p^{(k)}(t_p^R) = [T_p^R \pm \boldsymbol{\alpha}_p^R]$ of a non-faulty node p is $\boldsymbol{\pi}_{o,p}$-correct at t_p^R. Therefore, by using majorization in (3.67) and recalling (3.91), it follows that $\boldsymbol{C}_p^{(k)}(t)$ of any non-faulty node p, for any $t \leq t_p^R$ in round k, is $\boldsymbol{\pi}_{o,\cdot}$-correct with

$$\boldsymbol{\pi}_{o,\cdot} = \boldsymbol{\pi}_0 + P_S\boldsymbol{\rho}_{\max} + \boldsymbol{u}_{\max} + \boldsymbol{G}_{\boldsymbol{\rho}} + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}. \tag{3.92}$$

Assuming that the last resynchronizing node l in round k actually attains this maximum, we now consider the set $\boldsymbol{\mathcal{C}}$ containing just a single virtual clock $\boldsymbol{C}_p^{(k+1)}(t_p^R)$, of any non-faulty node $p \neq l$. From item (2) of Lemma 3.12 and weak monotonicity of $\boldsymbol{\Phi}_\pi(\cdot)$, we know that $\boldsymbol{C}_p^{(k+1)}(t_p^R)$ is $\boldsymbol{\Phi}_\pi(\boldsymbol{\mathcal{P}}, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I)$-correct w.r.t. internal global time of round k. Plugging this into (3.90) provides the observable precision

$$\begin{aligned} \boldsymbol{\pi}^M &= \boldsymbol{\Phi}_\pi(\boldsymbol{\mathcal{P}}, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I) + (E_{\max} - E_{\min})\boldsymbol{\rho}_{\max} + \boldsymbol{u}_{\max} + \boldsymbol{G} \\ &\quad + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max} \end{aligned} \tag{3.93}$$

of $\boldsymbol{C}_p^{(k+1)}(t_l^R)$ at time t_l^R, when the last node l is about to resynchronize. Of course, by the above reasoning, this precision is in fact valid for $\boldsymbol{C}_p^{(k+1)}(t)$ for any t satisfying $t_p^R \leq t \leq t_l^R$ as well, since granularity compensation $\boldsymbol{G}_{\boldsymbol{\rho}}$ is already incorporated in $\boldsymbol{\pi}^M$.

Applying Lemma 3.7 to $\boldsymbol{C}_l^{(k)}(t)$ and $\boldsymbol{C}_p^{(k+1)}(t)$ while using (3.92) and (3.93) reveals that the distance $\Upsilon(t) = \mathrm{ref}\big(\boldsymbol{C}_p^{(k+1)}(t)\big) - \mathrm{ref}\big(\boldsymbol{C}_l^{(k)}(t)\big)$ for any t with $t_p^R \leq t \leq t_l^R$ is bounded by

$$
\begin{aligned}
\Upsilon(t) &\in \boldsymbol{\pi}_{o,\cdot} + \overline{\boldsymbol{\pi}}_M \\
&\subseteq \boldsymbol{\pi}_0 + P_S \boldsymbol{\rho}_{\max} + \boldsymbol{u}_{\max} + \boldsymbol{G} + \overline{\boldsymbol{\Phi}_\pi}(\boldsymbol{\mathcal{P}}, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I) \\
&\quad + (E_{\max} - E_{\min}) \overline{\boldsymbol{\rho}}_{\max} + \overline{\boldsymbol{u}}_{\max} + \overline{\boldsymbol{G}} \\
&\quad + \mathcal{O}(P_S \rho_{\max} + G + \varepsilon_{\max})(\boldsymbol{\rho}_{\max} + \overline{\boldsymbol{\rho}}_{\max}) \\
&= \boldsymbol{\pi} + (E_{\max} - E_{\min}) \overline{\boldsymbol{\rho}}_{\max} + \overline{\boldsymbol{u}}_{\max} + [-G, G] \\
&\quad + \mathcal{O}(P_S \rho_{\max} + G + \varepsilon_{\max})[-\rho_{\max}, \rho_{\max}]
\end{aligned}
$$

by recalling the definition of $\boldsymbol{\pi}$ in (3.85).

Of course, the maximum of the positive and negative accuracies of the interval above gives the maximum observable precision for the "mixed" case, where virtual clocks of round k and $k+1$ are simultaneously alive. Thus, to determine $\pi_{\max}$, it only remains to find the maximum observable precision for the case where all nodes are still in round k, and to take the maximum of both cases. However, we have already established that $\boldsymbol{C}_p^{(k)}(t)$ and $\boldsymbol{C}_q^{(k)}(t)$ for any $t \leq \min\{t_p^R, t_q^R\}$ are at most $\boldsymbol{\pi}_{o,\cdot}$-correct, so that Lemma 3.7 in conjunction with (3.92) provides $\mho(t) = \mathrm{ref}\big(\boldsymbol{C}_p^k(t)\big) - \mathrm{ref}\big(\boldsymbol{C}_q^k(t)\big)$ bounded by

$$
\begin{aligned}
\mho(t) &\in [-\pi_0, \pi_0] + P_S[-\rho_{\max}, \rho_{\max}] + [-u_{\max}, u_{\max}] + [-G, G] \\
&\quad + \mathcal{O}(P_S \rho_{\max} + G + \varepsilon_{\max})[-\rho_{\max}, \rho_{\max}].
\end{aligned}
$$

Taking the maximum values of positive and negative accuracies of $\Upsilon(t)$ and $\mho(t)$ eventually provides (3.84).

Turning our attention to the last item, we know that $\boldsymbol{A}_q^{(k+1)}$ is $\boldsymbol{\Phi}_\pi(\boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I)$-correct w.r.t. $\tau^{(k)}$ by virtue of item (2) of Definition 3.11 and hence $\boldsymbol{\Phi}$-correct. Note that $\boldsymbol{\Phi} \subseteq \boldsymbol{\Phi}_\pi(\boldsymbol{\mathcal{P}}, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I)$ is a simple consequence of the weak monotonicity of $\boldsymbol{\Phi}_\pi(\cdot)$. Moreover, from item (3) of Definition 3.11, it follows that all $\boldsymbol{A}_p^{(k+1)}$ are $\boldsymbol{\pi}_0$-precise. Hence it is possible to choose $\tau^{(k+1)} \in \bigcap_p \hat{\boldsymbol{A}}_p^{(k+1)}$, as justified by Definition 3.2, and we claim that we may in fact choose $\tau^{(k+1)}$ such that

$$
-(\Phi^+ - \pi_0^+) \leq \tau^{(k+1)} - \tau^{(k)} \leq \Phi^- - \pi_0^-,
$$

which can also be written more elegantly as $\tau^{(k+1)} - \tau^{(k)} \in \overline{\boldsymbol{\Phi}} - \overline{\boldsymbol{\pi}}_0$. If this is not feasible, there would exist an interval $\hat{\boldsymbol{A}}_y^{(k+1)}$ of length π_0 with $\tau^{(k+1)} = \mathrm{left}(\hat{\boldsymbol{A}}_y^{(k+1)})$ or else $\tau^{(k+1)} = \mathrm{right}(\hat{\boldsymbol{A}}_y^{(k+1)})$ that satisfies $\tau^{(k)} \notin \hat{\boldsymbol{A}}_y^{(k+1)} + (\boldsymbol{\Phi} - \boldsymbol{\pi}_0) = \boldsymbol{\Phi}$. This, however, would contradict the $\boldsymbol{\Phi}$-correctness of $\boldsymbol{A}_y^{(k+1)}$.

The bound (3.88) is a simple consequence of the fact that internal global time progresses as real-time does during a round, so that the maximum deviation between internal global time and real-time remains the same during any round. Therefore, we just have to add up the worst case leap of internal global time at each round. The initial deviation is zero since choosing $\tau^{(0)}(t) = t$ is legitimate due to the initial synchronization assumption in Definition 3.7. Hence, to complete the proof of (3.88), it only remains to add the maximum deviation between $\tau^{(k+1)}$ and the reference point $T_q^{(k+1)}$ of $\boldsymbol{A}_q^{(k+1)}$, which is trivial since the latter is $\boldsymbol{\pi}_0$-correct.

To derive expression (3.89) for the rate of the synchronized clocks, we multiply (3.88) by (-1) to arrive at

$$(t_q^{R,(k)} - t_q^0) \in T_q^{(k+1)} - T_q^0 - (t_q^0 - T_q^0) + \boldsymbol{\pi}_0 + (k+1)\Big(\boldsymbol{\Phi} - \boldsymbol{\pi}_0\Big). \tag{3.94}$$

From the initial synchronization assumptions of Definition 3.7 we gather that $t_q^0 - T_q^0 \in \boldsymbol{\alpha}_q^0 \subseteq \boldsymbol{\pi}_0$. Moreover, from step (T) of the algorithm in conjunction with the fact that the maximum clock adjustment was shown to satisfy $\Upsilon_q \in \boldsymbol{\pi}$ in item (4) of this theorem, we obtain $T_q^{(k+1)} - T_q^0 = (k+1)P_S + \mathcal{O}(\pi)$. Plugging this into (3.94) yields

$$\frac{t_q^{R,(k)} - t_q^0}{T_q^{(k+1)} - T_q^0} \in 1 + \frac{[-\pi_0, \pi_0] + (k+1)\Big(\boldsymbol{\Phi} - \boldsymbol{\pi}_0\Big)}{(k+1)P_S + \mathcal{O}(\pi)}$$

and taking the limit for $k \to \infty$ eventually provides (3.89). This completes the proof of our theorem. □

The analysis of the accuracy bounds is done with both "halfs" $\Phi_\alpha^-(\cdot)$ and $\Phi_\alpha^+(\cdot)$, so that different worst case scenarios may be considered for the negative and positive accuracy. Note that in this case $\left|\left|\boldsymbol{\beta}_q^{(k+1)}\right|\right|$ is an overly conservative bound for the total length $\alpha_q^{-,(k+1)} + \alpha_q^{+,(k+1)}$, since they do not occur simultaneously.

3.5.3 Continuous Amortization Technique

Theorem 3.1 provides results for instantaneous correction, where the local interval clocks $\boldsymbol{C}_p^{(k)}(t_p^{R,(k)})$ are re-initialized to $\boldsymbol{C}_p^{k+1}(t_p^{R,k})$ at the end of round k. Since this simple approach could cause non-continuity and non-monotonicity of local time, applications usually demand some kind of continuous amortization. Such techniques are based on the idea of smoothing out the difference $C_p^{(k+1)}(t_p^{R,(k)}) - C_p^{(k)}(t_p^{R,(k)})$ by means of a suitable rate change. Continuous amortization has been studied in some detail in [58] and, in particular, in [56], where the non-interval based variant of our algorithm has been introduced and analyzed.

Adopting existing continuous amortization techniques to the interval-based framework involves intricate issues as already touched in our UTCSU specification in Chapter 2. First of all, continuous amortization only applies to the reference point $C_p(t)$ of a local interval clock

$$\boldsymbol{C}_p(t) = [L_p(t), U_p(t)] = [C_p(t) - \alpha_p^-(t), C_p(t) + \alpha_p^+(t)],$$

whereas the upper and lower envelope can be set instantaneously. However, since the envelopes are not maintained explicitly but rather implicitly via $\alpha_p^-(t)$ and $\alpha_p^+(t)$, it is necessary to compensate any change of $C_p(t)$ caused by continuous amortization by changing $\boldsymbol{\alpha}_p(t)$ appropriately.

Apart from that, it might also happen that instantaneous setting of lower or upper edge cause "negative" values of the accuracies α_p^- or α_p^+, since the reference point is not changed to its new value simultaneously with $\boldsymbol{\alpha}_p$. Of course, eventually, accuracies will become positive again, but one should ensure that applications read "negative" accuracy values as zero. Furthermore, the worst case accuracy bounds for the beginning of a round provided by Theorem 3.1 are not particularly meaningful anymore, except for the total length $\alpha_p = \alpha_p^- + \alpha_p^+$.

To cut the matter short, we just repeat the abstract specification and properties of our continuous amortization algorithm from [55]. Provided that the rate "boost" is chosen large enough, it turns out that continuous amortization does not impair worst cast precision and accuracy. Therefore, the following Theorem 3.2 imposes an upper bound on the length of the amortization period.

Definition 3.12 (Continuous Amortization Algorithm) *Setting node p's local interval clock $\boldsymbol{C}_p(t)$ displaying $[T \pm \boldsymbol{\alpha}]$ at real-time t to $[T' \pm \boldsymbol{\alpha}']$ (with $T' = T + \Upsilon$ being an integer multiple of G_S) is accomplished by*

(1) adjusting the (intrinsic) inverse rate r_p^{-1} of the local clock C_p to $(1 - \psi)r_p^{-1}$ for the next A clock ticks (according to the amortizing clock), where the amortization rate deviation *ψ and (local time) duration $A \cdot G$ of the* amortization period *are related by*

$$\psi = \frac{\Upsilon}{AG},$$

(2) instantaneously setting the clock's interval of accuracies to $\boldsymbol{\alpha}_p = \Upsilon + \boldsymbol{\alpha}'$, and modifying (intrinsic) deterioration, clock reading, etc. appropriately to keep away any effect of continuous amortization from the envelopes $L_p(t) = C_p(t) + \alpha_p^-(t)$ and $U_p(t) = C_p(t) + \alpha_p^+(t)$.

Theorem 3.2 (Continuous Amortization) *Given the clock synchronization algorithm of Definition 3.7 employing the continuous amortization algorithm of Definition 3.12 for setting the local interval clock in step (T), with*

[1] amortization rate deviation ψ_p of any non-faulty node p's clock satisfying

$$1 > |\psi_p| \geq \rho_{max},$$

[2] $P_S \geq \Lambda + \Omega + \Delta + E_{\max} + A_{\max} G$ with

$$A_{\max} \geq \left\lceil \frac{\max\{\pi^-, \pi^+\}}{G|\psi_{\min}|} \right\rceil, \tag{3.95}$$

where $|\psi_{\min}| = \min_p\{|\psi_p|\}$ and $\boldsymbol{\pi} = [-\pi^-, \pi^+]$ is the maximum adjustment applied to a non-faulty clock given by (3.85),

we obtain

(1) maximum precision $\pi^{\psi}_{\max}$ given by

$$\pi^{\psi}_{\max} = \pi_{\max} + \frac{\psi_{\max} u_{\max}}{(1 - \rho^{-}_{\max})(1 - \psi_{\max})} + \mathcal{O}\Big(G\rho_{\max}\Big), \tag{3.96}$$

for $\psi_{\max} = \max_p\{\psi_p\}$, where $\pi_{\max}$ defined in (3.84) is the maximum precision of the instantaneously corrected variant of the algorithm. The additional terms vanish in the remainder if $\psi_{\max} = \mathcal{O}(\rho_{\max})$.

(2) The accuracy interval $\boldsymbol{A}_q^{R,(k+1)} = [T_q^{R,(k+1)} \pm \boldsymbol{\alpha}_q^{R,(k+1)}]$ provided by the local interval clock of a non-faulty node q at the end of (and during) round $k+1$, $k \geq 0$, satisfies

$$\boldsymbol{\alpha}_q^{R,(k+1)} \subseteq \boldsymbol{\beta}_q^{(k+1)} + P_S \boldsymbol{\rho}_q + \boldsymbol{u}_q + \boldsymbol{G} + \mathcal{O}(P_S \rho_{\max} + G + \varepsilon_{\max}) \boldsymbol{\rho}_{\max},$$

where $\boldsymbol{\beta}_q^{(k+1)}$ is given by (3.69), which is the same bound as obtained for the instantaneously corrected variant of the algorithm.

3.6 Summary and Future Research

We introduced and rigorously analyzed a simple interval-based algorithm suitable for clock synchronization. Unlike usual internal synchronization approaches, our convergence function-based algorithm (dynamically) maintains both precision and accuracy w.r.t. an external time standard like UTC. To that end, each node keeps track of time by means of a local interval clock $\boldsymbol{C}(t)$, which is made up of an interval of accuracies taken relatively to the ordinary local clock's value. Our algorithm periodically exchanges local interval clock readings among all nodes in the system and employs an interval-valued convergence function to obtain and subsequently apply a clock correction value that enforces precision and accuracy. Clock correction can be done either instantaneously or by means of a certain continuous amortization algorithm.

The comprehensive analysis presented in the previous sections is generic w.r.t. the particular convergence function. It relies on a system model that considers many aspects usually abstracted away, like non-zero clock granularity, rate-adjustment uncertainties, and broadcast latencies. Technically, the analysis is based upon a novel, interval-based framework for providing worst case bounds for various parameters in terms of the characteristic functions of the convergence function. It manifests a striking conceptual beauty and expressive power, primarily by utilizing a suitable notion of internal global time. The results obtained include worst case bounds for initial and maximum precision, accuracy, maximum clock correction, and resynchronization tightness, for both instantaneous correction and linear continuous amortization. One of the surprising facts revealed by our analysis is the influence of rate adjustment uncertainties as well as the clock granularity.

Future theoretical research will be primarily devoted to the problem of integrating our algorithm(s) in the clock validation framework, which raises issues ranging from incorporating accuracy intervals from primary nodes up to suitable failure detectors for certain (unbounded accuracy) faults. Other subjects of interest include a more general investigation of the problem of optimality w.r.t. precision/accuracy besides the most challenging task of providing an average case analysis of precision and accuracy.

As far as practice is concerned, we are currently working on a fully engineered implementation (reviewed in Chapter 2), which will be mainly used for experimental evaluation. Besides confirming theoretical results experimentally, it will help us not to have overlooked important practical issues. Moreover, demonstrating the suitability of our concepts is mandatory for a certain industrial pilot application that could be carried out by using our approach.

◇

Chapter 4

INTERVAL-BASED CLOCK RATE SYNCHRONIZATION

4.1 Introduction

The concept of time allows us to reason about *events* and *durations*. For the purpose of quantification, designated *clocks* assign (real) numbers to them, so we are able to tell when a particular event occurs or how long a duration takes. Viewed from the other side, clocks can be characterized in two ways: An event draws a particular clock *state*, whereby a duration, measured as the difference between two corresponding clock states, enables us to determine a clock *rate* when compared against a reference clock. Unlike the states of clocks, however, the rates are not directly observable. In case of an ensemble of clocks, a simultaneous event could entail different clock states and a common duration different clock rates. This leads to the clock state/rate synchronization problem.

A clock $\mathcal{C}$ is regarded as an entity that reads clock state T at the non-directly observable real-time t in some meaningful Newtonian frame. In mathematical parlance, a clock can be modelled by a piecewise continuous function $C : t \mapsto T$. Ideally, $C(t)$ should be the identity function, but in a realistic system we have to deal with an approximation falling short of a perfect clock synchronization.

Commonly, clock synchronization is characterized in terms of clock states by two parameters: The maximum deviation between corresponding clock states and real-times on a single clock is called *accuracy* α, and the maximum clock state deviation between two different clocks in the distributed system at simultaneous real-times is called *precision* π. Maintaining accuracy resp. precision of an ensemble of clocks refers to the *external* resp. *internal* clock state synchronization problem.

Definition 4.1 (Clock Accuracy and Precision) *Let* $\mathcal{T}$ *be a non-empty (real-time) interval. A clock* $\mathcal{C}_p$ *has* accuracy α_p *during* $\mathcal{T}$ *iff* $|C_p(t) - t| \leq \alpha_p \; \forall t \in \mathcal{T}$. *Any two different clocks* $\mathcal{C}_p$ *and* $\mathcal{C}_q$ *have* precision π *during* $\mathcal{T}$ *iff* $|C_p(t) - C_q(t)| \leq \pi \; \forall t \in \mathcal{T}$.

However, clock synchronization can be viewed in terms of clock rates as well by considering the instantaneous clock rate $v(t) = dC(t)/dt$. Hence, we obtain another two parameters for characterization: The maximum deviation between the clock rate and the ideal rate 1 is called *drift*, and the maximum clock rate deviation between two different clocks in the distributed system at simultaneous real-times is called *consonance*. Maintaining drift resp. consonance of an ensemble of clocks refers to the *external* resp. *internal* clock rate synchronization problem.

Definition 4.2 (Clock Drift and Consonance) *Let $\mathcal{T}$ be a non-empty (real-time) interval. A clock $\mathcal{C}_p$ has* drift *δ_p during $\mathcal{T}$ iff $|v_p(t) - 1| \le \delta_p \ \forall t \in \mathcal{T}$. Any two different clocks $\mathcal{C}_p$ and $\mathcal{C}_q$ have* consonance *γ during $\mathcal{T}$ iff $|v_p(t) - v_q(t)| \le \gamma \ \forall t \in \mathcal{T}$.*

The significance of clock rate synchronization is motivated by the numerous distributed applications that are content with rate synchronized clocks, see [27]. Examples are algorithms based on timeouts (e.g., lifetime of Kerberos tickets [68]) or round-trips (e.g., NTP synchronization system [38]).

However, many applications require tight and robust state synchronized clocks, for instance distributed fault-tolerant real-time systems, see [24]. Most algorithms for clock state synchronization can be improved with an appropriate clock rate synchronization by achieving a better accuracy/precision or by saving communication bandwidth. Figure 4.1 helps to explain these improvements by regarding the precision development.

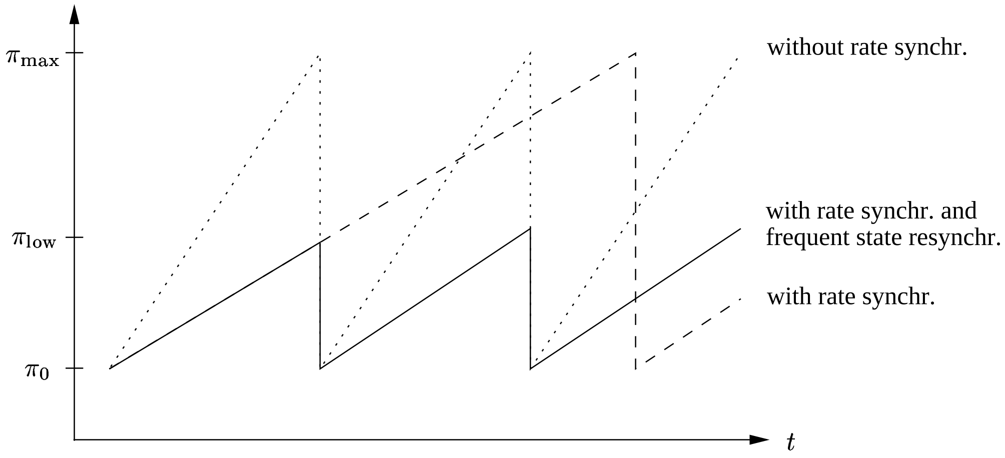


Figure 4.1: *Benefits of Rate Synchronized Clocks*

Let the dotted line represent the precision bound when no clock rate synchronization takes place, having a precision of at most π_0 immediately after state resynchronization instances and $\pi_{\max}$ just before it. If the clock rates are additionally synchronized beyond the usual manufacturer's drift specification, the precision deterioration of freely running clocks during consecutive state resynchronization instants can be reduced. This is shown by the dashed line with the same maximal precision, whereby the state resynchronization instants can occur less frequently (we estimate up to 10 times in the best case). On the other hand, adhering to the frequent state resynchronization instants leads to a lower worst case precision π_{low} as shown by the solid line. In order to achieve a highly accurate/precise state synchronization in the 1 μs-range as in [20], it is inevitable that clocks are additionally rate synchronized.

Last but not least, rate synchronization can be useful to achieve initial clock synchronization and or to implement fault detection mechanisms.

A vast material on clock state synchronization has been developed, see [65] for an overview or [58] for a generic paradigm, but not much research has been conducted on the problem of clock rate synchronization. The only important contribution is the work of [35], which includes a presentation of an interval-based algorithm for constant-rate clocks combined with a round-trip method for rate measurement. We extend those ideas, resulting in a framework for algorithms that deal with both drift and consonance on top of a realistic system model. In particular, a specification about the maximal rate change of a clock is taken into account and suitable intervals are used to capture the relevant rate information. The analysis is generic w.r.t. the employed convergence function, making our framework amenable to different failure assumptions. It makes use of the analysis techniques introduced in Chapter 3, which documents again their power and beauty.

This chapter is organized as follows: Section 4.2 introduces our system model concerning clocks and processors along with their means of communication. After a comprehensive preparatory work on notations and building blocks for rate synchronization in Section 4.3, the following Section 4.4 develops the theory of algorithms apt for both internal and external clock rate synchronization. Section 4.5 provides the analysis of consonance and drift in terms of properties of the employed convergence function. Section 4.6 closes this chapter by giving a summary and pointing out further research issues.

4.2 System Modelling

We assume a distributed system of $n \geq 2$ *nodes* connected by a *communication subsystem*, where each node p hosts a *processor* and a *clock* $\mathcal{C}_p$ driven by an *oscillator* $\mathcal{O}_p$. Figure 4.2 illustrates the components involved in steering a local clock, where a *clock state algorithm* (CSA) is concerned with accuracy/precision and a *clock rate algorithm* (CRA) with drift/consonance.

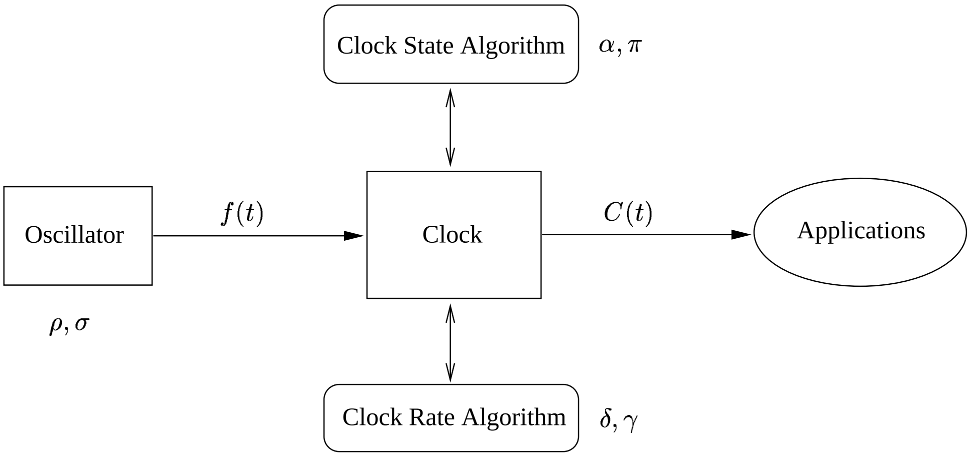


Figure 4.2: *Clock Steering*

4.2.1 Oscillator

The oscillator $\mathcal{O}_p$ indicates the progress of time with periodic ticks of *nominal frequency* f_p in Hz. Typically, the manufacturer specifies an *initial oscillator drift* ρ_p in ppm for shipment, hence the *instantaneous frequency* $f_p(t_0)$ at t_0 lies between $f_p(1-\rho_p)$ and $f_p(1+\rho_p)$. During operation the instantaneous frequency may change due to environmental influences and aging of the oscillator, so that a *maximal oscillator drift* $\rho_p^{\max}$ can be asserted, which is referred as “hardware drift rate” in the clock synchronization literature.

For clock rate synchronization it is crucial that the instantaneous frequency $f_p(t)$ does not alter too rapidly, otherwise rate resynchronization will not be effective. Fortunately, experiments have shown that oscillators keep up their frequency to some extent, see [69], [71], or [48]. Therefore, we are seeking for a dynamic characterization of $f_p(t)$ including deterministic influences (e.g., temperature, aging, pressure, humidity, magnetism, shock, power supply change, load impedance, radiation) and stochastic ones (e.g., white noise, flicker noise, random walk), see [1].

For our purpose, we capture the frequency variation as follows: Suppose oscillator $\mathcal{O}_p$ has instantaneous frequency $f_p(t)$ at some real-time t and $f_p(t+\Delta t)$ at $t+\Delta t$. We stipulate that $f_p(t+\Delta t)/f_p(t)$ is linearly bounded by $1 \pm \sigma_p \Delta t$. Parameter σ_p is called the *oscillator stability* measured in ppm/s. However, this simple characterization is only meaningful for certain durations $0 \leq \Delta t < \infty$. For very large ones, the term $\sigma_p \Delta t$ would become too pessimistic, since the specified maximum oscillator drift $\rho_p^{\max}$ prevents the frequency ratio on the long run from exceeding $1 \pm 2\rho_p^{\max}$. For very small durations, say a few oscillator periods $1/f_p$, glitches could violate the bounds, but we regard them as non-accumulating which can be confirmed by experiments (however the variance of the frequency deviation becomes larger). A more advanced way of characterization use an oscillator stability $\sigma_p(\Delta t)$ that is dependent on the duration Δt, see Figure 4.3 for a conceivable example. Still, we are interested in durations that are in the range of the resynchronization period, which should be safely within these extremes.

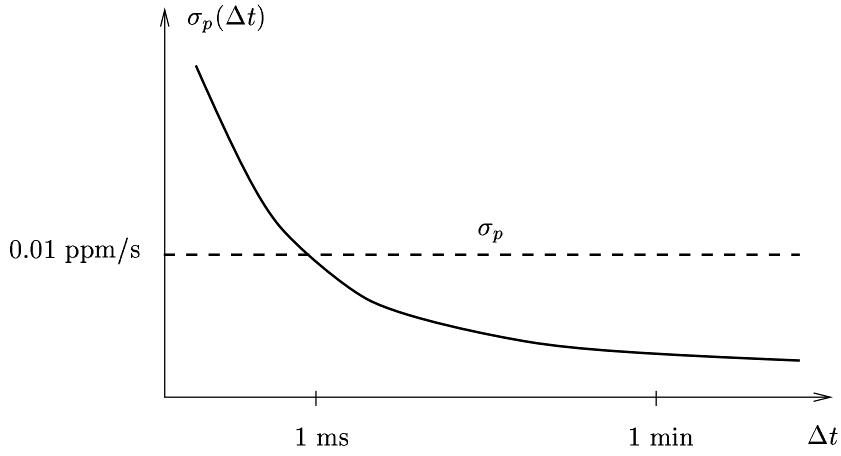


Figure 4.3: *Oscillator Stability*

In the following we present three examples of oscillators drawn from manufacturers data sheets to determine estimates on ρ and σ. For a more general overview take a look at the table presented in [73].

Example. *Uncompensated quartz (XO)*
SMI Inc. offers an quartz crystal 38STF327 with $f_p = 32.768$ kHz, where Typ B has an initial drift $\rho_p = 20 \text{ ppm} = 2 \cdot 10^{-5}$. The major influence for stability is temperature here, given as a parabolic curve for this X-cut quartz. From the characteristics we can obtain that a change of 1°C entails a frequency change of at most 3 ppm. Assuming that after a warm up phase the ambient temperature does not vary of more than 1°C within 10 sec, we end up with stability $\sigma_p = 0.3 \text{ ppm/s} = 3 \cdot 10^{-7} \text{ s}^{-1}$. In respect to this large value, all other influences can be neglected, like an aging of 3 ppm/year.

Example. *Temperature compensated quartz (TCXO)*
ACT offers a digital processing temperature compensated crystal oscillator DTCXO-07A with $f_p = 1$ MHz, where for Version A a drift of $0.1 \text{ ppm} = 1 \cdot 10^{-7}$ is specified over the temperature range from $-10°$ to $+60°$C. With the same ambient temperature assumptions as above, we get a stability of about 0.0002 ppm/s. An aging of 0.3 ppm/year can be neglected for our concerns. Additionally, a short term frequency stability of 0.0003 ppm/s is given, which covers the stochastic behavior (see next example). Combining both stability measures yields $\sigma_p = 0.001 \text{ ppm/s} = 1 \cdot 10^{-9} \text{ s}^{-1}$.

Example. *Rubidium gas cell (Ru)*
BALL-EFRATOM offers a subminiature rubidium oscillator FRS-C-1A8E4C with nominal frequency $f_p = 10$ MHz and initial drift $\rho_p = 5 \cdot 10^{-11}$. Again, an aging of 10^{-10}/day and 10^{-11}/year can be ignored. Also the environmental influences do not contribute significantly, but we have to account for stochastic effects. Those are characterized in the time domain with the help of the *Allan variance* that describes the variance of the frequency deviation during a particular measurement period, see [1]. This oscillator has an Allan variance of $(3.16 \cdot 10^{-11})^2$ at 10 seconds, so we can form a confidence interval for the frequency deviation with say three times of the standard deviation. Carrying over this into our notation, we get a stability $\sigma_p = (3.16 \cdot 10^{-11} \times 3)/10 \approx 1 \cdot 10^{-11}\ \mathrm{s}^{-1}$. Note again, that the nature of stochastic influences is primarily understood as non-accumulating.

Let us summarize the oscillator assumptions for our framework, which hold for proper parameter settings with $\max\{\sigma_p \Delta t, \rho_p\} < \rho_p^{\max}$.

Assumption 4.1 (Oscillator Drift and Stability) *Each non-faulty node p hosts an oscillator $\mathcal{O}_p$ with instantaneous frequency $f_p(t)$ subject to three conditions:*

(1) The initial drift condition

$$\left| \frac{f_p(t_0)}{f_p} - 1 \right| \leq \rho_p \tag{4.1}$$

bounds the instantaneous frequency at begin t_0, where f_p is the nominal oscillator frequency and ρ_p the initial oscillator drift.

(2) The overall drift condition

$$\left| \frac{f_p(t)}{f_p} - 1 \right| \leq \rho_p^{\max} \tag{4.2}$$

bounds the instantaneous frequency for any time $t \geq t_0$, where $\rho_p^{\max}$ is the maximal oscillator drift.

(3) The stability condition

$$\left| \frac{f_p(t + \Delta t)}{f_p(t)} - 1 \right| \leq \sigma_p \Delta t \tag{4.3}$$

bounds the variation of the instantaneous frequencies during $\Delta t \geq 0$ for any time $t \geq t_0$, where σ_p is the oscillator stability.

4.2.2 Local Clock

The distinction between oscillator $\mathcal{O}_p$ and clock $\mathcal{C}_p$ as shown in Figure 4.2 is the hook to introduce rate synchronization for clocks. In the absence of a CRA, the oscillator is directly coupled to the clock, which is usually a simple hardware counting device. Thus, the instantaneous clock rate $v_p(t)$ becomes to

$$v_p(t) = S_p f_p(t), \tag{4.4}$$

where *coupling factor* S_p in [sec] is the constant $1/f_p$. In such an arrangement the clock drift δ_p equals the maximal oscillator drift $\rho_p^{\max}$. Strictly speaking, clock rate $v(t)$ is not continuous since advancements happen only at oscillator ticks, but we regard them as evenly applied during the stalling periods in between.

It is vital to understand that a CRA tries to break up this rigid oscillator-clock coupling by getting a handle on S_p. In other words, even though $f_p(t)$ changes, the factor S_p should be set in such a way that the clock rate $v_p(t)$ remains approximately constant. Obviously, the realization of a local clock needs to provide some means to modify S_p, like the CSU from [24] or our UTCSU-ASIC, see Chapter 2. The feasible range of coupling factor S_p is given by $1/f_p(1 \pm \rho_p^{\max})$, through which the clock rate $v_p(t)$ of a non-faulty clock satisfies

$$|v_p(t) - 1| \leq 2\rho_p^{\max}. \tag{4.5}$$

A CRA has no means to affect the oscillator stability, hence a free running clock $\mathcal{C}_p$ inherits (4.3) for its rate as

$$\left| \frac{v_p(t + \Delta t)}{v_p(t)} - 1 \right| \leq \sigma_p \Delta t, \tag{4.6}$$

since the common coupling factor S_p falls out of the ratio. In the special case of $\sigma_p = 0$ we say the oscillator/clock is *stable*, thus $f_p(t) = \text{const}$. Moreover, we define

$$\sigma_{\max} = \max\{\sigma_p \ : \ 1 \leq p \leq n\} \tag{4.7}$$

as the uniform bound on the oscillator stability, and

$$\rho_{\max} = \max\{\rho_p^{\max} \ : \ 1 \leq p \leq n\} \tag{4.8}$$

as the uniform bound on the maximal oscillator drift.

Additionally, we presume a clock state synchronization with precision $\pi_{\max}$ as a result of the coexisting CSA. This facilitates a rounds-based fashion of a CRA, which entails an inexpensive implementation and renders the analysis easier. Note that we do not require an accuracy on the clocks.

Assumption 4.2 (Clock Precision) *The clocks in our system are synchronized by a clock state algorithm, where any non-faulty pair of clocks* $\mathcal{C}_p$ *and* $\mathcal{C}_q$ *satisfies the precision condition*

$$|C_p(t) - C_q(t)| \leq \pi_{\max} \qquad \forall\ t \geq t_0 \tag{4.9}$$

with $\sigma_{\max}\pi_{\max} \ll \rho_{\max}$.

4.2.3 Processor

Each node is equipped with a *processor* that is responsible to execute the CRA. All computations are essentially periodic and require integer arithmetic only. From our oscillator examples it appears reasonable to represent any rate related quantity in multiples of $2^{-43} \approx 1.1 \cdot 10^{-13}$ and limit it by $2^{-12} \approx 2.4 \cdot 10^{-4}$, hence a 32 bit integer arithmetic seems to be enough. In terms of the execution speed, memory and organization of the processor, we require that there are bounds on the execution time.

Assumption 4.3 (Processor Characteristics) *Each node in our system is equipped with a processor that ensures the following:*

(1) Rate related data is adequately represented in range and granularity.

(2) A single computation required for clock rate synchronization at any non-faulty node takes between $\eta_{\min}$ *and* $\eta_{\max}$ *seconds.*

4.2.4 Communication Subsystem

Nodes communicate with each other via a packet based *communication subsystem* that provides an (un)reliable broadcast primitive. It can either be a *fully connected point-to-point network* or a *broadcast-type network*, both with synchronous behavior and limited transmission capacity.

Assumption 4.4 (Communication Characteristics) *The nodes in our system communicate by via packets subject to the following conditions:*

(1) The maximum broadcast latency $\lambda_{\max}$ *delimits the time between initiating and actually sending a packet at any non-faulty sending node.*

(2) The maximum broadcast operation delay $\omega_{\max}$ *delimits the time to send out all packets at any non-faulty sending node.*

(3) If no transmission faults occur, a packet from node q to node $p \neq q$ experiences a delay $\Delta t'_{p,q}$ satisfying

$$\Delta t_{p,q} - \epsilon^-_{p,q} \leq \Delta t'_{p,q} \leq \Delta t_{p,q} + \epsilon^+_{p,q}, \tag{4.10}$$

where $\Delta t_{p,q}$ represents the deterministic part and $\epsilon^\pm_{p,q}$ the delivery uncertainties. For simplicity we will work with $\epsilon_{\max} = \max\{\epsilon^-_{p,q} + \epsilon^+_{p,q} : 1 \leq p, q \leq n\}$ and $\Delta t_{\max} = \max\{\Delta t_{p,q} : 1 \leq p, q \leq n\}$ satisfying $\min\{\Delta t_{p,q} : 1 \leq p, q \leq n\} > \epsilon_{\max}$ for logical reasons.

(4) Any packet can carry b bytes.

The first two parts allow to model a fully connected point-to-point network ($\lambda_{\max} = 0$, $\omega_{\max} \neq 0$) or a broadcast-type network ($\lambda_{\max} \neq 0$, $\omega_{\max} = 0$). Furthermore, the delivery uncertainties $\epsilon^\pm_{p,q}$ given in the third part can either be fixed a priori or, preferable, measured on-line with round-trip oriented protocols. For a long haul network it is typically in the ms-range, and for LANs with a shared broadcast channel in the μs-range. Timestamping uncertainties could be added to delivery uncertainties, since they arise primarily from non-zero clock granularities, see [55]. The last part accounts for a maximal data portion of a packet (e.g., for a CAN-Bus $b = 8$), so that only a certain amount of information can be passed on by one transmission.

4.2.5 Faults

The above made system assumptions are only meaningful in the absence of faults. However, when dealing with a realistic system, faults may occur and need to be considered both for developing and analyzing distributed algorithms. Faults can affect the clocks (e.g. stuck, jump, rate error), the processors (e.g. various crashes) or the communication subsystem (e.g. omissions, timing errors, value errors). Therefore, it becomes necessary to set up a proper *(abstract) fault model* $\mathcal{F}$ that specifies the prospective faults in our system, see Section 4.4.6. We do not focus on a specific $\mathcal{F}$ to keep our framework widely applicable.

4.3 Building Blocks

This section presents the building blocks for clock rate synchronization by introducing notations and definitions, and by proving useful technical lemmas with $\mathcal{O}()$-expressions, whose purpose are to specify the order of magnitude of terms that can be neglected in practice. Based on them will be the developed algorithm for clock rate synchronization in Section 4.4 along with its analysis in Section 4.5 but with relaxed $\mathcal{O}()$-expressions.

4.3.1 Interval Paradigm

Our framework uses *asymmetric intervals* to capture the necessary information, see [55] or earlier [50], [25] and [37]. We denote them with bold capital letters such as $\mathbf{I} = [x, r, y]$, where $r \geq 0$ is called the *reference point*, $x \geq 0$ the *left length* and $y \geq 0$ the *right length*. Such an asymmetric interval translates into a regular one $[r-x, r+y]$, where $(r-x)$ is the *left edge* and $(r+y)$ the *right edge*. In case that $x = y$ we also write $[r \pm x]$. An ordered set of them is written by calligraphic capital letters such as $\boldsymbol{\mathcal{I}}$. Operations on them are defined in a straightforward manner, summarized in the following extensive Definition.

Definition 4.3 (Operations on Asymmetric Intervals) *Given two asymmetric intervals* $\mathbf{I}_1 = [x_1, r_1, y_1]$ *and* $\mathbf{I}_2 = [x_2, r_2, y_2]$. *We define the*

(1) reference point *as* $\mathrm{ref}(\mathbf{I}_1) = r_1$,

(2) right edge *as* $\mathrm{right}(\mathbf{I}_1) = r_1 + y_1$,

(3) left edge *as* $\mathrm{left}(\mathbf{I}_1) = r_1 - x_1$,

(4) length $||\mathbf{I}_1||$ *as* $x_1 + y_1$,

(5) exchange of lengths *as* $\overline{\boldsymbol{I}}_1 = [y_1, r_1, x_1]$,

(6) sum $\mathbf{I}_1 + \mathbf{I}_2$ *as* $[x_1 + x_2, r_1 + r_2, y_1 + y_2]$,

(7) alignment *as* $\mathrm{align}(\mathbf{I}_1) = \mathbf{I}_1 - [0, r_1, 0]$,

(8) scalar product $s\mathbf{I}_1$ *as* $[sx_1, sr_1, sy_1]$ *for any real s,*

(9) normalization *as* $\mathrm{norm}(\mathbf{I}_1) = [x_1/r_1, 1, y_1/r_1]$,

(10) interval product $\mathbf{I}_1 \cdot \mathbf{I}_2$ *as* $[r_1x_2 + r_2x_1 - x_1x_2, r_1r_2, r_1y_2 + r_2y_1 + y_1y_2]$,

(11) intersection $\mathbf{I}_1 \cap \mathbf{I}_2$ *as* $[\max\{r_1 - x_1, r_2 - x_2\}, \min\{r_1 + y_1, r_2 + y_2\}]$,

(12) union $\mathbf{I}_1 \cup \mathbf{I}_2$ *as* $[\min\{r_1 - x_1, r_2 - x_2\}, \max\{r_1 + y_1, r_2 + y_2\}]$,

Note that in case of disjunct intervals the intersection delivers the *empty interval* $\emptyset$ and the union the closure of them. Furthermore, no reference point is explicitly given for these two operations, since several definitions are conceivable (e.g., midpoint setting).

4.3.2 Local Rate Intervals

For rate synchronization we have to find a way to capture the rate v_p of clock $\mathcal{C}_p$. Unfortunately, the rate of a clock cannot be observed directly, but we can postulate an asymmetric interval with reference point r_p and sufficiently long left/right lengths to include the ideal rate of 1. Both lengths and the reference point are given in multiples of the clock rate, hence $1 \in [v_p\vartheta_p^-, v_p r_p, v_p\vartheta_p^+]$, where ϑ_p^- and ϑ_p^+ are called *inverse rate drifts.* Note that this introduces a relative expression of clocks rate as opposed to an absolute expression of clock states. Generally speaking, "rates" are always regarded as the ideal rate 1 altered by the some "drifts". Dropping v_p leads to the definition of a *rate interval* $\mathbf{R}_p = [\vartheta_p^-, r_p, \vartheta_p^+]$, which contains enough information to run a clock rate algorithm.

Definition 4.4 (Correctness of Rate Intervals) *A rate interval* $\mathbf{R}_p = [\vartheta_p^-, r_p, \vartheta_p^+]$ *is* correct *during a non-empty (real-time) interval* $\mathcal{T}$ *iff*

$$1 \in v_p(t)\mathbf{R}_p \qquad \forall t \in \mathcal{T}. \tag{4.11}$$

In the special case that a rate interval $\mathbf{R}_p$ has 1 as reference point, we call it a *local rate interval.* These kind of rate intervals have great importance, since they can be maintained locally and possess many useful properties. The following lemma asserts that there exists a correspondence between local rate intervals and clock drifts.

Lemma 4.1 (Local Rate Interval vs. Clock Drift) *If clock* $\mathcal{C}_p$ *has clock drift* δ_p *then*

$$\mathbf{R}_p = \left[\frac{\delta_p}{1+\delta_p}, 1, \frac{\delta_p}{1-\delta_p}\right] \tag{4.12}$$

is guaranteed to be a correct local rate interval. On the other hand, if $\mathbf{R}_p = [\vartheta_p^-, 1, \vartheta_p^+]$ *is a correct local rate interval for clock* $\mathcal{C}_p$ *then the actual clock drift is at most*

$$\delta_p = \max\left\{\frac{\vartheta_p^+}{1+\vartheta_p^+}, \frac{\vartheta_p^-}{1-\vartheta_p^-}\right\}. \tag{4.13}$$

Proof. For the first part, we have to determine inverse rate drifts ϑ_p^- and ϑ_p^+ that satisfy

$$v_p(1-\vartheta_p^-) \le 1 \le v_p(1+\vartheta_p^+) \tag{4.14}$$

due to Definition 4.4. Given that Definition 4.2 assures that clock rate $v_p \in [1-\delta_p, 1+\delta_p]$, it is sufficient to draw upon the extreme values of v_p. Hence, plugging in $v_p = 1+\delta_p$ in (4.14) yields $\vartheta_p^- \ge \delta_p/(1+\delta_p)$ and $\vartheta_p^+ \ge 0$, and $v_p = 1-\delta_p$ delivers $\vartheta_p^- \ge 0$ and

$\vartheta_p^+ \geq \delta_p/(1-\delta_p)$. Putting these statements together, we obtain the desired local rate interval in (4.12).

For the second part of the lemma, we have to find a δ_p that matches the possible clock rates v_p induced by the local rate interval $\mathbf{R}_p$. For that purpose we transform (4.14) into

$$-\frac{\vartheta_p^+}{1+\vartheta_p^+} \leq v_p - 1 \leq \frac{\vartheta_p^-}{1-\vartheta_p^-}, \tag{4.15}$$

which can be rewritten as $|v_p - 1| \leq \max\{\vartheta_p^-/(1-\vartheta_p^-), \vartheta_p^+/(1+\vartheta_p^+)\} = \delta_p$ proving (4.13). □

The first part of Lemma 4.1 together with the specified initial oscillator drift ρ_p can be used to commence with a correct local rate interval at t_0, thus

$$\mathbf{R}_p(t_0) = \left[\frac{\rho_p}{1+\rho_p}, 1, \frac{\rho_p}{1-\rho_p}\right] \tag{4.16}$$

by a neutral setting of coupling factor $S_p = 1/f_p$. The second part of Lemma 4.1 can be used to assert a clock drift

$$\delta_p = \frac{||\mathbf{R}_p||}{1 - ||\mathbf{R}_p||}, \tag{4.17}$$

from which we can deduce that the current oscillator drift has to be within $[0 \pm \delta_p]$ in case of a feasible coupling factor S_p.

For further considerations it is important to relate an observable duration ΔT on a local clock with their real-time counterpart Δt and vice versa. Our next lemma establishes these relationships.

Lemma 4.2 (Duration Estimation) *Given a clock $\mathcal{C}_p$ paced by an oscillator with stability σ_p. Let t_1 resp. t_2 be real-times and $T_1 = C_p(t_1)$ resp. $T_2 = C_p(t_2)$ the corresponding clock states, where $t_1 \leq t_2$ and no resynchronization occurred in between. If clock $\mathcal{C}_p$ has rate $v_p(t_1)$ at t_1 then we have*

$$v_p(t_1)\left((t_2-t_1) - \frac{\sigma_p}{2}(t_2-t_1)^2\right) \leq T_2 - T_1 \leq v_p(t_1)\left((t_2-t_1) + \frac{\sigma_p}{2}(t_2-t_1)^2\right) \tag{4.18}$$

and the converse

$$t_2 - t_1 \leq \frac{T_2 - T_1}{v_p(t_1)} + \frac{\sigma_p(T_2-T_1)^2}{2v_p^2(t_1)} + \mathcal{O}\left(\sigma_p^2(T_2-T_1)^3\right) \tag{4.19}$$

$$t_2 - t_1 \geq \frac{T_2 - T_1}{v_p(t_1)} - \frac{\sigma_p(T_2-T_1)^2}{2v_p^2(t_1)} + \mathcal{O}\left(\sigma_p^2(T_2-T_1)^3\right). \tag{4.20}$$

Proof. We introduced the clock rate $v_p(t)$ as the derivative of the time-dependable function $C_p(t)$ of the clock state. Applying the integral from starting point t_1 to successive point $t_1 + \Delta t$, we get

$$\Delta T = C_p(t_1 + \Delta t) - C_p(t_1) = \int_0^{\Delta t} v_p(t_1 + \xi) d\xi. \tag{4.21}$$

In the absence of resynchronizations we derive from the stability condition of Assumption 4.1 part (3) that the clock rate at $t_1 + \xi$ satisfies

$$v_p(t_1)(1 - \sigma_p \xi) \leq v_p(t_1 + \xi) \leq v_p(t_1)(1 + \sigma_p \xi)$$

for any $\xi \geq 0$. Using these relations as majorants for the integrand in (4.21) and relying on the non-accumulating nature of short-term violations, we can bound the clock state difference by

$$\begin{aligned} \int_0^{\Delta t} v_p(t_1)(1 - \sigma_p \xi) d\xi \quad &\leq \Delta T \leq \quad \int_0^{\Delta t} v_p(t_1)(1 + \sigma_p \xi) d\xi \\ v_p(t_1) \left(\Delta t - \frac{\sigma_p}{2} \Delta t^2 \right) \quad &\leq \Delta T \leq \quad v_p(t_1) \left(\Delta t + \frac{\sigma_p}{2} \Delta t^2 \right), \end{aligned}$$

which proves (4.18) by replacing Δt with $t_2 - t_1$.

For the second part we treat the above relation as a quadratic equation and choose the corresponding roots in order to find bounds on Δt. This leads to

$$\frac{1}{\sigma_p} \left[-1 + \sqrt{1 + \frac{2\sigma_p \Delta T}{v_p(t_1)}} \right] \leq \Delta t \leq \frac{1}{\sigma_p} \left[1 - \sqrt{1 - \frac{2\sigma_p \Delta T}{v_p(t_1)}} \right],$$

where in our setting the second term under each root is small compared to 1. To simplify the bounds we use the asymptotic approximation

$$\sqrt{1 \pm x} = 1 \pm \frac{x}{2} \mp \frac{x^2}{8} + \mathcal{O}\left(x^3\right),$$

valid for $x \to 0$ and obtain after some algebraic manipulations that

$$\frac{\Delta T}{v_p(t_1)} - \frac{\sigma_p \Delta T^2}{2 v_p^2(t_1)} + \mathcal{O}\left(\frac{\sigma_p^2 \Delta T^3}{v_p^3(t_1)} \right) \leq \Delta t \leq \frac{\Delta T}{v_p(t_1)} + \frac{\sigma_p \Delta T^2}{2 v_p^2(t_1)} + \mathcal{O}\left(\frac{\sigma_p^2 \Delta T^3}{v_p^3(t_1)} \right).$$

Since clock rates v_p are very close to 1, we drop them in the $\mathcal{O}()$-terms and get (4.19)/(4.20) of the Lemma. □

Each node p maintains a local rate interval $\mathbf{R}_p$, which should remain correct as time proceeds. However, the oscillator stability requires to deteriorate them as illustrated in Figure 4.4. Let the local rate interval $\mathbf{R}_p$ be correct at t_1, hence $1 \in v_p(t_1)\mathbf{R}_p(t_1)$. The cone formed by the dotted lines indicates the feasible clock rates according to the stability condition of the attached oscillator, whereas the curved line represents an exemplary progress of the clock rate. At t_2 the local rate interval has to satisfy $1 \in v_p(t_2)\mathbf{R}_p(t_2)$.

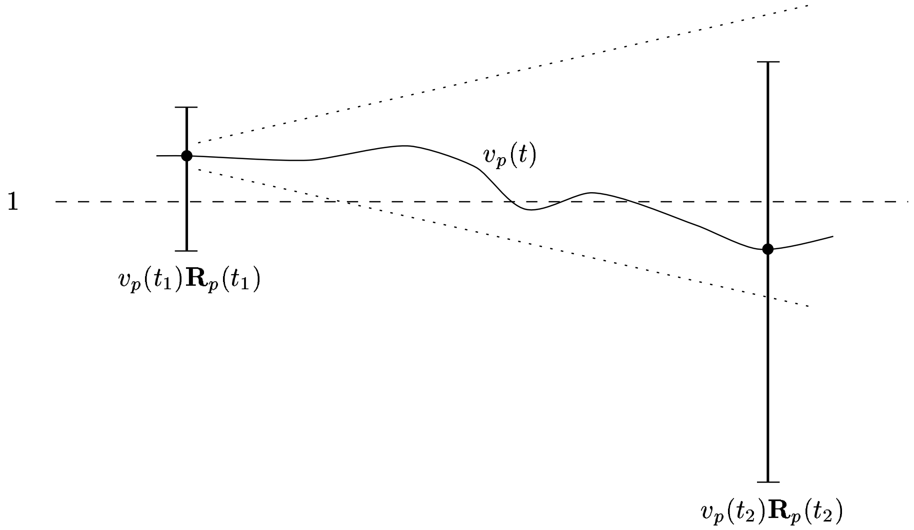


Figure 4.4: *Deterioration of Rate Intervals*

Lemma 4.3 (Deterioration of Rate Intervals) *Given a clock $\mathcal{C}_p$ paced by an oscillator with stability σ_p. If $\mathbf{R}_p$ is a correct local rate interval at T_1, then*

$$\mathbf{R}_p + \left[0 \pm \frac{\sigma_p(T_2 - T_1)}{1 - 2||\mathbf{R}_p||}\right] + \left[0 \pm \mathcal{O}\left(\sigma_p^2(T_2 - T_1)^2\right)\right] \tag{4.22}$$

is correct at $T_2 \geq T_1$, when no resynchronizations occurred in between.

Proof. Due to the correctness of $\mathbf{R}_p = [\vartheta_p^-, 1, \vartheta_p^+]$ at $T_1 = C_p(t_1)$ we know from Definition 4.4 that

$$v_p(t_1)(1 - \vartheta_p^-) \leq 1 \leq v_p(t_1)(1 + \vartheta_p^+). \tag{4.23}$$

In the absence of resynchronizations until $T_2 = C_p(t_2)$, the stability property (4.6) ensures that $|v_p(t_2)/v_p(t_1) - 1| \leq \sigma_p(t_2 - t_1)$. Making $v_p(t_1)$ explicit and using the asymptotic approximation $(1 \pm x)^{-1} = 1 \mp x + \mathcal{O}(x^2)$ valid for $x \to 0$ yields

$$\begin{aligned} v_p(t_1) &\leq v_p(t_2)\left(1 + \sigma_p(t_2 - t_1) + \mathcal{O}\left(\sigma_p^2(t_2 - t_1)^2\right)\right) \\ v_p(t_1) &\geq v_p(t_2)\left(1 - \sigma_p(t_2 - t_1) + \mathcal{O}\left(\sigma_p^2(t_2 - t_1)^2\right)\right). \end{aligned}$$

To bring in the observable duration $T_2 - T_1$ we consult Lemma 4.2 to bound $t_2 - t_1$, hence

$$v_p(t_1) \leq v_p(t_2)\left(1 + \frac{\sigma_p(T_2 - T_1)}{v_p(t_1)} + \mathcal{O}\left(\sigma_p^2(T_2 - T_1)^2\right)\right) \tag{4.24}$$

$$v_p(t_1) \geq v_p(t_2)\left(1 - \frac{\sigma_p(T_2 - T_1)}{v_p(t_1)} + \mathcal{O}\left(\sigma_p^2(T_2 - T_1)^2\right)\right). \tag{4.25}$$

Plugging $v_p(t_1)$ from (4.24)/(4.25) into (4.23) leads to

$$1 \leq v_p(t_2)\left(1 + \vartheta_p^+ + \frac{1 + \vartheta_p^+}{v_p(t_1)}\sigma_p(T_2 - T_1) + \mathcal{O}\left(\sigma_p^2(T_2 - T_1)^2\right)\right) \tag{4.26}$$

$$1 \geq v_p(t_2)\left(1 - \vartheta_p^- - \frac{1 - \vartheta_p^-}{v_p(t_1)}\sigma_p(T_2 - T_1) + \mathcal{O}\left(\sigma_p^2(T_2 - T_1)^2\right)\right). \tag{4.27}$$

To get rid of the denominator $v_p(t_1)$ we know from (4.17) that $v_p(t_1) \geq 1 - ||\mathbf{R}_p||/(1 - ||\mathbf{R}_p||)$ and utilizing $0 \leq \vartheta_p^-, \vartheta_p^+ < 1$ provides

$$\frac{1 - \vartheta_p^-}{v_p(t_1)} \leq \frac{1 + \vartheta_p^+}{v_p(t_1)} \leq \frac{1}{1 - 2||\mathbf{R}_p||}.$$

Eventually, we can transform (4.26)/(4.27) into

$$\begin{aligned} \frac{1}{v_p(t_2)} &\leq 1 + \vartheta_p^+ + \frac{\sigma_p(T_2 - T_1)}{1 - 2||\mathbf{R}_p||} + \mathcal{O}\left(\sigma_p^2(T_2 - T_1)^2\right) \\ \frac{1}{v_p(t_2)} &\geq 1 - \vartheta_p^- - \frac{\sigma_p(T_2 - T_1)}{1 - 2||\mathbf{R}_p||} + \mathcal{O}\left(\sigma_p^2(T_2 - T_1)^2\right), \end{aligned}$$

which proves the correctness of the deteriorated local rate intervals. □

The deterioration of rate intervals with reference point unequal 1 is of minor interest; it requires additional provisions and will show up in the proof of Lemma 4.15.

4.3.3 Relative Rate Measurement

Any clock rate algorithm has to work on the grounds of relative rates between clocks pairs. This restriction comes from the fact that a local clock is not able to determine its rate by itself, otherwise rate synchronization would be trivial. We capture the relative rate of a remote clock against a local one with a *quotient rate interval* $\mathbf{Q}_{p,q}$. The indexing follows the rule that the first one denotes the local node and the second the remote one. Such an interval quantifies how fast/slow a remote clock is in respect to its own.

Definition 4.5 (Quotient Rate Interval) *Given two clocks* $\mathcal{C}_p$ *resp.* $\mathcal{C}_q$ *with their rates* $v_p(t')$ *resp.* $v_q(t'')$ *during the non-empty (real-time) intervals* $\mathcal{T}_p$ *resp.* $\mathcal{T}_q$*. An asymmetric interval* $\mathbf{Q}_{p,q}$ *that satisfies*

$$\frac{v_q(t')}{v_p(t'')} \in \mathbf{Q}_{p,q} \qquad \forall t' \in \mathcal{T}_q \; \forall t'' \in \mathcal{T}_p \tag{4.28}$$

is a quotient rate interval of remote clock $\mathcal{C}_q$ *during* $\mathcal{T}_q$ *against local clock* $\mathcal{C}_p$ *during* $\mathcal{T}_p$*.*

We propose a simple protocol shown in Figure 4.5 to obtain $\mathbf{Q}_{p,q}$. It is based on repeated message pairs, such that remote node q broadcasts periodically a message that contains its current clock state $T_q = C_q(t_q)$. Local node p records its current clock state $T_p = C_p(t_p)$ upon message arrival. Therefore, node p can extract clock states T_q, T_p from the latter message M_q, and $T_q^{\prec}$, $T_p^{\prec}$ from the earlier message $M_q^{\prec}$. With these four clock states and taking into account the transmission characteristics from Assumption 4.4, node p can compute a quotient rate interval given by Lemma 4.5.

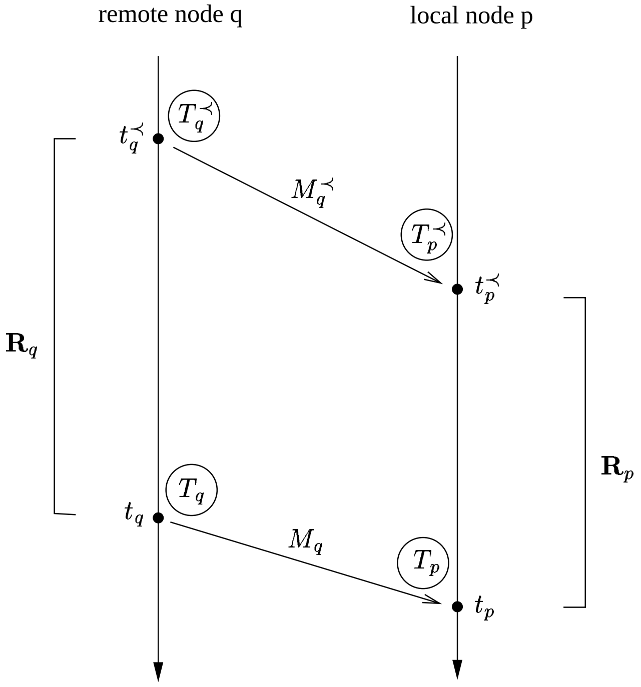


Figure 4.5: *Protocol for Relative Rate Measuring*

For preparation, we need the following technical Lemma 4.4, from which we can also learn that it is not meaningful to define a quotient rate interval on the ratio of clock rates at simultaneous real-times. This explains the relaxed condition (4.28) in Definition 4.5 with the dangling t' and t''.

Lemma 4.4 (Min/Max Clock Rate) *Given a clock $\mathcal{C}_p$ paced by an oscillator with stability σ_p and let t_1 resp. t_2 be real-times and $T_1 = C_p(t_1)$ resp. $T_2 = C_p(t_2)$ the corresponding clock states, where $t_1 \leq t_2$. When no resynchronization occurred in between, then clock rate $v_p(t)$ satisfies*

$$v_p(t) \in \frac{T_2 - T_1}{t_2 - t_1}\left[1 \pm \frac{\sigma_p}{2}(t_2 - t_1)\right] + \left[0 \pm \mathcal{O}\left(\sigma_p^2(t_2 - t_1)^2\right)\right] \tag{4.29}$$

for any $t \in [t_1, t_2]$.

Proof. We have to consider two cases, either where the clock rate increases or decreases maximally during the real-time duration $\Delta t = t_2 - t_1$ whereas the expired logical time still amounts to $\Delta T = T_2 - T_1$. The actual clock rate lies somewhere between these two extremes, which gives rise for bounding $v_p(t)$ $\forall t \in [t_1, t_2]$. In the first case (increasing rate), the clock has a minimum rate $v'_{\min}$ at t_1 related by

$$\Delta T = v'_{\min}\left(\Delta t + \frac{\sigma_p}{2}(\Delta t)^2\right) \tag{4.30}$$

according to (4.18) of Lemma 4.2. By virtue of Assumption 4.1 part (3) the maximum rate $v'_{\max}$ at t_2 is given by

$$v'_{\max} = v'_{\min}(1 + \sigma_p \Delta t). \tag{4.31}$$

For the second case (decreasing rate) we find similar expressions for the maximum rate

$$\Delta T = v''_{\max}\left(\Delta t - \frac{\sigma_p}{2}(\Delta t)^2\right) \tag{4.32}$$

and for the minimum rate

$$v''_{\min} = v''_{\max}(1 - \sigma_p \Delta t). \tag{4.33}$$

We can easily show that $v'_{\max} \leq v''_{\max}$ and $v'_{\min} \geq v''_{\min}$, but the asymptotic approximation $(1 \pm x)^{-1} = 1 \mp x + \mathcal{O}(x^2)$ valid for $x \to 0$ vanishes the differences. Hence, using (4.30) for the lower rate bound yields

$$v_p(t) \geq \frac{\Delta T}{\Delta t}\left(1 - \frac{\sigma_p}{2}\Delta t\right) + \mathcal{O}\left(\sigma_p^2 \Delta t^2\right)\frac{\Delta T}{\Delta t}$$

and (4.32) for the upper rate bound

$$v_p(t) \leq \frac{\Delta T}{\Delta t}\left(1 + \frac{\sigma_p}{2}\Delta t\right) + \mathcal{O}\left(\sigma_p^2 \Delta t^2\right)\frac{\Delta T}{\Delta t}$$

both for $t \in [t_1, t_2]$, which completes the proof of (4.29). □

Lemma 4.5 (Relative Rate Measurement) *Let $\mathcal{C}_p$ be the clock of local node p with stability σ_p and $\mathcal{C}_q$ be the clock of remote node q with stability σ_q. By executing the protocol given in Figure 4.5 relying on Assumption 4.4 part (3), and providing that local rate interval $\mathbf{R}_p$ is correct during $[t_p^{\prec}, t_p]$ and $\mathbf{R}_q$ is correct during $[t_q^{\prec}, t_q]$, then interval*

$$\begin{aligned}\mathbf{Q}_{p,q} \;=\; & \left[\frac{T_q - T_q^{\prec}}{T_p - T_p^{\prec}} \pm \left(\frac{(\sigma_p + \sigma_q)(T_q - T_q^{\prec})}{2(1 - ||\mathbf{R}_p||)} + \frac{\epsilon_{\max}(1 + ||\mathbf{R}_q||)}{T_p - T_p^{\prec}}\right)\right] \\ & + \left[0 \pm \mathcal{O}\left(\sigma_{\max}\epsilon_{\max} + \sigma_{\max}^2(T_p - T_p^{\prec})^2 + \sigma_{\max}(T_p - T_p^{\prec})||\mathbf{R}_p||^2 + \frac{\epsilon_{\max}||\mathbf{R}_q||^2}{T_p - T_p^{\prec}}\right)\right]\end{aligned} \tag{4.34}$$

is a quotient rate interval during $[t_p^{\prec}, t_p]$ resp. $[t_q^{\prec}, t_q]$.

Proof. First of all, we want to find a relation between the involved real-time duration $\Delta t_q = t_q - t_q^{\prec}$ at the sending side q, and $\Delta t_p = t_p - t_p^{\prec}$ at the receiving side p. Due to Assumption 4.4 part (3) the delivery delay for the earlier message $M_q^{\prec}$ can be captured by $\Delta t_{p,q} - \epsilon_{p,q}^- \leq t_p^{\prec} - t_q^{\prec} \leq \Delta t_{p,q} + \epsilon_{p,q}^+$, and for the latter message M_q by $\Delta t_{p,q} - \epsilon_{p,q}^- \leq t_p - t_q \leq \Delta t_{p,q} + \epsilon_{p,q}^+$. Subtracting them delivers

$$|\Delta t_q - \Delta t_p| \leq \epsilon_{p,q}^- + \epsilon_{p,q}^+ \leq \epsilon_{\max}. \tag{4.35}$$

Next we want to derive a lower bound on $\frac{v_q(t')}{v_p(t'')}$, where $t' \in [t_q^{\prec}, t_q]$ and $t'' \in [t_p^{\prec}, t_p]$. From Lemma 4.4 we know that the maximum clock rate $v_p^{\max}$ at node p during $[t_p^{\prec}, t_p]$ is related by

$$\Delta t_p = \frac{\Delta T_p}{v_p^{\max} - \frac{\sigma_p}{2}\Delta T_p + \mathcal{O}\left(\sigma_p^2 \Delta t_p^2\right)} \tag{4.36}$$

and the minimum clock rate $v_q^{\min}$ at node q during $[t_q^{\prec}, t_q]$ by

$$\Delta t_q = \frac{\Delta T_q}{v_q^{\min} + \frac{\sigma_q}{2}\Delta T_q + \mathcal{O}\left(\sigma_q^2 \Delta t_q^2\right)}. \tag{4.37}$$

Putting together (4.36) and (4.37) via (4.35) yields

$$\begin{aligned}\Delta T_p v_q^{\min} - \Delta T_q v_p^{\max} \geq & -\frac{\sigma_p + \sigma_q}{2}\Delta T_p \Delta T_q + \mathcal{O}\left(\sigma_q^2 \Delta t_q^2\right)\Delta T_p + \mathcal{O}\left(\sigma_p^2 \Delta t_p^2\right)\Delta T_q \\ & - \epsilon_{\max}\left(v_p^{\max} - \frac{\sigma_p}{2}\Delta T_p + \mathcal{O}\left(\sigma_p^2 \Delta t_p^2\right)\right)\left(v_q^{\min} + \frac{\sigma_q}{2}\Delta T_q + \mathcal{O}\left(\sigma_q^2 \Delta t_q^2\right)\right)\end{aligned}$$

and after some algebraic manipulations we get

$$\begin{aligned}\frac{v_q^{\min}}{v_p^{\max}} \;\geq\; & \frac{\Delta T_q}{\Delta T_p}\left(1 - \frac{(\sigma_p + \sigma_q)\Delta T_p}{2 v_p^{\max}}\right) - \left(\frac{v_q^{\min}}{\Delta T_p} + \frac{\sigma_q \Delta T_q}{2\Delta T_p}\right)\epsilon_{\max} \\ & + \mathcal{O}\left(\sigma_p^2 \Delta t_p^2 + \sigma_q^2 \Delta t_q^2\right).\end{aligned} \tag{4.38}$$

In the following, three manipulations on the right hand side of (4.38) are carried out. First, we are able give a lower bound on $v_p^{\max}$ by virtue of (4.17) since $\mathbf{R}_p$ is correct during $[t_p^{\prec}, t_p]$, thus $v_p^{\max} \geq 1 - ||\mathbf{R}_p|| + \mathcal{O}(||\mathbf{R}_p||^2)$. Alternatively, we could have used $\rho_p^{\max}$ according to (4.5), which exempts us from using the local rate interval of node p. By the same token, we obtain $v_q^{\min} \leq 1 + ||\mathbf{R}_q|| + \mathcal{O}(||\mathbf{R}_q||^2)$. Finally, the term $(\sigma_q \Delta T_q)/(2\Delta T_p)\epsilon_{\max}$ will be put into the $\mathcal{O}()$-term. After all, (4.38) can be rewritten as

$$\begin{aligned}\frac{v_q^{\min}}{v_p^{\max}} &\geq \frac{\Delta T_q}{\Delta T_p}\left(1 - \frac{(\sigma_p + \sigma_q)\Delta T_p}{2(1 - ||\mathbf{R}_p||)}\right) - \frac{\epsilon_{\max}(1 + ||\mathbf{R}_q||)}{\Delta T_p} \\ &\quad + \mathcal{O}\left(\sigma_q \epsilon_{\max} + \sigma_p^2 \Delta t_p^2 + \sigma_q^2 \Delta t_q^2 + (\sigma_p + \sigma_q)\Delta T_p ||\mathbf{R}_p||^2 + \frac{\epsilon_{\max}}{\Delta T_p}||\mathbf{R}_q||^2\right),\end{aligned}$$

which provides the left edge of $\mathbf{Q}_{p,q}$ in (4.34). A similar line of reasoning starting out with

$$\Delta t_p = \frac{\Delta T_p}{v_p^{\min} + \frac{\sigma_p}{2}\Delta T_p + \mathcal{O}\left(\sigma_p^2 \Delta t_p^2\right)} \tag{4.39}$$

and

$$\Delta t_q = \frac{\Delta T_q}{v_q^{\max} - \frac{\sigma_q}{2}\Delta T_q + \mathcal{O}\left(\sigma_q^2 \Delta t_q^2\right)} \tag{4.40}$$

yields the desired upper bound on $\frac{v_q(t')}{v_p(t'')}$, which completes the proof. □

Since relative rate measurements will take place less frequently than state resynchronizations, we have to care about interspersed state adjustments to get useful clock state differences ΔT_p and ΔT_q. In fact, we have to maintain and subsequently exchange the running sum of state adjustments on both sides, denoted as U_p and U_q, irrespective of being applied instantaneously or by continuous amortization, see [55].

By taking a look at (4.34) of the quotient rate interval in Lemma 4.5, we observe that the deterministic part $\Delta t_{p,q}$ of the message delivery delay does not appear, since it is included in the measured durations ΔT_p and ΔT_q. Furthermore, for a large measurement duration $\mathbf{Q}_{p,q}$ degrades to a point when the clocks are stable. In case of a short duration, the delivery uncertainty spoils the rate measurement. Thus, we can calculate a certain optimal duration

$$\Delta T_{\text{opt}} \approx \sqrt{\frac{\epsilon_{\max}}{\sigma_{\max}}},$$

where the interval length $||\mathbf{Q}_{p,q}||$ becomes smallest. Note that this is different to an CSA, where apart from bandwidth concerns there is no particular lower limit on the state resynchronization period.

Rate measurement requires the cooperation of two different nodes to compute $\mathbf{Q}_{p,q}$. In the special case that $q = p$, i.e. node p sends messages to itself, we get straight from Lemma 4.5 and by dropping delivery uncertainties that

$$\mathbf{Q}_{p,p} = \left[1 \pm \frac{\sigma_p \Delta T_p}{1 - ||\mathbf{R}_p||}\right] + \left[0 \pm \mathcal{O}\left(\sigma_p^2 \Delta T_p^2 + \sigma_p \Delta T_p ||\mathbf{R}_p||^2\right)\right] \tag{4.41}$$

for a duration ΔT_p. This reflects again the impossibility to acquire rate information about the own clock.

4.3.4 Remote Rate Intervals

Suppose node q maintains a correct local rate interval $\mathbf{R}_q$ of its clock $\mathcal{C}_q$. For synchronization purpose, we want to transfer it correctly to another node p resulting in the *remote rate interval* $\mathbf{R}_{p,q}$. This operation is similar to a CSA, where an accuracy interval from one node is passed on to another.

However, the way to carry over a rate interval from node q to a node p is more intricate than for an accuracy interval (recall delay and drift compensation, see [55]), since switching correctness involves the relative rate measurement via quotient rate interval $\mathbf{Q}_{p,q}$. The importance of a remote rate intervals come from the fact, that it expresses the ideal rate 1 in the "rate world" of local node p, though it stems from remote node q. The following lemma pins down this property in a formal way.

Lemma 4.6 (Remote Rate Intervals) *Let $\mathcal{C}_p$ and $\mathcal{C}_q$ be clocks driven by oscillators with stability σ_p and σ_q, respectively. As shown in Figure 4.5, the messages for the relative rate measurement are sent from node q at $T_q = C_q(t_q)$ resp. $T_q^{\prec} = C_q(t_q^{\prec})$ and received at node p at $T_p = C_p(t_p)$ resp. $T_p^{\prec} = C_p(t_p^{\prec})$. If $\mathbf{R}_q$ is a correct local rate interval at $t_q^{\prec}$ on remote node q, then*

$$\mathbf{R}_{p,q} = \mathbf{Q}_{p,q} \cdot \mathbf{R}_q \tag{4.42}$$

is a correct remote rate interval at t_p on local node p, where $\mathbf{Q}_{p,q}$ is a quotient rate interval during $[t_p^{\prec}, t_p]$ resp. $[t_q^{\prec}, t_q]$.

Proof. We string together the properties of intervals $\mathbf{Q}_{p,q}$ and $\mathbf{R}_q$ forming the remote rate interval $\mathbf{R}_{p,q}$, and show that this leads to the desired interval multiplication.

From Definition 4.4 we know that the local rate interval $\mathbf{R}_q = [\vartheta_q^-, 1, \vartheta_q^+]$ of clock $\mathcal{C}_q$ satisfies

$$v_q(t_q^{\prec})(1 - \vartheta_q^-) \leq 1 \leq v_q(t_q^{\prec})(1 + \vartheta_q^+), \tag{4.43}$$

and by specialization of Lemma 4.5 that the quotient rate interval $\mathbf{Q}_{p,q} = Q_{p,q}[u, 1, u]$, with $Q_{p,q} = \mathrm{ref}(\mathbf{Q}_{p,q})$ and $u = ||\mathbf{R}_q||/2$, holds

$$Q_{p,q}(1-u) \leq \frac{v_q(t_q^{\prec})}{v_p(t_p)} \leq Q_{p,q}(1+u). \tag{4.44}$$

Plugging in $v_q(t_q^{\prec})$ from (4.44) into (4.43) yields

$$v_p(t_p)Q_{p,q}(1-u)(1-\vartheta_q^-) \leq 1 \leq v_p(t_p)Q_{p,q}(1+u)(1+\vartheta_q^+).$$

We can translate the last equation into the asymmetric interval notation

$$1 \in v_p(t_p)Q_{p,q}[u + \vartheta_q^- - u\vartheta_q^-, 1, u + \vartheta_q^+ + u\vartheta_q^+],$$

which is according to Definition 4.3 item (10) equivalent to

$$1 \in v_p(t_p)Q_{p,q}[u, 1, u] \cdot [\vartheta_q^-, 1, \vartheta_q^+].$$

Finally, a resubstitution provides the desired property

$$1 \in v_p(t_p)(\mathbf{Q}_{p,q} \cdot \mathbf{R}_q)$$

that finishes the proof. □

In general, a remote rate interval does not have 1 as reference point, in fact

$$\mathrm{ref}(\mathbf{R}_{p,q}) = \mathrm{ref}(\mathbf{Q}_{p,q}). \tag{4.45}$$

In the special case that $q = p$ the ensuing remote rate interval $\mathbf{R}_{p,p}$ becomes to $\mathbf{Q}_{p,p} \cdot \mathbf{R}_p$ and using (4.41) shows that

$$\begin{aligned}
\mathbf{R}_{p,p} &= \left[1 \pm \left(\frac{\sigma_p \Delta T_p}{1 - ||\mathbf{R}_p||} + \mathcal{O}\left(\sigma_p^2 \Delta T_p^2 + \sigma_p \Delta T_p ||\mathbf{R}_p||^2\right)\right)\right] \cdot [\vartheta_p^-, 1, \vartheta_p^+] \\
&= \left[\vartheta_p^- + \frac{\sigma_p \Delta T_p (1 - \vartheta_p^-)}{1 - ||\mathbf{R}_p||}, 1, \vartheta_p^+ + \frac{\sigma_p \Delta T_p (1 + \vartheta_p^+)}{1 - ||\mathbf{R}_p||}\right] \\
&\quad + \left[0 \pm \mathcal{O}\left(\sigma_p^2 \Delta T_p^2 + \sigma_p \Delta T_p ||\mathbf{R}_p||^2\right)\right] \\
&\subseteq \mathbf{R}_p + \left[0 \pm \frac{\sigma_p \Delta T_p}{1 - 2||\mathbf{R}_p||}\right] + \left[0 \pm \mathcal{O}\left(\sigma_p^2 \Delta T_p^2 + \sigma_p \Delta T_p ||\mathbf{R}_p||^2\right)\right].
\end{aligned}$$

Note that this expression equals the deterioration given by (4.22), which makes perfect sense because measuring the rate by itself over a particular duration means simply to cope with the oscillator stability.

4.3.5 Consonance Intervals

For internal rate synchronization, consonance γ, which measures how close the clock rates are together, has to be expressed in terms of intervals. Returning to the definition of rate intervals, we introduce consonance by imposing distances upon their associated reference points.

Definition 4.6 ($\boldsymbol{\gamma}$-Consonance) *Given consonance interval* $\boldsymbol{\gamma} = [\gamma^-, 0, \gamma^+]$ *and an ensemble of clocks* $\mathcal{C}_1, \dots, \mathcal{C}_n$ *with their respective rates* $v_1(t), \dots, v_n(t)$*. An associated set of correct rate intervals* $\boldsymbol{\mathcal{R}} = \{\mathbf{R}_1, \dots, \mathbf{R}_n\}$ *is* $\boldsymbol{\gamma}$*-consonant during a non-empty (real-time) interval* $\mathcal{T}$ *iff*

$$\bigcap_{p=1}^{n} v_p(t)(\mathrm{ref}(\mathbf{R}_p) + \boldsymbol{\gamma}) \neq \emptyset \qquad \forall t \in \mathcal{T}. \tag{4.46}$$

As before, local rate intervals are playing a special role for establishing a relationship between the $\boldsymbol{\gamma}$-consonance property and consonance γ of the clock ensemble.

Lemma 4.7 ($\boldsymbol{\gamma}$-Consonance vs. Consonance γ) *Given a set of clocks* $\mathcal{C}_1, \dots, \mathcal{C}_n$ *each with drift* δ*. If such an ensemble has consonance* γ*, then a set* $\boldsymbol{\mathcal{R}} = \{\mathbf{R}_1, \dots, \mathbf{R}_n\}$ *of correct local rate intervals is* $\boldsymbol{\gamma}$*-consonant for any* $\boldsymbol{\gamma}$ *satisfying*

$$\boldsymbol{\gamma} \supseteq \left[0 \pm \frac{\gamma}{2(1-\delta)}\right]. \tag{4.47}$$

On the other hand, if the set of correct local rate intervals $\boldsymbol{\mathcal{R}}$ *is* $\boldsymbol{\gamma}$*-consonant, then the actual consonance is at most*

$$\gamma = (1+\delta)||\boldsymbol{\gamma}||. \tag{4.48}$$

Proof. For the first part of the Lemma we have to ensure that $v_p(1+\boldsymbol{\gamma})$ and $v_q(1+\boldsymbol{\gamma})$ intersect for any $1 \leq p, q \leq n$, since $\mathrm{ref}(\mathbf{R}_p) = \mathrm{ref}(\mathbf{R}_q) = 1$. Therefore we make $\boldsymbol{\gamma} = [\gamma^-, 0, \gamma^+]$ sufficiently large, in particular, $v_p\gamma^+ + v_q\gamma^- \geq \gamma$ has to hold if $v_p \leq v_q$, and $v_p\gamma^- + v_q\gamma^+ \geq \gamma$ if $v_p \geq v_q$. By solving these inequalities we easily get

$$\gamma^-, \gamma^+ \geq \frac{\gamma}{v_p + v_q}$$

and noting that the sum of any two clock rates cannot fall short of $2(1-\delta)$, we have shown (4.47) of our lemma.

For the second part we know from the non-empty intersection of $v_p(1+\boldsymbol{\gamma})$ and $v_q(1+\boldsymbol{\gamma})$ for any $1 \leq p, q \leq n$ that $v_p - v_q \leq v_p\gamma^- + v_q\gamma^+$ if $v_p \geq v_q$, and $v_q - v_p \leq v_p\gamma^+ + v_q\gamma^-$ if $v_p \geq v_q$. Combining them and recalling that no clock rate exceeds $(1+\delta)$ leads to

$$\begin{aligned} |v_p - v_q| &\leq \max\{v_p\gamma^- + v_q\gamma^+, v_p\gamma^+ + v_q\gamma^-\} \\ &= \gamma^-\max\{v_p, v_q\} + \gamma^+\max\{v_p, v_q\} \\ &\leq ||\boldsymbol{\gamma}||(1+\delta), \end{aligned}$$

which proves (4.48). □

Obviously, if an ensemble of clocks $\mathcal{C}_1, \ldots, \mathcal{C}_n$ has drift δ then their consonance is guaranteed to be 2δ. Therefore, applying (4.47) from the above lemma provides that an associated set $\boldsymbol{\mathcal{R}} = \{\mathbf{R}_1, \ldots, \mathbf{R}_n\}$ of correct local rate intervals is $\boldsymbol{\gamma}$-consonant for any $\boldsymbol{\gamma} \supseteq [0 \pm \delta/(1-\delta)]$.

In order to understand internal clock rate synchronization, we introduce the notion of *internal global rate* $\varphi(t)$ as analogue to internal global time as advocated in [55]. The idea is to define $\varphi(t)$ in such a way that $\varphi(t) \in \bigcap_{p=1}^{n} v_p(t)(1+\boldsymbol{\gamma})$ $\forall t \geq t_0$, which guarantees a particular consonance according to Lemma 4.7. In Section 4.5.1 it will become clear that $\varphi(t)$ is a piecewise constant function in the proximity of the ideal rate 1. We mention explicitly that the introduction of internal global rate is purely artificial, nevertheless, it allows us to reason about consonance by considering each clock separately, which provides great insights and simplifies the analysis tremendously. All further considerations about internal clock rate synchronization rely on the definition of $\boldsymbol{\gamma}$-correctness upon rate intervals.

Definition 4.7 ($\boldsymbol{\gamma}$-Correctness) *Given consonance interval* $\boldsymbol{\gamma}_p = [\gamma_p^-, 0, \gamma_p^+]$ *and a clock* $\mathcal{C}_p$ *with its rate* $v_p(t)$. *A rate interval* $\mathbf{R}_p$ *is* $\boldsymbol{\gamma}_p$*-correct w.r.t. internal global rate* $\varphi(t)$ *during a non-empty (real-time) interval* $\mathcal{T}$ *iff*

$$\varphi(t) \in v_p(t)\left(\mathrm{ref}(\mathbf{R}_p) + \boldsymbol{\gamma}_p\right) \qquad \forall t \in \mathcal{T}. \tag{4.49}$$

For an ensemble of clocks $\mathcal{C}_1, \ldots, \mathcal{C}_n$ *the associated set of correct rate intervals* $\boldsymbol{\mathcal{R}} = \{\mathbf{R}_1, \ldots, \mathbf{R}_n\}$ *is* $\boldsymbol{\gamma}$*-correct during* $\mathcal{T}$ *iff each of them is* $\boldsymbol{\gamma}$*-correct w.r.t. internal global rate* $\varphi(t)$ *during* $\mathcal{T}$.

Of course, $\boldsymbol{\gamma}$-correctness of rate intervals implies $\boldsymbol{\gamma}$-consonance but not necessarily the other way around. When the individual consonance and local rate intervals of clocks are available, the following lemma put back to back with Lemma 4.7 asserts the consonance of the ensemble.

Lemma 4.8 ($\boldsymbol{\gamma}$-Correctness of an ensemble) *If each correct local rate intervals* $\mathbf{R}_p$ *for all* $1 \leq p \leq n$ *is* $\boldsymbol{\gamma}_p$*-correct w.r.t. internal global rate* $\varphi(t)$ *during a non-empty (real-time) interval* $\mathcal{T}$*, then the set* $\{\mathbf{R}_1, \ldots, \mathbf{R}_n\}$ *is* $\boldsymbol{\gamma}$*-correct during* $\mathcal{T}$ *for any*

$$\boldsymbol{\gamma} \supseteq \bigcup_{p=1}^{n} \boldsymbol{\gamma}_p. \tag{4.50}$$

Proof. Let $\boldsymbol{\gamma}$ be an interval such that $\boldsymbol{\gamma} \supseteq \cup_{p=1}^{n} \boldsymbol{\gamma}_i$. Suppose $\bigcap_{p=1}^{n} v_p(1+\boldsymbol{\gamma}) = \emptyset$, then there exist at least two disjunct intervals $v_i(1+\boldsymbol{\gamma})$ and $v_j(1+\boldsymbol{\gamma})$. Since $\boldsymbol{\gamma}_i, \boldsymbol{\gamma}_j \subseteq \boldsymbol{\gamma}$ we conclude that $v_i(1+\boldsymbol{\gamma}_i) \cap v_j(1+\boldsymbol{\gamma}_j) = \emptyset$, which contradicts with the correctness assumption of either $\mathbf{R}_i$ or $\mathbf{R}_j$. □

Associated consonance intervals can be deteriorated locally and transferred between nodes in the same manner as rate intervals, cf. Lemma 4.3 and 4.6. This is obvious, since internal global rate $\varphi(t)$ takes over the role of the ideal rate 1 in the corresponding relations. The following two lemmas repeat these properties.

Lemma 4.9 (Deterioration of Consonance Intervals) *Given a clock* $\mathcal{C}_p$ *paced by an oscillator with stability* σ_p*. If local rate interval* $\mathbf{R}_p$ *is correct and* $\boldsymbol{\gamma}_p$*-correct at* T_1*, then it is*

$$\left(\boldsymbol{\gamma}_p + \left[0 \pm \frac{\sigma_p(T_2 - T_1)}{1 - 2||\mathbf{R}_p||}\right] + \left[0 \pm \mathcal{O}\left(\sigma_p^2(T_2 - T_1)^2\right)\right]\right) - \text{correct} \tag{4.51}$$

at $T_2 \geq T_1$*, when no resynchronizations occurred in between.*

Proof. The same line of reasoning as in the proof for Lemma 4.3 can be applied, but instead of the ideal rate 1 the internal global rate $\varphi(t)$ is enclosed, which is identical at T_1 and T_2 if no resynchronizations take place. □

Lemma 4.10 (Consonance of Remote Rate Interval) *Let* $\mathcal{C}_p$ *and* $\mathcal{C}_q$ *be clocks driven by oscillators with stability* σ_p *and* σ_q*, respectively. As shown in Figure 4.5, the messages for the relative rate measurement are sent from node* q *at* $T_q = C_q(t_q)$ *resp.* $T_q^{\prec} = C_q(t_q^{\prec})$ *and received at node* p *at* $T_p = C_p(t_p)$ *resp.* $T_p^{\prec} = C_p(t_p^{\prec})$*. Furthermore, let* $\mathbf{Q}_{p,q}$ *be a quotient rate interval during* $[t_p^{\prec}, t_p]$ *resp.* $[t_q^{\prec}, t_q]$*. If local rate interval* $\mathbf{R}_q$ *is* $\boldsymbol{\gamma}_q$*-correct at* $t_q^{\prec}$ *on remote node* q*, then* $\mathbf{R}_{p,q} = \mathbf{Q}_{p,q} \cdot \mathbf{R}_q$ *is* $\boldsymbol{\gamma}_{p,q}$*-correct at* t_p*, whereby*

$$\boldsymbol{\gamma}_{p,q} = \mathbf{Q}_{p,q} \cdot \boldsymbol{\gamma}_q \tag{4.52}$$

and no resynchronizations occurred in between.

Proof. Similar to the proof of Lemma 4.6 by starting out with $\varphi(t_q^{\prec}) \in v_q(t_q^{\prec})(1 + \boldsymbol{\gamma}_q)$, plugging in $v_q(t_q^{\prec})$ taken from (4.44), and noting that $\varphi(t_p) = \varphi(t_q^{\prec})$. □

4.4 Clock Rate Algorithm

In this section we develop an algorithm apt for both internal and external clock rate synchronization based on the building blocks from Section 4.3. As introduced in Section 4.2, we assume a distributed system of $n \geq 2$ nodes, connected by a communication subsystem (Assumption 4.4), where each node p hosts a processor (Assumption 4.3) and a clock $\mathcal{C}_p$ (Assumption 4.2) driven by an oscillator $\mathcal{O}_p$ (Assumption 4.1). Our starting point is the round structure of this algorithm followed by its generic description. A simple example provides a good insight how the algorithm works. Afterwards we focus on convergence functions responsible for internal clock rate synchronization, and on validation functions for external clock rate synchronization. Issues concerning abstract fault models (see Definition 4.11) can be found at the end of this section.

4.4.1 Round Structure

First we lay out the structure of our algorithm CRA, which is based on rounds as known from other distributed algorithms. With *rate resynchronization period* P_R each node executes the same algorithm, which consists basically of relative rate measurements and the computation of proper rate adjustments. The rounds are a product of a coexistent CSA with *state resynchronization period* P_S, whereby precision $\pi_{\max}$ does not need to be very small to make our algorithm working. In fact, a round-less version is also conceivable by an adaptation of the value of $\pi_{\max}$ and the ensuing analysis. To facilitate an easy implementation, P_R should be an integer multiple m of P_S, since the engendered clock synchronization traffic from the CSA can be reused by the CRA accordingly, see Figure 4.6.

Let us examine a particular rate round k, which starts at $(k-1)P_R + F + E$ lasting a duration of P_R logical seconds. The rates of the clocks are going to be adjusted at these points in time symbolized by circles in Figure 4.6. During a round, we need to carry out the protocol of Figure 4.5 in order to make relative rate measurements. More specifically, shortly before the beginning of round k, we initiate a *full message exchange* (FME) for the first messages, and near the end of the round for the second ones. These FMEs are already generated by the coexisting CSA that adjusts the state of the clocks at the times symbolized by both squares and circles in Figure 4.6. Intervening state synchronizations are allowed to occur during relative rate measurements, since they can be easily taken into account by summing up the appropriate state adjustments. Unfortunately, the rate adjustment at $(k-1)P_R + F + E$ happens during the relative rate measurement, but it has only a minor influence because $F + E \ll P_R$, see Section 4.5.2. As an alternative,

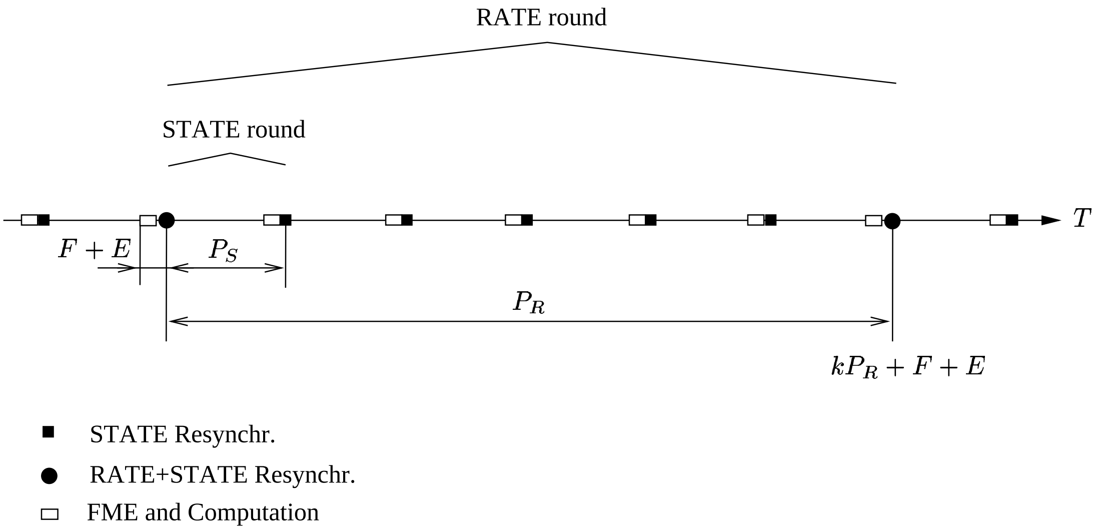


Figure 4.6: *Rounds*

we could initiate an additional FME shortly after $(k-1)P_R + F + E$ which increases the communication expense, or use the next FME for clock state synchronization which leaves a “hole” of size P_S in the measurements.

To make our algorithm running properly, we have to set up large enough delays to account for the longest possible duration of any FME and any execution of algorithm CRA. In particular, delay F covers the worst case FME duration and E the worst case execution time. The following lemma helps us to assign F and E.

Lemma 4.11 (FME and Execution Duration) *Complying to Assumptions 4.1–4.4, any full message exchange is completed within*

$$F = (1 + 2\rho_{\max}) \left(\frac{\pi_{\max}}{1 - 2\rho_{\max}} + \omega_{\max} + \lambda_{\max} + \Delta t_{\max} + \epsilon_{\max} \right) \tag{4.53}$$

logical seconds, and any single execution for the clock rate synchronization algorithm within

$$E = (1 + 2\rho_{\max})\eta_{\max} \tag{4.54}$$

logical seconds, when the participating nodes are non-faulty.

Proof. We make use of the assumed precision $\pi_{\max}$ of non-faulty clocks provided by Assumption 4.2. Suppose clock $\mathcal{C}_p$ is $\pi_{\max}$ ahead of clock $\mathcal{C}_q$, and the former is the fastest and the latter the slowest in our ensemble characterized by the uniform maximal oscillator drift $\rho_{\max}$ from (4.8), see also Assumption 4.1 part (2). Let t^I be the real-time when node p initiates as first one its broadcast of a particular FME. We know that $C_q(t^I) \geq C_p(t^I) - \pi_{\max}$, thus by virtue of (4.5) and Lemma 4.2 node q takes at most $\pi_q \leq \pi_{\max}/(1-2\rho_{\max})$ to initiate its broadcast as the latest participant of the same FME. Before the packet from node q gets transmitted, it can experiences a maximum broadcast operation delay $\omega_{\max}$ and a maximum broadcast latency $\lambda_{\max}$, see Assumption 4.4 part (1) and (2). Furthermore, the transmission itself can take as long as $\Delta t_{\max} + \epsilon_{\max}$ to reach its peer, see Assumption 4.4 part (3). Since we are interested in the worst case duration of any FME, we map the sum of expired real-times onto clock p, hence

$$F = C_p(t^I + \pi_q + \lambda_{\max} + \omega_{\max} + \Delta t_{\max} + \epsilon_{\max}) - C_p(t^I).$$

An application of Lemma 4.2 in combination with (4.5) proves (4.53).

For the second part, we know from Assumption 4.3 part (2) that any execution of algorithm CRA does not take longer than $\eta_{\max}$ seconds. Once again, by mapping this duration onto the fastest clock, we get the desired result (4.54). □

Given the worst case duration for both FME and execution, we are able to quantify the rate resynchronization period P_R. It cannot be smaller than the state resynchronization period P_S, which in turn has to be at least $F+E$ to manage a single FME along with the execution of the algorithm. Note that we did not consider any staggering of the initiation times of the broadcast to avoid a peak load in the communication subsystem.

4.4.2 Generic Algorithm

Now we have prepared all parts and join them together in the following algorithm.

Definition 4.8 (Clock Rate Algorithm) *Each node p in the system performs the following operations:*

(S) CSP Send: At kP_R initiate broadcast of timestamped packets $\langle \mathbf{R}_p, U_p, T_p \rangle$

(R) CSP Reception: Until $kP_R + F$ receive packets $\langle \mathbf{R}_q, U_q, T_q \rangle$ from nodes q and timestamp them by $T_{p,q}$

(C) Computation: At $kP_R + F$ compute

- *the set of quotient rate intervals $\mathfrak{Q}_p$, whereby*
  $\mathbf{Q}_{p,q} \leftarrow \left[\frac{T_q - T_q^{\prec} + U_q}{T_{p,q} - T_{p,q}^{\prec} + U_p} \pm \left(\frac{(\sigma_p + \sigma_q)(T_q - T_q^{\prec} + U_q)}{2} + \frac{\epsilon_{\max}}{T_{p,q} - T_{p,q}^{\prec} + U_p}\right)\right]$ *for* $q \neq p$

- *the set of remote rate intervals* $\boldsymbol{\mathcal{R}}_p$*, whereby*
  $\mathbf{R}_{p,q} \leftarrow \mathbf{Q}_{p,q} \cdot \mathbf{R}_q + [0 \pm (\sigma_p + \sigma_q)(F+E)]$ *for* $q \neq p$ *and* $\mathbf{R}_{p,p} \leftarrow \mathbf{R}_p$.
- *the new rate interval* $\tilde{\mathbf{R}}_p$ *as application of* $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ *and* $\boldsymbol{\mathcal{VAL}}_{\mathcal{F}}(\cdot)$ *over* $\boldsymbol{\mathcal{R}}_p$.
- *the new local rate interval* $\mathbf{R}_p \leftarrow \operatorname{norm}(\tilde{\mathbf{R}}_p) + [0 \pm \sigma_p P_R]$

(T) *Termination and Resynchronization:* *At* $kP_R + F + E$ *adjust clock rate by setting* $S_p \leftarrow \operatorname{ref}(\tilde{\mathbf{R}}_p) S_p$

Periodically, at kP_R each node p initiates a broadcast of timestamped packet(s), which contain the local rate interval $\mathbf{R}_p$ that is correct throughout round k, the sum U_p of applied state adjustments during round k, and the sending timestamp T_p, see Figure 4.6. Until $kP_R + F$ each node p receives packets from other nodes q containing as well the local rate interval $\mathbf{R}_q$ that is correct throughout round k, the sum U_q of applied state adjustments during round k, and the sending timestamp T_q. These packets will be timestamped with $T_{p,q}$ upon reception and collected for further processing. Remembering the timestamps $T_q^{\prec}$ and $T_{p,q}^{\prec}$ of the former rate round enables us to carry out the protocol for relative rate measurement, see Figure 4.5. The quotient rate interval $\mathbf{Q}_{p,q}$ can be computed by (4.34) of Lemma 4.5 in a simplified version; note also that $\mathbf{Q}_{p,p}$ is not explicitly required because $\mathbf{R}_{p,p}$ is just $\mathbf{R}_p$. Next the received local rate intervals $\mathbf{R}_q$ are transformed into remote rate intervals $\mathbf{R}_{p,q}$ by virtue of (4.42) of Lemma 4.6. An additional deterioration ensures a proper matching between the rate measurement period and the round endpoints in order make all $\mathbf{R}_{p,q}$ compatible with each other for the future resynchronization point $kP_R + F + E$, see Lemma 4.15.

Subsequently, the remote rate intervals $\mathbf{R}_{p,q}$ are fed into an interval-based *convergence function* $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ and/or *validation function* $\boldsymbol{\mathcal{VAL}}_{\mathcal{F}}(\cdot)$ to compute a new rate interval $\tilde{\mathbf{R}}_p$ for adjustment purposes. It should be both correct (required for external rate synchronization) and $\tilde{\gamma}$-correct (required internal rate synchronization) for a certain $\tilde{\gamma}$. Sections 4.4.4 and 4.4.5 are devoted to these functions.

Here it remains to explain how the algorithm accomplishes the rate adjustment at $kP_R + F + E$. In particular, we need to determine a new factor S_p for the oscillator-clock coupling and a new local rate interval $\mathbf{R}_p$. The first issue is straightforward, since we warrant that $1 \in v_p\tilde{\mathbf{R}}_p$ at $kP_R + F + E$. Our best approximation for the clock rate is given by $1/\operatorname{ref}(\tilde{\mathbf{R}}_p)$ according to Definition 4.4, hence we reset S_p in a multiplicative way by $\operatorname{ref}(\tilde{\mathbf{R}}_p)S_p$, i.e.,

$$S_p \leftarrow \operatorname{ref}(\tilde{\mathbf{R}}_p) S_p \tag{4.55}$$

to enforce the new clock rate. Recalling (4.5), we additionally check for $|S_p f_p - 1| \leq \rho_p^{\max}$ to ensure a feasible rate adjustment, otherwise the clock is declared as faulty[†]. The other issue deals with the computation of local rate interval $\mathbf{R}_p$ in respect to the new rate $\tilde{v}_p = \text{ref}(\tilde{\mathbf{R}}_p)v_p$. Because of $1 \in (\tilde{v}_p/\text{ref}(\tilde{\mathbf{R}}_p))\tilde{\mathbf{R}}_p$ we can set

$$\mathbf{R}_p = \frac{\tilde{\mathbf{R}}_p}{\text{ref}(\tilde{\mathbf{R}}_p)} = \text{norm}(\tilde{\mathbf{R}}_p). \tag{4.56}$$

A following deterioration of $\mathbf{R}_p$ via (4.22) of Lemma 4.3 again in a simplified version guarantees that this local rate interval is correct throughout the upcoming round $(k+1)$.

In the initial case $k = 0$ at t_0, we begin with coupling factor $S_p = 1/f_p$ and local rate interval $\mathbf{R}_p = [\rho_p/(1+\rho_p), 1, \rho_p/(1-\rho_p)]$ as justified by Lemma 4.1. The results of analyzing algorithm from Definition 4.8, in particular, the worst case consonance and the run of drifts will be given in Theorem 4.1 and 4.2, respectively.

4.4.3 Example

A simple example should demonstrate our algorithm. We assume three nodes with stable clocks ($\sigma_{\max} = 0$) having rates $v_1 = 0.8$, $v_2 = 0.9$ and $v_3 = 1.3$, respectively. Therefore the drift $\delta = 0.3$ for any clock and the consonance $\gamma = 0.5$ of the ensemble. Clearly, the clock rates are not directly observable, but local rate intervals $\mathbf{R}_1 = [0.15, 1, 0.3]$, $\mathbf{R}_2 = [0.2, 1, 0.4]$ and $\mathbf{R}_3 = [0.3, 1, 0.1]$ capture them by fulfilling condition $1 \in v_p\mathbf{R}_p$ for all $1 \leq p \leq 3$ from Definition 4.4.

The protocol for relative rate measurement together with Lemma 4.5 yields the quotient rate intervals. When neglecting message delivery uncertainties ($\epsilon_{\max} = 0$) we can immediately calculate $\mathbf{Q}_{p,q} = v_q/v_p$ for all $1 \leq p, q \leq 3$. In addition local rate intervals are exchanged and subsequently transformed into remote rate intervals by use of $\mathbf{R}_{p,q} = \mathbf{Q}_{p,q} \cdot \mathbf{R}_q$ from Lemma 4.6. The resulting matrix of remote rate interval reads

$$\mathbf{R} = \begin{pmatrix} [0.15, 1, 0.3] & [0.225, 1.125, 0.45] & [0.488, 1.625, 0.163] \\ [0.133, 0.889, 0.267] & [0.2, 1, 0.4] & [0.433, 1.444, 0.144] \\ [0.092, 0.615, 0.185] & [0.138, 0.692, 0.278] & [0.3, 1, 0.1] \end{pmatrix}.$$

Only local information was used to calculate the remote rate intervals. Figure 4.7 depicts them from a global perspective, where each dashed block represents the local view

---

[†] This operation touches an open problem, since due to a potential tradeoff between internal and external clock (rate) synchronization, it could become necessary to steer a clock beyond its maximum oscillator drift to meet the consonance requirement. In this case we have to be careful to keep up with the round structure, since delays F and E are depending on this drift.

of a node. Qualitatively we can say that nodes have a similar view up to a particular shift (nodes with faster clocks to the left) and stretch (nodes with slower clocks possess larger intervals).

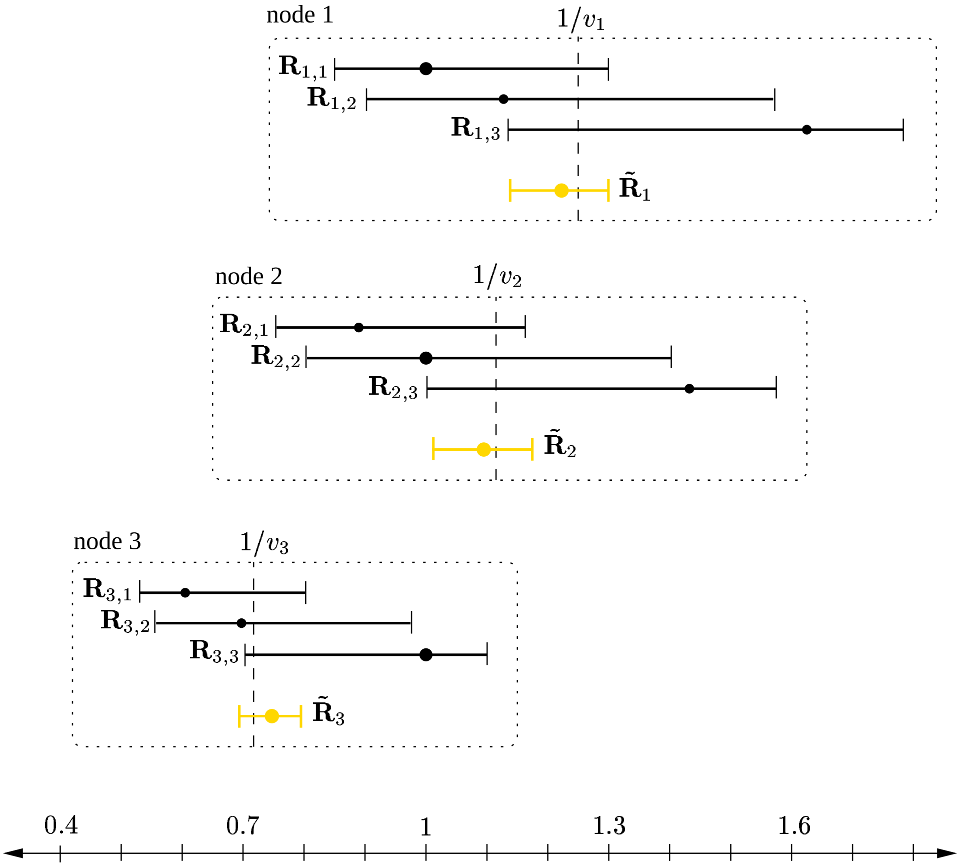


Figure 4.7: *Global View of Remote Rate Intervals*

Let's take a closer look at node 1. Since all its remote rate intervals include the desired reciprocal rate $1/v_1 = 1.25$ in our example, plain intersection is used as convergence function along with setting the reference point in the middle. We get $\tilde{\mathbf{R}}_1 = [0.082, 1.218, 0.082]$ as the new rate interval, which is rather short due to averaging effects. For adjusting the clock rate we multiply the oscillator-clock coupling with 1.218 leading to the new rate $\tilde{v}_1 = 0.9744$. Moreover the new local rate interval in respect to the changed rate becomes to $\mathbf{R}_1 = [0.067, 1, 0.067]$, which again goes hand in hand with Definition 4.4. Analogous results hold for the remaining nodes. Not surprisingly, these are excellent outcomes, but we should bear in mind the idealized assumptions.

4.4.4 Convergence Function

We know that each correct remote rate interval $\mathbf{R}_{p,q}$ computed by node p contains the searched value of $1/v_p$, which is the anchor for rate adjustments. Informally, the crux is to use a convergence function $\mathcal{CV}_{\mathcal{F}}(\cdot)$ for computing a new interval $\tilde{\mathbf{R}}_p$ out of these intervals that encloses $1/v_p$ more closely. However, in a realistic system failures may lead to erroneous intervals. An (abstract) fault model $\mathcal{F}$ has to be established that restricts the variety of faults in terms of the provided rate/consonance intervals, see Section 4.4.6. A suitable convergence function $\mathcal{CV}_{\mathcal{F}}(\cdot)$ is based on a such an abstract fault model, thus the subscript $\mathcal{F}$. Examples for $\mathcal{CV}_{\mathcal{F}}(\cdot)$ range from plain intersection, over Marzullo's function in [35], to the orthogonal-accuracy convergence function in [52] that, e.g., provides the usual 1/3 tolerance against arbitrary faults.

For a better understanding of internal rate synchronization, recall the notion of $\boldsymbol{\gamma}$-correctness introduced in Definition 4.7. Suppose that each local rate interval belonging to a non-faulty clock is $\boldsymbol{\gamma}_0$-correct w.r.t. $\varphi(t)$ at the beginning of a particular round k. During the round we have to deteriorate the consonance intervals in order to compensate for the stability of oscillators according to Lemma 4.9. Choosing two different non-faulty nodes p and q, which have acquired a set of remote rate intervals $\{\mathbf{R}_{p,1}, \ldots, \mathbf{R}_{p,n}\}$ and $\{\mathbf{R}_{q,1}, \ldots, \mathbf{R}_{q,n}\}$, respectively. The non-faulty ones among them are $\boldsymbol{\gamma}$-correct at the end of round k for a suitable $\boldsymbol{\gamma}$ according to Lemma 4.10.

Applying the convergence function at node p resp. q yields $\tilde{\mathbf{R}}_p = \mathcal{CV}_{\mathcal{F}}(\mathbf{R}_{p,1}, \ldots, \mathbf{R}_{p,n})$ resp. $\tilde{\mathbf{R}}_p = \mathcal{CV}_{\mathcal{F}}(\mathbf{R}_{q,1}, \ldots, \mathbf{R}_{q,n})$ that has to preserve/enhance consonance. More specifically, the clock rates have to be manipulated in such a way that the ensuing new local rate intervals become $\boldsymbol{\gamma}_0$-consonant with $\boldsymbol{\gamma}_0 \subset \boldsymbol{\gamma}$. We emphasize that they cannot be safely asserted as $\boldsymbol{\gamma}_0$-correct here, since it may be the case that the internal global rate $\varphi(t)$ from round k does not fit for the new consonance intervals. Defining a suitable new internal global rate resolves this deficiency and reassures $\boldsymbol{\gamma}_0$-correctness. As a result, our imaginary internal global rate $\varphi(t)$ makes discrete leaps at rate resynchronization instants and remains constant otherwise, see Section 4.5.1.

Formally, in continuation of [55] and earlier [58], a convergence function $\mathcal{CV}_{\mathcal{F}}(\cdot)$ for rate synchronization will be characterized by a *consonance preservation function* $\boldsymbol{\Phi}_{\gamma}(\cdot)$, a *consonance enhancement function* $\Psi_{\gamma}(\cdot)$, and *drift preservation functions* $\Phi_{\delta}^{\pm}(\cdot)$. Their arguments are consonance/rate intervals that specify the possible set of non-faulty remote rate intervals fed into $\mathcal{CV}_{\mathcal{F}}(\cdot)$. For the analysis of our algorithm from Definition 4.8 it suffices to require that a particular convergence function $\mathcal{CV}_{\mathcal{F}}(\cdot)$ is *translation invariant* as well as *weakly monotonic* in the sense of Definition 4.9 and complies to Definition 4.10.

Definition 4.9 (Translation Invariance, Weak Monotonicity) *Given two sets* $\boldsymbol{\mathcal{I}} = \{\mathbf{I}_1, \dots, \mathbf{I}_n\}$ *and* $\boldsymbol{\mathcal{J}} = \{\mathbf{J}_1, \dots, \mathbf{J}_n\}$ *of* $n \geq 1$ *intervals. An interval-valued function* $\boldsymbol{f}(\cdot)$ *is called* translation invariant *iff for any real* Δ

$$\boldsymbol{f}(\mathbf{I}_1 + [0, \Delta, 0], \dots, \mathbf{I}_n + [0, \Delta, 0]) = \boldsymbol{f}(\mathbf{I}_1, \dots, \mathbf{I}_n) + [0, \Delta, 0], \tag{4.57}$$

and weakly monotonic *iff* $\mathbf{I}_i \subseteq \mathbf{J}_i$ *with* $\mathrm{ref}(\mathbf{I}_i) = \mathrm{ref}(\mathbf{J}_i)$ *for all* $1 \leq i \leq n$ *implies*

$$\boldsymbol{f}(\mathbf{I}_1, \dots, \mathbf{I}_n) \subseteq \boldsymbol{f}(\mathbf{J}_1, \dots, \mathbf{J}_n). \tag{4.58}$$

Definition 4.10 (Generic Convergence Functions) *Let* $\boldsymbol{\mathcal{R}}_p = \{\mathbf{R}_{p,1}, \dots, \mathbf{R}_{p,n}\}$ *and* $\boldsymbol{\mathcal{R}}_q = \{\mathbf{R}_{q,1}, \dots, \mathbf{R}_{q,n}\}$ *be two ordered sets of remote rate intervals in accordance with a given abstract fault model* $\mathcal{F}$ *(see Definition 4.11), where non-faulty members* $\mathbf{R}_{p,i}$ *and* $\mathbf{R}_{q,i}$ *for suitable* $i \in I \subseteq \{1, \dots, n\}$ *are subject to the following preconditions:*

[1] $\mathbf{R}_{p,i}$ *is correct and* $\mathbf{R}_{q,i}$ *is correct,*

[2] $\mathbf{R}_{p,i}$ *is* $\boldsymbol{\gamma}_{p,i}$*-correct and* $\mathbf{R}_{q,i}$ *is* $\boldsymbol{\gamma}_{q,i}$*-correct,*

[3] $\{\mathbf{R}_{p,i}, \mathbf{R}_{q,i}\}$ *is* $\boldsymbol{\gamma}^i$*-correct,*

[4] $\bigcup_{i \in I} \boldsymbol{\gamma}^i \subseteq \boldsymbol{\gamma}_H$*,*

[5] $\{\mathbf{R}_{p,i}, \mathbf{R}_{q,i}\}$ *is* $\boldsymbol{\gamma}_I$*-consonant with* $||\boldsymbol{\gamma}_I|| < ||\boldsymbol{\gamma}_H||$*,*

[6] $\mathbf{R}_{p,i}$ *satisfies* $\mathrm{align}(\mathbf{R}_{p,i}) \subseteq \mathbf{V}_{p,i}$*, and* $\mathbf{R}_{q,i}$ *satisfies* $\mathrm{align}(\mathbf{R}_{q,i}) \subseteq \mathbf{V}_{q,i}$*.*

Provided that $\tilde{\mathbf{R}}_p = \boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\{\mathbf{R}_{p,1}, \dots, \mathbf{R}_{p,n}\})$ *and* $\tilde{\mathbf{R}}_q = \boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\{\mathbf{R}_{q,1}, \dots, \mathbf{R}_{q,n}\})$ *a generic convergence function* $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ *is characterized by*

(1) *the correctness of both* $\tilde{\mathbf{R}}_p$ *and* $\tilde{\mathbf{R}}_q$*,*

(2) *a weakly monotonic* consonance preservation function $\boldsymbol{\Phi}_\gamma(\cdot)$ *iff*

$$\tilde{\mathbf{R}}_p \text{ is } \boldsymbol{\Phi}_\gamma(\boldsymbol{\gamma}_{p,1}, \dots, \boldsymbol{\gamma}_{p,n}; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots) - \text{correct and} \tag{4.59}$$

$$\tilde{\mathbf{R}}_q \text{ is } \boldsymbol{\Phi}_\gamma(\boldsymbol{\gamma}_{q,1}, \dots, \boldsymbol{\gamma}_{q,n}; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots) - \text{correct} \tag{4.60}$$

with $||\boldsymbol{\Phi}_\gamma(\boldsymbol{\gamma}^1, \dots, \boldsymbol{\gamma}^n; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots)|| = \mathcal{O}(||\boldsymbol{\gamma}_H||)$*,*

(3) *a weakly monotonic* consonance enhancement function $\Psi_\gamma(\cdot)$ *iff*

$$\{\tilde{\mathbf{R}}_p, \tilde{\mathbf{R}}_q\} \textit{ is } \tilde{\boldsymbol{\gamma}} - \text{consonant satisfying } ||\tilde{\boldsymbol{\gamma}}|| = \Psi_\gamma(\boldsymbol{\gamma}^1, \dots, \boldsymbol{\gamma}^n; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots) \tag{4.61}$$

with $\Psi_\gamma(\boldsymbol{\gamma}^1, \dots, \boldsymbol{\gamma}^n; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots) < ||\boldsymbol{\gamma}_H||$*, and*

(4) weakly monotonic drift preservation functions $\Phi_\delta^\pm(\cdot)$ *iff*

$$\begin{aligned}\operatorname{align}(\tilde{\mathbf{R}}_p) \;\subseteq\; & \big[\Phi_\delta^-(\mathbf{V}_{p,1},\ldots,\mathbf{V}_{p,n};\boldsymbol{\gamma}_{p,1},\ldots,\boldsymbol{\gamma}_{p,n};\boldsymbol{\gamma}_H;\boldsymbol{\gamma}_I;\ldots),0,\\ & \quad \Phi_\delta^+(\mathbf{V}_{p,1},\ldots,\mathbf{V}_{p,n};\boldsymbol{\gamma}_{p,1},\ldots,\boldsymbol{\gamma}_{p,n};\boldsymbol{\gamma}_H;\boldsymbol{\gamma}_I;\ldots)\big] \qquad (4.62)\end{aligned}$$

and

$$\begin{aligned}\operatorname{align}(\tilde{\mathbf{R}}_q) \;\subseteq\; & \big[\Phi_\delta^-(\mathbf{V}_{q,1},\ldots,\mathbf{V}_{q,n};\boldsymbol{\gamma}_{q,1},\ldots,\boldsymbol{\gamma}_{q,n};\boldsymbol{\gamma}_H;\boldsymbol{\gamma}_I;\ldots),0,\\ & \quad \Phi_\delta^+(\mathbf{V}_{q,1},\ldots,\mathbf{V}_{q,n};\boldsymbol{\gamma}_{q,1},\ldots,\boldsymbol{\gamma}_{q,n};\boldsymbol{\gamma}_H;\boldsymbol{\gamma}_I;\ldots)\big] \qquad (4.63)\end{aligned}$$

Let us briefly make of few informally remarks about the above definition. Preconditions [1]–[6] specify the non-faulty remote rate intervals fed into the generic convergence function $\mathcal{CV}_\mathcal{F}(\cdot)$ at two different nodes p and q. Section 4.5.2 will bring up bounds on the respective parameters. Obviously, item (1) requires that the computed rate intervals $\tilde{\mathbf{R}}_p$ and $\tilde{\mathbf{R}}_q$ need to be correct for external clock rate synchronization. Item (2) helps to obtain the maximum amount of rate adjustments administered at the end of a round, since $\boldsymbol{\Phi}_\gamma(\cdot)$ expresses how well the new rate intervals fit into the former round. Item (3) holds the key to maintain the consonance of our clocks, because $\Psi_\gamma(\cdot)$ asserts a consonance property of the new rate intervals that allows to determine a new internal global rate. Finally, item (4) bounds the length of the new rate intervals via $\Phi_\delta^\pm(\cdot)$ to account for the growth of clock drifts separated in a left and right length, which allows a strengthened analysis.

The properties of a particular convergence function for clock rate synchronization can be either derived directly by adhering to the above characterization or, preferably, by taking over a convergence function for clock state synchronization as defined in Chapter 3, see also [55]. The following lemma provides the correspondence between them, showing that their characteristic functions translate with only minor modifications.

Lemma 4.12 (Correspond. between Rate and State Convergence Functions) *Given a convergence function $\mathcal{CV}_\mathcal{F}(\cdot)$ for clock state synchronization characterized by Definition 3.11 along with its* precision preservation function $\boldsymbol{\Phi}_\pi(\cdot)$, precision enhancement function $\Psi_\pi(\cdot)$, *and* accuracy preservation functions $\Phi_\alpha^\pm(\cdot)$. *Additionally, all functions are* multiplicative *in the sense that* $\boldsymbol{f}(s\mathbf{I}_1,\ldots,s\mathbf{I}_n) = s\boldsymbol{f}(\mathbf{I}_1,\ldots,\mathbf{I}_n)$ *for any* $s \geq 0$. *Using $\mathcal{CV}_\mathcal{F}(\cdot)$ as the convergence function for clock rate synchronization as specified in Definition 4.10 we get:*

(1) $\mathcal{CV}_\mathcal{F}(\cdot)$ preserves correctness,

(2) $\boldsymbol{\Phi}_\gamma(\boldsymbol{\gamma}_{p,1},\dots,\boldsymbol{\gamma}_{p,n};\boldsymbol{\gamma}_H;\boldsymbol{\gamma}_I;\dots) \equiv \boldsymbol{\Phi}_\pi(\underbrace{\{\boldsymbol{\gamma}_{p,1},\dots,\boldsymbol{\gamma}_{p,n}\}}_{\boldsymbol{\mathcal{P}}_p},\underbrace{z\boldsymbol{\gamma}_H}_{\boldsymbol{\pi}^H},\underbrace{z\boldsymbol{\gamma}_I}_{\boldsymbol{\pi}_I})$

and analogous for $\boldsymbol{\Phi}_\gamma(\boldsymbol{\gamma}_{p,1},\dots,\boldsymbol{\gamma}_{p,n};\boldsymbol{\gamma}_H;\boldsymbol{\gamma}_I;\dots)$*,*

(3) $\Psi_\gamma(\boldsymbol{\gamma}^1,\dots,\boldsymbol{\gamma}^n;\boldsymbol{\gamma}_H;\boldsymbol{\gamma}_I;\dots) \equiv \Psi_\pi(\underbrace{\{z\boldsymbol{\gamma}^1,\dots,z\boldsymbol{\gamma}^n\}}_{\boldsymbol{\mathcal{P}}},\underbrace{z\boldsymbol{\gamma}_H}_{\boldsymbol{\pi}^H},\underbrace{z\boldsymbol{\gamma}_I}_{\boldsymbol{\pi}_I})$*,*

(4) $\Big[\Phi_\delta^-(\mathbf{V}_{p,1},\dots,\mathbf{V}_{p,n};\boldsymbol{\gamma}_{p,1},\dots,\boldsymbol{\gamma}_{p,n};\boldsymbol{\gamma}_H;\boldsymbol{\gamma}_I;\dots),0,$
$\Phi_\delta^+(\mathbf{V}_{p,1},\dots,\mathbf{V}_{p,n};\boldsymbol{\gamma}_{p,1},\dots,\boldsymbol{\gamma}_{p,n};\boldsymbol{\gamma}_H;\boldsymbol{\gamma}_I;\dots)\Big] \equiv$
$\Big[-\Phi_\alpha^-(\underbrace{\{\mathbf{V}_{p,1},\dots,\mathbf{V}_{p,n}\}}_{\boldsymbol{\mathcal{B}}_p},\underbrace{\{\boldsymbol{\gamma}_{p,1},\dots,\boldsymbol{\gamma}_{p,n}\}}_{\boldsymbol{\mathcal{P}}_p},\underbrace{z\boldsymbol{\gamma}_H}_{\boldsymbol{\pi}^H},\underbrace{z\boldsymbol{\gamma}_I}_{\boldsymbol{\pi}_I},\underbrace{0}_{\iota}),$
$\Phi_\alpha^+(\underbrace{\{\mathbf{V}_{p,1},\dots,\mathbf{V}_{p,n}\}}_{\boldsymbol{\mathcal{B}}_p},\underbrace{\{\boldsymbol{\gamma}_{p,1},\dots,\boldsymbol{\gamma}_{p,n}\}}_{\boldsymbol{\mathcal{P}}_p},\underbrace{z\boldsymbol{\gamma}_H}_{\boldsymbol{\pi}^H},\underbrace{z\boldsymbol{\gamma}_I}_{\boldsymbol{\pi}_I},\underbrace{0}_{\iota})\Big]$

and analogous for $\Big[\Phi_\delta^-(\mathbf{V}_{q,1},\dots,\mathbf{V}_{q,n};\boldsymbol{\gamma}_{q,1},\dots,\boldsymbol{\gamma}_{q,n};\boldsymbol{\gamma}_H;\boldsymbol{\gamma}_I;\dots),0,\Phi_\delta^+(\mathbf{V}_{q,1},\dots,$
$\mathbf{V}_{q,n};\boldsymbol{\gamma}_{q,1},\dots,\boldsymbol{\gamma}_{q,n};\boldsymbol{\gamma}_H;\boldsymbol{\gamma}_I;\dots)\Big]$

where $z = \max\{\frac{v_p}{v_q},\frac{v_q}{v_p}\} = 1 + \mathcal{O}(\rho_{\max})$.

Proof. Based on the two sets $\boldsymbol{\mathcal{R}}_p$ and $\boldsymbol{\mathcal{R}}_q$ of remote rate intervals that meet preconditions [1]–[6] of Definition 4.10, we construct another two sets $\boldsymbol{\mathcal{Z}}_p$ and $\boldsymbol{\mathcal{Z}}_q$ of "pseudo" state intervals that meet preconditions [1]–[4] of Definition 3.11 by setting

$$\mathbf{Z}_{p,i} := v_p\mathbf{R}_{p,i} \text{ and } \mathbf{Z}_{q,i} := v_q\mathbf{R}_{q,i} \qquad \forall 1 \le i \le n.$$

[1]: If a remote rate interval $\mathbf{R}_{p,i}$ resp. $\mathbf{R}_{q,i}$ is correct in the context of rate synchronization, then interval $\mathbf{Z}_{p,i}$ resp. $\mathbf{Z}_{q,i}$ is correct in the context of state synchronization, since $1 \in v_p\mathbf{R}_{p,i}$ resp. $1 \in v_q\mathbf{R}_{q,i}$ where $t := 1$. Furthermore, if $\operatorname{align}(\mathbf{R}_{p,i}) \subseteq \mathbf{V}_{p,i}$ then $\operatorname{align}(\mathbf{Z}_{p,i}) \subseteq v_p\mathbf{V}_{p,i}$, and if $\operatorname{align}(\mathbf{R}_{q,i}) \subseteq \mathbf{V}_{q,i}$ then $\operatorname{align}(\mathbf{Z}_{q,i}) \subseteq v_q\mathbf{V}_{q,i}$.

[2]: If a remote rate interval $\mathbf{R}_{p,i}$ resp. $\mathbf{R}_{q,i}$ is $\boldsymbol{\gamma}_{p,i}$-correct resp. $\boldsymbol{\gamma}_{q,i}$-correct in the context of rate synchronization, then interval $\mathbf{Z}_{p,i}$ resp. $\mathbf{Z}_{q,i}$ is $v_p\boldsymbol{\gamma}_{p,i}$-correct resp. $v_q\boldsymbol{\gamma}_{q,i}$-correct in the context of state synchronization, since $\varphi \in \operatorname{ref}(v_p\mathbf{R}_{p,i}) + v_p\boldsymbol{\gamma}_{p,i}$ resp. $\varphi \in \operatorname{ref}(v_q\mathbf{R}_{q,i}) + v_q\boldsymbol{\gamma}_{q,i}$ where $\tau := \varphi$. When $\{\mathbf{R}_{p,i},\mathbf{R}_{q,i}\}$ is $\boldsymbol{\gamma}^i$-correct in the context of rate synchronization, then $\{\mathbf{Z}_{p,i},\mathbf{Z}_{q,i}\}$ is $(\max\{v_p,v_q\}\boldsymbol{\gamma}^i)$-correct in the context of state synchronization, since φ is lies in both $\operatorname{ref}(v_p\mathbf{R}_{p,i}) + \max\{v_p,v_q\}\boldsymbol{\gamma}^i$ and $\operatorname{ref}(v_q\mathbf{R}_{q,i}) + \max\{v_p,v_q\}\boldsymbol{\gamma}^i$.

[3]: By the same token, all non-faulty intervals from $\boldsymbol{\mathcal{Z}}_p \cup \boldsymbol{\mathcal{Z}}_q$ are $(\max\{v_p,v_q\}\boldsymbol{\gamma}_H)$-correct in the context of state synchronization.

[4]: If any pair $\{\mathbf{R}_{p,i}, \mathbf{R}_{q,i}\}$ of non-faulty remote rate interval is $\boldsymbol{\gamma}_I$-consonant, then $\{\mathbf{Z}_{p,i}, \mathbf{Z}_{q,i}\}$ is $(\max\{v_p, v_q\}\boldsymbol{\gamma}_I)$-precise because $(\mathrm{ref}(v_p\mathbf{R}_{p,i}) + \max\{v_p, v_q\}\boldsymbol{\gamma}_I) \cap (\mathrm{ref}(v_q\mathbf{R}_{q,i}) + \max\{v_p, v_q\}\boldsymbol{\gamma}_I) \neq \emptyset$.

After the preconditions on $\boldsymbol{\mathcal{Z}}_p$ and $\boldsymbol{\mathcal{Z}}_q$ are settled, we know that $\tilde{\mathbf{Z}}_p = \boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\boldsymbol{\mathcal{Z}}_p)$ and $\tilde{\mathbf{Z}}_q = \boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\boldsymbol{\mathcal{Z}}_q)$ satisfy items (1)–(3) of Definition 3.11. Exploiting them, it remains to derive items (1)–(4) of Definition 4.10 upon $\tilde{\mathbf{R}}_p = \boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\boldsymbol{\mathcal{R}}_p)$ and $\tilde{\mathbf{R}}_q = \boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\boldsymbol{\mathcal{R}}_q)$. By using the multiplicativity of $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ we get immediately

$$\tilde{\mathbf{Z}}_p = v_p\tilde{\mathbf{R}}_p \text{ and } \tilde{\mathbf{Z}}_q = v_q\tilde{\mathbf{R}}_q.$$

Item (1) is a trivial consequence, since $\tilde{\mathbf{Z}}_p$ resp. $\tilde{\mathbf{Z}}_p$ is correct in the context of state synchronization, hence $1 \in v_p\tilde{\mathbf{R}}_p$ resp. $1 \in v_q\tilde{\mathbf{R}}_q$.

Item (2) uses item (2) of Definition 3.11 by setting $\boldsymbol{\pi}^H := \max\{v_p, v_q\}\boldsymbol{\gamma}_H$, $\boldsymbol{\pi}_I := \max\{v_p, v_q\}\boldsymbol{\gamma}_I$, and $\boldsymbol{\mathcal{P}}_p := \{v_p\boldsymbol{\gamma}_{p,1}, \dots, v_p\boldsymbol{\gamma}_{p,n}\}$. Plugging them into the precision preservation function $\boldsymbol{\Phi}_\pi(\cdot)$ by considering the signature of this function (look at the underbraces of our lemma) yields in the context of state synchronization

$$\varphi \in \mathrm{ref}(\tilde{\mathbf{Z}}_p) + \boldsymbol{\Phi}_\pi\left(\{v_p\boldsymbol{\gamma}_{p,1}, \dots, v_p\boldsymbol{\gamma}_{p,n}\}, \max\{v_p, v_q\}\boldsymbol{\gamma}_H, \max\{v_p, v_q\}\boldsymbol{\gamma}_I\right).$$

Due to the multiplicativity of $\boldsymbol{\Phi}_\pi(\cdot)$ this can be transferred into the context of rate synchronization

$$\varphi \in v_p\left(\mathrm{ref}(\tilde{\mathbf{R}}_p) + \boldsymbol{\Phi}_\pi\left(\{\boldsymbol{\gamma}_{p,1}, \dots, \boldsymbol{\gamma}_{p,n}\}, \max\{1, \frac{v_q}{v_p}\}\boldsymbol{\gamma}_H, \max\{1, \frac{v_q}{v_p}\}\boldsymbol{\gamma}_I\right)\right),$$

and the analogous holds for $\tilde{\mathbf{R}}_q$. To unify both cases we use factor $z = \max\{\frac{v_p}{v_q}, \frac{v_q}{v_p}\}$, which is asymptotically $1 + \mathcal{O}(\rho_{\max})$ due to (4.8).

Item (3) follows the same line of reasoning as above by setting $\boldsymbol{\pi}^H := \max\{v_p, v_q\}\boldsymbol{\gamma}_H$, $\boldsymbol{\pi}_I := \max\{v_p, v_q\}\boldsymbol{\gamma}_I$, and $\boldsymbol{\mathcal{P}} := \{\max\{v_p, v_q\}\boldsymbol{\gamma}^1, \dots, \max\{v_p, v_q\}\boldsymbol{\gamma}^n\}$. Plugging them into the multiplicative precision enhancement function $\Psi_\pi(\cdot)$ specifies a $\tilde{\boldsymbol{\gamma}}$ with

$$||\tilde{\boldsymbol{\gamma}}|| = \Psi_\pi\left(\{z\boldsymbol{\gamma}^1, \dots, z\boldsymbol{\gamma}^n\}, z\boldsymbol{\gamma}_H, z\boldsymbol{\gamma}_I\right)$$

according to item (3), which leads to $v_p(\mathrm{ref}(\tilde{\mathbf{R}}_p) + \tilde{\boldsymbol{\gamma}}) \cap v_q(\mathrm{ref}(\tilde{\mathbf{R}}_q) + \tilde{\boldsymbol{\gamma}}) \neq \emptyset$.

Item (4) can be shown by using item (1) and setting $\boldsymbol{\mathcal{P}}_p$, $\boldsymbol{\pi}^H$, $\boldsymbol{\pi}_I$ as above and $\boldsymbol{\mathcal{B}}_p := \{v_p\mathbf{V}_{p,1}, \dots, v_p\mathbf{V}_{p,n}\}$. The length of the common intesection $\iota = 0$, since we are content with are more conservative analysis upon the clock drifts. Plugging them into the accuracy preservation functions $\Phi^{\pm}_{\alpha}(\cdot)$ yields

$$\begin{aligned}\operatorname{align}(\tilde{\mathbf{Z}}_p) \;\subseteq\; \Big[&-\Phi^{-}_{\alpha}\Big(\{v_p\mathbf{V}_{p,1},\dots,v_p\mathbf{V}_{p,n}\},\{v_p\boldsymbol{\gamma}_{p,1},\dots,v_p\boldsymbol{\gamma}_{p,n}\},\\ &\qquad \max\{v_p,v_q\}\boldsymbol{\gamma}_H,\max\{v_p,v_q\}\boldsymbol{\gamma}_I,0\Big),\\ &\Phi^{+}_{\alpha}\Big(\{v_p\mathbf{V}_{p,1},\dots,v_p\mathbf{V}_{p,n}\},\{v_p\boldsymbol{\gamma}_{p,1},\dots,v_p\boldsymbol{\gamma}_{p,n}\},\\ &\qquad \max\{v_p,v_q\}\boldsymbol{\gamma}_H,\max\{v_p,v_q\}\boldsymbol{\gamma}_I,0\Big)\Big]\end{aligned}$$

and by virtue of its multiplicativity

$$\begin{aligned}\operatorname{align}(\tilde{\mathbf{R}}_p) \;\subseteq\; &\Big[\Phi^{-}_{\alpha}\Big(\{\mathbf{V}_{p,1},\dots,\mathbf{V}_{p,n}\},\{\boldsymbol{\gamma}_{p,1},\dots,\boldsymbol{\gamma}_{p,n}\},\max\{1,\frac{v_q}{v_p}\}\boldsymbol{\gamma}_H,\max\{1,\frac{v_p}{v_q}\}\boldsymbol{\gamma}_I,0\Big),\\ &0,\\ &\Phi^{+}_{\alpha}\Big(\{\mathbf{V}_{p,1},\dots,\mathbf{V}_{p,n}\},\{\boldsymbol{\gamma}_{p,1},\dots,\boldsymbol{\gamma}_{p,n}\},\max\{1,\frac{v_q}{v_p}\}\boldsymbol{\gamma}_H,\max\{1,\frac{v_p}{v_q}\}\boldsymbol{\gamma}_I,0\Big)\Big].\end{aligned}$$

The analogous holds for $\operatorname{align}(\tilde{\mathbf{R}}_q)$ unified by factor z. This finishes the proof about our lemma on the correspondence between rate and state convergence functions. □

4.4.5 Validation Function

For external rate synchronization we have to devise a mechanism to inject rate information from *primary nodes* $\mathcal{N}'$ in our system. Such nodes have their clocks disciplined towards a reference frequency obtained from atomic clocks or receivers for WWV, DCF77, or GPS, see [5]. Obviously, their local rate intervals and consequently their contributing remote rate intervals are of short lengths compared to the ones from *non-primary nodes* $\mathcal{N}''$. If n stands for the total number of nodes in our system, we can say that

$$n = |\mathcal{N}'| + |\mathcal{N}''|, \tag{4.64}$$

where in a realistic setting $|\mathcal{N}'| \ll |\mathcal{N}''|$.

Figure 4.8 depicts exemplary remote rate intervals at node p, where three come from primary and seven from non-primary nodes, including faulty ones. It is also shown that the reference points of remote rate intervals from non-primary nodes are close to the reciprocal of the internal global rate $\varphi(t)$ which guarantees a certain consonance, see Definition 4.6.

Just focusing on the set $\boldsymbol{\mathcal{X}}_p$ of remote rate intervals from primary nodes appears to be a promising solution for external rate synchronization for a node p, since the new rate

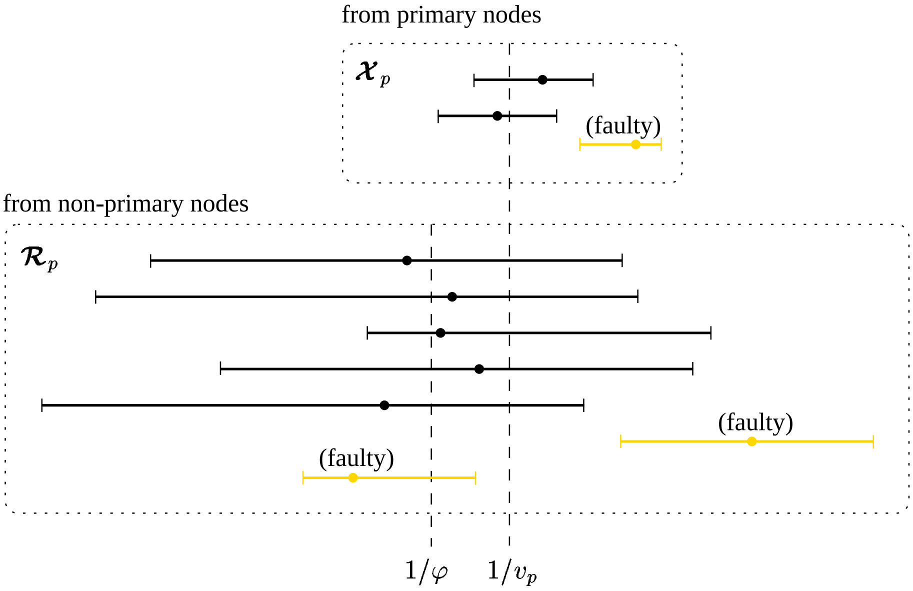


Figure 4.8: *Remote Rate Intervals from Primary and Non-primary Nodes*

interval will be of short length too. However, there are basically two difficulties to exploit them: First, they might not be continuously available which results in a toggling between internal and external rate synchronization. For justification, we carried out a 2-month continuous experimental evaluation of the output of six different GPS receivers, which revealed a wide variety of failures, see [16]. Second, an ordinary convergence function $\mathcal{CV}_{\mathcal{F}}(\cdot)$ is reluctant to obey them due to fault tolerance reasons.

One solution is the *clock validation* technique rooted in [50], which verifies whether the short but possibly faulty *reference* rate interval $\tilde{\mathbf{X}}_p$ stemming from nodes of $\mathcal{N}'$ is consistent with the longer but more reliable *validation* rate interval $\tilde{\mathbf{R}}_p$ from nodes of $\mathcal{N}''$. If the reference rate interval gets accepted, then clock $\mathcal{C}_p$ is adjusted by the clocks of $\mathcal{N}'$ accomplishing external rate synchronization. Otherwise the algorithm discards the reference rate interval and proceeds with the validation rate interval as it is the case of internal rate synchronization. Unfortunately, this straightforward technique works only in limited fault models $\mathcal{F}$, in particular, malicious faults could separate the ensemble into groups of clocks with different drifts, see [11] for further issues.

We are looking for new approaches embraced by a validation function $\mathcal{VAL}_{\mathcal{F}}(\cdot)$ that integrates these two sets of remote rate intervals in a meaningful way. For notational purpose, let $\mathcal{X}_p^{(k)} = \{\mathbf{X}_{p,q}^{(k)} : q \in \mathcal{N}'\}$ and $\mathcal{R}_p^{(k)} = \{\mathbf{R}_{p,q}^{(k)} : q \in \mathcal{N}''\}$ for round k. Moreover, inspired by [44] we are convinced that it is advantageous to consider the set $\{\mathbf{X}_{p,q} : q \in \mathcal{N}'\}$ from the preceding h rounds as well, which are maintained by virtue of Lemma 4.3

and denoted by $\boldsymbol{\mathcal{X}}_p^{(k-1)}, \ldots, \boldsymbol{\mathcal{X}}_p^{(k-h)}$. The validation function $\boldsymbol{\mathcal{VAL}}_{\mathcal{F}}(\cdot)$ in conjunction with a suitable convergence function $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ gets applied on all these sets of intervals in order to compute the rate interval $\tilde{\mathbf{R}}_p^{(k+1)}$ for the next round $k+1$, thus

$$\tilde{\mathbf{R}}_p^{(k+1)} = \boldsymbol{\mathcal{VAL}}_{\mathcal{F}}\left(\boldsymbol{\mathcal{R}}_p^{(k)}; \boldsymbol{\mathcal{X}}_p^{(k)}; \boldsymbol{\mathcal{X}}_p^{(k-1)}; \ldots; \boldsymbol{\mathcal{X}}_p^{(k-h)}\right). \tag{4.65}$$

4.4.6 Abstract Fault Model

So far we have only touched the issue of faults in our system by claiming a fault model $\mathcal{F}$ to keep our clock rate synchronization framework open for different fault assumptions. Such a fault model affects the choice of a particular convergence function $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ and validation function $\boldsymbol{\mathcal{VAL}}_{\mathcal{F}}(\cdot)$, since in case of simple fault scenarios a less sophisticated convergence/validation function should be adequate. Both functions call for certain fault-tolerance requirements, e.g. fraction of non-faulty primary or non-primary nodes.

In order to cope with the variety of system faults, we ignore the actual cause (e.g., faulty component or adversary) and pay attention on the resulting rate intervals. This explains the notion of an "abstract" fault model, see [55]. Rate intervals can be faulty in terms of their reference points and/or their lengths, leading to a catalogue of fault cases from which a specific $\mathcal{F}$ can be built. The following definition provides a taxonomy on faulty remote rate intervals; see Chapter 5 for further details.

Definition 4.11 (Abstract Fault Model) *An abstract fault model $\mathcal{F}$ specifies the type and number of potential faults in our system in terms of the obtained remote rate intervals during a round. Table 4.1 summarizes the elements of an abstract fault model, where the first column gives the name of the fault type, the second states the maximal number (a single prime refers to primary nodes and a double prime to non-primary nodes) of faulty remote rate intervals, the third provides a brief characterization, and the last column indicates whether such faulty remote rate intervals reside at a single node or at a pair of receiving nodes.*

Let us make a few remarks on the above introduced fault types. A miss fault at a single receiving node could be caused from omissive broadcasting nodes or from transient errors during message receptions. Note that two FMEs are necessary to compute the remote rate intervals. Such misses can be either consistent (crash fault) due to a node crash or inconsistent (omission fault) affecting different intervals at different nodes. The latter expands the traditionally view of missing intervals by considering the fact that most receive omissions occur independently. Also, a node that crashes during a broadcast operation might produce inconsistent receptions.

type	*max. number*	*remote rate interval(s)*	*receiving node(s)*
miss faults	n'_m, n''_m	missing	single
timing faults	n'_t, n''_t	non-correct and/or non-γ-correct due to faulty timing behavior	single
value faults	n'_v, n''_v	non-correct and/or non-γ-correct due to erroneous messages	single
crash faults	n'_c, n''_c	consistently missing	pair
omission faults	n'_o, n''_o	inconsistently missing	pair
restricted faults	n'_r, n''_r	consistently non-correct and/or non-γ-correct	pair
arbitrary faults	n'_a, n''_a	inconsistently non-correct and/or non-γ-correct	pair

Table 4.1: *Elements of an Abstract Fault Model*

Obtained remote rate intervals can be not correct according to Definition 4.4 (drift), to Definition 4.7 (consonance), or both. Here we do not examine further combinations, but we solely make the distinction whether they are caused by timing faults or by value faults. Examples for the former ones could be excessive delivery delays and for the latter damaged messages. In contrast to accuracy intervals, rate intervals are rather easy to check for meaningful lengths, so it is not necessary to consider *truncated*, *bounded* or even *unbounded* intervals, see [36].

Important for the selection of a suitable convergence/validation function is the assumption about the maximum number of consistent (restricted faults) and inconsistent (arbitrary faults) faulty remote rate intervals at two different receiving nodes having a broadcasting node in common. Arbitrary faults include the Byzantine case, which may be caused by nodes sending different messages to different receivers or by excessive transmission delays at receiving nodes. Such faults can also occur in broadcast type networks, since the broadcast operation is not assumed to be reliable, see Assumption 4.4. However, we rule out the possibility of *impersonating* other nodes or *jamming* the network.

4.5 Analysis

The analysis of our clock rate algorithm comes in five subsections concerning the round beginnings, the dissemination of rate and consonance intervals within a single round, the maximal applied rate adjustments, the worst case consonance (internal rate synchronization) and worst case drift (external rate synchronization) of clocks. The final results will be in terms of the characteristic functions of the employed convergence function along with a particular fault model. For practibility reasons, we work with "larger" $\mathcal{O}()$-terms in comparison to the ones of the building blocks in Section 4.3.

4.5.1 Round Beginnings

At the very first, we have to agree upon the real-time when a particular round starts, since clocks do not resynchronize simultaneously. According to Figure 4.6, the beginning of round $k \geq 0$ is given by $(k-1)P_R + F + E$, whereas the corresponding real-time of a non-faulty node p is $t_p^{(k)}$. Following the line of arguments from the proof of Lemma 4.11 and relying on the assumed precision $\pi_{\max}$ from Assumption 4.2, it is easy to show that

$$\left|t_p^{(k)} - t_q^{(k)}\right| \leq \frac{1+2\rho_{\max}}{1-2\rho_{\max}}\,\pi_{\max}, \tag{4.66}$$

where p and q are non-faulty nodes.

For the purpose of analysis, rate/consonance intervals are only meaningful when they refer to a common point of time. The beginning of round k denoted as $t^{(k)}$ is determined by the latest resynchronization time $t_p^{(k)}$ among non-faulty nodes $p \in \mathcal{N}^{(k)}$ during round k, thus

$$t^{(k)} = \max\{t_p^{(k)} : p \in \mathcal{N}^{(k)}\}. \tag{4.67}$$

Fortunately, since $\pi_{\max}$ is assumed to be very small compared to P_R we can justify that $v_p(t^{(k)}) = v_p(t_p^{(k)}) + \mathcal{O}(\sigma_p \pi_{\max})$ by rearranging (4.6) and recalling Assumption 4.1 item (3). Given this approximation, it is our goal to collapse the time span of resynchronizations of a particular round into a single point of time. In terms of rate intervals we assert that

$$\mathbf{R}_p(t^{(k)}) = \mathbf{R}_p(t_p^{(k)}) + [0 \pm \mathcal{O}(\sigma_p \pi_{\max})] \tag{4.68}$$

is correct at $t^{(k)}$ iff $\mathbf{R}_p(t_p^{(k)})$ is correct at $t_p^{(k)}$. Turning our attention to consonance intervals, we say the set $\boldsymbol{\mathcal{R}}^{(k)} = \{\mathbf{R}_p(t^{(k)}) : p \in \mathcal{N}^{(k)}\}$ of correct rate intervals is

$$\Big(\boldsymbol{\gamma}^{(k)} + [0 \pm \mathcal{O}(\sigma_{\max}\pi_{\max})]\Big) - \text{consonant} \tag{4.69}$$

at $t^{(k)}$ iff it holds that

$$\bigcap_{p \in \mathcal{N}^{(k)}} v_p(t_p^{(k)}) \left(\operatorname{ref}(\mathbf{R}_p(t_p^{(k)})) + \boldsymbol{\gamma}^{(k)}\right) \neq \emptyset$$

as a slight relaxation of Definition 4.6 in need of (4.66). Notice if there is no state synchronization or a too weak one in the sense that $\sigma_p \pi_{\max}$ cannot be put into an $\mathcal{O}()$-term, we are not entitled to make the above simplifications and need to redo the following analysis.

Now we are ready to define internal global rate $\varphi(t)$, which is the anchor for $\boldsymbol{\gamma}$-correctness, see Definition 4.7. As argued before, it is constant during a round and makes leaps at resynchronization instants. Formally, given that $\boldsymbol{\mathcal{R}}^{(k)} = \{\mathbf{R}_p(t^{(k)}) : p \in \mathcal{N}^{(k)}\}$ is $\boldsymbol{\gamma}^{(k)}$-consonant for all $k \geq 0$, internal global rate $\varphi(t) = \varphi^{(k)} = \varphi(t^{(k)})$ for $t^{(k)} \leq t < t^{(k+1)}$, whereby

$$\varphi(t^{(k)}) \;\in\; \bigcap_{p \in \mathcal{N}^{(k)}} v_p(t^{(k)}) \left(\mathrm{ref}(\mathbf{R}_p(t^{(k)})) + \boldsymbol{\gamma}^{(k)} + [0 \pm \mathcal{O}(\sigma_p \pi_{\max})]\right). \tag{4.70}$$

All this preparatory work can be concentrated in the following assumption, which serves as an induction hypotheses for the beginning of each round.

Assumption 4.5 (Round Beginnings) *At the beginning $t^{(k)}$ of each round $k \geq 0$, we assume that there exists a set $\boldsymbol{\mathcal{R}}^{(k)} = \{\mathbf{R}_p(t^{(k)}) : p \in \mathcal{N}^{(k)}\}$ of local rate intervals with the following properties:*

(1) $\mathbf{R}_p(t^{(k)}) + [0 \pm \mathcal{O}(\sigma_p \pi_{\max})]$ *is correct, and*

(2) $\mathbf{R}_p(t^{(k)})$ *is* $\left(\boldsymbol{\gamma}^{(k)} + [0 \pm \mathcal{O}(\sigma_p \pi_{\max})]\right)$*-correct w.r.t. internal global rate* $\varphi^{(k)}$*.*

4.5.2 Interval Dissemination

During each round the local rate interval $\mathbf{R}_q$ of a remote node q will be transferred to the remote rate interval $\mathbf{R}_{p,q}$ at local node p with the help of two FMEs. Since the computation of a remote rate interval involves the multiplication of the associated quotient rate interval $\mathbf{Q}_{p,q}$, we begin with obtaining properties on them. The final results on rate intervals appear in Lemma 4.15 and on consonance intervals in Lemma 4.16.

Lemma 4.13 (Quotient Rate Interval) *Given the algorithm from Definition 4.8 under Assumptions 4.1–4.4. If* $\mathbf{R}_q$ *resp.* $\mathbf{R}_p$ *is a correct local rate interval during a round at remote node q resp. local node p, then the computed quotient rate interval* $\mathbf{Q}_{p,q}$ *at local node* $p \neq q$ *has the following properties at the end of this round:*

(1) $$\mathrm{align}(\mathbf{Q}_{p,q}) \subseteq \left[0 \pm \left(\tfrac{\sigma_p + \sigma_q}{2}(P_R + B) + \tfrac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\right)\right] + \left[0 \pm \mathcal{O}\left(\left(\sigma_{\max} P_R + \tfrac{\epsilon_{\max}}{P_R}\right)\left(\sigma_{\max} P_R + ||\mathbf{R}_p + \mathbf{R}_q||\right)\right)\right]$$

(2) $\mathrm{ref}(\mathbf{Q}_{p,q}) \in \Big[1 \pm \Big(||\mathbf{R}_p + \mathbf{R}_q|| + \frac{\sigma_p+\sigma_q}{2} P_R + \frac{2B+\epsilon_{\max}+4(F+E)\rho_{\max}}{P_R}\Big)\Big]$

$+ \Big[0 \pm \mathcal{O}\Big(\Big(||\mathbf{R}_p + \mathbf{R}_q|| + \sigma_{\max} P_R + \frac{B+\epsilon_{\max}+(F+E)\rho_{\max}}{P_R}\Big)^2\Big)\Big]$

with maximum logical broadcast delay $B = (1 + 2\rho_{\max})(\omega_{\max} + \lambda_{\max})$.

Proof. Looking at the formula for $\mathbf{Q}_{p,q}$ in the algorithm from Definition 4.8, our first step is devoted to find bounds on the duration $\Delta T_q = T_q - T_q^{\prec}$ and the corresponding duration $\Delta T_{p,q} = T_{p,q} - T_{p,q}^{\prec}$. The accumulated state adjustments U_p and U_q can be ignored, since we assume a transparent clock state synchronization. The idea is to carry over bounds on ΔT_q to the real-time counterpart $\Delta t_q = t_q - t_q^{\prec}$ and further to $\Delta t_{p,q} = t_{p,q} - t_{p,q}^{\prec}$ at node p, which will finally give us bounds on $\Delta T_{p,q}$. See also Figure 4.5 for a better understanding of the involved points of time.

To get a handle on ΔT_q, we have to figure out the range of sending timestamps at remote node q belonging to the two FMEs enclosing an arbitrary round k. Complying to Assumption 4.4 item (1) and (2), at any non-faulty node the maximal logical time between initiating an FME and drawing a sending timestamp is given by

$$B = (1 + 2\rho_{\max})(\omega_{\max} + \lambda_{\max}), \tag{4.71}$$

called *maximum logical broadcast delay.*

Unfortunately, one rate adjustment takes place at $t_q^{(k)}$ during the relative rate measurement, see Figure 4.6. Therefore we split ΔT_q into a two durations

$$\Delta T_q = L_q^{\prec} + L_q^{\succ}, \tag{4.72}$$

where $L_q^{\prec} = (k-1)P_R + F + E - T_q^{\prec}$ reflects the duration from the first timestamp $T_q^{\prec}$ until $(k-1)P_R + F + E$, and $L_q^{\succ} = T_q - (k-1)P_R - F - E$ the duration from $(k-1)P_R + F + E$ until the second timestamp T_q. Directly from the algorithm and consulting the proof of Lemma 4.11 we see that $0 \le T_q^{\prec} - (k-1)P_R \le B$ and $0 \le T_q - kP_R \le B$, which proves

$$(F + E) - B \le L_q^{\prec} \le (F + E) \tag{4.73}$$

and

$$(P_R - F - E) \le L_q^{\succ} \le (P_R - F - E) + B. \tag{4.74}$$

Adding (4.73) and (4.74) leads to

$$|\Delta T_q - P_R| \le B. \tag{4.75}$$

For the sake of simplicity, whilst $L_q^{\prec}$ we work with the maximal oscillator drift $\rho_q^{\max}$ as given in (4.5), so we get limits for the corresponding real-time $l_q^{\prec}$ as

$$\frac{L_q^{\prec}}{1+2\rho_q^{\max}} \leq l_q^{\prec} \leq \frac{L_q^{\prec}}{1-2\rho_q^{\max}} \tag{4.76}$$

by exploiting (4.21) from the proof of Lemma 4.2.

In order to obtain bounds on the corresponding real-time $l_q^{\succ}$ of $L_q^{\succ}$, it is necessary to have limits on the clock rate v_q whilst $L_q^{\succ}$. Since local rate interval $\mathbf{R}_q$ is correct during $\left[t^{(k)}, t^{(k+1)}\right]$, we can assert by virtue of (4.17) falling out from Lemma 4.1 that

$$|v_q(t) - 1| \leq ||\mathbf{R}_q|| + \mathcal{O}(||\mathbf{R}_q||^2) \tag{4.77}$$

during this interval. From Lemma 4.4 we can obtain the relation

$$\frac{L_q^{\succ}}{l_q^{\succ}}(1 - \frac{\sigma_q}{2} l_q^{\succ}) + \mathcal{O}\left(\sigma_q^2 (l_q^{\succ})^2\right) \leq v_q(t) \leq \frac{L_q^{\succ}}{l_q^{\succ}}(1 + \frac{\sigma_q}{2} l_q^{\succ}) + \mathcal{O}\left(\sigma_q^2 (l_q^{\succ})^2\right), \tag{4.78}$$

which holds for the same interval. Making $l_q^{\succ}$ explicit by combining (4.77) and (4.78), followed by an asymptotic approximation yields

$$\begin{aligned} l_q^{\succ} &\geq L_q^{\succ}\left(1 - \frac{\sigma_q L_q^{\succ}}{2} - ||\mathbf{R}_q|| + \mathcal{O}\left(||\mathbf{R}_q||^2 + \sigma_q^2 (L_q^{\succ})^2\right)\right) && (4.79)\\ l_q^{\succ} &\leq L_q^{\succ}\left(1 + \frac{\sigma_q L_q^{\succ}}{2} + ||\mathbf{R}_q|| + \mathcal{O}\left(||\mathbf{R}_q||^2 + \sigma_q^2 (L_q^{\succ})^2\right)\right). && (4.80)\end{aligned}$$

Now we are able to compute Δt_q as the sum of $l_q^{\prec}$ and $l_q^{\succ}$. For that purpose, we add up (4.76) and (4.79)/(4.80) accordingly, and plug in (4.73) and (4.74) as needed. A subsequent application of (4.35) gives us the bounds on the corresponding real-time $\Delta t_{p,q}$ at node p, thus

$$\begin{aligned} \Delta t_{p,q} &\geq (P_R - F - E)\left(1 - \frac{\sigma_q (P_R - F - E)}{2} - ||\mathbf{R}_q||\right) + \frac{F + E - B}{1 + 2\rho_q^{\max}} - \epsilon_{\max} \\ &\quad + \mathcal{O}\left(||\mathbf{R}_q||^2 + \sigma_q^2 P_R^2\right) P_R \\ \Delta t_{p,q} &\leq (P_R - F - E + B)\left(1 + \frac{\sigma_q (P_R - F - E + B)}{2} + ||\mathbf{R}_q||\right) + \frac{F + E}{1 - 2\rho_q^{\max}} + \epsilon_{\max} \\ &\quad + \mathcal{O}\left(||\mathbf{R}_q||^2 + \sigma_q^2 P_R^2\right) P_R. \end{aligned}$$

Next they have to be mapped onto clock $\mathcal{C}_p$ in order to obtain the aspired bounds on $\Delta T_{p,q}$. Since we are again facing the problem with the interspersed rate adjustment at

$t_p^{(k)}$, the mapping happens in two portions: For term $(F+E)/(1 \pm 2\rho_q^{\max})$ it sufficient to work with the maximal oscillator drift $\rho_p^{\max}$ as given in (4.5) and by exploiting (4.21). For the remaining term we use the clock rate induced by the correct local rate interval $\mathbf{R}_p$ analogous to (4.79)/(4.80). Focusing on the lower bound, we get

$$\begin{aligned}
\Delta T_{p,q} &\geq P_R\left(1-\frac{\sigma_q P_R}{2}-||\mathbf{R}_q||\right)\left(1-\frac{\sigma_p P_R}{2}-||\mathbf{R}_p||\right) \\
&\quad -B-\epsilon_{\max}+(F+E)\left(\frac{1-2\rho_p^{\max}}{1+2\rho_q^{\max}}-1\right) \\
&\quad +\mathcal{O}\left(||\mathbf{R}_q||^2+\sigma_p P_R||\mathbf{R}_q||+\sigma_{\max}^2 P_R^2\right)P_R \\
&\geq P_R\left(1-||\mathbf{R}_p||-||\mathbf{R}_q||-\frac{\sigma_p+\sigma_q}{2}P_R\right) \qquad (4.81)\\
&\quad -B-\epsilon_{\max}-4(F+E)\rho_{\max} \\
&\quad +\mathcal{O}\left((||\mathbf{R}_p||+||\mathbf{R}_q||)^2+\sigma_{\max}P_R(||\mathbf{R}_p||+||\mathbf{R}_q||)+\sigma_{\max}^2 P_R^2\right)P_R
\end{aligned}$$

and for the upper bound

$$\begin{aligned}
\Delta T_{p,q} &\leq P_R\left(1+\frac{\sigma_q P_R}{2}+||\mathbf{R}_q||\right)\left(1+\frac{\sigma_p P_R}{2}+||\mathbf{R}_p||\right) \\
&\quad +B+\epsilon_{\max}+(F+E)\left(\frac{1+2\rho_p^{\max}}{1-2\rho_q^{\max}}-1\right) \\
&\quad +\mathcal{O}\left(||\mathbf{R}_q||^2+\sigma_p P_R||\mathbf{R}_q||+\sigma_{\max}^2 P_R^2\right)P_R \\
&\leq P_R\left(1+||\mathbf{R}_p||+||\mathbf{R}_q||+\frac{\sigma_p+\sigma_q}{2}P_R\right) \qquad (4.82)\\
&\quad +B+\epsilon_{\max}+4(F+E)\rho_{\max} \\
&\quad +\mathcal{O}\left((||\mathbf{R}_p||+||\mathbf{R}_q||)^2+\sigma_{\max}P_R(||\mathbf{R}_p||+||\mathbf{R}_q||)+\sigma_{\max}^2 P_R^2\right)P_R
\end{aligned}$$

After this preparatory work we are ready to attack $\mathbf{Q}_{p,q}$ itself. Let us begin with the alignment of it from (4.34) in Lemma 4.5, whereby the left and right lengths are equal. We use (4.75) for an upper bound on ΔT_q, and (4.81) as a lower bound on $\Delta T_{p,q}$, hence

$$\begin{aligned}
\frac{1}{2}||\mathbf{Q}_{p,q}|| &\leq \frac{\sigma_p+\sigma_q}{2}(P_R+B)+\frac{\epsilon_{\max}}{P_R-B-\epsilon_{\max}-4(F+E)\rho_{\max}} \\
&\quad +\mathcal{O}\left(\sigma_{\max}\epsilon_{\max}+\sigma_{\max}^2 P_R^2+\left(\sigma_{\max}P_R+\frac{\epsilon_{\max}}{P_R}\right)(||\mathbf{R}_p||+||\mathbf{R}_q||)\right),
\end{aligned}$$

which proves item (1) of our Lemma.

For the reference point $\mathrm{ref}(\mathbf{Q}_{p,q})=\frac{\Delta T_q}{\Delta T_{p,q}}$ from (4.34) in Lemma 4.5, we apply (4.75) on ΔT_q, and (4.81)/(4.82) on $\Delta T_{p,q}$. A lower bound becomes to

$$\begin{aligned}
\mathrm{ref}(\mathbf{Q}_{p,q}) &\geq \frac{P_R - B}{P_R\left(1 + ||\mathbf{R}_p|| + ||\mathbf{R}_q|| + \frac{\sigma_p+\sigma_q}{2}P_R\right) + B + \epsilon_{\max} + 4(F+E)\rho_{\max}} \\
&\quad \mathcal{O}\left((||\mathbf{R}_p + \mathbf{R}_q|| + \sigma_{\max}P_R)^2\right) \\
&\geq 1 - ||\mathbf{R}_p + \mathbf{R}_q|| - \frac{\sigma_p + \sigma_q}{2}P_R - \frac{2B + \epsilon_{\max} + 4(F+E)\rho_{\max}}{P_R} \\
&\quad + \mathcal{O}\left(\left(||\mathbf{R}_p + \mathbf{R}_q|| + \sigma_{\max}P_R + \frac{B + \epsilon_{\max} + (F+E)\rho_{\max}}{P_R}\right)^2\right)
\end{aligned}$$

and an analogous upper bound, which completes the proof of item (2). □

Lemma 4.14 (Interval Multiplication Bounds) *If* $\mathbf{I} = [x, 1, y]$ *and* $\mathbf{J} = [r \pm u]$ *then*

(1) $\mathrm{align}(\mathbf{J}) + \mathrm{left}(\mathbf{J})\mathrm{align}(\mathbf{I}) \subseteq \mathrm{align}(\mathbf{I} \cdot \mathbf{J}) \subseteq \mathrm{align}(\mathbf{J}) + \mathrm{right}(\mathbf{J})\mathrm{align}(\mathbf{I})$

(2) $\mathrm{ref}(\mathbf{I} \cdot \mathbf{J}) = \mathrm{ref}(\mathbf{J})$

Proof. First we have $\mathrm{align}(\mathbf{I} \cdot \mathbf{J}) = [u + x(r-u), 0, u + y(r+u)]$. The lower bounding interval is given by $[0 \pm u] + (r-u)[x, 0, y]$ and the upper one by $[0 \pm u] + (r+u)[x, 0, y]$. A re-substitution according to Definition 4.3 delivers the claimed result. The second item is a trivial conclusion from Definition 4.3 item (10). □

Lemma 4.15 (Rate Interval Dissemination) *Given the algorithm from Definition 4.8 under Assumptions 4.1–4.4. If* $\mathbf{R}_q$ *resp.* $\mathbf{R}_p$ *is a correct local rate interval during round* k *at remote node* q *resp. local node* $p \neq q$*, then the computed remote rate interval* $\mathbf{R}_{p,q}$ *has the following properties at the end of round* k*:*

(1) $\mathrm{align}(\mathbf{R}_{p,q}) \subseteq \mathrm{align}(\mathbf{R}_q)$

$$+ \left[0 \pm \left(\tfrac{\sigma_p+\sigma_q}{2}(P_R + 2(F+E) + B) + \tfrac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\right)\right]$$

$$+ \left[0 \pm \mathcal{O}\left(G_{p,q} + \sigma_{\max}\pi_{\max}\right)\right]$$

(2) $\mathrm{ref}(\mathbf{R}_{p,q}) \in \left[1 \pm \left(||\mathbf{R}_p + \mathbf{R}_q|| + \tfrac{\sigma_p+\sigma_q}{2}P_R + \tfrac{2B + \epsilon_{\max} + 4(F+E)\rho_{\max}}{P_R}\right)\right]$

$$+ [0 \pm \mathcal{O}(G_{p,q})]$$

with error term $G_{p,q} = (||\mathbf{R}_p + \mathbf{R}_q|| + \sigma_{\max}P_R + (B + \epsilon_{\max} + (F+E)\rho_{\max})/P_R)^2$.

Proof. Let us begin with the alignment of $\mathbf{R}_{p,q}$ addressed by item (1). From the formula of $\mathbf{R}_{p,q}$ in the algorithm from Definition 4.8 we have to deal with the interval multiplication of $\mathbf{Q}_{p,q}$ and $\mathbf{R}_q$, whereby Lemma 4.14 certifies an upper bound as

$$\operatorname{align}(\mathbf{Q}_{p,q} \cdot \mathbf{R}_q) \subseteq \operatorname{align}(\mathbf{Q}_{p,q}) + \operatorname{right}(\mathbf{Q}_{p,q})\operatorname{align}(\mathbf{R}_q). \tag{4.83}$$

Focusing on $\operatorname{right}(\mathbf{Q}_q)$ we get by virtue of Lemma 4.13 item (1) and (2) that

$$\begin{aligned} \operatorname{right}(\mathbf{Q}_q) &\leq 1 + ||\mathbf{R}_p + \mathbf{R}_q|| + \frac{\sigma_p + \sigma_q}{2} P_R + \frac{\sigma_p + \sigma_q}{2}(P_R + B) \\ &\quad + \frac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}} + \frac{2B + \epsilon_{\max} + 4(F+E)\rho_{\max}}{P_R} \\ &\quad + \mathcal{O}\left(\left(||\mathbf{R}_p + \mathbf{R}_q|| + \sigma_{\max} P_R + \frac{B + \epsilon_{\max} + (F+E)\rho_{\max}}{P_R}\right)^2\right) \\ &\leq 1 + ||\mathbf{R}_p + \mathbf{R}_q|| + (\sigma_p + \sigma_q)(P_R + B) \\ &\quad + \frac{2B + 2\epsilon_{\max} + 4(F+E)\rho_{\max}}{P_R} + \mathcal{O}(G_{p,q}) \end{aligned} \tag{4.84}$$

where $G_{p,q}$ stands for $(||\mathbf{R}_p + \mathbf{R}_q|| + \sigma_{\max} P_R + (B + \epsilon_{\max} + (F+E)\rho_{\max})/P_R)^2$ as above.

Next we justify and analyze the deterioration interval

$$\mathbf{R}_{p,q}^{\mathrm{FME}} = [0 \pm (\sigma_p + \sigma_p)(F + E)] \tag{4.85}$$

as drawn from the formula for $\mathbf{R}_{p,q}$ in our algorithm. The reason for it lies in the mismatch between the endpoints of the relative rate measurement period and the corresponding round. More specifically, at remote node q the first sending timestamp $T_q^{\succ}$ is taken before the beginning of round k when the correctness of the current local rate interval $\mathbf{R}_q$ starts, however, Lemma 4.6 requires $\mathbf{R}_q$ to be correct at the moment of timestamping. Therefore, we need to deteriorate $\mathbf{R}_q$ "backwards" for as much as $F+E$ following Lemma 4.3 to make it correct at $(k-1)P_R$ and afterwards. A multiplication with $\mathbf{Q}_{p,q}$ provides a temporary remote rate interval

$$\begin{aligned} \mathbf{R}'_{p,q} &= \mathbf{Q}_{p,q} \cdot \Big(\mathbf{R}_q + [0 \pm \sigma_q(F+E)] + \Big[0 \pm \mathcal{O}\left(\sigma_q^2 (F+E)^2 + \sigma_q(F+E)||\mathbf{R}_q||\right)\Big]\Big) \\ &= \mathbf{Q}_{p,q} \cdot \mathbf{R}_q + [0 \pm \operatorname{right}(\mathbf{Q}_{p,q})\sigma_q(F+E)] \\ &\quad + \Big[0 \pm \mathcal{O}\left(\sigma_q^2 (F+E)^2 + \sigma_q(F+E)||\mathbf{R}_q||\right)\Big], \end{aligned}$$

which in turn needs to be deteriorated by as much as $F+E$ on node p to end up with a correct $\mathbf{R}_{p,q}$ at $kP_R + F + E$. However, the reference point of $\mathbf{R}'_{p,q}$ is not necessarily 1, so we cannot immediately apply Lemma 4.3. Therefore, going back to (4.26)/(4.27) in the proof of Lemma 4.3, replacing 1 by $\operatorname{ref}(\mathbf{R}'_{p,q})$ and knowing that the clock rate v_p is at least $1/\operatorname{right}(\mathbf{R}'_{p,q})$ at kP_R from Definition 4.4, yields a deterioration interval

$$\begin{aligned}
\Delta\mathbf{R}'_{p,q} &= \left[0 \pm (\operatorname{right}(\mathbf{R}'_{p,q}))^2\sigma_p(F+E)\right] + \left[0 \pm \mathcal{O}\left(\sigma_p^2(F+E)^2\right)\right] \\
&= \left[0 \pm (\operatorname{right}(\mathbf{Q}_{p,q})\operatorname{right}(\mathbf{R}_q))^2\,\sigma_p(F+E)\right] + \left[0 \pm \mathcal{O}\left(\sigma_{\max}^2(F+E)^2\right)\right] \\
&= [0 \pm \sigma_p(F+E)] + [0 \pm \mathcal{O}(G_{p,q})]
\end{aligned}$$

by relying on $\operatorname{right}(\mathbf{Q}_{p,q}) \le 1 + \mathcal{O}\left(||\mathbf{R}_p + \mathbf{R}_q|| + \sigma_{\max}P_R + (B + \epsilon_{\max} + (F+E)\rho_{\max})/P_R\right)$ from (4.84), and $\operatorname{right}(\mathbf{R}_q) \le 1 + \mathcal{O}(||\mathbf{R}_q||)$ by definition. Eventually, adding $\mathbf{R}'_{p,q}$ and $\Delta\mathbf{R}'_{p,q}$ explains $\mathbf{R}^{\mathrm{FME}}_{p,q}$ as given in (4.85), whereby an upper bounding interval for it can be easily expressed as

$$\operatorname{align}(\mathbf{R}^{\mathrm{FME}}_{p,q}) \subseteq [0 \pm (\sigma_p + \sigma_q)(F+E)] + [0 \pm \mathcal{O}(G_{p,q})]\,. \tag{4.86}$$

Finishing up the proof for $\operatorname{align}(\mathbf{R}_{p,q})$, we know that $\mathbf{R}_{p,q} = \mathbf{Q}_{p,q} \cdot \mathbf{R}_q + \mathbf{R}^{\mathrm{FME}}_{p,q}$ as provided by the algorithm, hence combining (4.83), (4.84), (4.86), and item (1) from Lemma 4.13 leads to

$$\begin{aligned}
\operatorname{align}(\mathbf{R}_{p,q}) &\subseteq \operatorname{align}(\mathbf{Q}_{p,q}) + \operatorname{right}(\mathbf{Q}_{p,q})\operatorname{align}(\mathbf{R}_q) + \operatorname{align}(\mathbf{R}^{\mathrm{FME}}_{p,q}) \\
&\subseteq \left[0 \pm \left(\frac{\sigma_p + \sigma_q}{2}(P_R + B) + \frac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4\rho_{\max}(F+E)}\right)\right] \\
&\quad + \operatorname{align}(\mathbf{R}_q) + [0 \pm (\sigma_p + \sigma_q)(F+E)] \\
&\quad + [0 \pm \mathcal{O}(G_{p,q})] \\
&\subseteq \operatorname{align}(\mathbf{R}_q) + \left[0 \pm \frac{\sigma_p + \sigma_q}{2}(P_R + B + 2(F+E))\right] \\
&\quad + \left[0 \pm \frac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4\rho_{\max}(F+E)}\right] + [0 \pm \mathcal{O}(G_{p,q})]
\end{aligned}$$

proving item (1) of our lemma by adding $[0 \pm \mathcal{O}(\sigma_{\max}\pi_{\max})]$ to agree with Assumption 4.5 item (2).

Item (2) of our lemma follows immediately from Lemma 4.14 item (2), since the deterioration term $\mathbf{R}^{\mathrm{FME}}_{p,q}$ does not affect the reference point, i.e., $\operatorname{ref}(\mathbf{R}_{p,q}) = \operatorname{ref}(\mathbf{Q}_{p,q})$. Therefore it is sufficient to take over the interval from item (2) of Lemma 4.13. □

The case $q = p$ is trivial, since the algorithm sets $\mathbf{R}_{p,p}$ to the local rate interval $\mathbf{R}_p$, hence $\operatorname{align}(\mathbf{R}_{p,p}) = \operatorname{align}(\mathbf{R}_p)$ and $\operatorname{ref}(\mathbf{R}_{p,p}) = 1$.

At this point we should elaborate on the limited transmission capacity b of our communication subsystem, see Assumption 4.4 item (4). Our algorithm requires to put the timestamp T_q (e.g., 4 bytes), the local rate interval $\mathbf{R}_q$ (e.g., 2×4 bytes), and the sum of state adjustments U_p (e.g., 2 bytes) into a packet for an FME. If b is sufficiently large, then there is no need to modify the algorithm nor its analysis. When b becomes stringent, the data (in the first place $\mathbf{R}_q$) could be divided into several packets, otherwise the analysis needs to be adapted to account for a reduced granularity.

Lemma 4.16 (Consonance Interval Dissemination) *Given the algorithm from Definition 4.8 under Assumptions 4.1–4.4. Let* $\mathcal{N}^{(k)}$ *be the set of non-faulty nodes during round k. If the set of local rate intervals* $\boldsymbol{\mathcal{R}}^{(k)} = \{\mathbf{R}_p : p \in \mathcal{N}^{(k)}\}$ *is correct during round k and* $\boldsymbol{\gamma}^{(k)}$*-correct w.r.t. internal global rate* $\varphi^{(k)}$ *at the beginning of round k, then the remote rate intervals* $\mathbf{R}_{p,q}$ *for* $p, q \in \mathcal{N}^{(k)}$ *have the following properties at the end of the round k:*

(1) Any remote rate interval $\mathbf{R}_{p,q}$ *is* $\boldsymbol{\gamma}_{p,q}$*-correct for* $p \neq q$ *with*

$$\boldsymbol{\gamma}_{p,q} \subseteq \boldsymbol{\gamma}^{(k)} + \left[0 \pm \left(\tfrac{\sigma_p+\sigma_q}{2}(P_R + 2(F+E) + B) + \tfrac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\right)\right] + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})],$$

and any local rate interval $\mathbf{R}_p = \mathbf{R}_{p,p}$ *is* $\boldsymbol{\gamma}_{p,p}$*-correct with*

$$\boldsymbol{\gamma}_{p,p} \subseteq \boldsymbol{\gamma}^{(k)} + [0 \pm \sigma_p P_R] + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})].$$

(2) The set of remote rate intervals $\boldsymbol{\mathcal{R}}_p = \{\mathbf{R}_{p,q} : q \in \mathcal{N}^{(k)}\}$ *at receiving node p supplied by broadcasting nodes* $q \in \mathcal{N}^{(k)}$ *is* $\boldsymbol{\gamma}_p^H$*-correct with*

$$\boldsymbol{\gamma}_p^H \subseteq \boldsymbol{\gamma}^{(k)} + \left[0 \pm \left(\tfrac{\sigma_p+\sigma_{\max}}{2}(P_R + 2(F+E) + B) + \tfrac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\right)\right] + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})].$$

(3) The set of all remote rate intervals $\boldsymbol{\mathcal{R}} = \bigcup_{p,q \in \mathcal{N}^{(k)}} \mathbf{R}_{p,q}$ *at non-faulty nodes is* $\boldsymbol{\gamma}_H$*-correct with*

$$\boldsymbol{\gamma}_H \subseteq \boldsymbol{\gamma}^{(k)} + \left[0 \pm \left(\sigma_{\max}(P_R + 2(F+E) + B) + \tfrac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\right)\right] + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})].$$

(4) The set of remote rate intervals $\boldsymbol{\mathcal{R}}^q = \{\mathbf{R}_{p,q} : p \in \mathcal{N}^{(k)}\}$ *at receiving nodes* $p \in \mathcal{N}^{(k)}$ *from a broadcasting node q is* $\boldsymbol{\gamma}^q$*-correct with*

$$\boldsymbol{\gamma}^q \subseteq \boldsymbol{\gamma}^{(k)} + \left[0 \pm \left(\tfrac{\sigma_{\max}+\sigma_q}{2}(P_R + 2(F+E) + B) + \tfrac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\right)\right] + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})],$$

and $\boldsymbol{\gamma}_I$*-consonant with*

$$\boldsymbol{\gamma}_I \subseteq \left[0 \pm \left(\tfrac{\sigma_{\max}+\sigma_q}{2}(P_R + 2(F+E)) + \tfrac{\sigma_{\max}+3\sigma_q}{2}B + \tfrac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\right)\right] + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}(\pi_{\max} + B\rho_{\max}))].$$

Error term $G_{\max}$ *is given as* $\left(||\mathbf{R}_{\max}|| + \sigma_{\max} P_R + \frac{B+\epsilon_{\max}+(F+E)\rho_{\max}}{P_R}\right)^2$*, where* $\mathbf{R}_{\max}$ *is the largest correct local rate interval.*

Proof. Item (1) of our lemma is analogous to item (1) of Lemma 4.15, but instead of rate intervals the associated consonance intervals are analyzed. Suppose node $q \in \mathcal{N}^{(k)}$ maintains a correct local rate interval $\mathbf{R}_q$ during round k, which is also $\boldsymbol{\gamma}^{(k)}$-correct at the beginning of round k satisfying $\varphi^{(k)} \in v_p(t^{(k)})(1+\boldsymbol{\gamma}^{(k)})$, see Definition 4.7. It remains to quantify the $\boldsymbol{\gamma}_{p,q}$-correctness of $\mathbf{R}_{p,q}$. Due to Lemma 4.10, we can use the same line of reasoning as for $\operatorname{align}(\mathbf{R}_{p,q})$ in Lemma 4.15 in order to find a bounding interval on $\boldsymbol{\gamma}_{p,q}$. In particular, equation (4.83) can be rewritten as

$$\mathbf{Q}_{p,q} \cdot \boldsymbol{\gamma}^{(k)} \subseteq \operatorname{align}(\mathbf{Q}_{p,q}) + \operatorname{right}(\mathbf{Q}_{p,q})\boldsymbol{\gamma}^{(k)},$$

the upper bound on $\operatorname{right}(\mathbf{Q}_{p,q})$ stays (4.84), and the deterioration interval to bridge $F+E$ on both ends is the same as (4.85) relying on Lemma 4.9 and retrenched Lemma 4.3. Eventually, we arrive at $\varphi^{(k)} \in v_q(t^{(k+1)})(\operatorname{ref}(\mathbf{R}_{p,q}) + \boldsymbol{\gamma}_{p,q})$ with the properties from item (1) of Lemma 4.15 after applying two substitutions, namely $\operatorname{align}(\mathbf{R}_{p,q})$ becomes to $\boldsymbol{\gamma}_{p,q}$ and $\operatorname{align}(\mathbf{R}_q)$ to $\boldsymbol{\gamma}^{(k)}$. In preparation for further operations, we want to get independent of the corresponding local rate intervals in expression $G_{p,q}$, so we define a uniform $G_{\max}$ —also referred as rate granularity— as

$$G_{\max} = \left(||\mathbf{R}_{\max}|| + \sigma_{\max} P_R + \frac{B+\epsilon_{\max}+(F+E)\rho_{\max}}{P_R}\right)^2 \tag{4.87}$$

with $\mathbf{R}_{\max}$ as the largest correct local rate interval. The case $p = q$ is a trivial consequence of Lemma 4.9 and by considering Assumption 4.5 item (2).

In item (2) the $\boldsymbol{\gamma}_p^H$-correctness of set $\boldsymbol{\mathcal{R}}_p$ follows from a straight majorization over the non-faulty broadcasting nodes, since each remote rate interval is $\boldsymbol{\gamma}_{p,q}$-correct as asserted by item (1). More specifically, calculating $\boldsymbol{\gamma}_p^H = \bigcup_{q\in\mathcal{N}^k\setminus\{p\}} \boldsymbol{\gamma}_{p,q}$ results in the given formula, whereas the consonance interval $\boldsymbol{\gamma}_{p,p}$ is also subsumed since $\boldsymbol{\gamma}_{p,p} \subseteq \boldsymbol{\gamma}_p^H$ holds.

Item (3) embraces all appearing remote rate intervals at non-faulty nodes, whose $\boldsymbol{\gamma}^H$-correctness can easily be proved by doing the union of $\boldsymbol{\gamma}_{p,q}$ from item (1) over all $p, q \in \mathcal{N}^{(k)}$. Effortlessly, we get the claimed formula.

The first part of item (4) results from a majorization over the receiving nodes, i.e., $\boldsymbol{\gamma}^q = \bigcup_{p\in\mathcal{N}^{(k)}\setminus\{q\}} \boldsymbol{\gamma}_{p,q}$, whereas $\boldsymbol{\gamma}_{q,q} \subseteq \boldsymbol{\gamma}^q$ holds as well.

For the second part, we have to calculate the consonance of the set of remote rate intervals $\boldsymbol{\mathcal{R}}^q$ stemming from a common broadcasting node q. For that purpose, we attach a pseudo consonance interval $\boldsymbol{\kappa}_q$ to the local rate interval $\mathbf{R}_q$. It is initialized to $\emptyset$ when the first message leaves node q during $(k-1)P_R$ and $(k-1)P_R + F$. Covering a point-to-point and a broadcast-type network uniformly, we deteriorate it by the maximum logical broadcast delay B to take care for the last leaving message, thus

$$\boldsymbol{\kappa}_q = [0 \pm \sigma_q B] + \left[0 \pm \mathcal{O}\left(\sigma_q^2 B^2 + \sigma_q B \rho_q^{\max}\right)\right]$$

by exploiting Lemma 4.3. Replacing $\mathbf{R}_q$ with $\boldsymbol{\kappa}_q$ in item (1) of Lemma 4.15 yields the corresponding pseudo consonance interval $\boldsymbol{\kappa}_{p,q}$ valid at $t^{(k+1)}$ with

$$\begin{aligned} \boldsymbol{\kappa}_{p,q} \;\subseteq\; & \left[0 \pm \left(\frac{\sigma_p}{2}(P_R + 2(F+E) + B) + \frac{\sigma_q}{2}(P_R + 2(F+E) + 3B)\right)\right] \\ & + \left[0 \pm \mathcal{O}\left(G_{\max} + \sigma_{\max}(\pi_{\max} + B\rho_{\max})\right)\right] \end{aligned}$$

for all $p, q \in \mathcal{N}^{(k)}$. A majorization over the receiving nodes provides the $\boldsymbol{\gamma}_I$-consonance of $\boldsymbol{\mathcal{R}}^q$, hence $\boldsymbol{\gamma}_I = \bigcup_{p \in \mathcal{N}^{(k)} \setminus \{q\}} \boldsymbol{\kappa}_{p,q}$ leads to the formula given in the second part of item (4), and noting that $\boldsymbol{\kappa}_{q,q} \subseteq \boldsymbol{\gamma}_I$. □

This eventually completes the issues about disseminating of rate/consonance intervals within a single round. We have managed to express the properties of the remote rate interval $\mathbf{R}_{p,q}$ at the end of a particular round k in the terms of the local rate intervals $\mathbf{R}_p$ and $\mathbf{R}_q$ correct during round k, and consonance interval $\boldsymbol{\gamma}^{(k)}$ at the beginning of round k. The actual remote rate intervals are fed into the convergence/validation function, computing a new local rate interval for the next round after adjusting the clock rate accordingly. The remaining sections focus on the evaluation and effects of these functions based on the bounding intervals characterized by Lemma 4.15 and 4.16.

4.5.3 Rate Adjustments

We investigate the amounts of rate adjustments administered at the end of each round by changing instantaneously the oscillator-clock coupling factor S_p of the algorithm from Definition 4.8. Since this change happens in a multiplicative way, we aim to find bounds on the ratio between the new and old one. An additional sanity check for feasible values of S_p helps to exclude faulty clocks. Our analysis is grounded on the consonance preservation function $\mathbf{\Phi}_\gamma(\cdot)$ of the utilized convergence function $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$, see Definition 4.10 item (2).

Lemma 4.17 (Rate Adjustment) *Given the algorithm from Definition 4.8 under Assumptions 4.1–4.4 using convergence function* $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ *subject to fault model* $\mathcal{F}$. *Let* $\mathcal{N}^{(k)}$ *be the set of non-faulty nodes during round* k. *If the set of local rate intervals* $\boldsymbol{\mathcal{R}}^{(k)} = \{\mathbf{R}_p : p \in \mathcal{N}^{(k)}\}$ *is correct during round* k *and* $\boldsymbol{\gamma}^{(k)}$*-correct w.r.t. internal global* *rate* $\varphi^{(k)}$ *at the beginning of round* k*, then the rate adjustment at the end of the round* k*, expressed as the ratio between the new and old oscillator-clock coupling factor, is limited by*

$$\begin{aligned}\frac{S_p^{(k+1)}}{S_p^{(k)}} \in \boldsymbol{\Theta}_p \;\; &= \;\; 1+\boldsymbol{\gamma}^{(k)} + [0 \pm \sigma_p P_R] + \overline{\boldsymbol{\Phi}_\gamma}(\boldsymbol{\gamma}_{p,1}, \dots, \boldsymbol{\gamma}_{p,n}; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots) \\ &\quad + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})] \end{aligned} \tag{4.88}$$

for node $p \in \mathcal{N}^{(k)}$*, whereby* $\boldsymbol{\Phi}_\gamma(\boldsymbol{\gamma}_{p,1}, \dots, \boldsymbol{\gamma}_{p,n}; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots)$ *is a weakly monotonic consonance preservation function of* $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ *supplied with*

(1) $\boldsymbol{\gamma}_{p,q} = \boldsymbol{\gamma}^{(k)} + \Big[0 \pm \Big(\frac{\sigma_p+\sigma_q}{2}(P_R + 2(F+E) + B) + \frac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\Big)\Big]$
$+ [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})]$,

(2) $\boldsymbol{\gamma}_H = \boldsymbol{\gamma}^{(k)} + \Big[0 \pm \Big(\sigma_{\max}(P_R + 2(F+E) + B) + \frac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\Big)\Big]$
$+ [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})]$,

(3) $\boldsymbol{\gamma}_I = \Big[0 \pm \Big(\sigma_{\max}\,(P_R + 2(F+E+B)) + \frac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\Big)\Big]$
$+ [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}(\pi_{\max} + B\rho_{\max}))]$,

and error term $G_{\max} = \Big(||\mathbf{R}_{\max}|| + \sigma_{\max}P_R + \frac{B+\epsilon_{\max}+(F+E)\rho_{\max}}{P_R}\Big)^2$ *with* $\mathbf{R}_{\max}$ *as the largest correct local rate interval.*

Proof. At the beginning $t^{(k)}$ of round k, suppose clock $\mathcal{C}_p$ of node $p \in \mathcal{N}^{(k)}$ is steered with coupling factor $S_p^{(k)}$ and the corresponding local rate interval $\mathbf{R}_p$ is $\boldsymbol{\gamma}^{(k)}$-correct w.r.t. $\varphi^{(k)}$, see Definition 4.7. From item (1) of Lemma 4.16 we know that $\mathbf{R}_{p,p}$ is $\boldsymbol{\gamma}_{p,p}$-correct as shown there. Hence, if $v_p(t^{(k+1)})$ is the clock rate just before the rate adjustment takes place to switch over to round $k+1$, it follows that

$$\begin{aligned}\varphi^{(k)} \;\; &\in \;\; v_p(t^{(k+1)})\left(1+\boldsymbol{\gamma}^{(k)} + [0 \pm \sigma_p P_R]\right) \\ &\quad + \Big[0 \pm \mathcal{O}\left(\sigma_p^2 P_R^2 + \sigma_p P_R ||\mathbf{R}_p|| + \sigma_p \pi_{\max}\right)\Big]. \end{aligned} \tag{4.89}$$

Furthermore, node p computes a new correct rate interval $\tilde{\mathbf{R}}_p$ valid for $t^{(k+1)}$ by virtue of the convergence function $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$, whose characterization with the consonance preservation function $\boldsymbol{\Phi}_\gamma(\cdot)$ ensures that $\tilde{\mathbf{R}}_p$ is $\boldsymbol{\Phi}_\gamma(\boldsymbol{\gamma}_{p,1}, \dots, \boldsymbol{\gamma}_{p,n}; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots)$-correct w.r.t. $\varphi^{(k)}$, see Definition 4.10 item (2). Due to the the weak monotonicity of $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ and $\boldsymbol{\Phi}_\gamma(\cdot)$, we get

$$\begin{aligned}\varphi^{(k)} \;\; &\in \;\; v_p(t^{(k+1)})\Big(\mathrm{ref}(\tilde{\mathbf{R}}_p) + \boldsymbol{\Phi}_\gamma(\boldsymbol{\gamma}_{p,1}, \dots, \boldsymbol{\gamma}_{p,n}; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots) \\ &\quad + [0 \pm \mathcal{O}(\sigma_p \pi_{\max})]\Big) \end{aligned} \tag{4.90}$$

with the above calculated intervals for $\boldsymbol{\gamma}_{p,1}, \dots, \boldsymbol{\gamma}_{p,n}$, $\boldsymbol{\gamma}_H$ and majorized $\boldsymbol{\gamma}_I$ taken over as bounds from item (1), (3) and (4) of Lemma 4.16. Since the calculated intervals in (4.89) and (4.90) intersect, we can derive

$$\begin{aligned} \mathrm{ref}(\tilde{\mathbf{R}}_p) \;\in\;& 1 + \boldsymbol{\gamma}^{(k)} + [0 \pm \sigma_p P_R] + \overline{\boldsymbol{\Phi}_\gamma}(\boldsymbol{\gamma}_{p,1}, \dots, \boldsymbol{\gamma}_{p,n}; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots) \\ & + [0 \pm \mathcal{O}(G_{\max} + \sigma_p \pi_{\max})] \,. \end{aligned}$$

Recalling $S_p^{(k+1)} = S_p^{(k)} \mathrm{ref}(\tilde{\mathbf{R}}_p)$ from (4.55) finishes the proof. □

Note that above limits on the amounts of rate adjustments are only true for straight internal rate synchronization. If there are adjustments meant for external rate synchronization, we have to consider properties of the employed validation function.

4.5.4 Consonance

For internal rate synchronization, we are interested how the consonance of our clocks evolve. In particular, we study the maintenance of the consonance, which is primarily determinated by the consonance enhancement function $\Psi_\gamma(\cdot)$ of $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$, see Definition 4.10 item (3). The following theorem expresses the maximal lengths of the ensuing consonance intervals at the beginning of each round, whereby the conversion to the usual consonance γ can be done via Lemma 4.7.

Theorem 4.1 (Consonance) *Given the algorithm from Definition 4.8 under Assumptions 4.1–4.4 using convergence function $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ subject to fault model $\mathcal{F}$. Let $\boldsymbol{\gamma}^*$ be a solution of equation*

$$||\boldsymbol{\gamma}^*|| \;=\; \Psi_\gamma(\boldsymbol{\gamma}^1, \dots, \boldsymbol{\gamma}^n; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots), \tag{4.91}$$

whereby $\Psi_\gamma(\boldsymbol{\gamma}^1, \dots, \boldsymbol{\gamma}^n; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots)$ is a weakly monotonic consonance enhancement function of $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ supplied with

(1) $\boldsymbol{\gamma}^q = \boldsymbol{\gamma}^* + \left[0 \pm \left(\frac{\sigma_{\max}+\sigma_q}{2}(P_R + 2(F+E) + B) + \frac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\right)\right]$
$+ [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})]$ *for* $1 \le q \le n$,

(2) $\boldsymbol{\gamma}_H = \boldsymbol{\gamma}^* + \left[0 \pm \left(\sigma_{\max}(P_R + 2(F+E) + B) + \frac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\right)\right]$
$+ [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})]$,

(3) $\boldsymbol{\gamma}_I = \left[0 \pm \left(\sigma_{\max}\,(P_R + 2(F+E+B)) + \frac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\right)\right]$
$+ [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}(\pi_{\max} + B\rho_{\max}))]$,

and error term $G_{\max} = \left(||\mathbf{R}_{\max}|| + \sigma_{\max}P_R + \frac{B+\epsilon_{\max}+(F+E)\rho_{\max}}{P_R}\right)^2$. *If each local rate interval* $\mathbf{R}_p$ *of a non-faulty node* p *is correct during all rounds* $k \geq 0$ *with* $\mathbf{R}_p \subseteq \mathbf{R}_{\max}$ *and* $\boldsymbol{\gamma}^{(0)}$*-correct with* $\boldsymbol{\gamma}^{(0)} \subseteq \boldsymbol{\gamma}^*$ *at* $t^{(0)} = t_0$*, then* $\mathbf{R}_p$ *is* $(\boldsymbol{\gamma}^{(k)}+[0\pm\mathcal{O}(G_{\max}+\sigma_{\max}\pi_{\max})])$*-correct with*

$$\boldsymbol{\gamma}^{(k)} \subseteq \boldsymbol{\gamma}^* \tag{4.92}$$

at the beginning $t^{(k)}$ *of round* k *for all* $k \geq 1$.

Proof. We establish this result by conducting an induction proof on round k with Assumption 4.5 item (2) as hypothesis. The initial case $k = 0$ is already implied by the above assumption. Therefore, we have to show that if the set of correct local rate intervals $\boldsymbol{\mathcal{R}}^{(k)}$ is $\boldsymbol{\gamma}^{(k)}$-correct w.r.t. internal global rate $\varphi^{(k)}$ at $t^{(k)}$ for some $k \geq 0$, then $\boldsymbol{\mathcal{R}}^{(k+1)}$ is $\boldsymbol{\gamma}^{(k+1)}$-correct w.r.t. the newly internal global rate $\varphi^{(k+1)}$ at $t^{(k+1)}$, whereby both $\boldsymbol{\gamma}^{(k)}$ and $\boldsymbol{\gamma}^{(k+1)}$ have to be enclosed by a consonance interval $\boldsymbol{\gamma}_0^*$, which is a solution of equation (4.91). For a better comprehension of the involved consonance intervals take a look at Figure 4.9.

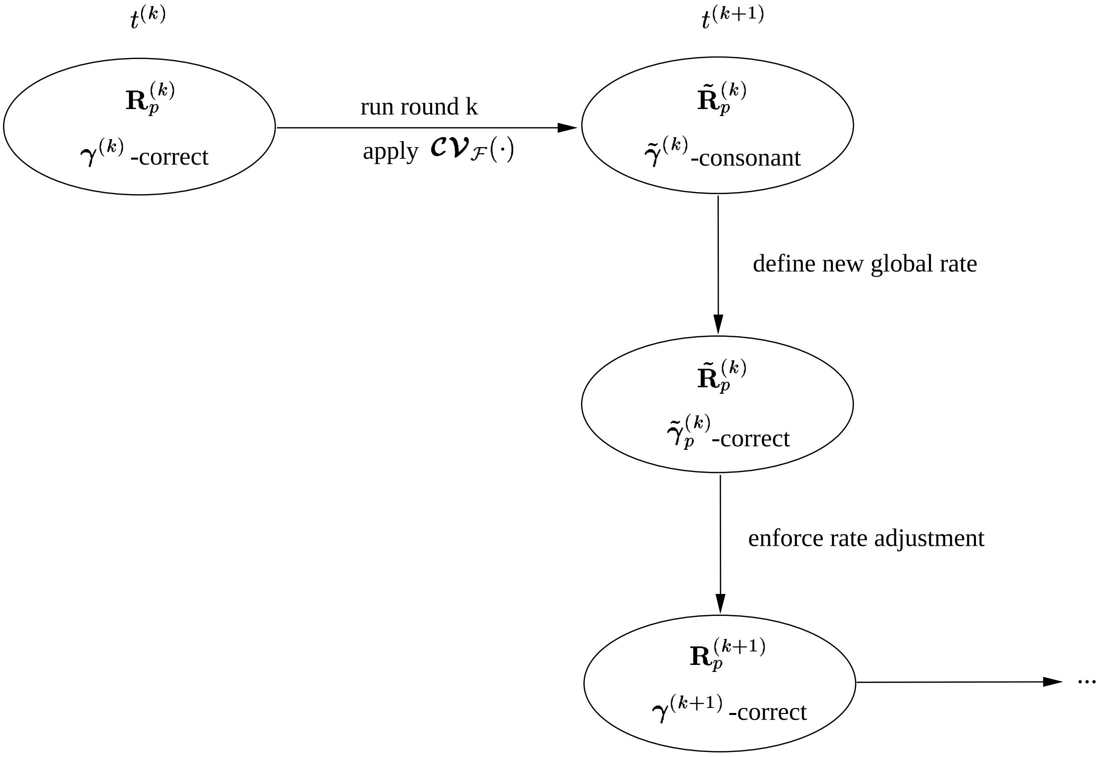

$t^{(k)}$
$t^{(k+1)}$
$\mathbf{R}_p^{(k)}$
$\gamma^{(k)}$ -correct
run round k
apply $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$
$\tilde{\mathbf{R}}_p^{(k)}$
$\tilde{\gamma}^{(k)}$-consonant
define new global rate
$\tilde{\mathbf{R}}_p^{(k)}$
$\tilde{\gamma}_p^{(k)}$-correct
enforce rate adjustment
$\mathbf{R}_p^{(k+1)}$
$\gamma^{(k+1)}$ -correct
...

Figure 4.9: *Succession of Consonance Intervals*

Suppose each node p from set $\mathcal{N}^{(k)}$ of non-faulty nodes during round k has computed its new rate interval $\tilde{\mathbf{R}}_p^{(k)}$ valid for $t^{(k+1)}$ when round k ends. Due to the consonance enhancement function $\Psi_\gamma(\cdot)$ of the employed convergence function $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$, there exists a $\tilde{\boldsymbol{\gamma}}^{(k)}$ such that the set $\{\tilde{\mathbf{R}}_p^{(k)} : p \in \mathcal{N}^{(k)}\}$ is $\tilde{\boldsymbol{\gamma}}^{(k)}$-consonant, see Definition 4.10 item (3). Since $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ and $\Psi_\gamma(\cdot)$ are weakly monotonic, the above calculated intervals for $\boldsymbol{\gamma}^1, \dots, \boldsymbol{\gamma}^n$, $\boldsymbol{\gamma}_H$ and majorized $\boldsymbol{\gamma}_I$ taken over as bounds from item (3) and (4) of Lemma 4.16 determine a maximum on $||\tilde{\boldsymbol{\gamma}}^{(k)}||$ when plugged into $\Psi_\gamma(\cdot)$. This maximum depends primarily on $\boldsymbol{\gamma}^{(k)}$, which indicates the $\boldsymbol{\gamma}^{(k)}$-correctness w.r.t. internal global rate $\varphi^{(k)}$ at $t^{(k)}$ of the preceding set $\{\mathbf{R}_p^{(k)} : p \in \mathcal{N}^{(k)}\}$ of correct local rate intervals, hence we simply write

$$\left|\left|\tilde{\boldsymbol{\gamma}}^{(k)}\right|\right| \le \Psi_\gamma\left(\boldsymbol{f}_{\boldsymbol{\gamma}^1}(\boldsymbol{\gamma}^{(k)}), \dots, \boldsymbol{f}_{\boldsymbol{\gamma}^n}(\boldsymbol{\gamma}^{(k)}); \boldsymbol{f}_{\boldsymbol{\gamma}_H}(\boldsymbol{\gamma}^{(k)}); \boldsymbol{\gamma}_I; \dots\right). \tag{4.93}$$

For our interval-based rate synchronization approach it is vital to understand how to carry over the $\tilde{\boldsymbol{\gamma}}^{(k)}$-consonance of $\{\tilde{\mathbf{R}}_p^{(k)} : p \in \mathcal{N}^{(k)}\}$ to the $\tilde{\boldsymbol{\gamma}}_p^{(k)}$-correctness of $\tilde{\mathbf{R}}_p^{(k)}$ for all $p \in \mathcal{N}^{(k)}$. It is solely accomplished by "choosing" a new internal global rate $\varphi^{(k+1)}$ as an arbitrary point within the intersection

$$\bigcap_{p \in \mathcal{N}^{(k)}} v_p(t^{(k+1)}) \left(\mathrm{ref}(\tilde{\mathbf{R}}_p^{(k)}) + \tilde{\boldsymbol{\gamma}}^{(k)} + [0 \pm \mathcal{O}(\sigma_{\max}\pi_{\max})]\right),$$

which is non-empty due to Definition 4.6. For the other modalities of internal global rate $\varphi(t)$ turn to Section 4.5.1. Consequently, to guarantee a $\tilde{\boldsymbol{\gamma}}_p^{(k)}$-correctness of $\tilde{\mathbf{R}}_p^{(k)}$ w.r.t. $\varphi^{(k+1)}$ it is sufficient to set $\tilde{\boldsymbol{\gamma}}_p^{(k)} = \tilde{\boldsymbol{\gamma}}^{(k)}$ for all $p \in \mathcal{N}^{(k)}$, leading to

$$\left|\left|\tilde{\boldsymbol{\gamma}}_p^{(k)}\right|\right| \le \Psi_\gamma\left(\boldsymbol{f}_{\boldsymbol{\gamma}^1}(\boldsymbol{\gamma}^{(k)}), \dots, \boldsymbol{f}_{\boldsymbol{\gamma}^n}(\boldsymbol{\gamma}^{(k)}); \boldsymbol{f}_{\boldsymbol{\gamma}_H}(\boldsymbol{\gamma}^{(k)}); \boldsymbol{\gamma}_I; \dots\right). \tag{4.94}$$

Next we examine the enforcement of the rate adjustment at $t^{(k+1)}$ of our algorithm. Recalling (4.55) and (4.56), the new clock rate $\tilde{v}_p(t^{(k+1)})$ turns out to be $\mathrm{ref}(\tilde{\mathbf{R}}_p^{(k)})v_p(t^{(k+1)})$ at node p, which assures

$$\varphi^{(k+1)} \in \tilde{v}_p(t^{(k+1)}) \left(1 + \frac{\tilde{\boldsymbol{\gamma}}_p^{(k)}}{\mathrm{ref}\left(\tilde{\mathbf{R}}_p^{(k)}\right)} + [0 \pm \mathcal{O}(\sigma_{\max}\pi_{\max})]\right) \qquad \forall\, p \in \mathcal{N}^{(k)}$$

and further that the new local rate interval $\mathbf{R}_p^{(k+1)} = \mathrm{norm}(\tilde{\mathbf{R}}_p^{(k)})$ is $\boldsymbol{\gamma}^{(k+1)}$-correct after the rate adjustment took place, whereby

$$\left|\left|\boldsymbol{\gamma}^{(k+1)}\right|\right| = \max\left\{\frac{||\tilde{\boldsymbol{\gamma}}_p^{(k)}||}{\mathrm{ref}\left(\tilde{\mathbf{R}}_p^{(k)}\right)} : p \in \mathcal{N}^{(k)}\right\} + \mathcal{O}(\sigma_{\max}\pi_{\max}). \tag{4.95}$$

The additional deterioration of $[0 \pm \sigma_p P_R]$ by our algorithm is not relevant for internal rate synchronization. In order to bound the maximum in (4.95), we need to find a minimum on $\mathrm{ref}(\tilde{\mathbf{R}}_p^{(k)})$ over all $p \in \mathcal{N}^{(k)}$. With the help of Lemma 4.17 and using $||\mathbf{\Phi}_\gamma(\boldsymbol{\gamma}^1, \dots, \boldsymbol{\gamma}^n; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots)|| = \mathcal{O}(||\boldsymbol{\gamma}_H||)$ from Definition 4.10 item (2) we derive

$$\begin{aligned}
\mathrm{ref}(\tilde{\mathbf{R}}_p^{(k)}) &\geq 1 - ||\boldsymbol{\gamma}^{(k)}|| - \sigma_p P_R - \left|\left|\mathbf{\Phi}_\gamma\left(\boldsymbol{f}_{\boldsymbol{\gamma}_{p,1}}(\boldsymbol{\gamma}^{(k)}), \dots, \boldsymbol{f}_{\boldsymbol{\gamma}_{p,n}}(\boldsymbol{\gamma}^{(k)}); \boldsymbol{f}_{\boldsymbol{\gamma}_H}(\boldsymbol{\gamma}^{(k)}); \boldsymbol{\gamma}_I; \dots\right)\right|\right| \\
&\quad + \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max}) \\
&\geq 1 + \mathcal{O}\left(||\boldsymbol{\gamma}^{(k)}|| + ||\mathbf{R}_{\max}|| + \sigma_{\max} P_R + \frac{B + \epsilon_{\max} + (F+E)\rho_{\max}}{P_R}\right). \qquad (4.96)
\end{aligned}$$

Obviously, this is a conservative estimation, but a more detailed analysis renders the resulting expressions rather complicated due of the arising consonance preservation function $\mathbf{\Phi}_\gamma(\cdot)$ without providing any further insight. Anyway, combining (4.94) and (4.96) yields

$$\begin{aligned}
\left|\left|\boldsymbol{\gamma}^{(k+1)}\right|\right| &\leq \Psi_\gamma\left(\boldsymbol{f}_{\boldsymbol{\gamma}^1}(\boldsymbol{\gamma}^{(k)}), \dots, \boldsymbol{f}_{\boldsymbol{\gamma}^n}(\boldsymbol{\gamma}^{(k)}); \boldsymbol{f}_{\boldsymbol{\gamma}_H}(\boldsymbol{\gamma}^{(k)}); \boldsymbol{\gamma}_I; \dots\right) \\
&\quad + \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max}) \qquad (4.97)
\end{aligned}$$

relying on $\Psi_\gamma(\boldsymbol{\gamma}^1, \dots, \boldsymbol{\gamma}^n; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots) < ||\boldsymbol{\gamma}_H||$ from Definition 4.10 item (3), and $||\boldsymbol{\gamma}^{(k)}|| = \mathcal{O}(||\mathbf{R}_{\max}||)$ as a consequence of Lemma 4.4 and 4.7.

To make our induction working, we require that $||\boldsymbol{\gamma}^{(k+1)}|| \leq ||\boldsymbol{\gamma}^{(k)}||$ for any $k \geq 0$, which leads to equation (4.91) with the unknown $\boldsymbol{\gamma}^*$. If it exists, the induction step becomes true, and the weak monotonicity of $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ makes it applicable to any consonance interval enclosed by $\boldsymbol{\gamma}^*$ at the beginning of a round. This eventually finishes the proof of our theorem about consonance. □

Since the above calculated consonance interval $\boldsymbol{\gamma}^*$ holds only at the beginning of a round, we have to establish a worst case consonance interval $\boldsymbol{\gamma}_{\max}$ just before the end of a round. For that purpose imagine two nodes: One has not yet resynchronized, therefore its local rate interval is $(\boldsymbol{\gamma}^* + [0 \pm \sigma_{\max} P_R] + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})])$-correct according to Lemma 4.16 item (1). Another one has already resynchronized, so its local rate interval is $(\mathbf{\Phi}_\gamma(\dots; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots) + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})])$-correct with respect to the same internal global rate by exploiting (4.90) and (4.96). As argued in Section 4.5.1, we neglect the impact of different round beginnings. Combining these intervals by virtue of Lemma 4.8, we finally get for the worst case consonance interval

$$\begin{aligned}
\boldsymbol{\gamma}_{\max} &= (\boldsymbol{\gamma}^* + [0 \pm \sigma_{\max} P_R]) \cup \mathbf{\Phi}_\gamma(\dots; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots) \\
&\quad + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})], \qquad (4.98)
\end{aligned}$$

which holds now for all $t \geq t_0$.

An open issue remains the achievement of the initial consonance $\boldsymbol{\gamma}^{(0)} \subseteq \boldsymbol{\gamma}^*$ at the beginning t_0. Basically, there are three avenues: Either the initial oscillator drifts ρ_p are small enough, or an external rate synchronization takes care for it. Otherwise, we have to devise alternative means (e.g., rate validation as sketched in Section 4.4.5) to meet this initial condition.

4.5.5 Drift

For external rate synchronization, we are interested how the clock drifts evolve. Such a result is not only useful in case there are no correct remote rate intervals available from primary nodes, but for the mechanism of the validation function $\boldsymbol{\mathcal{VAL}}(\cdot)$ as well, see Section 4.4.5. The following theorem expresses the lengths of the resulting local rate intervals at round beginnings, whereby the conversion to the usual clock drift δ_p can be done via Lemma 4.1.

Theorem 4.2 (Drift) *Given the algorithm from Definition 4.8 under Assumptions 4.1–4.4 using convergence function $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ subject to fault model $\mathcal{F}$. Let all local rate intervals $\mathbf{R}_p$ correct at $t^{(0)}$ with* $\operatorname{align}(\mathbf{R}_p) \subseteq \mathbf{V}_p^{(0)}$*. If node p is non-faulty up to the beginning $t^{(k)}$ of round $k \geq 1$, then $\mathbf{R}_p$ is correct at $t^{(k)}$ and satisfies*

$$\operatorname{align}(\mathbf{R}_p) \subseteq \mathbf{V}_p^{(k)} + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})], \tag{4.99}$$

with $\mathbf{V}_p^{(k)} = [\Phi_\delta^-(\mathbf{V}_{p,1}^{(k)}, \ldots, \mathbf{V}_{p,n}^{(k)}; \boldsymbol{\gamma}_{p,1}, \ldots, \boldsymbol{\gamma}_{p,n}; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \ldots), 0, \Phi_\delta^+(\mathbf{V}_{p,1}^{(k)}, \ldots, \mathbf{V}_{p,n}^{(k)}; \boldsymbol{\gamma}_{p,1}, \ldots, \boldsymbol{\gamma}_{p,n}; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \ldots)]$, whereby $\Phi_\delta^\pm(\mathbf{V}_{p,1}^{(k)}, \ldots, \mathbf{V}_{p,n}^{(k)}; \boldsymbol{\gamma}_{p,1}, \ldots, \boldsymbol{\gamma}_{p,n}; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \ldots)$ are the weakly monotonic drift preservation functions of $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ supplied with

(1) $\boldsymbol{\gamma}_{p,q} = \boldsymbol{\gamma}^* \; + \Big[0 \pm \Big(\frac{\sigma_p+\sigma_q}{2}(P_R + 2(F+E) + B) + \frac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\Big)\Big]$
$+ [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})]$ *with $\boldsymbol{\gamma}^*$ from (4.91),*

(2) $\boldsymbol{\gamma}_H = \boldsymbol{\gamma}^* \; + \Big[0 \pm \Big(\sigma_{\max}(P_R + 2(F+E) + B) + \frac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\Big)\Big]$
$+ [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})]$ *with $\boldsymbol{\gamma}^*$ from (4.91),*

(3) $\boldsymbol{\gamma}_I = \Big[0 \pm \Big(\sigma_{\max}\,(P_R + 2(F+E+B)) + \frac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\Big)\Big]$
$+ [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}(\pi_{\max} + B\rho_{\max}))]$,

(4) $\mathbf{V}_{p,q}^{(k)} = \mathbf{V}_q^{(k-1)} + \Big[0 \pm \Big(\frac{(\sigma_p+\sigma_q)(P_R+2(F+E)+B)}{2} + \frac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\Big)\Big]$

and error term $G_{\max} = \Big(||\mathbf{R}_{\max}|| + \sigma_{\max}P_R + \frac{B+\epsilon_{\max}+(F+E)\rho_{\max}}{P_R}\Big)^2$ with $\mathbf{R}_{\max}$ as the largest correct local rate interval.

Proof. We carry out an induction proof on round k about $\operatorname{align}(\mathbf{R}_p^{(k)})$ with Assumption 4.5 item (1) as hypothesis. The initial case $k = 0$ is already implied by the above assumption, where $[\rho_p/(1+\rho_p), 1, \rho_p/(1-\rho_p)]$ can be used for $\mathbf{V}_p^{(0)}$ justified by Lemma 4.1.

For the induction step, assume that $\mathbf{R}_q^{(k-1)}$ is correct during round $k-1$ with $k \geq 1$ for $q \in \mathcal{N}^{(k-1)}$ and satisfies $\operatorname{align}(\mathbf{R}_q^{(k-1)}) \subseteq \mathbf{V}_q^{(k-1)}$. From item (1) of Lemma 4.15 we know that

$$\operatorname{align}(\mathbf{R}_{p,q}^{(k-1)}) \subseteq \mathbf{V}_q^{(k-1)} + \Delta\mathbf{V}_{p,q} \tag{4.100}$$

for a remote rate interval $\mathbf{R}_{p,q}$ on local node $p \in \mathcal{N}^{(k-1)}$ with $p \neq q$, where

$$\begin{aligned}\Delta\mathbf{V}_{p,q} \;=\; & \left[0 \pm \left(\frac{\sigma_p + \sigma_q}{2}(P_R + 2(F+E) + B) + \frac{\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\right)\right] \\ & + \left[0 \pm \mathcal{O}\left(G_{p,q} + \sigma_{\max}\pi_{\max}\right)\right].\end{aligned}$$

In case of $q = p$, we use $\mathbf{R}_{p,p} = \mathbf{R}_p$ straight from our algorithm.

The application of convergence function $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$ outputs a correct rate interval $\tilde{\mathbf{R}}_p^{(k)}$ that satisfies

$$\begin{aligned}\operatorname{align}(\tilde{\mathbf{R}}_p^{(k)}) \;\subseteq\; & \big[\Phi_\delta^-(\mathbf{V}_{p,1}^{(k)}, \dots, \mathbf{V}_{p,n}^{(k)}; \boldsymbol{\gamma}_{p,1}, \dots, \boldsymbol{\gamma}_{p,n}; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots), 0, \\ & \Phi_\delta^+(\mathbf{V}_{p,1}^{(k)}, \dots, \mathbf{V}_{p,n}^{(k)}; \boldsymbol{\gamma}_{p,1}, \dots, \boldsymbol{\gamma}_{p,n}; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots)\big]\end{aligned} \tag{4.101}$$

according to item (1) and (4) of Definition 4.10, whereby

$$\mathbf{V}_{p,q}^{(k)} = \mathbf{V}_q^{(k-1)} + \Delta\mathbf{V}_{p,q}$$

taken from (4.100), and consonance intervals $\boldsymbol{\gamma}_{p,1}, \dots, \boldsymbol{\gamma}_{p,n}$, $\boldsymbol{\gamma}_H$ and majorized $\boldsymbol{\gamma}_I$ are taken over as bounds from item (1), (3) and (4) of Lemma 4.16; recalling the weak monotonicity of $\Phi_\delta^{\pm}(\cdot)$. The encountered $\boldsymbol{\gamma}^{(k)}$ is bounded by $\boldsymbol{\gamma}^*$, which is justified by (4.91) and (4.92) of Theorem 4.1. This proves the inputs (1)–(4) for $\Phi_\delta^{\pm}(\cdot)$ in our theorem.

It remains to analyze $\operatorname{align}(\mathbf{R}_p^{(k)})$ as a result of the rate adjustment at $t^{(k)}$, which is given by

$$\operatorname{align}(\mathbf{R}_p^{(k)}) = \frac{\operatorname{align}(\tilde{\mathbf{R}}_p^{(k)})}{\operatorname{ref}(\tilde{\mathbf{R}}_p^{(k)})}$$

due to (4.56). Recalling estimation (4.96) for $\operatorname{ref}(\tilde{\mathbf{R}}_p^{(k)})$ and (4.101) for $\operatorname{align}(\tilde{\mathbf{R}}_p^{(k)})$ provides

$$\begin{aligned}\operatorname{align}(\mathbf{R}_p^{(k)}) \;\subseteq\; & \big[\Phi_\delta^-(\mathbf{V}_{p,1}^{(k)}, \dots, \mathbf{V}_{p,n}^{(k)}; \boldsymbol{\gamma}_{p,1}, \dots, \boldsymbol{\gamma}_{p,n}; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots), 0, \\ & \Phi_\delta^+(\mathbf{V}_{p,1}^{(k)}, \dots, \mathbf{V}_{p,n}^{(k)}; \boldsymbol{\gamma}_{p,1}, \dots, \boldsymbol{\gamma}_{p,n}; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \dots)\big] \\ & + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})]\end{aligned}$$

which finally proves (4.99) of our theorem. □

Admittedly, this result is overly conservative since in case of multiple rounds, only the worst case enlargements of the rate intervals would be added up. For an improvement we need to consider two facts: First, the nominal oscillator stability σ_p decreases for longer durations Δt, so we are better off when working with $\sigma_p(\Delta t)$, see again Figure 4.3. Second, the maximal rate adjustment cannot take place more than once in consecutive rounds, hence analogous to clock state synchronization "conditioned" drift preservation functions $\Phi_{\delta}^{\pm}(\cdot)$ seem to be useful here.

It should be clear that there is no conflict with the optimal result of [66], since the "hardware drift rate" is now affected by the CRA but not the oscillator stability.

4.6 Summary and Future Research

Clock rate synchronization is useful for plenty of distributed applications and supports algorithms for clock state synchronization. Their communication and computation costs are low, although a rate adjustable local clock is required. We crafted a framework for algorithms synchronizing both internally and externally the rates of clocks in a fault-tolerant distributed system. Its interval-based paradigm is surprisingly similar in many aspects to ours for clock state synchronization presented in [55]. In fact, the same convergence/validation functions can be used to operate on intervals that captured synchronization quantities like clock rates or real-time. These intervals in turn need to be sufficiently enlarged to account for uncertainties arising from the communication subsystem or the clocks. For the latter we have discovered that their stabilities have great importance for (internal) clock synchronization instead of their conservative maximum drift.

Much work remains to be done. In particular, we are about to investigate the initialization/joining problem and to analyze promising convergence/validation functions along with abstract fault models. Moreover, we are interested in optimality results for clock rate synchronization. Last but not least, experimental measurements on oscillators are required to obtain their drift and stability parameters.

◇

Chapter 5

INTERVAL-BASED CONVERGENCE FUNCTIONS

5.1 Introduction

Traditional internal synchronization techniques turned out to be ill-suited for coping with this both precision and accuracy requirement, see Chapter 3. In fact, although worst case accuracy bounds have been provided for most existing synchronization algorithms, it is nevertheless true that a static worst case bound is not representative for the "average" execution. A promising alternative are interval-based algorithms, where local time information relating to real-time t (usually UTC) is represented by an interval that is supposed to contain t. Given a set of such intervals from different nodes, a usually smaller interval that contains UTC can be determined by means of a suitable interval-valued convergence function. Since accuracy bounds are hence maintained "on-line" here, they are of course representative for the average execution.

In Chapter 3 resp. 4, we presented and analyzed a generic interval-based framework for clock state resp. rate synchronization algorithms. According to our exposition above, this implies that it is capable of maintaining precision resp. consonance when started from an initially synchronized state resp. rate and takes care of accuracy resp. drift as well. The algorithms are generic in the sense that it left unspecified the convergence function $\mathcal{CV}_{\mathcal{F}}(\cdot)$ employed for computing the clock state/rate adjustments. As in [58], all results (worst-case precision and consonance, accuracy and drift, maximum adjustments, etc.) are expressed in terms of a few characteristic parameters of the convergence function, see Definition 3.11 and 4.10. In order to determine the performance of a particular instance of the algorithm, one has to determine the characteristic parameters of the particular convergence function and to plug them into the generic results: Theorem 3.1 on precision/accuracies with instantaneous state corrections, Theorem 3.2 on continuous amortization, Theorem 4.1 on the consonance of clocks, and Theorem 4.2 on the drift of clocks.

Although our interval-based framework does of course not provide entirely new results for worst case precision and consonance, it nevertheless surpasses traditional approaches due to its conceptual beauty and high flexibility w.r.t. incorporating features like clock granularity, broadcast latencies, etc. This is primarily a consequence of our notion of internal global time and rate, which allows us to reason about precision and consonance by considering each local interval clock separately, i.e., without explicitly relating it to the other clocks in the system. Even more, there is no need to consider the "absolute position" of intervals, i.e., clock state and rate, at all.

This chapter provides description and detailed analysis of the *optimal precision algorithms* OP-STATE and OP-RATE obtained by employing the *optimal precision convergence function* $\mathcal{OP}(\cdot)$ in the generic clock state and rate algorithm, respectively. It is organized as follows: Section 5.2 is devoted to the investigation of Marzullo's function $\mathcal{M}(\cdot)$, which plays a central role in the analysis of $\mathcal{OP}(\cdot)$ contained in Section 5.3. The latter also includes the major results, namely, worst case bounds for precision and accuracy of OP-STATE, and worst case bounds for consonance and drift of OP-RATE. As usual, a summary and directions of further research are appended in Section 5.4.

5.2 Marzullo's Function

This section is devoted to an in-depth investigation of a certain fault-tolerant intersection function $\mathcal{M}(\cdot)$, which plays a central role in our optimal precision convergence function. It was introduced in Marzullo's thesis [35] and termed *Marzullo function* in [25]. However, its relevance was recognized in the context of replicated sensors only, see [36] and [3]. For clock synchronization purposes, a number of additional properties are required that will be established subsequently. More specifically, we start with a definition and Subsection 5.2.1 and 5.2.2 analyze the properties in regard of accuracy and precision, respectively. Subsection 5.2.3 concludes with a discussion why Marzullo's function alone is not sufficient for internal clock synchronization. For ease of presentation the following definitions and lemmas will be in the context of clock state synchronization, see Chapter 3.

Definition 5.1 (Marzullo Function) *Given a set* $\boldsymbol{\mathcal{I}} = \{\boldsymbol{I}_1, \dots, \boldsymbol{I}_n\}$ *of* $n \geq 1$ *non-empty compatible intervals with at least* $n - f \geq 1$ *of the intervals being accurate,* $\mathcal{M}_n^{n-f}(\boldsymbol{\mathcal{I}})$ *is defined as the largest interval whose edges lie in the intersection of at least* $n - f$ *different* $\boldsymbol{I}_j$*'s.*

Therefore, to compute the left resp. right edge of $\mathcal{M}_n^{n-f}(\boldsymbol{\mathcal{I}})$, one has to "sweep" over the set of intervals from left to right resp. right to left and stop when $n - f$ intervals intersect for the first time. Thus, $\mathcal{M}(\cdot)$ can be computed in $\mathcal{O}(n \log n)$ time by sorting the intervals' edges, see [36]. The following Figure 5.1 shows an example for $n = 4$ and $f = 1$. Note that the unknown t cannot lie in the region between right($\boldsymbol{I}_3$) and left($\boldsymbol{I}_2$) in this example. However, since there is no way to decide whether t lies in the area left or right of this region, both areas must be covered by $\mathcal{M}_4^3(\cdot)$.

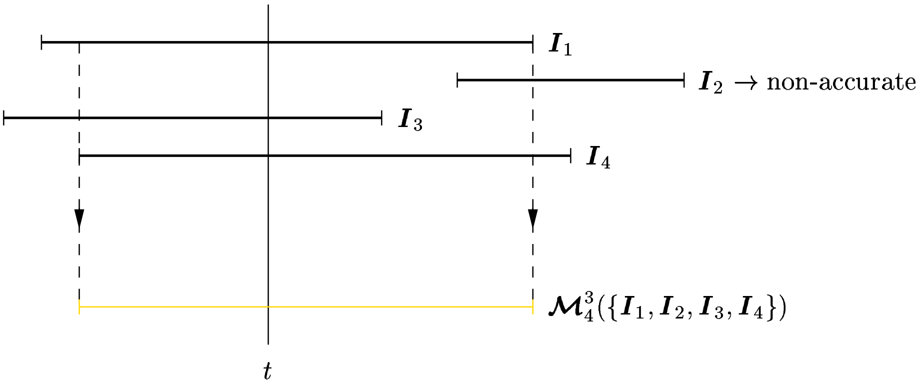

$\boldsymbol{I}_1$
$\boldsymbol{I}_2$ → non-accurate
$\boldsymbol{I}_3$
$\boldsymbol{I}_4$
$\mathcal{M}_4^3(\{\boldsymbol{I}_1, \boldsymbol{I}_2, \boldsymbol{I}_3, \boldsymbol{I}_4\})$
t

Figure 5.1: *Example of Marzullo's Function*

It is immediately apparent from Definition 5.1 that $\mathcal{M}_n^1(\{\boldsymbol{I}_1, \dots, \boldsymbol{I}_n\}) = \bigcup_{1 \le j \le n} \boldsymbol{I}_j$ and $\mathcal{M}_n^n(\{\boldsymbol{I}_1, \dots, \boldsymbol{I}_n\}) = \bigcap_{1 \le j \le n} \boldsymbol{I}_j$, hence $\mathcal{M}_n^{n-f}(\cdot)$ "changes" from union to intersection as $n - f$ goes from 1 to n.

When Marzullo's function $\mathcal{M}(\cdot)$ is a constituent part of a particular convergence function, we need to prove the following straightforward properties, cf. Definition 3.11 and Lemma 4.12.

Lemma 5.1 (Translation Invariance, Multiplicativity of Marzullo's Function) *Let $\boldsymbol{\mathcal{I}} = \{\boldsymbol{I}_1, \dots, \boldsymbol{I}_n\}$ be a set of $n > f \ge 0$ compatible non-empty accuracy intervals representing t, where up to f faulty ones are among them. Then,*

(1) $\mathcal{M}_n^{n-f}(\{\boldsymbol{I}_1 + \Delta, \dots, \boldsymbol{I}_n + \Delta\}) = \mathcal{M}_n^{n-f}(\{\boldsymbol{I}_1, \dots, \boldsymbol{I}_n\}) + \Delta$ *for any* Δ, *and*

(2) $\mathcal{M}_n^{n-f}(\{s\boldsymbol{I}_1, \dots, s\boldsymbol{I}_n\}) = s\mathcal{M}_n^{n-f}(\{\boldsymbol{I}_1, \dots, \boldsymbol{I}_n\})$ *for any* $s \ge 0$.

Proof. Let $\boldsymbol{I}_j = [x_j, x_{j+n}]$ for any $1 \le j \le n$, then there exists a partial order of the endpoints such that $x_{r(1)} \le \ldots \le x_{r(2n)}$. Adding a Δ or a multiplication with any $s \ge 0$ keeps this partial order, thus $x_{r(1)} + \Delta \le \ldots \le x_{r(2n)} + \Delta$ and $sx_{r(1)} \le \ldots \le sx_{r(2n)}$, which reflects the input of the l.h.s. of item (1) and (2), respectively. Since only the partial order of the endpoints are relevant for the computation of Marzullo's function $\mathcal{M}(\cdot)$, the properties of translation invariance in item (1) and multiplicativity in item (2) follow immediately. $\square$

5.2.1 Accuracy Properties

The most important feature of $\boldsymbol{\mathcal{M}}(\cdot)$ is fault-tolerance w.r.t. faulty input intervals. For example, Figure 5.1 shows that $\boldsymbol{\mathcal{M}}_4^3(\cdot)$ provides an accurate result despite of the fact that $\boldsymbol{I}_2$ was non-accurate. We will now embark formally on $\boldsymbol{\mathcal{M}}(\cdot)$'s capabilities in this respect. To that end, we introduce the following classification of faults:

Definition 5.2 (Single Faults) *An interval* $\boldsymbol{I}$ *representing* t *can suffer from the following faults:*

(1) Omission*: missing interval expressed by* $\boldsymbol{I} = \emptyset$.

(2) Non-accurate interval*:* $t \notin \boldsymbol{I}$

(3) Unbounded accuracy*:* $t \in \boldsymbol{I}$ *but* $||\boldsymbol{I}||$ *too large according to some condition (that need not be known explicitly).*

Whereas it is usually impossible to decide locally whether an interval $\boldsymbol{I}$ is accurate or not, it is of course possible to detect omission faults. Hence, given a set $\boldsymbol{\mathcal{I}}$ of $n \geq 1$ compatible intervals with $o' \geq 0$ of them exhibiting omission faults, it is trivial to discard the o' omissive ones from $\boldsymbol{\mathcal{I}}$ and to proceed with the reduced set $\boldsymbol{\mathcal{J}}$ containing $n' = n - o'$ non-empty intervals only.

Masking or detecting —and thus ruling them out completely— unbounded accuracy faults is impossible in most circumstances. Indeed, apart from the fact that it is sometimes difficult to determine the exact border between faulty and non-faulty accuracy values locally, even enforcing a certain worst-case bound —by limiting α^-, α^+— cannot prevent faulty nodes from considerably spoiling the "average" behavior.

The following lemma reveals how $\boldsymbol{\mathcal{M}}(\cdot)$ behaves in the presence of faults according to Definition 5.2. Extending the accomplishments of [35] and [36], it answers the question of how "severe" a particular fault is w.r.t. the number of non-faulty intervals required for tolerating it. Moreover, both an upper and a lower bound on the length of the provided interval is given.

Lemma 5.2 (Accuracy of Marzullo's Function) *Let* $\boldsymbol{\mathcal{J}} = \{\boldsymbol{J}_1, \dots, \boldsymbol{J}_n\}$ *be a set of* $n \geq 1$ *non-empty compatible accuracy intervals representing* t*, and define* w^h *to be the length of the largest intersection of* $h \geq 1$ *non-faulty intervals among them, i.e.,* $w^h = \max\{||\boldsymbol{W}|| : \boldsymbol{W} \in \boldsymbol{\mathcal{W}}^h\}$ *for*

$$\boldsymbol{\mathcal{W}}^h = \{\boldsymbol{W} \ : \ \boldsymbol{W} = \bigcap_{i=1}^{h} \boldsymbol{J}_{w_i} \text{ with } w_i \neq w_j \text{ for } i \neq j \text{ and } \boldsymbol{J}_{w_i} \in \boldsymbol{\mathcal{J}} \text{ being non-faulty}\}.$$

If $a' \geq 0$ of the $\boldsymbol{J}_j$ suffer from unbounded accuracy faults and $d' \geq 0$ are non-accurate, where $a' \leq a$ and $d' \leq d$ with $a' + d' = f' \leq a + d = f < n$, then

(1) $\boldsymbol{M} = \boldsymbol{\mathcal{M}}_n^{n-f}(\boldsymbol{\mathcal{J}})$ *is accurate and contains any intersection* $\boldsymbol{W} \in \boldsymbol{\mathcal{W}}^{n-f}$ *of* $n - f \geq 1$ *different non-faulty input intervals, i.e.,*

$$\boldsymbol{W} = \bigcap_{j=1}^{n-f} \boldsymbol{J}_{w_j} \subseteq \boldsymbol{M}, \tag{5.1}$$

so that $||\boldsymbol{M}|| \geq w^{n-f}$ *(minimal intersection property),*

(2) there are at least $n - 2f - a' \geq n - 2f - a$ *different non-faulty input intervals* $\boldsymbol{J}_{b_1}, \ldots, \boldsymbol{J}_{b_{n-2f-a'}} \in \boldsymbol{\mathcal{J}}$ *such that*

$$\boldsymbol{M} \subseteq \bigcap_{j=1}^{n-2f-a'} \boldsymbol{J}_{b_j} \subseteq \bigcap_{j=1}^{n-2f-a} \boldsymbol{J}_{b'_j}, \tag{5.2}$$

where the sequence $\{b'_j\}_{1 \leq j \leq n-2f-a}$ *is obtained from* $\{b_j\}_{1 \leq j \leq n-2f-a'}$ *by discarding* $a - a'$ *arbitrary intervals. Hence,* $||\boldsymbol{M}|| \leq w^{n-2f-a'} \leq w^{n-2f-a}$.

(3) there are at least $f - f' + 1 \geq 1$ *non-faulty intervals* $\boldsymbol{J}_{\ell_k}$ *resp.* $\boldsymbol{J}_{r_k}$ *in* $\boldsymbol{\mathcal{J}}$ *satisfying* $\mathrm{left}(\boldsymbol{M}) \leq \mathrm{left}(\boldsymbol{J}_{\ell_k})$ *resp.* $\mathrm{right}(\boldsymbol{M}) \geq \mathrm{right}(\boldsymbol{J}_{r_k})$.

Proof. Since $\boldsymbol{M} = \boldsymbol{\mathcal{M}}_n^{n-f}(\boldsymbol{\mathcal{J}})$ contains any intersection of at least $n - f$ input intervals by definition, it obviously contains any intersection of $n - f$ non-faulty intervals $\boldsymbol{W} = \bigcap_{j=1}^{n-f} \boldsymbol{J}_{w_j} \in \boldsymbol{\mathcal{W}}^{n-f}$. Therefore, it follows that $t \in \boldsymbol{M}$ and $||\boldsymbol{M}|| \geq w^{n-f}$ as asserted in item (1) of the lemma.

Turning our attention to item (2), it follows that the total number of intersections of left and right edge of $\boldsymbol{M}$ with non-faulty input intervals is $g'_l + g'_r \geq 2(n - f) - 2a' - d'$, because an interval $\boldsymbol{J}_j$ suffering from an unbounded accuracy fault (a') could intersect with both edges of $\boldsymbol{M}$ whereas a non-accurate interval (d') can only intersect with one edge of $\boldsymbol{M}$ due to $t \notin \boldsymbol{J}_j$ but $t \in \boldsymbol{M}$. However, since there are only $g' = n - f'$ different non-faulty intervals in $\boldsymbol{\mathcal{J}} = \{\boldsymbol{J}_1, \ldots, \boldsymbol{J}_n\}$, the pigeonhole principle reveals that

$$g'_l + g'_r - g' \geq 2n - 2f - 2a' - d' - n + f' = n - 2f - a'$$

of the intersected accurate intervals, say $\boldsymbol{J}_{b_1}, \ldots, \boldsymbol{J}_{b_{n-2f-a'}}$, must be the same. Therefore, $\boldsymbol{M}$ must lie in the intersection of those intervals and $||\boldsymbol{M}|| \leq w^{n-2f-a'}$ as asserted. A simple majorization easily establishes (5.2).

Finally, to prove item (3), consider w.l.o.g. the left edge of $\boldsymbol{M}$. We show by contradiction that the left edge of at least $f - f' + 1$ non-faulty intervals lies at or right of left($\boldsymbol{M}$): If there were at most $f - f'$ such intervals, all $n - f' - (f - f') = n - f$ remaining non-faulty intervals would have their left edge strictly left of left($\boldsymbol{M}$). However, this would contradict the minimal intersection property established in item (1), completing the proof of our lemma. □

The lower bound on $||\boldsymbol{M}||$ in item (1) expresses the rather obvious fact that $\boldsymbol{\mathcal{M}}(\cdot)$ cannot improve the accuracy beyond the one "hidden" in the input intervals; the term *minimal intersection property* was coined in [35]. Note that $\boldsymbol{M}$ contains any intersection of $n - f$ intervals, hence also intersections involving unbounded accuracy faults.

Interpreting item (2) of Lemma 5.2 in terms of the fault-tolerance degree, it follows that $n \geq o' + 2f + a' + 1$ nodes are required to guarantee that $\boldsymbol{M}$ remains bounded by the length of at least one non-faulty input interval. Hence, as many as

$$n \geq \begin{cases} o' + 1 & \text{for } o' \text{ omission faults,} \\ 2d + 1 & \text{for } d' \leq d \text{ non-accurate intervals,} \\ 2a + a' \leq 3a & \text{for } a' \leq a \text{ unbounded accuracy faults} \end{cases}$$

nodes are required for tolerating faults of the given type. It is thus apparent that $\boldsymbol{\mathcal{M}}(\cdot)$ can tolerate $\lfloor (n-1)/2 \rfloor$ non-accurate intervals but only $\lfloor (n-1)/3 \rfloor$ intervals that suffer from unbounded accuracy faults, cf. [36]. Note carefully that the numbers above do not solely depend on the *actual* number of faults (e.g., a'), but also on their maximum number (e.g., a); this is due to the fact that the latter is compiled into the superscript argument of $\boldsymbol{\mathcal{M}}(\cdot)$.

Item (3) just says that $\boldsymbol{M}$ contains the left and right edge of at least one (not necessarily the same) non-faulty interval.

Lemma 5.3 (Monotonicity of Marzullo's Function) *Let* $\boldsymbol{\mathcal{I}} = \{\boldsymbol{I}_1, \ldots, \boldsymbol{I}_n\}$ *be a set of* $n > f \geq 0$ *compatible non-empty accuracy intervals representing* t, *with* f', $0 \leq f' \leq f$, *faulty ones among them. Then,* $\boldsymbol{\mathcal{M}}_n^{n-f}(\boldsymbol{\mathcal{I}})$ *is accurate and satisfies the following monotonicity relations:*

(1) $\boldsymbol{\mathcal{M}}_n^{n-f}(\boldsymbol{\mathcal{I}}) \subseteq \boldsymbol{\mathcal{M}}_n^{n-(f+k)}(\boldsymbol{\mathcal{I}})$ *for any integer* k *with* $0 \leq k < n - f$,

(2) $\boldsymbol{\mathcal{M}}_n^{n-f}(\boldsymbol{\mathcal{I}}) \subseteq \boldsymbol{\mathcal{M}}_n^{n-f}(\boldsymbol{\mathcal{J}})$ *for any* $\boldsymbol{\mathcal{J}} = \{\boldsymbol{J}_1, \ldots, \boldsymbol{J}_n\}$ *with* $\boldsymbol{I}_l \subseteq \boldsymbol{J}_l$ *for* $1 \leq l \leq n$,

(3) For $f \geq f' \geq 1$, *if* $\boldsymbol{\mathcal{L}} = \boldsymbol{\mathcal{I}} \backslash \{\boldsymbol{I}_j\}$ *is obtained by discarding some faulty interval* $\boldsymbol{I}_j$ *from* $\boldsymbol{\mathcal{I}}$, $\boldsymbol{\mathcal{M}}_{n-1}^{(n-1)-(f-1)}(\boldsymbol{\mathcal{L}}) = \boldsymbol{\mathcal{M}}_{n-1}^{n-f}(\boldsymbol{\mathcal{L}})$ *is accurate and satisfies*

$$\boldsymbol{\mathcal{M}}_{n-1}^{n-f}(\boldsymbol{\mathcal{L}}) \subseteq \boldsymbol{\mathcal{M}}_n^{n-f}(\boldsymbol{\mathcal{I}}). \tag{5.3}$$

Proof. In the proof of item (1) of Lemma 5.2 we argued that $\boldsymbol{M} = \boldsymbol{\mathcal{M}}_n^{n-f}(\boldsymbol{\mathcal{I}})$ contains any intersection of at least $n-f$ input intervals by definition, hence must be accurate. Since the interval containing any intersection of $n-f-k$ input intervals obviously contains any intersection of $n-f$ input intervals, item (1) of the lemma follows.

Turning our attention to item (2), it is clear that any particular intersection of $n-f$ intervals in $\boldsymbol{\mathcal{J}}$ contains the intersection of the corresponding intervals in $\boldsymbol{\mathcal{I}}$ if it is non-empty at all. By the same token as before, it hence follows that $\boldsymbol{\mathcal{M}}_n^{n-f}(\boldsymbol{\mathcal{J}})$ must contain $\boldsymbol{\mathcal{M}}_n^{n-f}(\boldsymbol{\mathcal{I}})$ as well.

For proving item (3), the same argument is used again. First of all, $n > f \geq 1$ implies that $n-1 > f-1 \geq 0$, hence $\boldsymbol{\mathcal{M}}_{n-1}^{(n-1)-(f-1)}(\boldsymbol{\mathcal{L}})$ is accurate. Moreover, discarding a faulty $\boldsymbol{I}_j$ in $\boldsymbol{\mathcal{I}}$ and simultaneously reducing f by 1 leaves the superscript argument $(n-1)-(f-1) = n-f$ in $\boldsymbol{\mathcal{M}}_{n-1}^{n-f}(\boldsymbol{\mathcal{L}})$ unchanged. Since the interval containing any intersection of $n-f$ input intervals in $\boldsymbol{\mathcal{I}}$ must contain the interval containing any intersection of $n-f$ intervals in $\boldsymbol{\mathcal{L}} \subseteq \boldsymbol{\mathcal{I}}$, the statement given in item (3) of our lemma follows. $\square$

It is not difficult to show that $\boldsymbol{\mathcal{M}}(\cdot)$ is optimal w.r.t. worst case accuracy for non-accurate intervals (but not for unbounded accuracy faults) among all interval-valued functions of n interval arguments, cf. [25]: Suppose there were a function $\boldsymbol{\mathcal{F}}$ that provides an accurate interval satisfying $\boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{I}}) \not\subseteq \boldsymbol{\mathcal{F}}(\boldsymbol{\mathcal{I}})$, then $\boldsymbol{M}' = \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{I}}) \setminus \big(\boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{I}}) \cap \boldsymbol{\mathcal{F}}(\boldsymbol{\mathcal{I}})\big) \neq \emptyset$ and there must be an intersection of $n-f$ accurate intervals $\boldsymbol{A} = \bigcap_{i=1}^{n-f} \boldsymbol{I}_{b_i}$ so that $\boldsymbol{A} \cap \boldsymbol{M}' \neq \emptyset$. However, the (valid) assumption that $t \in \boldsymbol{A} \cap \boldsymbol{M}'$ reveals that $\boldsymbol{\mathcal{F}}(\boldsymbol{\mathcal{I}})$ cannot be accurate, providing the required contradiction.

Regarding item (2) of Lemma 5.3, it is important to note that enlarging an input interval (even by a minor amount) can cause a discontinuous jump of an edge of $\boldsymbol{\mathcal{M}}_n^{n-f}(\boldsymbol{\mathcal{J}})$ if a "new" intersection of $n-f$ intervals comes up. Just consider shrinking or moving right the faulty interval $\boldsymbol{I}_2$ in Figure 5.1, which causes the right edge of $\boldsymbol{\mathcal{M}}_4^3(\cdot)$ to shrink to right$(\boldsymbol{I}_3)$ as soon as left$(\boldsymbol{I}_2) >$ right$(\boldsymbol{I}_1)$. This implies that $\boldsymbol{\mathcal{M}}(\cdot)$ does not satisfy a Lipschitz condition w.r.t. moving (edges of) input intervals, as already noted in [25].

Item (3) of Lemma 5.3 implies that one should always try to detect and discard faulty intervals before $\boldsymbol{\mathcal{M}}(\cdot)$ is applied, since this can only improve the result.

5.2.2 Precision Properties

For establishing precision results, we also require some "global" properties, i.e., statements relating the results $\boldsymbol{M}_p$ and $\boldsymbol{M}_q$ of $\boldsymbol{\mathcal{M}}(\cdot)$ computed at different nodes p and q. Of course, since $\boldsymbol{M}_p$ and $\boldsymbol{M}_q$ are both accurate and hence contain t, Lemma 5.2 implies certain results on $\boldsymbol{M}_p \cup \boldsymbol{M}_q$ as well; for example, $||\boldsymbol{M}_p \cup \boldsymbol{M}_q|| \leq 2w^{n-2f-a}$. However, this would be too weak for a statement about precision enhancement.

As a first step, the single-interval faults of Definition 5.2 must be complemented by faults of *pairs of intervals* $\boldsymbol{I}_p^s$ resp. $\boldsymbol{I}_q^s$ obtained at nodes p resp. q in a broadcast from node s. In addition, we account for the case that a more severe fault comes out as a less severe on by introducing a convenient hierarchy of faults.

Definition 5.3 (Pairwise Faults) *A pair of compatible accuracy intervals* $\{\boldsymbol{I}_p^s, \boldsymbol{I}_q^s\}$ *representing* t *suffers from*

(1) *a* crash fault *iff* $\boldsymbol{I}_p^s = \boldsymbol{I}_q^s = \emptyset$,

(2) *a* symmetric fault *iff either*

- *both* $\boldsymbol{I}_p^s$ *and* $\boldsymbol{I}_q^s$ *are not accurate in the sense of* $t < \text{left}(\boldsymbol{I}_p^s)$ *and* $t < \text{left}(\boldsymbol{I}_q^s)$, *or else* $t > \text{right}(\boldsymbol{I}_p^s)$ *and* $t > \text{right}(\boldsymbol{I}_q^s)$,
- *without loss of generality,* $\boldsymbol{I}_p^s = \emptyset$ *and* $\boldsymbol{I}_q^s \neq \emptyset$ *does not exhibit an unbounded accuracy fault.*

(3) *an* asymmetric fault *iff either*

- *both* $\boldsymbol{I}_p^s$ *and* $\boldsymbol{I}_q^s$ *are not accurate in the sense of* $t > \text{right}(\boldsymbol{I}_p^s)$ *and* $t < \text{left}(\boldsymbol{I}_q^s)$ *or else* $t > \text{right}(\boldsymbol{I}_q^s)$ *and* $t < \text{left}(\boldsymbol{I}_p^s)$ *(true Byzantine fault),*
- *without loss of generality,* $\boldsymbol{I}_p^s \neq \emptyset$ *is faulty and* $\boldsymbol{I}_q^s \neq \emptyset$ *is arbitrary.*

(4) *a* restricted fault *if it suffers from a symmetric fault according to item (2) or a crash fault according to item (1).*

(5) *an* arbitrary fault *if it suffers from an asymmetric fault according to item (3) or a restricted fault according to item (4).*

Separating different classes of faults as in Definition 5.3 yields a *hybrid fault model* similar to the one of [70], see also [2] for an overview and more recent extensions. There are intricate differences, though. Most importantly, we consider pairwise perceptions $\boldsymbol{\mathcal{I}}_p$ and $\boldsymbol{\mathcal{I}}_q$ at two different receiving nodes p and q *in isolation*. By the precision requirement, we only have to ensure that any two non-faulty clocks satisfy $|C_p(t) - C_q(t)| \leq \pi$, independently of the other clocks in the system. Hence, system-global fault assumptions —like the one that all receivers perceive a symmetric fault consistently— are not required. Note that we do not even need the common assumption that the total number of faulty nodes during a round does not exceed, say, $\lfloor (n-1)/3 \rfloor$; it is only the maximum number of faulty intervals in a pairwise perception that counts.

Our fault model is thus of wider applicability than conventional hybrid fault models. Most importantly, we do not require that faults caused by receiving nodes —like receive omissions— are artificially attributed to (a small number of) sending nodes. Instead, we credit each receiving node a certain number of omissions that may affect arbitrary senders, independently of receive omissions at other receivers, simply by counting them as symmetric faults. If modelled in the conventional way, this could easily lead to a situation where all n nodes of the distributed system appear faulty.

Now we are ready for the following "precision lemma", which is an advanced version of a lemma introduced in [50]. Note that it also takes into account that two different nodes p and q usually receive slightly different intervals in the broadcast of a node s even if there is no fault.

Lemma 5.4 (Precision of Marzullo's Function) *Let* $\boldsymbol{\mathcal{I}}_p = \{\boldsymbol{I}_p^1, \ldots, \boldsymbol{I}_p^n\}$ *and* $\boldsymbol{\mathcal{I}}_q = \{\boldsymbol{I}_q^1, \ldots, \boldsymbol{I}_q^n\}$ *be two ordered sets of* $n > e' + d'$ *compatible (or empty) accuracy intervals representing* t*, where* e' *resp.* d' *of the* n *pairs of intervals* $\{\boldsymbol{I}_p^i, \boldsymbol{I}_q^i\}$ *exhibit arbitrary resp. restricted faults, and the remaining ones are non-faulty. Define* u^h *resp.* v^h *to be the length of the largest intersection of* $h \geq 1$ *unions resp. intersections of pairs of non-faulty intervals, formally* $u^h = \max\{||\boldsymbol{U}|| : \boldsymbol{U} \in \boldsymbol{\mathcal{U}}_{pq}^h\}$ *resp.* $v^h = \max\{||\boldsymbol{V}|| : \boldsymbol{V} \in \boldsymbol{\mathcal{V}}_{pq}^h\}$ *for*

$$\begin{aligned}
\boldsymbol{\mathcal{U}}_{pq}^h &= \{\boldsymbol{U} : \boldsymbol{U} = \bigcap_{i=1}^{h} \boldsymbol{I}_p^{u_i} \cup \boldsymbol{I}_q^{u_i} \text{ with } u_i \neq u_j,\ i \neq j, \text{ and non-faulty } \boldsymbol{I}_p^{u_i} \in \boldsymbol{\mathcal{I}}_p,\ \boldsymbol{I}_q^{u_i} \in \boldsymbol{\mathcal{I}}_q\} \\
\boldsymbol{\mathcal{V}}_{pq}^h &= \{\boldsymbol{V} : \boldsymbol{V} = \bigcap_{i=1}^{h} \boldsymbol{I}_p^{v_i} \cap \boldsymbol{I}_q^{v_i} \text{ with } v_i \neq v_j,\ i \neq j, \text{ and non-faulty } \boldsymbol{I}_p^{v_i} \in \boldsymbol{\mathcal{I}}_p,\ \boldsymbol{I}_q^{v_i} \in \boldsymbol{\mathcal{I}}_q\}.
\end{aligned}$$

Let $\boldsymbol{\mathcal{J}}_p = \{\boldsymbol{J}_1, \ldots, \boldsymbol{J}_{n_p}\}$ *be the set of* $n_p = n - o_p$ *non-empty intervals obtained from* $\boldsymbol{\mathcal{I}}_p$ *by discarding any of the* o_p *empty intervals caused by omissions. Using the upper bound* $f_p = d + e - o_p$ *on the number of intervals in* $\boldsymbol{J}_p$ *that (still) may be faulty in presence of* o_p *omissions, define*

$$\boldsymbol{M}_p = \boldsymbol{\mathcal{M}}_{n_p}^{n-d-e}(\boldsymbol{\mathcal{J}}_p) \qquad \text{and analogously} \qquad \boldsymbol{M}_q = \boldsymbol{\mathcal{M}}_{n_q}^{n-d-e}(\boldsymbol{\mathcal{J}}_q). \tag{5.4}$$

Then,

(1) both $\boldsymbol{M}_p$ *and* $\boldsymbol{M}_q$ *are accurate and*

$$\boldsymbol{M}_p \cap \boldsymbol{M}_q \supseteq \bigcap_{j=1}^{n-d-e} \boldsymbol{I}_p^{v_j} \cap \boldsymbol{I}_q^{v_j} = \boldsymbol{V} \tag{5.5}$$

for any possible subset $\boldsymbol{V} \in \boldsymbol{\mathcal{V}}_{pq}^{n-d-e}$*, so that* $||\boldsymbol{M}_p \cap \boldsymbol{M}_q|| \geq v^{n-d-e}$ *(distributed minimal intersection property),* *(2) if $e' \leq e$ and $d' \leq d$, there are at least $n-2d-2e-e' \geq n-2d-3e$ pairs of non-faulty intervals $\{\boldsymbol{I}_p^{u_k}, \boldsymbol{I}_q^{u_k}\}$ with $\boldsymbol{I}_p^{u_k} \in \boldsymbol{\mathcal{J}}_p$ and $\boldsymbol{I}_q^{u_k} \in \boldsymbol{\mathcal{J}}_q$ such that*

$$\boldsymbol{M}_p \cup \boldsymbol{M}_q \subseteq \bigcap_{k=1}^{n-2d-2e-e'} \boldsymbol{I}_p^{u_k} \cup \boldsymbol{I}_q^{u_k} \subseteq \bigcap_{k=1}^{n-2d-3e} \boldsymbol{I}_p^{u'_k} \cup \boldsymbol{I}_q^{u'_k}, \tag{5.6}$$

where the sequence $\{u'_k\}_{1\leq k\leq n-2d-2e-e'}$ by discarding $e-e'$ arbitrary elements. Hence, $||\boldsymbol{M}_p \cup \boldsymbol{M}_q|| \leq u^{n-2d-2e-e'} \leq u^{n-2d-3e}$.

Proof. First of all, we note that f_p gives indeed an upper bound on the number of intervals in $\boldsymbol{\mathcal{J}}_p$ that still may be faulty in presence of $o_p \leq d' \leq d$ omissions, since it holds that

$$n_p - f_p = n - d - e \tag{5.7}$$

and the same for $n_q - f_q$. Hence, Lemma 5.2 is applicable, and it follows that $\boldsymbol{M}_p$ and $\boldsymbol{M}_q$ are both accurate and satisfy the (local) minimal intersection property. That is, $\boldsymbol{M}_p$ contains any intersection of $n_p - f_p$ non-faulty intervals present in $\boldsymbol{\mathcal{J}}_p$. If $\{v_j\}_{1\leq j\leq n_p-f_p}$ denotes any sequence of different indices of non-faulty pairs of intervals $\boldsymbol{I}_p^{v_j} \in \boldsymbol{\mathcal{I}}_p$, $\boldsymbol{I}_q^{v_j} \in \boldsymbol{\mathcal{I}}_q$ (of course also present in $\boldsymbol{\mathcal{J}}_p$, $\boldsymbol{\mathcal{J}}_q$), we thus have

$$\boldsymbol{W}_p = \bigcap_{j=1}^{n_p-f_p} \boldsymbol{I}_p^{v_j} \subseteq \boldsymbol{M}_p$$

and, for the same sequence $\{v_j\}$, $\boldsymbol{W}_q = \bigcap_{j=1}^{n_p-f_p} \boldsymbol{I}_q^{v_j} \subseteq \boldsymbol{M}_q$. By elementary set algebra, it thus follows that $\boldsymbol{V} = \boldsymbol{W}_p \cap \boldsymbol{W}_q \in \boldsymbol{\mathcal{V}}^{n_p-f_p}$ satisfies (5.5). Finally, $||\boldsymbol{M}_p \cap \boldsymbol{M}_q|| \geq v^{n_p-f_p}$ is a simple consequence of the definition of v^h as the maximum length of $\boldsymbol{V} \in \boldsymbol{\mathcal{V}}_{pq}^h$. This completes the proof of item (1).

For item (2), we distinguish two cases:

1. If w.l.o.g. $\boldsymbol{M}_p$ determines both left and right edge of $\boldsymbol{M}_p \cup \boldsymbol{M}_q$, Lemma 5.2 applies with $n := n_p$, $f := f_p$, and $a' \leq e'$ (as well as $a \leq e$). Hence, by item (2) of this lemma in conjunction with (5.7), there are at least

$$n_p - 2f_p - a' \geq n - 2d - 2e - e'$$

non-faulty intervals $\boldsymbol{I}_p^{b_i}$ in $\boldsymbol{\mathcal{J}}_p$, the intersection of which majorizes $\boldsymbol{M}_p$. Of course, any corresponding $\boldsymbol{I}_q^{b_i}$ is present in $\boldsymbol{\mathcal{J}}_q$, so that we can write

$$\begin{aligned}\boldsymbol{M}_p \cup \boldsymbol{M}_q \subseteq \boldsymbol{M}_p &\subseteq \bigcap_{j=1}^{n-2d-2e-e'} \boldsymbol{I}_p^{b_j} \\ &\subseteq \bigcap_{j=1}^{n-2d-2e-e'} \boldsymbol{I}_p^{b_j} \cup \boldsymbol{I}_q^{b_j} \in \boldsymbol{\mathcal{U}}_{pq}^{n-2d-2e-e'},\end{aligned} \tag{5.8}$$

which shows (5.6) in this case.

2. If w.l.o.g. the left resp. right edge of $\boldsymbol{M}_p \cup \boldsymbol{M}_q$ is determined by $\boldsymbol{M}_p$ resp. $\boldsymbol{M}_q$, where the left resp. right edge of $\boldsymbol{M}_p$ resp. $\boldsymbol{M}_q$ intersects with $g_{p,l}$ resp. $g_{q,r}$ intervals belonging to a non-faulty pair of intervals in $calBI_p$, $\boldsymbol{\mathcal{I}}_q$, we easily find

$$\begin{aligned} g_{p,l} &\geq n_p - f_p - e' - (d_{\text{left}} - d_{p,\text{left}}) \geq n - d - e - e' - d_{\text{left}} \\ g_{q,r} &\geq n_q - f_q - e' - (d_{\text{right}} - d_{q,\text{right}}) \geq n - d - e - e' - d_{\text{right}}, \end{aligned}$$

where $d_{\text{left}} + d_{\text{right}} = d' \leq d$ are the number of restricted faulty pairs of intervals lying left resp. right of t, and $d_{p,\text{left}} \leq d_p$, $d_{q,\text{right}} \leq d_q$ denote the number of omissions among them at node p resp. q. The lower bounds given above come immediately from (5.7). On the other hand, we have only $g \leq n - d' - e'$ different non-faulty pairs of intervals. Thus, the usual pigeonhole argument reveals that

$$\begin{aligned} g_{p,l} + g_{q,r} - g &\geq 2n - 2d - 2e - 2e' - d' - n + d' + e' \\ &\geq n - 2d - 2e - e' \\ &\geq n - 2d - 3e \end{aligned}$$

of them must be the same. We can conclude that there are $n - 2d - 2e - e'$ (pairs of) accurate intervals $\boldsymbol{I}_p^{b_1} \cup \boldsymbol{I}_q^{b_1}, \ldots, \boldsymbol{I}_p^{b_{n-2d-2e-e'}} \cup \boldsymbol{I}_q^{b_{n-2d-2e-e'}}$ with $\boldsymbol{I}_p^{b_i} \in \boldsymbol{\mathcal{J}}_p$ and $\boldsymbol{I}_q^{b_i} \in \boldsymbol{\mathcal{J}}_q$ satisfying

$$\boldsymbol{M}_p \cup \boldsymbol{M}_q \subseteq \bigcap_{j=1}^{n-2d-2e-e'} \boldsymbol{I}_p^{b_j} \cup \boldsymbol{I}_q^{b_j} \in \boldsymbol{\mathcal{U}}_{pq}^{n-2d-2e-e'}, \tag{5.9}$$

which proves (5.6) for this case as well.

The condition $||\boldsymbol{M}_p \cup \boldsymbol{M}_q|| \leq u^{n-2d-2e-e'} \leq u^{n-2d-3e}$ is a trivial consequence of (5.8) and (5.9) finishing up the proof. □

Interpreting the accomplishments of Lemma 5.4 and the previous comment in terms of the fault-tolerance degree, it turns out that $n \geq 2d + 2e + e'$ nodes are required to guarantee that $\boldsymbol{M}_p \cup \boldsymbol{M}_q$ remains bounded by the length of the union of at least one pair of non-faulty input intervals $\boldsymbol{I}_p^j \cup \boldsymbol{I}_q^j$. Hence, as much as

$$n \geq \begin{cases} 2d + 1 & \text{for } d' \leq d \text{ restricted faults,} \\ 2e + e' + 1 \leq 3e + 1 & \text{for } e' \leq e \text{ arbitrary faults} \end{cases}$$

nodes are required for tolerating faults of the given type.

5.2.3 Reference Point Setting

Whereas $\mathcal{M}(\cdot)$ is optimal w.r.t. the length of the resulting accuracy interval, it can nevertheless exhibit large discontinuous jumps even for minor shifts of a faulty input interval, recall the comments on Lemma 5.3. For that reason, it has been claimed in [25] that $\mathcal{M}(\cdot)$ is incapable of guaranteeing small precision if the input intervals are large. The following example proves this claim: Consider a system of four nodes A, B, C, D with interval clocks having the following characteristics:

- A's interval clock is deteriorated by ±2 units during the resynchronization period P, but actually runs at perfect rate,
- B's interval clock is deteriorated by ±1 unit and is actually one unit ahead of real-time after one period P,
- C's interval clock is deteriorated by ±1 unit and is actually one unit behind of real-time after one period P,
- D's clock exhibits arbitrarily faulty behavior in the sense that D's accuracy interval received by node p mimics p's current interval clock.

Assuming fault-free, zero-delay communication and initially perfectly synchronized interval clocks $\boldsymbol{C}_p(0) = [0 \pm \mathbf{1}]$ for $p = A, B, C$, we obtain the scenario depicted in Figure 5.2. Each node receives the interval clocks $\boldsymbol{C}_p$ of the non-faulty nodes A, B, C exactly as they are shown above at $t = kP$, $k \geq 1$. One observes that the interval $\boldsymbol{M}_p$ obtained by applying $\mathcal{M}_4^3(\cdot)$ to the $\boldsymbol{I}_p^q$'s received by node p is always exactly $\boldsymbol{C}_p$, hence no precision enhancement will ever take place here, and the reference points of $\boldsymbol{C}_B$ and $\boldsymbol{C}_C$ will drift apart perpetually.

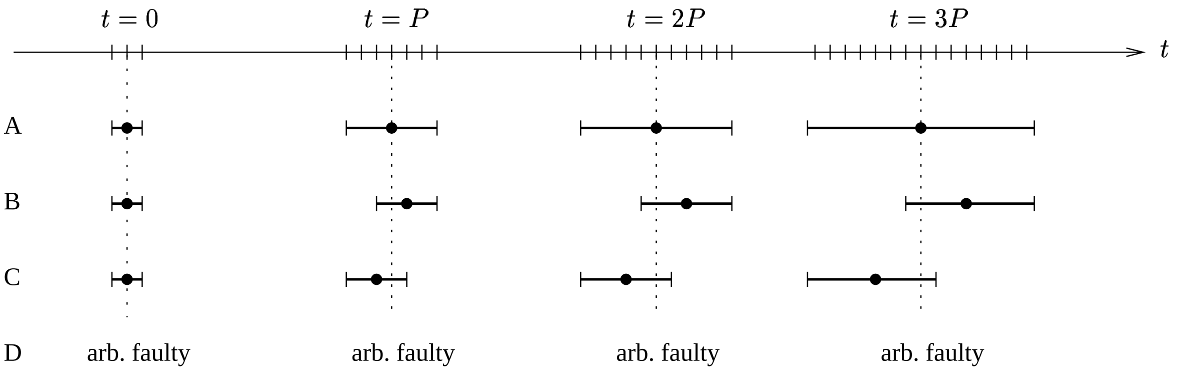


Figure 5.2: *Lacking Precision Enhancement Property*

This behavior can be avoided if one applies $\boldsymbol{\mathcal{M}}_4^3(\cdot)$ to the associated $\boldsymbol{\pi}^H$-precision intervals $\hat{\boldsymbol{I}}_p^q$ instead of $\boldsymbol{I}_p^q$, recall Definition 3.2. In the example of Figure 5.2, $\boldsymbol{\pi}^H$ must satisfy $||\boldsymbol{\pi}^H|| = 2 + 4$ units because of the initial precision (= 2 units) plus twice the maximum deterioration during P (= $2 \cdot 2$ units). The intervals fed into $\boldsymbol{\mathcal{M}}_4^3(\cdot)$ are hence trimmed to length π^H, and considering the resulting intervals $\hat{\boldsymbol{M}}_p$ and $\hat{\boldsymbol{M}}_q$ at two different nodes p and q, one finds that the reference points cannot be further apart than $\pi^H/2$ (centerpoint setting assumed), since $\hat{\boldsymbol{M}}_p \cup \hat{\boldsymbol{M}}_q$ has a length of at most π^H by item (2) of Lemma 5.4.

Since the above considerations are only meaningful for maintaining precision, i.e., setting the reference point of $\boldsymbol{C}_p$, the optimality of $\boldsymbol{\mathcal{M}}(\cdot)$ recommends its use for determining left and right edge of $\boldsymbol{C}_p$. However, this requires some care, because the reference point computed via the associated precision intervals might lie outside of the accuracy interval. This is demonstrated by the following example: Reconsider our system of four nodes A, B, C, D, now with the following characteristics:

- A's interval clock is deteriorated by ±1 unit and is actually one unit ahead of real-time after one period P,
- B's, C's interval clocks are deteriorated by ±1 unit and are actually one unit behind real-time after period P,
- D's clock exhibits restricted faults.

Assuming fault-free, zero-delay communication and initially synchronized clocks satisfying $\boldsymbol{C}_p(t_0) = \hat{\boldsymbol{C}}_p(t_0)$ for $p = A, B, C$ and $\boldsymbol{\pi}_0 = [-1, 1]$, so that $\boldsymbol{\pi}^H = [-2, 2]$ (since maximum deterioration during P is ± 1), consider the evolution of accuracy intervals during two rounds depicted in Figure 5.3. At t_1, applying $\boldsymbol{\mathcal{M}}_4^3(\cdot)$ to the received accuracy intervals resp. the associated $\boldsymbol{\pi}^H$-precision intervals (which satisfy $\boldsymbol{C}_q^p(t_1) = \hat{\boldsymbol{C}}_q^p(t_1)$ for $p, q \in \{A, B, C\}$ here) yields $\boldsymbol{C}_A(t_1) = [t_1 \pm \boldsymbol{0}]$ and $\boldsymbol{C}_B(t_1) = \boldsymbol{C}_C(t_1) = [t_1 - 2 \pm \boldsymbol{2}]$ at the respective nodes; recall that the reference point of $\boldsymbol{C}_q$ is computed as the centerpoint of $\boldsymbol{\mathcal{M}}(\cdot)$ applied to the $\hat{\boldsymbol{C}}_q^p$'s. In order to ensure $\boldsymbol{\pi}_0$-correctness of clock A and B, internal global time τ_1 must be set to $t_1 - 1$ to lie in the intersection of the renewed $\boldsymbol{\pi}_0$-precision intervals. Although τ_1 does not lie in $\boldsymbol{C}_A(t_1)$, this situation is still feasible due to the fact that we decoupled precision and accuracy in the definition of $\boldsymbol{\pi}$-precision intervals. However, a problem shows up when setting the reference point of clock A at t_2: Applying $\boldsymbol{\mathcal{M}}(\cdot)$ to the received accuracy intervals provides $[t_2 \pm \boldsymbol{0}]$, but the reference point evaluates to $t_2 - 1$, which lies outside. If we ignored this, that is, if we set the reference point to t_2, precision would be violated. Therefore, the accuracy interval must be enlarged to the left by 1 to include the reference point. Note also that internal global time τ_2 must be set to $t_2 - 2$ here.

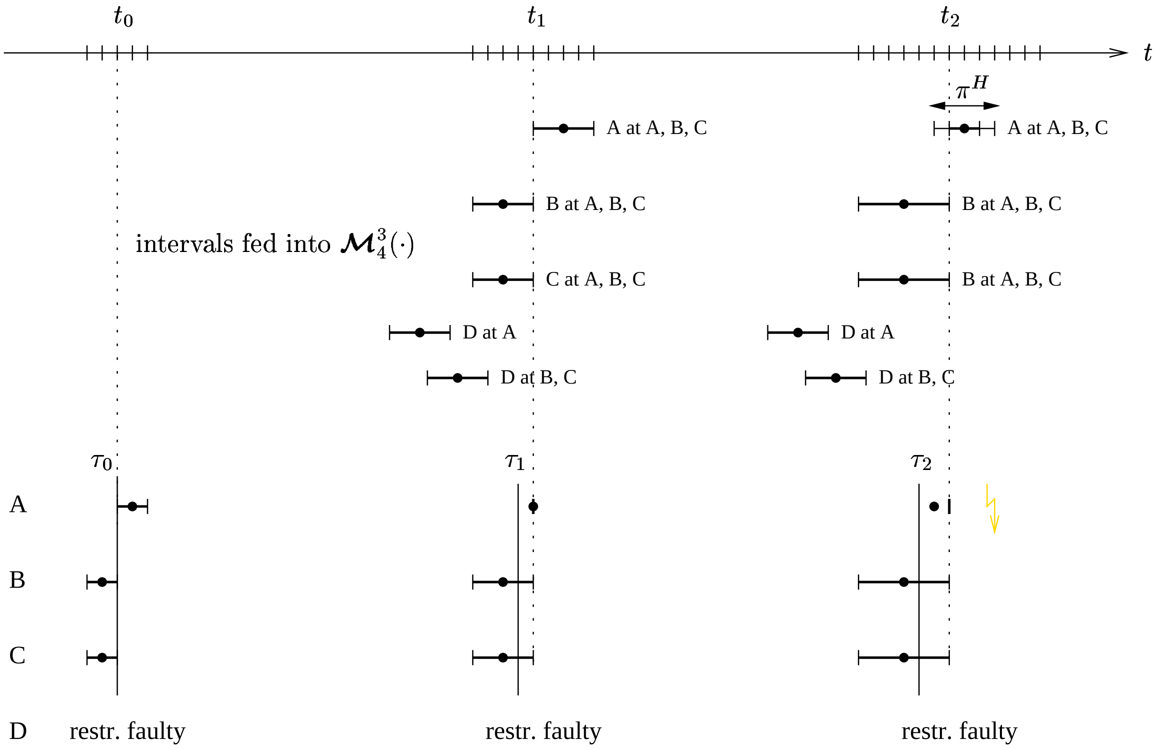


Figure 5.3: *Reference Point Setting outside an Accuracy Interval*

Viewed from a different angle, this problem can be seen as a consequence of the fact that internal global time may drift away from real-time. In Figure 5.3, it is node D's faulty behavior that slows down the overall progress of internal global time relative to real-time. This reveals that limiting $\boldsymbol{\pi}$-precision intervals to subintervals of the accuracy interval, i.e., by defining $\hat{\boldsymbol{I}} = [r \pm \boldsymbol{\pi}] \cap \boldsymbol{I}$ for an interval $\boldsymbol{I} = [r \pm \boldsymbol{\alpha}]$, does not work in general. Alternatively, we could pad any accuracy interval $[r \pm \boldsymbol{\alpha}]$ computed by $\boldsymbol{\mathcal{M}}(\cdot)$ to enforce the condition $\boldsymbol{\pi} \subseteq \boldsymbol{\alpha}$, so that $\tau_1 \in \boldsymbol{C}_A(t_1)$ in Figure 5.3 can be guaranteed, however, this approach spoils accuracy more. For that reason, we eventually decoupled accuracy and precision, thus viewing them as "orthogonal" issues. Note that this opens up the possibility of employing virtually any internal synchronization algorithm for maintaining precision and to enlarge accuracy as needed.

In any case, we need some means to set the reference point, and since we assumed a typical processor with integer arithmetic only, the following discrete asymmetric reference point setting operation turned out to be useful.

Definition 5.4 (Discrete Reference Point Setting) *Let an interval* $\boldsymbol{I} = [a, b]$ *with* a, b *being integer multiples of* $G_S > 0$ *and some arbitrary* $\boldsymbol{\pi} = [-\pi^-, \pi^+]$ *satisfying* $\pi = \pi^- + \pi^+ > 0$ *be given. With* $\lfloor x \rfloor_{G_S}$ *denoting truncation of* x *to the next integer multiple of* G_S *being* $\leq x$*, and* $\lceil x \rceil_{G_S}$ *denoting rounding up* x *to the next integer multiple of* G_S *being* $\geq x$*, we define*

$$\boldsymbol{\pi}\text{−center}_{G_S}(\boldsymbol{I}) = \left\lfloor \frac{\pi^- b + \pi^+ a}{\pi} \right\rfloor_{G_S}. \tag{5.10}$$

Note that this operation provides a reference point that partitions $\boldsymbol{I}$ according to the proportion of $\pi^- : \pi^+$. The following two technical lemmas are dealing with the properties of $\boldsymbol{\pi}\text{−center}_{G_S}$ operation.

Lemma 5.5 (Accuracies Discrete Reference Point Setting) *Let* $\boldsymbol{I} = [a, b]$ *with* $0 \leq a \leq b$ *being integer multiples of* $G_S > 0$ *and an arbitrary interval* $\boldsymbol{\pi} = [-\pi^-, \pi^+]$ *with* $\pi = \pi^- + \pi^+ > 0$ *be given. Then,*

$$\boldsymbol{\pi}\text{−center}_{G_S}([a, b]) \leq \boldsymbol{\pi}\text{−center}_{G_S}([a + x, b + y]) \tag{5.11}$$

for any $x, y \geq 0$ *being integer multiples of* G_S*, and the accuracies in the interval* $[r \pm \boldsymbol{\alpha}]$ *obtained from* $\boldsymbol{I}$ *by setting the reference point to* $r = \boldsymbol{\pi}\text{−center}_{G_S}(\boldsymbol{I})$ *satisfy*

$$\alpha^- = \left\lfloor \frac{\pi^-}{\pi} ||\boldsymbol{I}|| \right\rfloor_{G_S}, \tag{5.12}$$

$$\alpha^+ = \left\lceil \frac{\pi^+}{\pi} ||\boldsymbol{I}|| \right\rceil_{G_S}. \tag{5.13}$$

Proof. Monotonicity (5.11) follows immediately from the definition (5.10) of $\boldsymbol{\pi}\text{−center}_{G_S}$ and monotonicity of $\lfloor x \rfloor$. Using the well-known fact $-\lceil x \rceil = \lfloor -x \rfloor$, the expressions for negative and positive accuracy yield

$$\alpha^- = \left\lfloor \frac{\pi^- b + \pi^+ a}{\pi} \right\rfloor_{G_S} - a = \left\lfloor \frac{\pi^- b + \pi^+ a - \pi a}{\pi} \right\rfloor_{G_S} = \left\lfloor \frac{\pi^-}{\pi} ||\boldsymbol{I}|| \right\rfloor_{G_S}$$

and

$$\alpha^+ = b - \left\lfloor \frac{\pi^- b + \pi^+ a}{\pi} \right\rfloor_{G_S} = b + \left\lceil -\frac{\pi^- b + \pi^+ a}{\pi} \right\rceil_{G_S} = \left\lceil \frac{\pi^+}{\pi} ||\boldsymbol{I}|| \right\rceil_{G_S}.$$

□

Lemma 5.6 (Reference Point Precision) *Let* $\boldsymbol{I}_p$*,* $\boldsymbol{I}_q$ *be two consistent intervals with length* $0 \leq ||\boldsymbol{I}_p|| \leq \bar{\pi}_p$*,* $0 \leq ||\boldsymbol{I}_q|| \leq \bar{\pi}_q$ *being integer multiples of* G_S*, and* $\text{ref}(\boldsymbol{I}_p) =$

$\boldsymbol{\pi}_p$−center$_{G_S}(\boldsymbol{I}_p)$, $\mathrm{ref}(\boldsymbol{I}_q) = \boldsymbol{\pi}_q$−center$_{G_S}(\boldsymbol{I}_q)$ for some $\boldsymbol{\pi}_p = [-\pi_p^-, \pi_p^+]$ and $\boldsymbol{\pi}_q = [-\pi_q^-, \pi_q^+]$. If $||\boldsymbol{I}_p \cup \boldsymbol{I}_q|| \leq \pi$ with $\max\{\bar{\pi}_p, \bar{\pi}_q\} \leq \pi \leq \bar{\pi}_p + \bar{\pi}_q$, then

$$||\mathrm{ref}(\boldsymbol{I}_p) - \mathrm{ref}(\boldsymbol{I}_q)|| \leq \begin{cases} \max\Big\{ \Big\lceil \frac{\pi_q^-}{\pi_q}\bar{\pi}_q + \frac{\pi_p^+}{\pi_p}(\pi - \bar{\pi}_q) \Big\rceil_{G_S}, \Big\lceil \frac{\pi_p^-}{\pi_p}\bar{\pi}_p + \frac{\pi_q^+}{\pi_q}(\pi - \bar{\pi}_p) \Big\rceil_{G_S} \Big\} \\ \qquad \text{if } \frac{\pi_q^-}{\pi_q} \geq \frac{\pi_p^+}{\pi_p}, \\ \max\Big\{ \Big\lceil \frac{\pi_p^+}{\pi_p}\bar{\pi}_p + \frac{\pi_q^-}{\pi_q}(\pi - \bar{\pi}_p) \Big\rceil_{G_S}, \Big\lceil \frac{\pi_q^+}{\pi_q}\bar{\pi}_q + \frac{\pi_p^-}{\pi_p}(\pi - \bar{\pi}_q) \Big\rceil_{G_S} \Big\} \\ \qquad \text{otherwise.} \end{cases}$$

Proof. Apart from discreteness of $\boldsymbol{\pi}_p$−center$_{G_S}$, this is a straightforward linear programming problem. One has to look out for an arrangement of the consistent intervals $\boldsymbol{I}_p = [a, b]$ and $\boldsymbol{I}_q = [c, d]$ that maximizes the distance of their reference points. The full-length proof can be found in the technical report [52]. □

5.3 Optimal Precision Convergence Function

The *optimal precision convergence function* $\mathcal{OP}(\cdot)$, originally introduced in [51], treats precision and accuracy independently of each other and works as follows: At any node q, Marzullo's function $\mathcal{M}(\cdot)$ is applied to the set $\mathcal{I}_q$ of preprocessed accuracy intervals $\boldsymbol{I}_q^p$ to compute a new accuracy interval for q's interval clock $\boldsymbol{C}_q$; In addition, we consider the fact that it does not make sense to "correct" a non-faulty clock to an interval that is worse than its current one. Consequently, it is perfectly reasonable to use the intersection $\mathcal{M}(\mathcal{I}_q) \cap \boldsymbol{I}_q^q$ for clock correction at node q, where $\boldsymbol{I}_q^q$ denotes the accuracy interval originating in the node's own clock. This intersection is guaranteed to be non-empty for a non-faulty node q, since $t \in \boldsymbol{I}_q^q$ and also $t \in \mathcal{M}(\mathcal{I}_q)$. Due to the fact that no network transmission and hence delay compensation operation is required for $\boldsymbol{I}_q^q$ (and certain granularity effects do not show up either), this interval is usually considerably smaller than remote intervals $\boldsymbol{I}_q^p$ that ultimately determine $\mathcal{M}(\mathcal{I}_q)$. As a result, the achievable accuracy will be improved appropriately. Without the intersection of $\boldsymbol{I}_q^q$ we are dealing with the precursor of $\mathcal{OP}(\cdot)$, called the *orthogonal accuracy convergence function* $\mathcal{OA}(\cdot)$, see [52].

Of course, exactly the same reasoning also applies to the precision part of $\mathcal{OP}(\cdot)$. More specifically, the reference point is set to the midpoint of the intersection $\hat{\boldsymbol{C}}_q = \mathcal{M}(\hat{\mathcal{I}}_q) \cap \check{\boldsymbol{I}}_q^q$, where $\check{\boldsymbol{I}}_q^q$ denotes the $\boldsymbol{\pi}_q^q$-precision interval originating in the own node q's clock. Usually, $\boldsymbol{\pi}_q^q$ is also much smaller than the precision $\boldsymbol{\pi}_q^p$ of a remote precision interval $\hat{\boldsymbol{I}}_q^p$, which immediately carries over to the above intersection. This ultimately leads to a considerably improved precision.

This section provides the formal definition of $\mathcal{OP}(\cdot)$, whose characteristic functions are derived in Subsection 5.3.1. Instantiating it as convergence function for clock state synchronization leads to algorithm OP-STATE and for clock rate synchronization to algorithm OP-RATE, which are fully analyzed in Subsection 5.3.2 and 5.3.3, respectively.

Definition 5.5 (Optimal Precision Function) *Let $\boldsymbol{\mathcal{I}}$ be a set of n compatible accuracy intervals and $\boldsymbol{\mathcal{J}} \subseteq \boldsymbol{\mathcal{I}}$ with $|\boldsymbol{\mathcal{J}}| = n' \leq n$ be the set of non-empty ones among them. Moreover, with $\boldsymbol{I}_o^o \in \boldsymbol{\mathcal{J}}$ denoting the interval originating in the own node's clock, let $\hat{\boldsymbol{\mathcal{J}}} = \{\hat{\boldsymbol{J}}^1, \ldots, \hat{\boldsymbol{J}}^{n'-1}, \check{\boldsymbol{I}}_o^o\}$ be the set of associated precision intervals utilizing a given precision $\boldsymbol{\pi}^H$ for $\hat{\boldsymbol{J}}^i$ and $\boldsymbol{\pi}^o \subset \boldsymbol{\pi}^H$ for $\check{\boldsymbol{I}}_o^o$, where π^{H-}, π^{H+}, π^{o-}, π^{o+} are integer multiples of G_S. For some* a priori *given fault-tolerance parameter f and a maximum correction bound $\Upsilon_{\max}$, the* optimal precision convergence function $\boldsymbol{\mathcal{OP}}_{n-f}^{\boldsymbol{\pi}^o, \boldsymbol{\pi}^H, \Upsilon_{\max}}(\cdot)$, *herein abbreviated by $\boldsymbol{\mathcal{OP}}(\cdot)$, is defined by*

$$\operatorname{ref}\Big(\boldsymbol{\mathcal{OP}}(\boldsymbol{\mathcal{J}})\Big) = \begin{cases} r_n & \text{if } |r_n - r_o| \leq \Upsilon_{\max}, \\ r_o + \Upsilon_{\max} & \text{if } r_n - r_o > \Upsilon_{\max}, \\ r_o - \Upsilon_{\max} & \text{if } r_n - r_o < -\Upsilon_{\max}, \end{cases} \tag{5.14}$$

where

$$\begin{aligned} r_o &= \operatorname{ref}(\boldsymbol{I}_o^o), \\ r_n &= \boldsymbol{\pi}^o\text{−center}_{G_S}\Big(\boldsymbol{\mathcal{M}}_{n'}^{n-f}(\hat{\boldsymbol{\mathcal{J}}}) \cap \check{\boldsymbol{I}}_o^o\Big), \end{aligned} \tag{5.15}$$

and

$$\boldsymbol{\mathcal{OP}}(\boldsymbol{\mathcal{J}}) = \Big(\boldsymbol{\mathcal{M}}_{n'}^{n-f}(\boldsymbol{\mathcal{J}}) \cap \boldsymbol{I}_o^o\Big) \cup \Big[\operatorname{ref}\Big(\boldsymbol{\mathcal{OP}}(\boldsymbol{\mathcal{J}})\Big) \pm \boldsymbol{0}\Big]. \tag{5.16}$$

That is, the result of $\boldsymbol{\mathcal{OP}}(\cdot)$ is the interval provided by $\boldsymbol{\mathcal{M}}(\cdot)$ applied to the accuracy intervals in $\boldsymbol{\mathcal{J}}$, possibly extended appropriately to include the reference point; the latter is set to the $\Upsilon_{\max}$-bounded $\boldsymbol{\pi}^o$−center$_{G_S}$ of $\boldsymbol{\mathcal{M}}(\cdot)$ applied to the associated precision intervals. Obviously, $\Upsilon_{\max} = \infty$ means that the new reference point is always set to r_n, whereas a finite $\Upsilon_{\max}$ implies that the maximum clock correction applied to any non-faulty clock is limited to $-\Upsilon_{\max}$ resp. $+\Upsilon_{\max}$. Inspired by [10], we will show in Lemma 5.8 that $\Upsilon_{\max}$ can be reduced up to the provably necessary maximum clock correction established in [9].

Before characterizing $\boldsymbol{\mathcal{OP}}(\cdot)$ in detail, we point out that it is obviously translation invariant due to item (1) of Lemma 5.1 and the fact that $(\boldsymbol{I}+\Delta)\cap(\boldsymbol{I}'+\Delta) = (\boldsymbol{I}\cap\boldsymbol{I}')+\Delta$ and $(\boldsymbol{I}+\Delta)\cup(\boldsymbol{I}'+\Delta) = (\boldsymbol{I}\cup\boldsymbol{I}')+\Delta$. Moreover, for the purpose of clock rate synchronization, we make the remark that $\boldsymbol{\mathcal{OP}}(\cdot)$ is multiplicative due to item (2) of Lemma 5.1 and the fact that $s\boldsymbol{I} \cap s\boldsymbol{I}' = s(\boldsymbol{I} \cap \boldsymbol{I}')$ and $s\boldsymbol{I} \cup s\boldsymbol{I}' = s(\boldsymbol{I} \cup \boldsymbol{I}')$.

5.3.1 Characteristic Functions

In the remainder of this section, we will analyze the worst case behavior of this convergence function by evaluating its characteristic functions according to Definition 3.11. For this purpose, we have to provide an appropriate fault model $\mathcal{F}$ first, which is a refinement of Definition 5.3 that takes into account that faults may occur both in accuracy and precision intervals. Actually, we have to distinguish faults affecting an accuracy interval and its associated precision interval either in the same way or differently: Let a single accuracy interval $\boldsymbol{I}$ that is faulty w.r.t. t and/or τ be called *t/τ-symmetrically faulty* if $t < \mathrm{left}(\boldsymbol{I})$ and $\tau < \mathrm{left}(\hat{\boldsymbol{I}})$ or either $t > \mathrm{right}(\boldsymbol{I})$ and $\tau > \mathrm{right}(\hat{\boldsymbol{I}})$, and *$t/\tau$-asymmetrically faulty* otherwise; a set $\boldsymbol{\mathcal{I}}$ of faulty accuracy intervals is *identically* t/τ-symmetrically faulty if t is either to the left or to the right for all members of $\boldsymbol{\mathcal{I}}$ identically.

Assumption 5.1 (Hybrid Fault Model) *A pair of compatible accuracy intervals $\{\boldsymbol{I}_p^s, \boldsymbol{I}_q^s\}$ representing t, with the associated precision intervals $\{\hat{\boldsymbol{I}}_p^s, \hat{\boldsymbol{I}}_q^s\}$ representing τ, suffers from a*

(1) restricted fault *if it suffers from a crash fault or a symmetric fault w.r.t. t and/or τ and all faulty intervals involved are identically t/τ-symmetrically faulty,*

(2) arbitrary fault *if it suffers either from an asymmetric fault w.r.t. t and/or τ, a symmetric fault involving at least one t/τ-asymmetrically faulty interval, or a restricted fault.*

Let $\boldsymbol{\mathcal{I}}_p = \{\boldsymbol{I}_p^1, \ldots, \boldsymbol{I}_p^n\}$ and $\boldsymbol{\mathcal{I}}_q = \{\boldsymbol{I}_q^1, \ldots, \boldsymbol{I}_q^n\}$ be the sets of intervals obtained at any two non-faulty nodes p and q after reception and preprocessing of the accuracy intervals disseminated in an FME. We assume that at most $e' \le e$ and $d' \le d$ of the pairs of intervals $\{\boldsymbol{I}_p^s, \boldsymbol{I}_q^s\}$, $1 \le s \le n$, suffer from arbitrary and restricted faults, respectively, where e and d are such that

$$n \geq 3e + 2d + 1. \tag{5.17}$$

Now we are ready for the lemma providing the most important properties of $\boldsymbol{\mathcal{OP}}(\cdot)$, which are analyzed according to the framework in Chapter 3. More specifically, we provide expressions for the (conditional) accuracy preservation functions $\Phi_\alpha^\pm(\cdot)$, precision preservation function $\boldsymbol{\Phi}_\pi(\cdot)$, precision enhancement function $\Psi_\pi(\cdot)$, and (conditional) intersection enhancement functions $\Psi_\iota^\pm(\cdot)$ of $\boldsymbol{\mathcal{OP}}(\cdot)$ for the special case $\Upsilon_{\max} = \infty$ first. In Lemma 5.8 we will show that our results remain valid for (certain) finite settings of $\Upsilon_{\max}$ as well.

Lemma 5.7 (Optimal Precision Function for $\Upsilon_{\max} = \infty$) *Let $\boldsymbol{\mathcal{I}}_p = \{\boldsymbol{I}_p^1, \dots, \boldsymbol{I}_p^n\}$ resp. $\boldsymbol{\mathcal{I}}_q = \{\boldsymbol{I}_q^1, \dots, \boldsymbol{I}_q^n\}$ be two ordered sets of n compatible accuracy intervals obtained at nodes p resp. q at the end of a round, which are in accordance with the fault model of Assumption 5.1. Moreover, let the subsets of non-empty accuracy intervals among them be $\boldsymbol{\mathcal{J}}_p \subseteq \boldsymbol{\mathcal{I}}_p$, $|\boldsymbol{\mathcal{J}}_p| = n_p \le n$, and $\boldsymbol{\mathcal{J}}_q \subseteq \boldsymbol{\mathcal{I}}_q$, $|\boldsymbol{\mathcal{J}}_q| = n_q \le n$, respectively, with $\boldsymbol{I}_p^p \in \boldsymbol{\mathcal{J}}_p$ and $\boldsymbol{I}_q^q \in \boldsymbol{\mathcal{J}}_q$ denoting the interval originating in the own node's clock. Assume further that*

[1] the accuracies of any non-faulty $\boldsymbol{I}_p^i = [T_p^i \pm \boldsymbol{\alpha}_p^i]$ are integer multiples of G_S satisfying $\boldsymbol{\alpha}_p^i \subseteq \boldsymbol{\beta}_p^i \in \boldsymbol{\mathcal{B}}_p$ for a given set of accuracy bounds $\boldsymbol{\mathcal{B}}_p = \{\boldsymbol{\beta}_p^1, \dots, \boldsymbol{\beta}_p^n\}$, and analogous for $\boldsymbol{I}_q^i$ with set of accuracy bounds $\boldsymbol{\mathcal{B}}_q$,

[2] any non-faulty $\boldsymbol{I}_p^i \in \boldsymbol{\mathcal{I}}_p$ as well as any non-faulty $\boldsymbol{I}_q^i \in \boldsymbol{\mathcal{I}}_q$ is $\boldsymbol{\pi}^H$-correct for some given $\boldsymbol{\pi}^H$ with π^{H+}, π^{H-} being integer multiples of G_S, and both $\boldsymbol{I}_p^p$ and $\boldsymbol{I}_q^q$ are $\boldsymbol{\pi}^o$-correct for some $\boldsymbol{\pi}^o \subset \boldsymbol{\pi}^H$ with π^{o-}, π^{o+} being integer multiples of G_S,

[3] any pair of non-faulty intervals $\{\boldsymbol{I}_p^i, \boldsymbol{I}_q^i\}$ is $\boldsymbol{\pi}_I$-precise for some given $\boldsymbol{\pi}_I \subseteq \boldsymbol{\pi}^H$, where π_I is an integer multiple of G_S,

[4] for any $s \in \{1, \dots, n\}$ with both $\boldsymbol{I}_p^s$ and $\boldsymbol{I}_q^s$ being non-faulty, the common intersection of the associated precision intervals $\hat{\boldsymbol{I}}_p^s \cap \hat{\boldsymbol{I}}_q^s \cap \hat{\boldsymbol{I}}_p^{\min_p} \cap \hat{\boldsymbol{I}}_q^{\min_q}$ resp. $\hat{\boldsymbol{I}}_p^s \cap \hat{\boldsymbol{I}}_q^s \cap \hat{\boldsymbol{I}}_p^{\max_p} \cap \hat{\boldsymbol{I}}_q^{\max_q}$ has length at least $\iota_s^+ \ge 0$ resp. $\iota_s^- \ge 0$ (all integer multiples of G_S), where $\min_x$ resp. $\max_x$ represents that non-faulty node that leads to the leftmost $\mathrm{right}(\hat{\boldsymbol{I}}_x^{\min_x})$ resp. the rightmost $\mathrm{left}(\hat{\boldsymbol{I}}_x^{\max_x})$ for $x \in \{p, q\}$.

The convergence function $\boldsymbol{\mathcal{OP}}_{n-d-e}^{\boldsymbol{\pi}^o, \boldsymbol{\pi}^H, \infty}(\cdot)$ applied to $\boldsymbol{\mathcal{J}}_p$ resp. $\boldsymbol{\mathcal{J}}_q$ at node p resp. q provides accurate intervals $\boldsymbol{S}_p = \boldsymbol{\mathcal{OP}}(\boldsymbol{\mathcal{J}}_p) = [T'_p \pm \boldsymbol{\alpha}'_p]$ resp. $\boldsymbol{S}_q = \boldsymbol{\mathcal{OP}}(\boldsymbol{\mathcal{J}}_q) = [T'_q \pm \boldsymbol{\alpha}'_q]$ with reference point and accuracies being integer multiples of G_S. It has a worst case computational complexity of $\mathcal{O}(n \log n)$, is translation invariant and weakly monotonic. The characteristic functions $\Phi_\alpha(\cdot)$, $\boldsymbol{\Phi}_\pi(\cdot)$ and $\Psi_\pi(\cdot)$ are weakly monotonic w.r.t. any interval argument, are as follows:

(1) The conditional accuracy preservation functions $\Phi_\alpha^\pm(\cdot)$, *which ensure that the accuracies $\boldsymbol{\alpha}'_p$ in $\boldsymbol{S}_p$ satisfy $\boldsymbol{\alpha}'_p \subseteq [-\Phi_\alpha^-(\boldsymbol{\mathcal{B}}_p, \boldsymbol{\pi}^H, \boldsymbol{\pi}^o, \forall s : \iota_s^-), \Phi_\alpha^+(\boldsymbol{\mathcal{B}}_p, \boldsymbol{\pi}^H, \boldsymbol{\pi}^o, \forall s : \iota_s^+)]$, read*

$$\Phi_\alpha^-(\boldsymbol{\mathcal{B}}_p, \boldsymbol{\pi}^H, \boldsymbol{\pi}^o, \forall s : \iota_s^-) = \begin{cases} \beta_p^{p,-} + \Upsilon_p^- & \\ \beta_p^{x,-} + \pi^{H+} - \left\lceil \frac{\pi^{o+}}{\pi^o} \iota_x^- \right\rceil_{G_S} & \text{if } \Delta\beta_p^- \le 0, \end{cases} \tag{5.18}$$

$$\Phi_\alpha^+(\boldsymbol{\mathcal{B}}_p, \boldsymbol{\pi}^H, \boldsymbol{\pi}^o, \forall s : \iota_s^+) = \begin{cases} \beta_p^{p,+} + \Upsilon_p^+ & \\ \beta_p^{x,+} + \pi^{H-} - \left\lfloor \frac{\pi^{o-}}{\pi^o} \iota_x^+ \right\rfloor_{G_S} & \text{if } \Delta\beta_p^+ \le 0, \end{cases} \tag{5.19}$$

where

$$\Upsilon_p^- = \pi^{o+} + \left\lceil \frac{\pi^{o-}}{\pi^o} \min\{0, \Delta\beta_p^-\} \right\rceil_{G_S} - \left\lceil \frac{\pi^{o+}}{\pi^o} \iota_p^- \right\rceil_{G_S} \tag{5.20}$$

$$\Upsilon_p^+ = \pi^{o-} + \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{0, \Delta\beta_p^+\} \right\rceil_{G_S} - \left\lfloor \frac{\pi^{o-}}{\pi^o} \iota_p^+ \right\rfloor_{G_S} \tag{5.21}$$

are bounds on the (absolute value of the) maximum clock correction applied to node p's clock on the occurrence of a worst case accuracy setting w.r.t. α'^-_p *resp.* α'^+_p*, and*

$$\Delta\beta_p^- = \beta_p^{x,-} - \beta_p^{p,-} + \pi^{H+} - \pi^{o+} \tag{5.22}$$

$$\Delta\beta_p^+ = \beta_p^{x,+} - \beta_p^{p,+} + \pi^{H-} - \pi^{o-}, \tag{5.23}$$

where x *denotes the node with the* $n-2d-3e$*-largest accuracy bounds among* $\boldsymbol{\mathcal{B}}_p$*, i.e.,* $\beta_p^{x,-} = \max_{i:n-2d-3e}\{\beta_p^{i,-}\}$ *resp.* $\beta_p^{x,+} = \max_{i:n-2d-3e}\{\beta_p^{i,+}\}$ *with* $\max_{i:m} \mathcal{B}$ *denoting the m-th largest element of the set* $\mathcal{B} = \{b_i : 1 \leq i \leq n\}$.

(2) The precision preservation function $\boldsymbol{\Phi}_\pi(\cdot)$*, which ensures that* $\boldsymbol{S}_p$ *and* $\boldsymbol{S}_q$ *are* $\boldsymbol{\Phi}_\pi(\boldsymbol{\pi}^o)$*-correct, is*

$$\boldsymbol{\Phi}_\pi(\boldsymbol{\pi}^o) = \boldsymbol{\pi}^o, \tag{5.24}$$

(3) The precision enhancement function $\Psi_\pi(\cdot)$*, which ensures that the set* $\{\boldsymbol{S}_p, \boldsymbol{S}_q\}$ *is* $\boldsymbol{\pi}_0$*-precise with* $\pi_0 = \Psi_\pi(\boldsymbol{\pi}^H, \boldsymbol{\pi}^o, \boldsymbol{\pi}_I) \leq \pi^o < \pi^H$*, evaluates to*

$$\Psi_\pi(\boldsymbol{\pi}^H, \boldsymbol{\pi}^o, \boldsymbol{\pi}_I) = \begin{cases} \max\Big\{ \left\lceil \pi^{o+} + \frac{\pi^{o-}}{\pi^o}(\pi^H - \pi^o + \pi_I) \right\rceil_{G_S}, \\ \qquad \left\lceil \pi^{o-} + \frac{\pi^{o+}}{\pi^o}(\pi^H - \pi^o + \pi_I) \right\rceil_{G_S} \Big\} \\ \qquad\qquad \text{if } \pi^H + \pi_I \leq 2\pi^o, \\ \pi^o \qquad\qquad \text{otherwise.} \end{cases} \tag{5.25}$$

(4) The conditional intersection enhancement functions $\Psi_\iota^-(\cdot)$ *resp.* $\Psi_\iota^+(\cdot)$*, which ensure that the set* $\{\boldsymbol{S}_p, \boldsymbol{S}_q\}$ *is* $\boldsymbol{\pi}_0^{\overset{-}{\iota}_{pq}}$*-precise resp.* $\boldsymbol{\pi}_0^{\overset{+}{\iota}_{pq}}$*-precise with* $\pi_0^{\overset{-}{\iota}_{pq}} = \Psi_\iota^-(\boldsymbol{\mathcal{B}}_p, \boldsymbol{\pi}^H, \boldsymbol{\pi}^o, \boldsymbol{\pi}_I, \forall s : \iota_s^-)$ *resp.* $\pi_0^{\overset{+}{\iota}_{pq}} = \Psi_\iota^+(\boldsymbol{\mathcal{B}}_p, \boldsymbol{\pi}^H, \boldsymbol{\pi}^o, \boldsymbol{\pi}_I, \forall s : \iota_s^+)$ *for worst case settings w.r.t. negative resp. positive accuracy of* $\boldsymbol{S}_p$*, evaluate to*

$$\pi_0^{\overset{-}{\iota}_{pq}} = \begin{cases} \left\lceil \frac{\pi^{o-}}{\pi^o} \min\{\pi_I + \Delta\beta_p^-, \pi^o - \iota_p^-\} \right\rceil_{G_S} - \left\lfloor \frac{\pi^{o-}}{\pi^o} \min\{0, \Delta\beta_p^-\} \right\rfloor_{G_S} \\ \left\lceil \frac{\pi^{o-}}{\pi^o} \min\{\pi_I, \pi^o - \iota_x^-\} \right\rceil_{G_S} \qquad \text{if } \Delta\beta_p^- \leq 0, \end{cases} \tag{5.26}$$

$$\pi_0^{\iota_{pq}^+} = \begin{cases} \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{\pi_I + \Delta\beta_p^+, \pi^o - \iota_p^+\} \right\rceil_{G_S} - \left\lfloor \frac{\pi^{o+}}{\pi^o} \min\{0, \Delta\beta_p^+\} \right\rfloor_{G_S} \\ \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{\pi_I, \pi^o - \iota_x^+\} \right\rceil_{G_S} \qquad \textit{if } \Delta\beta_p^+ \leq 0. \end{cases} \tag{5.27}$$

which are in fact independent of q.

Proof. Requiring two evaluations of $\boldsymbol{\mathcal{M}}(\cdot)$, the worst case computational complexity of $\boldsymbol{\mathcal{OP}}(\cdot)$ for n nodes is $\mathcal{O}(n \log n)$, recall Definition 5.1. Since $\boldsymbol{\mathcal{M}}(\cdot)$ is translation invariant and monotonic by item (2) of Lemma 5.3, the same is true for $\boldsymbol{\mathcal{OP}}(\cdot)$, at least if no enlargement due to the inclusion of an excessive reference point according to (5.16) occurs. However, since we only have to establish weak monotonicity, where the reference points of all input intervals are invariant, this enlargement cannot cause any problem since it depends solely on the reference points.

We first approach the precision statements given in items (2) and (3). Using the notation introduced in $\boldsymbol{\mathcal{OP}}(\cdot)$'s Definition 5.5, we have $\tilde{\boldsymbol{S}}_p = \tilde{\boldsymbol{M}}_p \cap \check{\boldsymbol{I}}_p^p$ with $\tilde{\boldsymbol{M}}_p = \boldsymbol{\mathcal{M}}_{n_p}^{n-d-e}(\hat{\boldsymbol{\mathcal{J}}}_p)$, where $\check{\boldsymbol{I}}_p^p$ denotes the $\boldsymbol{\pi}^o$-precision interval associated with $\boldsymbol{I}_p^p$. Now, since $\boldsymbol{I}_p^p$ is $\boldsymbol{\pi}^o$-correct according to precondition [2], we obviously have $\tau \in \check{\boldsymbol{I}}_p^p$. In addition, any non-faulty $\boldsymbol{I}_p^i$ was assumed to be $\boldsymbol{\pi}^H$-correct, hence $\tau \in \hat{\boldsymbol{I}}_p^i$ and $||\hat{\boldsymbol{I}}_p^i|| \leq \pi^H$, which implies that any intersection of such intervals has these properties as well. Lemma 5.2 applies with $n := n_p$ and $f := d + e - (n - n_p)$, thus it follows that $\tau \in \tilde{\boldsymbol{M}}_p$ by its item (1) and $||\tilde{\boldsymbol{M}}_p|| \leq \pi^H$ by its item (2) since

$$n_p - 2f - a' \geq n - 2d - 3e + n - n_p \geq n - 2d - 3e \geq 1, \tag{5.28}$$

recall Assumption 5.1. Hence we conclude that $\tau \in \tilde{\boldsymbol{S}}_p$ as well, and since $\boldsymbol{\mathcal{OP}}(\cdot)$ sets the reference point T_p' of $\tilde{\boldsymbol{S}}_p = [T_p' \pm \boldsymbol{\pi}_p]$ to $\boldsymbol{\pi}^o$$-$center$_{G_S}(\tilde{\boldsymbol{S}}_p)$, applying Lemma 5.5 yields

$$\begin{aligned} \pi_p^- &= \left\lfloor \frac{\pi^{o-}}{\pi^o} ||\tilde{\boldsymbol{S}}_p|| \right\rfloor_{G_S} \leq \frac{\pi^{o-}}{\pi^o} ||\tilde{\boldsymbol{S}}_p|| \leq \pi^{o-} \\ \pi_p^+ &= \left\lceil \frac{\pi^{o+}}{\pi^o} ||\tilde{\boldsymbol{S}}_p|| \right\rceil_{G_S} \leq \left\lceil \frac{\pi^{o+}}{\pi^o} \pi^o \right\rceil_{G_S} = \pi^{o+} \end{aligned}$$

by recalling that π^{o+} was assumed to be an integer multiple of G_S. Since obviously $\mathrm{ref}(\boldsymbol{S}_p) = \mathrm{ref}(\tilde{\boldsymbol{S}}_p)$, the asserted $\boldsymbol{\pi}^o$-correctness of $\boldsymbol{S}_p$ and hence expression (5.24) for $\boldsymbol{\Phi}_\pi(\cdot)$ is evident.

The required weak monotonicity of $\boldsymbol{\Phi}_\pi(\boldsymbol{\pi}^o)$ w.r.t. $\boldsymbol{\pi}^o$ follows immediately from the monotonicity of the above expressions w.r.t. $||\tilde{\boldsymbol{S}}_p||$. Note carefully that the (apparently problematic) fractions $\frac{\pi^{o-}}{\pi^o}$ and $\frac{\pi^{o+}}{\pi^o}$ originate in the $\boldsymbol{\pi}^o$-center operation compiled statically into $\boldsymbol{\mathcal{OP}}$. Hence, they are fixed and thus irrelevant for weak monotonicity.

Of course, exactly the same reasoning holds for $\boldsymbol{S}_q$, completing the proof of item (2).

As far as item (3) is concerned, we first recall that any pair of $\boldsymbol{\pi}^H$-correct intervals $\boldsymbol{I}_p^i \in \boldsymbol{\mathcal{J}}_p$ and $\boldsymbol{I}_q^i \in \boldsymbol{\mathcal{J}}_q$ was assumed to be $\boldsymbol{\pi}_I$-precise in precondition [3]. Hence it follows that $||\hat{\boldsymbol{I}}_p^i \cup \hat{\boldsymbol{I}}_q^i|| \leq \pi^H + \pi_I$, since $||\hat{\boldsymbol{I}}_p^i||, ||\hat{\boldsymbol{I}}_q^i|| \leq \pi^H$ and $|\mathrm{ref}(\boldsymbol{I}_p^i) - \mathrm{ref}(\boldsymbol{I}_q^i)| \leq \pi_I$ by item (2) of Lemma 3.3. This implies $||\tilde{\boldsymbol{S}}_p \cup \tilde{\boldsymbol{S}}_q|| \leq ||\tilde{\boldsymbol{M}}_p \cup \tilde{\boldsymbol{M}}_q|| \leq \pi^H + \pi_I$ due to $\tilde{\boldsymbol{M}}_p \cup \tilde{\boldsymbol{M}}_q \subseteq \bigcap_{k=1}^{n-2d-3e}(\hat{\boldsymbol{I}}_p^{i_k} \cup \hat{\boldsymbol{I}}_q^{i_k})$ by virtue of item (2) of Lemma 5.4 in conjunction with (5.28). Now, if

$$\pi^H + \pi_I \leq 2\pi^o \tag{5.29}$$

holds, Lemma 5.6 with $\pi_p = \pi_q = \bar{\pi}_p = \bar{\pi}_q := \pi^o$ and $\pi = \pi^H + \pi_I$ applies and yields

$$\begin{aligned} |\mathrm{ref}(\boldsymbol{S}_p) - \mathrm{ref}(\boldsymbol{S}_q)| \leq \max\Bigg\{ & \left\lceil \frac{\pi^{o+}}{\pi^o}\pi^o + \frac{\pi^{o-}}{\pi^o}(\pi^H - \pi^o + \pi_I) \right\rceil_{G_S}, \\ & \left\lceil \frac{\pi^{o-}}{\pi^o}\pi^o + \frac{\pi^{o+}}{\pi^o}(\pi^H - \pi^o + \pi_I) \right\rceil_{G_S} \Bigg\}. \end{aligned} \tag{5.30}$$

If, on the other hand, (5.29) does not hold, then no precision enhancement takes place in the worst case. From $\boldsymbol{\pi}^o$-correctness established in item (2) it follows that $|\mathrm{ref}(\boldsymbol{S}_p) - \mathrm{ref}(\boldsymbol{S}_q)| = |\mathrm{ref}(\tilde{\boldsymbol{S}}_p) - \mathrm{ref}(\tilde{\boldsymbol{S}}_q)| \leq \pi^o$ here. This finally establishes the expression given for $\pi_0 = \Psi_\pi(\cdot)$ in item (3) of our lemma. Plugging (5.29) into (5.30) and using $\boldsymbol{\pi}^o \subset \boldsymbol{\pi}^H$ from precondition [2] confirms the asserted condition $\pi_0 \leq \pi^o < \pi^H$ as well.

It only remains to show that $\Psi_\pi(\boldsymbol{\pi}^H, \boldsymbol{\pi}^o, \boldsymbol{\pi}_I)$ is weakly monotonic w.r.t. $\boldsymbol{\pi}^H$, $\boldsymbol{\pi}^o$, and $\boldsymbol{\pi}_I$. The only non-trivial part of this proof is to show that (5.30) does not increase when $\boldsymbol{\pi}^o$ decreases. As in the proof of item (2), we use the fact that the (potentially problematic) fractions $\frac{\pi^{o-}}{\pi^o}$ and $\frac{\pi^{o+}}{\pi^o}$ are irrelevant for weak monotonicity, since they stem from the parameters $\pi_p = \pi_q := \pi^o$ that respresent the $\boldsymbol{\pi}^o$-center operation compiled statically into $\boldsymbol{\mathcal{OP}}(\cdot)$. Now, if (5.29) is still satisfied when π^o is decreased, it is not hard to see that the maximum value of the two terms on the r.h.s. of (5.30) is the first one if $\frac{\pi^{o+}}{\pi^o} \geq \frac{\pi^{o-}}{\pi^o}$ and the second one otherwise, invariant w.r.t. weak monotonicity. Therefore, the relevant term is monotonically decreasing when π^o is decreasing as required. If, on the other hand, (5.29) is no longer satisfied, $\Psi_\pi(\cdot) = \pi^o$ and weak monotonicity follows trivially. This eventually completes the proof of item (3).

Now we will turn our attention to item (1) of our lemma. From Definition 5.5, it is evident that the accuracies α'^{+}_p, α'^{-}_p as well as the reference point of $\boldsymbol{S}_p$ are integer multiples of G_S. Moreover, item (1) of Lemma 5.2 applied to (5.16) in conjunction with the fact that the own node was assumed to be non-faulty reveals that $\boldsymbol{S}_p$ is accurate.

Hence, it only[†] remains to bound $\boldsymbol{\alpha}'_p$. For this purpose, we need an arrangement of input intervals that maximizes, say, the positive accuracy α'^{+}_p subject to the given lower bounds $\forall s : \iota^+_s$ on the intersection of certain non-faulty input precision intervals.

A typical worst case scenario for α'^{+}_p is depicted in Figure 5.4. It will be used subsequently for establishing bounds on the quantities of primay interest, namely, the worst case accuracy $\beta' = \Phi^+_\alpha(\cdot)$, the maximum clock correction Υ^+, and the resulting precision $\zeta' = \Psi^+_\iota(\cdot)$. First of all, it reveals the crucial role of the common intersection lengths $\forall s : \iota^+_s$ defined in precondition [4] of our lemma: With the dotted vertical line R marking the ridge edge of the utmost left non-faulty input precision interval $\hat{\boldsymbol{I}}^{\min}_{p/q}$ at either node p or q, parameter ι^+_s gives the lower bound on how far left of it the left edge of both $\hat{\boldsymbol{I}}^{s}_p$ and $\hat{\boldsymbol{I}}^{s}_q$ may lie; for comparison see also Figure 3.5. Hence, boundary R in conjunction with $\forall s : \iota^+_s$ effectively allows us to relate the many different intervals involved in the worst case scenario. It will turn out, the depending on the value of ι^+ and ι^+_x, we have to distinguish two cases:

(I) $\iota^+ \leq \iota^+_x$, which is the case shown in Figure 5.4.

(II) $\iota^+ \geq \iota^+_x$, which is the situation faced by the orthogonal accuracy convergence function analyzed in [52].

Clearly, our analysis has to provide results that are valid in either case.

We will provide the detailed argument for α'^{+}_p only; α'^{-}_p is derived analogously. First, we define the *mixed interval* $\vec{\boldsymbol{I}}$ of an arbitrary accuracy interval $\boldsymbol{I} = [T \pm \boldsymbol{\alpha}]$ with its associated $\boldsymbol{\pi}$-precision interval $\hat{\boldsymbol{I}} = [T \pm \boldsymbol{\pi}]$ as $\vec{\boldsymbol{I}} = [T - \pi^-, T + \alpha^+]$. Mixed intervals are in fact ideally suited for attacking our problem: Since left and right edge of the result of $\boldsymbol{\mathcal{M}}(\cdot)$, and hence $\boldsymbol{\mathcal{OP}}(\cdot)$, are computed independently of each other, and right$(\vec{\boldsymbol{I}}^{s}_p)$ = right$(\boldsymbol{I}^s_p)$ resp. left$(\vec{\boldsymbol{I}}^{s}_p)$ = left$(\hat{\boldsymbol{I}}^{s}_p)$, it follows that right$(\vec{\boldsymbol{S}}_p)$ = right$(\boldsymbol{S}_p)$ resp. left$(\vec{\boldsymbol{S}}_p)$ = left$(\hat{\boldsymbol{S}}_p)$ as well. Analyzing the result of $\boldsymbol{\mathcal{M}}(\cdot)$ in terms of mixed intervals, however, is easy since the hybrid fault model in Assumption 5.1 guarantees that the sets of mixed input intervals $\vec{\boldsymbol{\mathcal{I}}}^{s}_p$, $\vec{\boldsymbol{\mathcal{I}}}^{s}_q$ are in accordance with the fault model of Definition 5.3, which underlies the properties of Marzullo's function $\boldsymbol{\mathcal{M}}(\cdot)$.

In order to determine the worst case accuracy bound β', we need those settings of the edges of $\vec{\boldsymbol{S}}_p$ and $\tilde{\boldsymbol{S}}_p$ that maximize $\boldsymbol{S}_p$'s positive accuracy. Analogous to the proof of the precision enhancement function $\Psi_\pi(\cdot)$, we know from item (2) of Lemma 5.4 in conjunction with (5.28) that there are at least $n - 2d - 3e$ pairs of non-faulty intervals $\boldsymbol{I}^{u_k}_p$, $\boldsymbol{I}^{u_k}_q$ present in $\boldsymbol{\mathcal{J}}_p \times \boldsymbol{\mathcal{J}}_q$ such that $\vec{\boldsymbol{S}}_p \cup \vec{\boldsymbol{S}}_q \subseteq \vec{\boldsymbol{M}}_p \cup \vec{\boldsymbol{M}}_q \subseteq \bigcap_{k=1}^{n-2d-3e} (\vec{\boldsymbol{I}}^{u_k}_p \cup \vec{\boldsymbol{I}}^{u_k}_q)$. The

---

[†] Without loss of generality, since the analogous result for $\boldsymbol{S}_q$ is obtained by exchanging p and q.

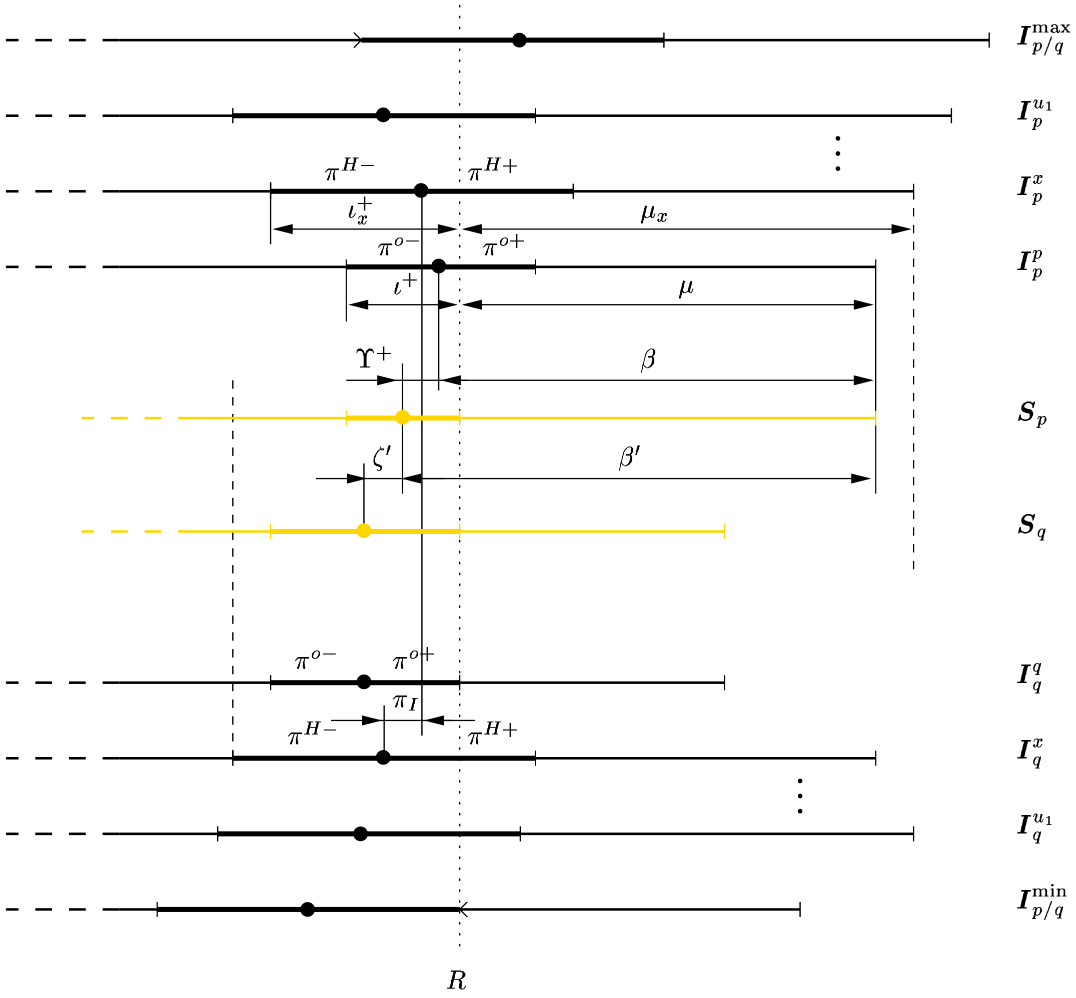


Figure 5.4: *A Worst Case Scenario for Positive Accuracy*

reference points (and hence the left edges of the mixed intervals) of any non-faulty pair $\boldsymbol{I}_p^s$, $\boldsymbol{I}_q^s$ can be at most π_I apart since they are $\boldsymbol{\pi}_I$-precise according to precondition [3]. Let $x = u_{n-2d-3e}$ be the node that produces the $n-2d-3e$-largest positive accuracy $\beta_x = \beta_p^x = \max_{i:n-2d-3e}\{\beta_p^{u_i}\}$ among all $\boldsymbol{I}_p^{u_i}$. Evidently, maximizing their intersection's length requires $\vec{\boldsymbol{I}}_p^x$ to be entirely contained in all the larger $\vec{\boldsymbol{I}}_p^{u_i}$, which eventually leads to the left resp. right edge of $\bigcap_{k=1}^{n-2d-3e}(\vec{\boldsymbol{I}}_p^{u_k} \cup \vec{\boldsymbol{I}}_q^{u_k})$ as marked with a vertical dashed line in Figure 5.4, and eventually to

$$\operatorname{left}(\vec{\boldsymbol{S}}_p) \geq \max\Big\{\operatorname{left}(\vec{\boldsymbol{I}}_p^p), \operatorname{left}(\vec{\boldsymbol{I}}_p^x)\Big\} \tag{5.31}$$

$$\operatorname{right}(\vec{\boldsymbol{S}}_p) \leq \min\Big\{\operatorname{right}(\vec{\boldsymbol{I}}_p^p), \operatorname{right}(\vec{\boldsymbol{I}}_p^x)\Big\}. \tag{5.32}$$

From

$$\mathrm{ref}(\boldsymbol{I}_p^p) - R = \pi^{o-} - \iota^+ \tag{5.33}$$

together with the fact that $\vec{\boldsymbol{S}}_p$ cannot be larger than $\beta + \pi^{o-}$, since the precision interval $\hat{\boldsymbol{S}}_p$ resp. the accuracy interval $\boldsymbol{S}_p$ has been intersected with $\check{\boldsymbol{I}}_p^p$ resp. $\boldsymbol{I}_p^p$, it hence follows that the worst case of $\vec{\boldsymbol{S}}_p$ w.r.t. β' is characterized by

$$\begin{align}
\mathrm{right}(\vec{\boldsymbol{S}}_p) - \mathrm{ref}(\boldsymbol{I}_p^p) &= \mathrm{right}(\vec{\boldsymbol{S}}_p) - R - (\mathrm{ref}(\boldsymbol{I}_p^p) - R) \tag{5.34}\\
&\le \min\{\mu, \mu_x\} - (\pi^{o-} - \iota^+) \notag\\
&= \min\{\beta, \beta_x + \pi^{H-} - \pi^{o-} - (\iota_x^+ - \iota^+)\}, \tag{5.35}
\end{align}$$

see Figure 5.4 for the definition of μ, μ_x.

As far as the worst case position of $\mathrm{right}(\tilde{\boldsymbol{S}}_p)$, we know already that no non-faulty precision interval $\hat{\boldsymbol{I}}_p^s$, $\hat{\boldsymbol{I}}_q^s$ (including $\check{\boldsymbol{I}}_p^p$ and $\check{\boldsymbol{I}}_q^q$) can have a right edge left of R, recall our earlier comments on precondition [4]. From the distributed minimal intersection property of $\boldsymbol{\mathcal{M}}(\cdot)$ stated in item (1) of Lemma 5.4, it thus follows that both $\tilde{\boldsymbol{M}}_p$ and $\tilde{\boldsymbol{M}}_q$ must have this property as well. Consequently, neither $\mathrm{right}(\tilde{\boldsymbol{S}}_p)$ nor $\mathrm{right}(\tilde{\boldsymbol{S}}_q)$ can be left of R, see Figure 5.4. This implies that we have to consider $\mathrm{right}(\tilde{\boldsymbol{S}}_p) = R$ for worst case settings w.r.t. β' only, since the monotonicity property (5.11) of $\boldsymbol{\pi}-\mathrm{center}_{G_S}$ ensures that setting $\mathrm{right}(\tilde{\boldsymbol{S}}_p)$ further right would provide a smaller β' only. Using (5.33) and $R - \mathrm{left}(\vec{\boldsymbol{S}}_p) \le \min\{\iota^+, \iota_x^+\}$ by (5.31) in conjunction with (5.13) easily yields

$$\begin{align}
\Upsilon^+ &= \mathrm{ref}(\boldsymbol{I}_p^p) - \mathrm{ref}(\boldsymbol{S}_p) = \mathrm{ref}(\boldsymbol{I}_p^p) - R + (R - \mathrm{ref}(\boldsymbol{S}_p)) \tag{5.36}\\
&\le \pi^{o-} - \iota^+ + \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{\iota^+, \iota_x^+\} \right\rceil_{G_S} \tag{5.37}\\
&= \pi^{o-} + \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{0, \iota_x^+ - \iota^+\} - \frac{\pi^{o-}}{\pi^o}\iota^+ \right\rceil_{G_S} \notag\\
&\le \pi^{o-} + \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{0, \iota_x^+ - \iota^+\} \right\rceil_{G_S} - \left\lfloor \frac{\pi^{o-}}{\pi^o}\iota^+ \right\rfloor_{G_S}. \tag{5.38}
\end{align}$$

For this derivation, we exploited the fact that ι^+ is an integer multiple of G_S and used the well-known relations $\lceil x + y \rceil \le \lceil x \rceil + \lceil y \rceil$ and $-\lceil x \rceil = \lfloor -x \rfloor$, whose justifications can be found in Chapter 3 of [14]. Adding (5.34) and (5.36), we eventually obtain

$$\begin{align}
\beta' &= \mathrm{right}(\vec{\boldsymbol{S}}_p) - \mathrm{ref}(\boldsymbol{S}_p) \notag\\
&\le \min\{\beta, \beta_x + \pi^{H-} - \pi^{o-} - (\iota_x^+ - \iota^+)\} + \Upsilon^+ \tag{5.39}\\
&\le \min\{\beta + \pi^{o-}, \beta_x + \pi^{H-} - (\iota_x^+ - \iota^+)\} \notag\\
&\quad + \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{0, \iota_x^+ - \iota^+\} \right\rceil_{G_S} - \left\lfloor \frac{\pi^{o-}}{\pi^o}\iota^+ \right\rfloor_{G_S}. \tag{5.40}
\end{align}$$

Now we turn our attention to a bound on the precision ζ' of the set $\{\boldsymbol{S}_p, \boldsymbol{S}_q\}$ in worst case settings for β'. As for $\mathrm{right}(\tilde{\boldsymbol{S}}_p)$ before, we only have to consider $\mathrm{right}(\tilde{\boldsymbol{S}}_q) = R$ here, since a right edge located further right can only provide a smaller ζ' by virtue of monotonicity (5.11). Due to the containment property established earlier and the obvious $\boldsymbol{\pi}^o$-correctness of $\boldsymbol{S}_q$, we hence get

$$R - \mathrm{left}(\vec{\boldsymbol{S}}_q) \leq \min\{\iota_x^+ + \pi_I, \pi^o\}. \tag{5.41}$$

Using (5.33), (5.36) and the above relation in conjunction with (5.13) reveals that

$$\begin{aligned} \zeta' &= \mathrm{ref}(\boldsymbol{S}_p) - \mathrm{ref}(\boldsymbol{S}_q) = \mathrm{ref}(\boldsymbol{I}_p^p) - R + (R - \mathrm{ref}(\boldsymbol{S}_q)) - \Upsilon^+ \\ &\leq \pi^{o-} - \iota^+ + \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{\iota_x^+ + \pi_I, \pi^o\} \right\rceil_{G_S} - \Upsilon^+. \end{aligned} \tag{5.42}$$

However, a uniformly valid upper bound would require a lower bound on Υ^+, which is not available. Nevertheless, it is apparent from (5.39) that the maximum of β' occurs for maximal Υ^+ only. Hence, for our purpose, we can safely insert the upper bound (5.37) on Υ^+ into (5.42), arriving

$$\begin{aligned} \zeta' &\leq \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{\iota_x^+ + \pi_I, \pi^o\} \right\rceil_{G_S} - \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{\iota^+, \iota_x^+\} \right\rceil_{G_S} && (5.43) \\ &\leq \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{\iota_x^+ + \pi_I, \pi^o\} - \frac{\pi^{o+}}{\pi^o} \min\{\iota^+, \iota_x^+\} \right\rceil_{G_S} && (5.44) \\ &\leq \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{\iota_x^+ - \iota^+ + \pi_I, \pi^o - \iota^+\} \right\rceil_{G_S} - \left\lfloor \frac{\pi^{o+}}{\pi^o} \min\{0, \iota_x^+ - \iota^+\} \right\rfloor_{G_S} && (5.45) \end{aligned}$$

by the same devices as used for the derivation of (5.38). Note that our bound (5.45) on ζ' does not depend on q, i.e., it is uniformly valid.

To complete our worst case analysis, we have to show that the worst case scenario for β' constructed above is also valid w.r.t. *multiple rounds*. It is of course valid for a single round, in the sense that there is no scenario that provides a worse β' for the given ι^+. However, the resulting ζ' is quite small, and since ζ' will determine ι^+ —and hence β'— in the next round, the question arises whether a non-worst case setting could provide a worse overall accuracy. Now, it follows by construction that $\beta' + \zeta'$ is also maximal in the worst case scenario considered above, see Figure 5.4. In fact, adding (5.39) and (5.42) and employing the same derivation that led to (5.38) yields an upper bound

$$
\begin{aligned}
\beta' + \zeta' &= \mathrm{right}(\vec{\boldsymbol{S}}_p) - \mathrm{ref}(\boldsymbol{S}_q) \\
&\leq \min\{\beta, \beta_x + \pi^{H-} - \pi^{o-} - (\iota_x^+ - \iota^+)\} \\
&\quad + \pi^{o-} - \iota^+ + \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{\iota_x^+ + \pi_I, \pi^o\} \right\rceil_{G_S} && (5.46) \\
&\leq \min\{\beta + \pi^{o-}, \beta_x + \pi^{H-} - (\iota_x^+ - \iota^+)\} \\
&\quad + \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{\iota_x^+ - \iota^+ + \pi_I, \pi^o - \iota^+\} \right\rceil_{G_S} - \left\lfloor \frac{\pi^{o-}}{\pi^o} \iota^+ \right\rfloor_{G_S}, && (5.47)
\end{aligned}
$$

which holds for any scenario (including non-worst case ones, as caused by a smaller Υ^+). It follows that the occurrence of some ζ' exceeding our bound (5.43) can only occur in scenarios where β' falls below (5.40) at least by the same amount. However, (5.40) in conjunction with $\pi^{o-}/\pi^o \leq 1$ demonstrates that the resulting decrease in ι^+ for the next round cannot catch up the decrease β' experienced in the current round. Thus, our single round worst case setting also implies a multiple round one.

Nevertheless, the worst case bounds on Υ^+, β', and ζ' established above depend upon something that is actually not available: Both the definition of the conditional accuracy preservation functions $\Phi_\alpha^+(\cdot)$ as well as the conditional intersection enhancement functions $\Psi_\iota^+(\cdot)$ relies upon lower bounds $\tilde{\iota}^+ \leq \iota^+$ and $\tilde{\iota}_x^+ \leq \iota_x^+$ only. Hence, in order to derive the required expressions (5.19) and (5.27), we have to modify our results to utilize the lower bounds $\tilde{\iota}^+$ resp. $\tilde{\iota}_x^+$ instead of the actual values ι^+ resp. ι_x^+.

First, looking at our expressions (5.38), (5.40), (5.45), and (5.47), it is apparent that they actually depend upon ι^+ and $\Delta\iota^+ = \iota_x^+ - \iota^+$. Referring to ι^+, they are all monotonically decreasing, which means that we can safely replace ι^+ by its lower bound $\tilde{\iota}^+$. Dealing with $\Delta\iota^+$, however, is more complicated since (the sign of) $\Delta\tilde{\iota}^+ = \tilde{\iota}_x^+ - \tilde{\iota}^+$ is not representative for (the sign of) $\Delta\iota^+$. For our argument, we exploit the fact that the maximum of (5.47) occurs for $\Delta\iota^+ = \Delta\beta = \beta_x - \beta + \pi^{H-} - \pi^{o-}$. For, if $\Delta\beta \geq 0$, it is apparent that $\beta + \pi^{o-}$ determines the first min-term of (5.47) provided that $\Delta\iota^+ \leq \Delta\beta$, whereas the whole expression is monotonically decreasing for $\Delta\iota^+ > \Delta\beta$. If, on the other hand, $\Delta\beta \leq 0$, then $\beta_x + \pi^{H-} - (\iota_x^+ - \iota^+)$ determines this min-term provided that $\Delta\iota^+ \geq \Delta\beta$; clearly, this term is monotonically increasing in $|\Delta\iota^+|$ here. Hence, it follows that the maximum of (5.47) occurs when $\Delta\iota^+ = \Delta\beta$ in any case. Plugging $\Delta\iota^+ = \Delta\beta$ in (5.40), (5.38), and (5.45) immediately leads to the bounds (5.19), (5.21), and (5.27), respectively, as stated in items (1) and (4) of our lemma.

Actually, the above bounds can be improved in case of $\Delta\beta \le 0$. Stepping back to (5.39) resp. (5.46) and pulling $-(\iota_x^+ - \iota^+)$ out of the first min-term easily yields

$$\begin{aligned}
\beta' &\le \min\{\beta + \iota_x^+ - \iota^+, \beta_x + \pi^{H-} - \pi^{o-}\} + \iota^+ - \iota_x^+ + \Upsilon^+ && (5.48)\\
&\le \min\{\beta + \pi^{o-} + \iota_x^+ - \iota^+, \beta_x + \pi^{H-}\}\\
&\quad + \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{\iota^+ - \iota_x^+, 0\} \right\rceil_{G_S} - \left\lfloor \frac{\pi^{o-}}{\pi^o} \iota_x^+ \right\rfloor_{G_S} && (5.49)
\end{aligned}$$

resp.

$$\begin{aligned}
\beta' + \zeta' &\le \min\{\beta + \iota_x^+ - \iota^+, \beta_x + \pi^{H-} - \pi^{o-}\}\\
&\quad + \pi^{o-} - \iota_x^+ + \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{\iota_x^+ + \pi_I, \pi^o\} \right\rceil_{G_S}\\
&\le \min\{\beta + \pi^{o-} + \iota_x^+ - \iota^+, \beta_x + \pi^{H-}\}\\
&\quad + \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{\pi_I, \pi^o - \iota_x^+\} \right\rceil_{G_S} - \left\lfloor \frac{\pi^{o-}}{\pi^o} \iota_x^+ \right\rfloor_{G_S} && (5.50)
\end{aligned}$$

by analogous manipulations as used for deriving (5.38). Similarly, by pulling out ι_x^+ in (5.44) it easily follows that

$$\zeta' \le \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{\pi_I, \pi^o - \iota_x^+\} \right\rceil_{G_S} - \left\lfloor \frac{\pi^{o+}}{\pi^o} \min\{\iota^+ - \iota_x^+, 0\} \right\rfloor_{G_S}. \qquad (5.51)$$

This time, our expressions depend upon ι_x^+ and $\Delta\iota^+ = \iota_x^+ - \iota^+$. Again, replacing ι_x^+ by its lower bound $\tilde{\iota}_x^+$ poses no problem since (5.49), (5.50), and (5.51) are all monotonically decreasing in ι_x^+. From (5.49) it is apparent that the maximum value of β' occurs when $\Delta\iota^+ = 0$, since the first min-term is determinated by $\beta_x + \pi^{H-}$ due to $\Delta\beta \le 0$. Note that (5.38) and (5.50) also attain their maximum value for $\Delta\iota^+ = 0$. Plugging $\Delta\iota^+ = 0$ in our expressions confirms statements (5.19), (5.21), and (5.27) for $\Delta\beta_p^+ \le 0$ as well.

We still have to show that the expressions established before are weakly monotonic w.r.t. any interval argument. This is obvious for $\Phi_\alpha^+(\cdot)$ and Υ^+ given by (5.19) and (5.21), respectively, but in fact not true for $\Psi_\iota^+(\cdot)$ given by (5.27): Increasing π^{o-} and/or $\beta_p^{p,+}$ decreases $\Delta\beta_p^+$. By the same reasoning as used for carrying over the single-round worst case to a multiple round one, however, it can be justified that weak monotonicity is only required for $\Phi_\alpha^+(\cdot) + \Psi_\iota^+(\cdot)$, and this is of course guaranteed by virtue of (5.47) and (5.50).

It only remains to provide the analogous expressions for $\Phi_\alpha^-(\cdot)$ resp. $\Psi_\iota^-(\cdot)$, which can be done by considering Figure 5.4 mirrored at the vertical line R while exchanging $\boldsymbol{I}_{p/q}^{\min} \leftrightarrow \boldsymbol{I}_{p/q}^{\max}$, $\pi^{H+} \leftrightarrow \pi^{H-}$, $\pi^{o+} \leftrightarrow \pi^{o-}$ and further $\iota^+ \leftrightarrow \iota^-$, $\iota_x^+ \leftrightarrow \iota_x^-$. According to (5.12), we also have to replace $\lceil\cdot\rceil$ by $\lfloor\cdot\rfloor$ in our starting equations (5.37) and (5.42).

Rewriting the derivation of (5.38) accordingly results in

$$\begin{aligned}
\Upsilon^- &\le \pi^{o+} - \iota^- + \left\lfloor \frac{\pi^{o-}}{\pi^o} \min\{\iota^-, \iota_x^-\} \right\rfloor_{G_S} \\
&= \pi^{o+} + \left\lfloor \frac{\pi^{o-}}{\pi^o} \min\{0, \iota_x^- - \iota^-\} - \frac{\pi^{o+}}{\pi^o} \iota^- \right\rfloor_{G_S} \\
&\le \pi^{o+} + \left\lceil \frac{\pi^{o-}}{\pi^o} \min\{0, \iota_x^- - \iota^-\} \right\rceil_{G_S} - \left\lceil \frac{\pi^{o+}}{\pi^o} \iota^- \right\rceil_{G_S},
\end{aligned}$$

where we used the relation $\lfloor x \rfloor - \lfloor y \rfloor \le \lceil x - y \rceil$. The latter follows from combining $\lfloor x \rfloor + \lfloor y \rfloor \le \lfloor x + y \rfloor$ and $-\lfloor y - x \rfloor = \lceil x - y \rceil$, see again [14]. An analogous reasoning as carried out for the positive accuracy part leads to the expressions (5.18) and (5.20) stated in item (1) of our lemma.

Similarly, rewriting (5.42) accordingly and using the relation introduced above, it follows that

$$\begin{aligned}
\zeta' &\le \left\lfloor \frac{\pi^{o-}}{\pi^o} \min\{\iota_x^- + \pi_I, \pi^o\} \right\rfloor_{G_S} - \left\lfloor \frac{\pi^{o-}}{\pi^o} \min\{\iota^-, \iota_x^-\} \right\rfloor_{G_S} \\
&\le \left\lceil \frac{\pi^{o-}}{\pi^o} \min\{\iota_x^- + \pi_I, \pi^o\} - \frac{\pi^{o-}}{\pi^o} \min\{\iota^-, \iota_x^-\} \right\rceil_{G_S} \\
&\le \left\lceil \frac{\pi^{o-}}{\pi^o} \min\{\iota_x^- - \iota^- + \pi_I, \pi^o - \iota^-\} \right\rceil_{G_S} - \left\lfloor \frac{\pi^{o-}}{\pi^o} \min\{0, \iota_x^- - \iota^-\} \right\rfloor_{G_S}.
\end{aligned}$$

From here, an analogous derivation as conducted for the positive accuracy part leads to the expression (5.26), finally completing the proof. □

The following lemma shows that all results of Lemma 5.7 can at least be improved by using certain finite settings of $\Upsilon_{\max}$.

Lemma 5.8 (Optimal Precision Function for $\Upsilon_{\max} < \infty$) *With the notations and preconditions of Lemma 5.7, and if $\Upsilon_{\max}$ is chosen to satisfy*

$$\Upsilon_{\max} \ge \pi^o - \pi_0, \tag{5.52}$$

all results of Lemma 5.7 remain valid. The conditional accuracy preservation functions $\Phi_\alpha^\pm(\cdot)$ improve to

$$\Phi_\alpha^-(\boldsymbol{\mathcal{B}}_p, \boldsymbol{\pi}^H, \boldsymbol{\pi}^o, \forall s : \iota_s^-) = \beta_p^{p,-} + \min\Big\{\Upsilon_p^-, \Upsilon_{\max}\Big\} \tag{5.53}$$

$$\Phi_\alpha^+(\boldsymbol{\mathcal{B}}_p, \boldsymbol{\pi}^H, \boldsymbol{\pi}^o, \forall s : \iota_s^+) = \beta_p^{p,+} + \min\Big\{\Upsilon_p^+, \Upsilon_{\max}\Big\} \tag{5.54}$$

with (unchanged) maximum clock correction bounds

$$\Upsilon_p^- = \left\lceil \frac{\pi^{o+}}{\pi^o}(\pi^o - \iota_p^-) \right\rceil_{G_S} + \left\lceil \frac{\pi^{o-}}{\pi^o} \min\{0, \Delta\beta_p^-\} \right\rceil_{G_S} \tag{5.55}$$

$$\Upsilon_p^+ = \left\lfloor \frac{\pi^{o-}}{\pi^o}(\pi^o - \iota_p^+) \right\rfloor_{G_S} + \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{0, \Delta\beta_p^+\} \right\rceil_{G_S}, \tag{5.56}$$

and the conditional intersection enhancement functions $\Psi_\iota^\pm(\cdot)$ *improve to*

$$\Psi_\iota^-(\boldsymbol{\mathcal{B}}_p, \boldsymbol{\pi}^H, \boldsymbol{\pi}^o, \boldsymbol{\pi}_I, \forall s : \iota_s^-) = \left\lceil \frac{\pi^{o-}}{\pi^o} \min\{\pi_I + \Delta\beta_p^-, \pi^o - \iota_p^-\} \right\rceil_{G_S} - \left\lfloor \frac{\pi^{o-}}{\pi^o} \min\{0, \Delta\beta_p^-\} \right\rfloor_{G_S} - \min\Big\{0, \Upsilon_{\max} - \Upsilon_p^-\Big\} \tag{5.57}$$

$$\Psi_\iota^+(\boldsymbol{\mathcal{B}}_p, \boldsymbol{\pi}^H, \boldsymbol{\pi}^o, \boldsymbol{\pi}_I, \forall s : \iota_s^+) = \left\lceil \frac{\pi^{o+}}{\pi^o} \min\{\pi_I + \Delta\beta_p^+, \pi^o - \iota_p^+\} \right\rceil_{G_S} - \left\lfloor \frac{\pi^{o+}}{\pi^o} \min\{0, \Delta\beta_p^+\} \right\rfloor_{G_S} - \min\Big\{0, \Upsilon_{\max} - \Upsilon_p^+\Big\}. \tag{5.58}$$

Proof. From $\boldsymbol{\mathcal{OP}}(\cdot)$'s Definition 5.5, we recall that limiting $\Upsilon_{\max}$ to finite values means limiting the correction $|\Upsilon_p| = |T'_p - T_p|$ applied to the clock at any node p; herein, $T_p = \mathrm{ref}(\boldsymbol{I}_p^p)$ resp. $T'_p = \mathrm{ref}(\boldsymbol{S}_p)$ denote the reference point of old resp. new clock state at the end of some round k at node p in case of $\Upsilon_{\max} = \infty$.

Let $\tau = \tau^{(k)}$ resp. $\tau' = \tau^{(k+1)}$ be the instances of internal global time for round k resp. $k+1$, and $\boldsymbol{S}_p^\Upsilon$ with $T_p^\Upsilon = \mathrm{ref}(\boldsymbol{S}_p^\Upsilon)$ be the accuracy interval computed by $\boldsymbol{\mathcal{OP}}(\cdot)$ in case of finite $\Upsilon_{\max}$. First of all, it is obvious that the precision preservation function (5.24) —stating that $\boldsymbol{S}_p$ is $\boldsymbol{\pi}^o$-correct w.r.t. τ— remains unaltered by limiting $\Upsilon_{\max}$, since $\boldsymbol{I}_p^p$ is $\boldsymbol{\pi}^o$-correct by precondition [2] of Lemma 5.7, so $\tilde{\boldsymbol{S}}_p \subseteq \check{\boldsymbol{I}}_p^p$ and $\tau \in \tilde{\boldsymbol{S}}_p$. Putting this together, it is clear that $\tau \in [T_p^\Upsilon \pm \boldsymbol{\pi}^o]$ for any $T_p^\Upsilon \in [T_p, T'_p]$, which confirms item (2) of the foregoing lemma.

Turning our attention to the precision enhancement function $\Psi_\pi(\cdot)$ given by (5.25), we show that it is feasible to limit the maximum correction to $\Upsilon_{\max} = \pi^o - \pi_0$ without imparing the $\boldsymbol{\pi}_0$-correctness of the resulting $\boldsymbol{S}_p^\Upsilon$ w.r.t. τ'. First of all, it is easy to see that (in case of $\Upsilon_{\max} = \infty$) the new internal global time τ' may be chosen such that it is not too far away from τ. More specifically, since $\boldsymbol{S}_p$ and $\boldsymbol{S}_q$ are both $\boldsymbol{\pi}^o$-correct w.r.t. τ and $\boldsymbol{\pi}_0$-correct w.r.t. τ', a choice of τ' such that

$$\pi^{o+} - \pi_0^+ \geq -(\tau' - \tau) \geq -(\pi^{o-} - \pi_0^-) \tag{5.59}$$

is legitimate. If this is not feasible, one of the intervals $\boldsymbol{S}_p$, $\boldsymbol{S}_q$, say $\boldsymbol{S}_q$ w.l.o.g., is responsible that the (new) associated $\boldsymbol{\pi}_0$-precision interval $\check{\boldsymbol{S}}_p$ satisfies $\tau' = \mathrm{left}(\check{\boldsymbol{S}}_q) > \mathrm{left}(\check{\boldsymbol{S}}_p)$

resp. $\tau' = \mathrm{right}(\check{\boldsymbol{S}}_q) < \mathrm{right}(\check{\boldsymbol{S}}_p)$, since otherwise we could choose a smaller resp. larger τ', but then $\mathrm{ref}(\check{\boldsymbol{S}}_q) - \tau = \tau' + \pi_0^- - \tau > \pi^{o-} - \pi_0^- + \pi_0^- = \pi^{o-}$ resp. $\tau - \mathrm{ref}(\check{\boldsymbol{S}}_q) = \tau - (\tau' - \pi_0^+) > \pi^{o+} - \pi_0^+ + \pi_0^+ = \pi^{o+}$. This, however, contradicts the $\boldsymbol{\pi}^o$-correctness of $\boldsymbol{S}_q$ w.r.t. τ.

When $|T_p' - T_p|$ is pressed down by $\Upsilon_{\max}$, we need to consider two cases. Assuming that we encounter $T_p' - T_p > \Upsilon_{\max}$ but limiting it to $T_p^\Upsilon = T_p + \Upsilon_{\max}$ enforces $\boldsymbol{S}_p^\Upsilon$ to be not $\boldsymbol{\pi}_0$-correct. However, this can only happen if

$$\tau' > T_p + \Upsilon_{\max} + \pi_0^+ \geq T_p + \pi^o - \pi_0^- \tag{5.60}$$

and hence

$$\tau = \tau' - (\tau' - \tau) > T_p + \pi^o - \pi_0^- - (\pi^{o-} - \pi_0^-) = T_p + \pi^{o+}$$

by (5.60) and (5.59), which contradicts the $\boldsymbol{\pi}^o$-correctness of $\boldsymbol{I}_p^p$.

By the same token, if $T_p' - T_p < -\Upsilon_{\max}$ but limiting it to $T_p^\Upsilon = T_p - \Upsilon_{\max}$ leads to $\boldsymbol{R}_p^\Upsilon$ being not $\boldsymbol{\pi}_0$-correct, we must have

$$\tau' < T_p - \Upsilon_{\max} - \pi_0^- \leq T_p - \pi^o + \pi_0^+ \tag{5.61}$$

and hence

$$\tau = \tau' - (\tau' - \tau) < T_p - \pi^o + \pi_0^+ + \pi^{o+} - \pi_0^+ = T_p - \pi^{o-}$$

by (5.61) and (5.59), which again contradicts the $\boldsymbol{\pi}^o$-correctness of $\boldsymbol{I}_p^p$. This eventually confirms item (3) of the foregoing lemma.

It only remains to establish the modified values of the conditional accuracy preservation functions $\Phi_\alpha^\pm(\cdot)$ and intersection enhancement functions $\Psi_\iota^\pm(\cdot)$ as given in our Lemma. For that purpose, we have to consider the effect of limiting Υ to $\Upsilon_{\max}$ in the proof of Lemma 5.7. However, it is obvious that replacing Υ in (5.39) by $\min\{\Upsilon, \Upsilon_{\max}\}$ simply translates to adding the difference $\min\{0, \Upsilon_{\max} - \Upsilon\}$ to the accuracy preservation functions (5.18) resp. (5.19), which immediately leads to (5.53) resp. (5.54). Our expressions (5.55) resp. (5.56) are just a slight modification of (5.20) resp. (5.21) based on the fact that both π^{o-}, π^{o+} are assumed to be integer multiples of G_S. By the same token, in view of (5.42), it is clear that subtracting $\min\{0, \Upsilon_{\max} - \Upsilon\}$ from the intersection enhancement functions (5.26) resp. (5.27) provides the corresponding expressions (5.57) resp. (5.58). This finally completes the proof of our lemma. □

In the following we make some remarks about the accomplishments of the above two lemmas:

Observe that $\Psi_\pi(\boldsymbol{\pi}^H, \boldsymbol{\pi}^o, \boldsymbol{\pi}_I)$ given by (5.25) is minimized when $\boldsymbol{\pi}^o$ is symmetric, i.e., when $\pi^{o+} = \pi^{o-}$. In that case, we obtain

$$\pi_0 = \left\lceil \frac{\pi^H + \pi_I}{2} \right\rceil_{G_S} \leq \frac{\pi^H + \pi_I}{2} + \frac{G_S}{2}.$$

This respresents the maximum *precision enhancement* of our convergence function, see [58]. The convergence factor is 1/2, which is the same as provided by the well-known fault-tolerant midpoint convergence function [30].

A violation of the condition $\pi^H + \pi_I \leq 2\pi^o$ in item (3) of Lemma 5.7 would indicate too frequent resynchronizations, i.e., a too large $\boldsymbol{\varepsilon}_{\max}$ compared to the maximum possible clock drift $P_S \boldsymbol{\rho}_{\max}$ during a round. In this case, no precision enhancement might take place, i.e., only $|\mathrm{ref}(\boldsymbol{S}_p) - \mathrm{ref}(\boldsymbol{S}_q)| \leq \pi^o$ can be guaranteed here.

The fact that $\boldsymbol{S}_p$ is $\boldsymbol{\pi}^o$-correct easily provides the "accuracy" α of our convergence function in the terminology of [58], which gives the maximum amount the resulting clock value can differ from any non-faulty input clock value. More specifically, since any non-faulty input interval is $\boldsymbol{\pi}^H$-correct, it follows that $|\alpha| \leq \max\{\pi^{H+} + \pi^{o-}, \pi^{H-} + \pi^{o+}\} \leq \pi^H$. Hence, $\mathcal{OP}(\cdot)$ provides a considerably better "accuracy" than most other convergence functions.

Although both accuracy preservation and intersection enhancement functions are modified in Lemma 5.8, it is nevertheless true that the corresponding results of Lemma 5.7 can be literally applied to $\Upsilon_{\max} < \infty$ as well. The reason is that (5.54) resp. (5.53) can only be less or equal than (5.19) resp. (5.18), whereas the sums of (5.54) and (5.58) resp. (5.53) and (5.57) are the same. Thus, accumulated over multiple rounds, the results obtained via Lemma 5.7 majorize the ones achieved via Lemma 5.8.

We did not carry over the improved bounds of item (1) and (4) from Lemma 5.7 for $\Delta\beta_p^\pm \leq 0$ to Lemma 5.8, since the latter is only encountered when node p's clock has bad accuracy (large drift $\boldsymbol{\rho}_p$), communication is good (low uncertainty $\boldsymbol{\varepsilon}_{qp}$), and there are many (at least $2d + 3e$) high-accuracy clocks in the system. It does not seem worth while to squeeze out our formulae for this rarely encountered situation.

5.3.2 Clock State Algorithm

In this section, we will plug in the results obtained for the optimal precision convergence function $\mathcal{OP}(\cdot)$ into the generic expressions for precision, accuracy, etc. of Theorem 3.1. This yields a complete characterization of the worst case performance of the optimal precision algorithm OP-STATE for clock state synchronization by means of instantaneous correction, see Definition 3.7. Most results carry over to continuous amortization via Theorem 3.2.

Theorem 5.1 (Precision of OP-STATE) *For the system model complying to Assumptions 3.1–3.4 and to the fault model in Assumption 5.1, the optimal precision clock state algorithm* OP-STATE *for instantaneous clock correction, with transmission delay compensation*

$$\begin{aligned}\Delta &\geq 2\varepsilon_{\max} + \varepsilon^{+}_{\max} + (H+3)u_{\max} + 2G + G_S + \delta_{\max}(1+\rho^{-}_{\max}) \\ &\quad + (2P_S + \Lambda + \Omega + 2E_{\max} - 2E_{\min} - 2\delta_{\min})\rho_{\max} \\ &\quad + \mathcal{O}(P_S\rho^2_{\max} + G\rho_{\max} + \varepsilon_{\max}\rho_{\max}), \qquad (5.62)\end{aligned}$$

and

$$\boldsymbol{\pi}^o = \boldsymbol{\pi}_0 + \boldsymbol{u}_{\max} + P_S\boldsymbol{\rho}_{\max} + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max} \qquad (5.63)$$

$$\begin{aligned}\boldsymbol{\pi}^H &= \boldsymbol{\pi}_0 + 2\boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{\max} + (P_S + E_{\max} - E_{\min} - \delta_{\min})\boldsymbol{\rho}_{\max} \\ &\quad + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max} \qquad (5.64)\end{aligned}$$

$$\Upsilon_{\max} \geq \pi^o - \pi_0 = P_S\rho_{\max} + u_{\max} + \mathcal{O}(P_S\rho^2_{\max} + G\rho_{\max} + \varepsilon_{\max}\rho_{\max}) \qquad (5.65)$$

used in $\boldsymbol{\mathcal{OP}}^{\boldsymbol{\pi}^o,\boldsymbol{\pi}^H,\Upsilon_{\max}}_{n-d-e}(\cdot)$ *requires a computation time of* $\mathcal{O}(n\log n)$ *to state synchronize the (non-faulty) clocks of* n *nodes to the initial worst case precision (i.e., the precision at the beginning of each round for the last non-faulty clock)*

$$\begin{aligned}\pi_{0,\max} &= \pi_0 + u_{\max} + G + (E_{\max} - E_{\min})\rho_{\max} \\ &\quad + \mathcal{O}(P_S\rho^2_{\max} + G\rho_{\max} + \varepsilon_{\max}\rho_{\max}) \qquad (5.66)\end{aligned}$$

where $\boldsymbol{\pi}_0 = [-\pi_0^-, \pi_0^+]$ *is given by*

$$\begin{aligned}\pi_0^- &= \frac{1}{2}\Big(2\varepsilon_{\max} + (H+2)u_{\max} + 2G + G_S + u^{+}_{\max} - u^{-}_{\max} \\ &\quad + (P_S + \Lambda + \Omega + \Delta + 2E_{\max} - E_{\min} - 2\delta_{\min})\rho_{\max} + P_S(\rho^{+}_{\max} - \rho^{-}_{\max})\Big) \\ &\quad + \mathcal{O}(P_S\rho^2_{\max} + G\rho_{\max} + \varepsilon_{\max}\rho_{\max}) \qquad (5.67)\end{aligned}$$

$$\begin{aligned}\pi_0^+ &= \frac{1}{2}\Big(2\varepsilon_{\max} + (H+2)u_{\max} + 2G + G_S - u^{+}_{\max} + u^{-}_{\max} \\ &\quad + (P_S + \Lambda + \Omega + \Delta + 2E_{\max} - E_{\min} - 2\delta_{\min})\rho_{\max} - P_S(\rho^{+}_{\max} - \rho^{-}_{\max})\Big) \\ &\quad + \mathcal{O}(P_S\rho^2_{\max} + G\rho_{\max} + \varepsilon_{\max}\rho_{\max}), \qquad (5.68)\end{aligned}$$

so that

$$\begin{aligned}\pi_0 &= 2\varepsilon_{\max} + (H+2)u_{\max} + 2G + G_S \\ &\quad + (P_S + \Lambda + \Omega + \Delta + 2E_{\max} - E_{\min} - 2\delta_{\min})\rho_{\max} \\ &\quad + \mathcal{O}(P_S\rho_{\max}^2 + G\rho_{\max} + \varepsilon_{\max}\rho_{\max}). \end{aligned} \tag{5.69}$$

Moreover, OP-STATE *guarantees an overall worst case precision* $\pi_{\max}$ *satisfying*

$$\begin{aligned}\pi_{\max} &= 2\varepsilon_{\max} + (H+3)u_{\max} + 3G + G_S \\ &\quad + (2P_S + \Lambda + \Omega + \Delta + 2E_{\max} - E_{\min} - 2\delta_{\min})\rho_{\max} \\ &\quad + \max\left\{u^+_{\max} + (E_{\max} - E_{\min})\rho^+_{\max}, u^-_{\max} + (E_{\max} - E_{\min})\rho^-_{\max}\right\} \\ &\quad + \mathcal{O}(P_S\rho_{\max}^2 + G\rho_{\max} + \varepsilon_{\max}\rho_{\max}), \end{aligned} \tag{5.70}$$

and any two non-faulty nodes p, q resynchronize within real-time $t_p^R - t_q^R$ *satisfying*

$$E_p - E_q - \pi_P \le t_p^R - t_q^R \le E_p - E_q + \pi_P$$

for

$$\pi_P = \pi_0 + u_{\max} + P_S\rho_{\max} + \mathcal{O}(P_S\rho_{\max}^2 + G\rho_{\max} + \varepsilon_{\max}\rho_{\max}), \tag{5.71}$$

where clock state adjustments

$$\Upsilon_q \in \boldsymbol{\pi}^o \cap [-\Upsilon_{\max}, \Upsilon_{\max}] \tag{5.72}$$

are applied to the local clock of any non-faulty node q.

Proof. The computational complexity of the optimal precision algorithm is primarily determined by the complexity of computing the convergence function $\mathcal{OP}(\cdot)$, which is $\mathcal{O}(n\log n)$ by Lemma 5.7.

By item (2) of Theorem 3.1, $\boldsymbol{\pi}_0$ is the solution of the equation $||\boldsymbol{\pi}_0|| = \Psi_\pi(\boldsymbol{\pi}^H, \boldsymbol{\pi}^o, \boldsymbol{\pi}_I)$ involving $\mathcal{OP}(\cdot)$'s precision enhancement function $\Psi_\pi(\cdot)$ given by (5.25). Precision enhancement is optimal if $\boldsymbol{\pi}^o$ given by (5.63) is a symmetric interval, recall the remarks following Lemma 5.8. Hence, writing $\boldsymbol{\pi}^H = \boldsymbol{\pi}_0 + \boldsymbol{\pi}_1$ and $\boldsymbol{\pi}^o = \boldsymbol{\pi}_0 + \boldsymbol{\pi}_2$ with

$$\begin{aligned}\boldsymbol{\pi}_1 &= 2\boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{\max} + (P_S + E_{\max} - E_{\min} - \delta_{\min})\boldsymbol{\rho}_{\max} \\ &\quad + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}, \end{aligned} \tag{5.73}$$

$$\boldsymbol{\pi}_2 = \boldsymbol{u}_{\max} + P_S\boldsymbol{\rho}_{\max} + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max} \tag{5.74}$$

according to (3.82) and (3.81), we exploit our freedom of choosing an arbitrary reference point of $\boldsymbol{\pi}_0$ to enforce this symmetry: Setting

$$\boldsymbol{\pi}_0 = \left[-\left(\left\lceil\frac{\pi_I}{2}\right\rceil_{G_S} + \frac{\pi_1 + \pi_2^+ - \pi_2^-}{2}\right), \left\lceil\frac{\pi_I}{2}\right\rceil_{G_S} + \frac{\pi_1 + \pi_2^- - \pi_2^+}{2}\right] \tag{5.75}$$

such that $\pi_0 = 2\lceil\pi_I/2\rceil_{G_S} + \pi_1$ provides a symmetric interval

$$\boldsymbol{\pi}^o = \boldsymbol{\pi}_0 + \boldsymbol{\pi}_2 = \left[-\left(\left\lceil\frac{\pi_I}{2}\right\rceil_{G_S} + \frac{\pi_1 + \pi_2}{2}\right), \left\lceil\frac{\pi_I}{2}\right\rceil_{G_S} + \frac{\pi_1 + \pi_2}{2}\right] \tag{5.76}$$

and $\pi^H = \pi_0 + \pi_1 = 2\lceil\pi_I/2\rceil_{G_S} + 2\pi_1$. Evaluating the precision preservation function $\Psi_\pi(\cdot)$ given by (5.25) yields

$$\begin{aligned} \Psi_\pi(\boldsymbol{\pi}^H, \boldsymbol{\pi}^o, \boldsymbol{\pi}_I) &= \left\lceil\frac{\pi^H + \pi_I}{2}\right\rceil_{G_S} = \left\lceil \lceil\pi_I/2\rceil_{G_S} + \pi_1 + \pi_I/2 \right\rceil_{G_S} \\ &= 2\lceil\pi_I/2\rceil_{G_S} + \pi_1 = \pi_0 \end{aligned} \tag{5.77}$$

as required; recall that π_1 and all other precision values are integer multiples of G_S since its constituting parameters have this property, see Definition 3.7 of our algorithm. The condition $2\pi^o \geq \pi^H + \pi_I$ required for validity of the used expression for $\Psi_\pi(\cdot)$ is easily verified, since $\pi^o \geq \pi_0 \geq (\pi^H + \pi_I)/2$. Plugging in π_1^+, π_1^- from (5.73), π_2^+, π_2^- from (5.74), and $\lceil\pi_I/2\rceil_{G_S} \leq \pi_I/2 + G_S/2$ with $\pi_I = |\boldsymbol{\pi}_I|$ from (3.83) into (5.75) confirms the values of π_0^-, π_0^+ given in (5.68) and (5.67). An addition provides the value of π_0 stated in (5.69), and $\pi_{0,\max}$ given by (5.66) is only a restatement of (3.78).

Next, the value π_P given in (5.71) is obtained by plugging in (conservative) maximum bounds for $\boldsymbol{u}_p, \boldsymbol{u}_q$ and $\boldsymbol{\rho}_p, \boldsymbol{\rho}_q$ in (3.86). Inserting $\overline{\boldsymbol{\Phi}_\pi}(\boldsymbol{\pi}^o) = \overline{\boldsymbol{\pi}}^o$ according to item (2) of Lemma 5.7 and the definition (5.63) of $\boldsymbol{\pi}^o$ into (3.85) provides

$$\begin{aligned} \boldsymbol{\pi} &= \overline{\boldsymbol{\pi}}_0 + \overline{\boldsymbol{u}}_{max} + \Big(P_S + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\Big)\overline{\boldsymbol{\rho}}_{\max} \\ &\quad + \boldsymbol{\pi}_0 + \boldsymbol{u}_{\max} + \Big(P_S + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\Big)\boldsymbol{\rho}_{\max}, \end{aligned}$$

so that actually

$$\begin{aligned} \pi^+ = \pi^- &= 2\varepsilon_{\max} + (H+3)u_{\max} + 2G + G_S \\ &\quad + (2P_S + \Lambda + \Omega + \Delta + 2E_{\max} - E_{\min} - 2\delta_{\min})\rho_{\max} \\ &\quad + \mathcal{O}(P_S\rho_{\max}^2 + G\rho_{\max} + \varepsilon_{\max}\rho_{\max}) \end{aligned} \tag{5.78}$$

Now it is possible to evaluate (3.84) in Theorem 3.1, which confirms the value of $\pi_{\max}$ stated in (5.70).

To show the maximum clock correction bound (5.72), we do not use the bounds from (3.87) of the generic Theorem 3.1, since they would be overly conservative. This is due to the fact that nothing but consistency w.r.t. $\tau^{(k)}$ of the "old" $\boldsymbol{\pi}^o$-precision interval $\check{\boldsymbol{I}}_q^q$ and the newly computed $\boldsymbol{\Phi}_\pi(\cdot)$-precise $\boldsymbol{S}_q$ was assumed in the proof of Theorem 3.1. However, in case of $\boldsymbol{\mathcal{OP}}(\cdot)$, we have the additional input that $\mathrm{ref}(\boldsymbol{S}_q) \in \tilde{\boldsymbol{S}}_q \subseteq \hat{\boldsymbol{I}}_q^q$, recall (5.15) in Definition 5.5. This obviously implies that the maximum clock adjustment satisfies $\Upsilon_q \in \boldsymbol{\pi}^o$, unless $\Upsilon_{\max}$ sets an even stricter limit; this is conveniently expressed by the intersection $\boldsymbol{\pi}^o \cap [-\Upsilon_{\max}, \Upsilon_{\max}]$ as asserted.

It remains to justify the value of the transmission delay compensation Δ. Plugging in the expressions for π_0 and $\pi^- = \mathcal{O}(\varepsilon_{\max} + G + P_S\rho_{\max})$ according to (5.78) into the definition of Δ in (3.21), we easily obtain

$$\begin{aligned}
\Delta &\geq \frac{1}{1+\rho_{\max}^+}\Big(2\varepsilon_{\max} + (H+3)u_{\max} + 2G + G_S + \delta_{\max} + \varepsilon_{\max}^+ \\
&\qquad +(2P_S + \Lambda + \Omega + \Delta + 2E_{\max} - 2E_{\min} - 2\delta_{\min})\rho_{\max} \\
&\qquad +\mathcal{O}(P_S\rho_{\max}^2 + G\rho_{\max} + \varepsilon_{\max}\rho_{\max})\Big) \\
&= \frac{Z + \rho_{\max}\Delta}{1+\rho_{\max}^+}
\end{aligned}$$

Solving this for Δ yields

$$\begin{aligned}
\Delta &\geq \frac{Z}{1-\rho_{\max}^-} = Z\Big(1 + \rho_{\max}^- + \mathcal{O}\big((\rho_{\max}^-)^2\big)\Big) \\
&= 2\varepsilon_{\max} + \varepsilon_{\max}^+ + (H+3)u_{\max} + 2G + G_S + \delta_{\max} \\
&\quad + (2P_S + \Lambda + \Omega + 2E_{\max} - 2E_{\min} - 2\delta_{\min})\rho_{\max} \\
&\quad + \delta_{\max}\rho_{\max}^- + \mathcal{O}(P_S\rho_{\max}^2 + G\rho_{\max} + \varepsilon_{\max}\rho_{\max}),
\end{aligned}$$

which confirms the value of Δ given in (5.62). This eventually completes the proof of Theorem 5.1. □

The precision results above are valid for any setting of the parameters defined in the system model. That is, our analysis provides the worst case behavior of the algorithm under the worst setting of parameters. Note that one could derive improved worst case results for more relaxed parameterizations. Unfortunately, our algorithm does not benefit much from such situations unless refined worst case bounds are compiled into it — after all, $\boldsymbol{\mathcal{OP}}(\cdot)$ depends on $\boldsymbol{\pi}^o$ and $\boldsymbol{\pi}^H$.

With respect to precision, our OP-STATE algorithm achieves optimal performance like the optimal algorithm of [10]. More specifically, it has the same computational complexity

$\mathcal{O}(n \log n)$ and the same worst case precision $\pi_{\max} \approx 4\varepsilon + 4P_S\rho$ (in a comparable setting); our respective terminology is related by $\pi_{\max} = \delta$, $\varepsilon_{\max} = 2\Lambda$, $P_S = r_{\max}$, and $\rho_{\max} = 2\rho$. Both algorithms require initially synchronized clocks.

It is interesting to compare the fundamentals of our approaches, which are —despite of their similarity— based on totally different paradigms. In essence, optimality of the extended fault-tolerant midpoint algorithm of [10] rests upon enforcing the so-called *nested adjustment condition*: Any newly resynchronized clock is guaranteed to lie between two non-faulty "old" clocks in the system. Whereas this condition is automatically maintained when the remote clock readings are reasonably apart, it must be secured if all of them are close to each other. This is accomplished by modifying the well-known *fault-tolerant midpoint* (FTM) convergence function [30] to incorporate the "Λ-extended" clock of the own node as well. This way, if all remote clock readings surviving FTM are almost the same, it is the own "old" clock that sustains the nested adjustment condition.

By contrast, our optimal precision algorithm guarantees optimality by ensuring that a newly computed clock value is not farther away from internal global time than the worst of the non-faulty old clocks. This is accomplished by improving the relatively large precision interval computed from remote accuracy intervals by the smaller one —not spoiled from the remote clock reading error ε— from the own clock, simply by intersecting them. We think that this explains optimality in a more natural way than the somewhat artificial Λ-extension of [10] does, although our approaches are clearly related. Moreover, our notion of internal global time facilitates a considerably simpler proof (apart from analyzing additional accuracy quantities).

Theorem 5.2 (Accuracy of OP-STATE) *For the system model complying to Assumptions 3.1–3.4 and to the fault model in Assumption 5.1, the accuracies* $\alpha_q^{-,(k+1)}$, $\alpha_q^{+,(k+1)}$ *of a non-faulty node q's accuracy interval* $\boldsymbol{A}_q^{(k+1)}(t_q^{(k+1)}) = [T_q^{(k+1)} \pm \boldsymbol{\alpha}_q^{(k+1)}]$ *at the beginning of round* $k+1$, $k \geq 0$, *as computed by the the optimal precision clock state algorithm* OP-STATE *for instantaneous clock correction, and with transmission delay compensation* Δ *given by (5.62),* $\boldsymbol{\pi}^o$ *given by (5.63),* $\boldsymbol{\pi}^H$ *given by (5.64), and* $\Upsilon_{\max}$ *according to (5.65), are integer multiples of* G_S *that obey the bounds* $\boldsymbol{\alpha}_q^{(k+1)} \subseteq \boldsymbol{\beta}_q^{(k+1)}$ *with*

$$\begin{aligned}
\beta_q^{-,(k+1)} &= \beta_q^{-,(k)} + P_S\rho_q^- + u_q^- \\
&\quad + \min\left\{ \left\lceil \frac{P_S\rho_q + u_q + I_q^{-,(k)}}{2} \right\rceil_{G_S} + \left\lceil \frac{\min\{0, D_q^{-,(k)}\}}{2} \right\rceil_{G_S}, \Upsilon_{\max} \right\} \\
&\quad + \mathcal{O}(P_S\rho_{\max}^2 + G\rho_{\max} + \varepsilon_{\max}\rho_{\max}) && (5.79) \\
&\leq \beta_q^{-,(k)} + P_S\rho_q^- + u_q^- + \Upsilon_{\max} + \mathcal{O}(P_S\rho_{\max}^2 + G\rho_{\max} + \varepsilon_{\max}\rho_{\max}), && (5.80) \\
\beta_q^{-,(0)} &= \alpha_q^{-,0}, && (5.81)
\end{aligned}$$

$$\begin{aligned}\beta_q^{+,(k+1)} &= \beta_q^{+,(k)} + P_S\rho_q^+ + u_q^+ \\ &\quad + \min\left\{\left\lceil\frac{P_S\rho_q + u_q + I_q^{+,(k)}}{2}\right\rceil_{G_S} + \left\lceil\frac{\min\{0, D_q^{+,(k)}\}}{2}\right\rceil_{G_S}, \Upsilon_{\max}\right\} \\ &\quad + \mathcal{O}(P_S\rho_{\max}^2 + G\rho_{\max} + \varepsilon_{\max}\rho_{\max})\end{aligned} \tag{5.82}$$

$$\leq \beta_q^{+,(k)} + P_S\rho_q^+ + u_q^+ + \Upsilon_{\max} + \mathcal{O}(P_S\rho_{\max}^2 + G\rho_{\max} + \varepsilon_{\max}\rho_{\max}), \tag{5.83}$$

$$\beta_q^{+,(0)} = \alpha_q^{+,0}, \tag{5.84}$$

where $\boldsymbol{\alpha}_q^0 \subseteq \boldsymbol{\pi}_0$ *with* π_0 *given by (5.69) denotes the initial interval of accuracies,*

$$D_q^{-,(k)} = \beta_q^{x,-,(k)} - \left(\beta_q^{-,(k)} + P_S\rho_q^- + u_q^-\right) + \pi^{H+} - \pi^{o+} \tag{5.85}$$

$$D_q^{+,(k)} = \beta_q^{x,+,(k)} - \left(\beta_q^{+,(k)} + P_S\rho_q^+ + u_q^+\right) + \pi^{H-} - \pi^{o-} \tag{5.86}$$

with

$$\begin{aligned}\beta_q^{x,-,(k)} = \max_{p:n-2d-3e}&\Bigg\{\Big\{\beta_p^{-,(k)} + u_p^- + u_q^- + G + G_A + \varepsilon_{pq}^- + (P_S - \Delta - E_p)\rho_p^- \\ &\qquad + (E_q + \Delta - \delta_{pq})\rho_q^- + (\Lambda + \Omega)\max\{\rho_q^- - \rho_p^-, 0\}\Big\}_{p\neq q} \\ &\quad \cup \Big\{\beta_q^{-,(k)} + u_q^- + P_S\rho_q^-\Big\}\Bigg\} \\ &+ \mathcal{O}(P_S\rho_{\max}^2 + G\rho_{\max} + \varepsilon_{\max}\rho_{\max})\end{aligned} \tag{5.87}$$

$$\begin{aligned}\beta_q^{x,+,(k)} = \max_{p:n-2d-3e}&\Bigg\{\Big\{\beta_p^{+,(k)} + u_p^+ + u_q^+ + G_A + \varepsilon_{pq}^+ + (P_S - \Delta - E_p)\rho_p^+ \\ &\qquad + (E_q + \Delta - \delta_{pq})\rho_q^+ + (\Lambda + \Omega)\max\{\rho_q^+ - \rho_p^+, 0\}\Big\}_{p\neq q} \\ &\quad \cup \Big\{\beta_q^{+,(k)} + u_q^+ + P_S\rho_q^+\Big\}\Bigg\} \\ &+ \mathcal{O}(P_S\rho_{\max}^2 + G\rho_{\max} + \varepsilon_{\max}\rho_{\max})\end{aligned} \tag{5.88}$$

for $\max_{p:m}\mathcal{B}$ *denoting the m-th largest element of the set* $\mathcal{B} = \{b_p : 1 \leq p \leq n\}$, *and*

$$\begin{aligned}I_q^{-,(k+1)} &= \left\lceil\frac{\min\{\pi_I + D_q^{-,(k)}, P_S\rho_q + u_q + I_q^{-,(k)}\}}{2}\right\rceil_{G_S} - \left\lfloor\frac{\min\{0, D_q^{-,(k)}\}}{2}\right\rfloor_{G_S} \\ &\quad - \min\left\{0, \Upsilon_{\max} - \left\lceil\frac{P_S\rho_q + u_q + I_q^{-,(k)}}{2}\right\rceil_{G_S}\right\} \\ &\quad + \mathcal{O}(P_S\rho_{\max}^2 + G\rho_{\max} + \varepsilon_{\max}\rho_{\max})\end{aligned} \tag{5.89}$$

$$I_q^{-,(0)} = \pi_0, \tag{5.90}$$

$$\begin{aligned} I_q^{+,(k+1)} &= \left\lceil \frac{\min\{\pi_I + D_q^{+,(k)}, P_S\rho_q + u_q + I_q^{+,(k)}\}}{2} \right\rceil_{G_S} - \left\lfloor \frac{\min\{0, D_q^{+,(k)}\}}{2} \right\rfloor_{G_S} \\ &\quad - \min\left\{0, \Upsilon_{\max} - \left\lfloor \frac{P_S\rho_q + u_q + I_q^{+,(k)}}{2} \right\rfloor_{G_S} \right\} \\ &\quad + \mathcal{O}(P_S\rho_{\max}^2 + G\rho_{\max} + \varepsilon_{\max}\rho_{\max}) \qquad (5.91) \\ I_q^{+,(0)} &= \pi_0 \qquad (5.92) \end{aligned}$$

with

$$\begin{aligned} \pi_I &= \varepsilon_{\max} + Hu_{\max} + G + (\Lambda + \Omega + \Delta + E_{\max} - \delta_{\min})\rho_{\max} \\ &\quad + \mathcal{O}(P_S\rho_{\max}^2 + G\rho_{\max} + \varepsilon_{\max}\rho_{\max}). \qquad (5.93) \end{aligned}$$

The traditional accuracy $\aleph^{(k+1)} = T_q^{(k+1)} - t_q^{R,(k)}$ *at the beginning of round* $k+1$, $k \geq 0$, *satisfies*

$$\begin{aligned} \aleph^{(k+1)} &\in \left[-\pi_0^+ - (k+1)(P_S\rho_{\max}^+ + u_{\max}^+), \pi_0^- + (k+1)(P_S\rho_{\max}^- + u_{\max}^-)\right] \\ &\quad + (k+1)\mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max} \qquad (5.94) \end{aligned}$$

for π_0^- *resp.* π_0^+ *given by (5.67) resp. (5.68). The inverse rate* r_q^{-1} *of the synchronized clock at node* q *evaluates to*

$$r_q^{-1} \in \left[1 \pm \boldsymbol{\rho}_{\max} + \frac{\boldsymbol{u}_{\max}}{P_S} + \mathcal{O}(P_S\rho_{\max} + G + \varepsilon_{\max})\boldsymbol{\rho}_{\max}\right]. \qquad (5.95)$$

Proof. We showed in the proof of Theorem 5.1 that $\boldsymbol{\pi}^o$ given by (5.63) is a symmetric interval, hence $\pi^{o+}/\pi^o = \pi^{o-}/\pi^o = 1/2$. According to item (1) of Theorem 3.1, the bounds $\beta_q^{-,(k+1)}$ resp. $\beta_q^{+,(k+1)}$ are just $\boldsymbol{\mathcal{OP}}(\cdot)$'s accuracy preservation functions $\Phi_\alpha^-(\cdot)$ resp. $\Phi_\alpha^+(\cdot)$ of Lemma 5.8 applied to the accuracy bounds $\boldsymbol{\beta}_q^{p,(k+1)} = \boldsymbol{\beta}_p^{(k)} + \boldsymbol{\zeta}_q^p$ valid at the end of round k.

To explain expressions (5.79) resp. (5.82), we first note that our definitions of $D_q^{-,(k)}$ in (5.85) resp. $D_q^{+,(k)}$ in (5.86) follow immediately from the ones of $\Delta\beta_p^-$ in (5.22) resp. $\Delta\beta_p^+$ in (5.23). Herein, we utilized the abbreviations $\beta_q^{x,-,(k)}$ resp. $\beta_q^{x,+,(k)}$ based upon (3.71) and plugged in $\beta_q^{-,(k)} + \zeta_q^{q,-}$ resp. $\beta_q^{+,(k)} + \zeta_q^{q,+}$ given by (3.72). In addition, denoting the outcome of the conditional intersection enhancement functions at the previous resynchronization instant by $I_q^{-,(k)}$ resp. $I_q^{+,(k)}$, we obtain $\pi^o - \iota_q^{-,(k)} = \pi^o - \pi_0 + I_q^{-,(k)} = P_S\rho_q - u_q + I_q^{-,(k)}$ resp. $\pi^o - \iota_q^{+,(k)} = P_S\rho_q - u_q + I_q^{+,(k)}$ by (3.75) resp. (3.76) in conjunction with the definition (5.63) of $\boldsymbol{\pi}^o$; note that the remainder term can be safely omitted here, since it is already present when the above equations are actually employed. More specifically, plugging the above results into (5.18) resp. (5.19) immediately leads to (5.79) resp. (5.82). Similarly, the expressions (5.89) for $I_q^{-,(k+1)}$ resp. (5.91) for $I_q^{+,(k+1)}$ are confirmed by employing those devices in (5.57) resp. (5.58); the expression (5.93) for π_I follows immediately from (3.83).

The initial values (5.81) and (5.84) for $\boldsymbol{\beta}_q^0$ are a simple consequence of the initial synchronization assumption in our generic algorithm's Definition 3.7. In addition, the initial values (5.90) of $I_q^{-,(0)}$ and (5.92) of $I_q^{+,(0)}$ are implied by (3.77) together with the fact that the initial synchronization assumption forces us to assume a zero-length common intersection $\forall s: \iota_s^{-,(0)} = \iota_s^{+,(0)} = 0$ at the beginning of round 0.

Finally, to prove the statement for traditional accuracy $\aleph^{(k+1)}$, we first bring $\boldsymbol{\mathcal{OP}}(\cdot)$'s precision preservation function $\boldsymbol{\Phi}_\pi(\boldsymbol{\pi}^o) = \boldsymbol{\pi}^o$ into (3.88) to obtain

$$T_q^{(k+1)} - t_q^{R,(k)} \in \overline{\boldsymbol{\pi}}_0 + (k+1)(\overline{\boldsymbol{\pi}}^o - \overline{\boldsymbol{\pi}}_0);$$

inserting the (swapped) expression for $\boldsymbol{\pi}^o - \boldsymbol{\pi}_0$ from (5.63) easily yields (5.94). Plugging $\boldsymbol{\pi}^o - \boldsymbol{\pi}_0$ into (3.89) also justifies expression (5.95) for the inverse rate of the synchronized clock, completing the proof of Theorem 5.2. □

It is important to note that accuracy intervals can grow faster than traditional accuracy, which reveals some sub-optimality of the optimal precision algorithm: According to (5.94), traditional accuracy increases resp. decreases at most by $P_S\rho_{\max}^- + u_{\max}^-$ resp. $P_S\rho_{\max}^+ + u_{\max}^+$ during each round. However, (5.80) and (5.83) reveal that both positive and negative accuracy exceed this growth by $\Upsilon_{\max} = P_S\rho_{\max} + u_{\max}$ (in case of the optimal choice of $\Upsilon_{\max}$). Hence, both α^+ and α^- can grow by more than $\pi_0 + (k+1)(P_S\rho_{\max} + u_{\max})$, although this cannot happen simultaneously for both negative and positive accuracy. Intuitively, this is due to the fact that the reference point and any edge could move into opposite directions at resynchronization, because faulty intervals might affect the accuracy and precision algorithm differently. Remember that it can even happen that the reference point is placed outside the originally computed accuracy interval.

In general, it is not apparent whether there is much laxness in accuracy when using the much simpler bounds (5.80) resp. (5.83) instead of the exhaustive ones (5.79) resp. (5.82), since parameter settings for either case are conceivable.

As the optimal algorithm of [10] and the one of [66], our OP-STATE algorithm provides optimal worst case traditional accuracy and drift of the synchronized clocks. More specifically, the latter is at most the maximum rate deviation $\rho_{\max}$ of the (worst) physical clock, where it has been proved in [66] that the worst case global rate cannot be better than the rate of the underlying physical clocks.

5.3.3 Clock Rate Algorithm

In this section, we will plug in the results obtained for the optimal precision convergence function $\boldsymbol{\mathcal{OP}}(\cdot)$ into the generic expressions for consonance of Theorem 4.1 and for drift of Theorem 4.2. This yields a complete characterization of the worst case performance of the optimal precision algorithm OP-RATE for clock rate synchronization, see Definition 4.8.

Since the analysis of $\boldsymbol{\mathcal{OP}}(\cdot)$ was done in the context of clock state synchronization, we need to adapt the fault model in Assumption 5.1 in the following way: Let a single rate interval $\boldsymbol{R}$ that is faulty w.r.t. v and/or φ be called *v/φ-symmetrically faulty* if $1/v <$ left$(\boldsymbol{R})$ and $\varphi/v <$ left(ref$(\boldsymbol{R})+\boldsymbol{\gamma}$) or either $1/v >$ right$(\boldsymbol{R})$ and $\varphi/v >$ right(ref$(\boldsymbol{R})+\boldsymbol{\gamma}$), and *$v/\varphi$-asymmetrically faulty* otherwise; a set $\boldsymbol{\mathcal{R}}$ of faulty rate intervals is *identically* v/φ-symmetrically faulty if $1/v$ is either to the left or to the right for all members of $\boldsymbol{\mathcal{R}}$ identically. For a better understanding reconsider Definition 4.11.

Theorem 5.3 (Consonance of OP-RATE) *For the system model complying to Assumptions 4.1–4.4 and to the fault model in Assumption 5.1 (adapted to the rate framework), the optimal precision clock rate algorithm* OP-RATE *with*

$$\begin{aligned}\boldsymbol{\gamma}^o &= \left[0 \pm \left(\sigma_{\max}\,(3P_R + 4(F+E) + 3B) + \frac{2\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\right)\right] \\ &\quad + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}(\pi_{\max} + B\rho_{\max}))] \qquad (5.96)\end{aligned}$$

$$\begin{aligned}\boldsymbol{\gamma}_H &= \left[0 \pm \left(\sigma_{\max}\,(3P_R + 6(F+E) + 4B) + \frac{3\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\right)\right] \\ &\quad + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})], \qquad (5.97)\end{aligned}$$

$$\Theta_{\max} \ \geq \ 2\sigma_{\max}P_R + \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max}) \qquad (5.98)$$

used in $\boldsymbol{\mathcal{OP}}_{n-d-e}^{\boldsymbol{\gamma}^o,\boldsymbol{\gamma}_H,\Theta_{\max}}(\cdot)$ *requires a computation time of* $\mathcal{O}(n\log n)$ *to rate synchronize the (non-faulty) clocks of* n *nodes to the initial worst case consonance (i.e., the consonance at the beginning of each round for the last non-faulty clock)*

$$\begin{aligned}\gamma_{0,\max} &= \sigma_{\max}\,(4P_R + 8(F+E) + 6B) + \frac{4\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}} \\ &\quad + \mathcal{O}(G_{\max} + \sigma_{\max}(\pi_{\max} + B\rho_{\max}))] \qquad (5.99)\end{aligned}$$

with error term $G_{\max} = \left(\rho_{\max} + \sigma_{\max}P_R + \frac{B+\epsilon_{\max}+(F+E)\rho_{\max}}{P_R}\right)^2$. *Moreover,* OP-RATE *guarantees an overall worst case consonance* $\gamma_{\max}$ *satisfying*

$$\begin{aligned}\gamma_{\max} &= \sigma_{\max}\,(6P_R + 8(F+E) + 6B) + \frac{4\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}} \\ &\quad + \mathcal{O}(G_{\max} + \sigma_{\max}(\pi_{\max} + B\rho_{\max}))] \qquad (5.100)\end{aligned}$$

and the ratio of successive coupling factors for the local clock of any non-faulty node lies within

$$\begin{aligned}\boldsymbol{\Theta}_{\max} &= \left[1 \pm \min\left\{\frac{||\boldsymbol{\gamma}^o||}{2}, \Theta_{\max}\right\}\right] \\ &\quad + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}(\pi_{\max} + B\rho_{\max}))]. \qquad (5.101)\end{aligned}$$

Proof. The computational complexity of the optimal precision algorithm is primarily determined by the complexity of computing the convergence function $\boldsymbol{\mathcal{OP}}(\cdot)$, which is $\mathcal{O}(n \log n)$ by Lemma 5.7.

According to Theorem 4.1, the consonance interval $\boldsymbol{\gamma}^*$ is the solution of the equation $||\boldsymbol{\gamma}^*|| = \Psi_\gamma(\boldsymbol{\gamma}^1, \ldots, \boldsymbol{\gamma}^n; \boldsymbol{\gamma}_H; \boldsymbol{\gamma}_I; \ldots)$ involving the consonance enhancement function $\Psi_\gamma(\cdot)$. It equals $\boldsymbol{\mathcal{OP}}(\cdot)$'s precision enhancement function $\Psi_\pi(\cdot)$ from (5.25) up to factor $(1 + \mathcal{O}(\rho_{\max}))$, which is certified by item (3) from Lemma 4.12 and its apparent multiplicativity, hence by setting $G_S = 0$ here

$$\Psi_\gamma(\boldsymbol{\gamma}_H, \boldsymbol{\gamma}^o, \boldsymbol{\gamma}_I) = \max\left\{\gamma^{o+} + \frac{\gamma^{o-}}{\gamma^o}(\gamma_H - \gamma^o + \gamma_I), \gamma^{o-} + \frac{\gamma^{o+}}{\gamma^o}(\gamma_H - \gamma^o + \gamma_I)\right\}(1 + \mathcal{O}(\rho_{\max}))$$

if $\gamma_H + \gamma_I \leq 2\gamma^o$. Again, consonance enhancement is optimal if $\gamma^{o-} = \gamma^{o+}$, leading to

$$\Psi_\gamma(\boldsymbol{\gamma}_H, \boldsymbol{\gamma}^o, \boldsymbol{\gamma}_I) = \frac{\gamma_H + \gamma_I}{2}(1 + \mathcal{O}(\rho_{\max})). \qquad (5.102)$$

From item (1) of Lemma 4.16 we know that $\boldsymbol{\gamma}^o = \boldsymbol{\gamma}^* + [0 \pm \sigma_{\max} P_R] + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})]$, which enforces $\boldsymbol{\gamma}^*$ to be symmetric as well, i.e., $\boldsymbol{\gamma}^* = [0 \pm \gamma^{*,\pm}]$. Therefore, combining (5.102) and (4.91) provides the equation

$$4\gamma^{*,\pm} = (\gamma_H + \gamma_I)(1 + \mathcal{O}(\rho_{\max})),$$

where $\gamma_H = ||\boldsymbol{\gamma}_H||$ from item (2) and $\gamma_I = ||\boldsymbol{\gamma}_I||$ from item (3) of Theorem 4.1. Solving this equation yields

$$\begin{aligned}\boldsymbol{\gamma}^* &= \left[0 \pm \left(2\sigma_{\max}\left(P_R + 2(F + E) + \frac{3}{2}B\right) + \frac{2\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F + E)\rho_{\max}}\right)\right] \\ &\quad + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}(\pi_{\max} + B\rho_{\max}))] \qquad (5.103)\end{aligned}$$

with error term $G_{\max} = \left(||\mathbf{R}_{\max}|| + \sigma_{\max} P_R + \frac{B + \epsilon_{\max} + (F+E)\rho_{\max}}{P_R}\right)^2$. Now we are ready to calculate $\boldsymbol{\gamma}^o$ from item (1) of Lemma 4.16, and $\boldsymbol{\gamma}_H$ from item (2) of Theorem 4.1, which justifies (5.96) and (5.97), respectively. This confirms also the needed condition $\gamma_H + \gamma_I \leq 2\gamma^o$ for the application of $\Psi_\gamma(\cdot)$ above. As stated in Lemma 5.8, the choice of parameter $\Theta_{\max}$ for $\boldsymbol{\mathcal{OP}}(\cdot)$ has to be at least $||\boldsymbol{\gamma}^o - \boldsymbol{\gamma}^*||$, which readily checks (5.98).

The calculated $\boldsymbol{\gamma}^*$ in (5.103) can be turned to the worst case consonance $\gamma_{0,\max}$ at the beginning $t^{(k)}$ of round $k \geq 0$ by evaluating (4.48) from Lemma 4.7. This ends up in (5.99) by noting that $\delta||\boldsymbol{\gamma}^*|| = \mathcal{O}(||\mathbf{R}_{\max}||(\sigma_{\max} P_R + \frac{\epsilon_{\max}}{P_R}))$ and $\mathcal{O}(||\mathbf{R}_{\max}||) = \mathcal{O}(\rho_{\max})$.

Instead of applying Lemma 4.17 to bound the rate adjustments $S_p^{(k+1)}/S_p^{(k)} \in \boldsymbol{\Theta}_{\max}$ for a non-faulty node p when commencing round $k+1$, $k \geq 0$, we obain a better result by exploiting $\boldsymbol{\mathcal{OP}}(\cdot)$'s construction. Going back to Definition 5.5 we simple have $r_o = 1$ and $r_n \in 1 + \boldsymbol{\gamma}^o$ with $\boldsymbol{\gamma}^o$ from (5.96). Together with $\Theta_{\max}$ from (5.98) we can simply argue that the computed reference point $\text{ref}(\tilde{\mathbf{R}}_p)$ lies in the intersection $(1 + \boldsymbol{\gamma}^o) \cap [1 \pm \Theta_{\max}]$, which justifies (5.101).

By virtue of item (2) from Lemma 4.12, the consonance preservation function $\boldsymbol{\Phi}_\gamma(\cdot)$ equals the (obviously multiplicative) precision preservation function $\boldsymbol{\Phi}_\pi(\cdot)$ of $\boldsymbol{\mathcal{OP}}(\cdot)$ from (5.24) up to factor $(1+\mathcal{O}(\rho_{\max}))$, hence $\boldsymbol{\Phi}_\gamma(\boldsymbol{\gamma}^o) = \boldsymbol{\gamma}^o(1+\mathcal{O}(\rho_{\max}))$. Now we can calculate the overall worst case consonance interval $\boldsymbol{\gamma}_{\max}$ with the help of (4.98) as

$$\begin{aligned}
\boldsymbol{\gamma}_{\max} &= (\boldsymbol{\gamma}^* + [0 \pm \sigma_{\max} P_R]) \cup \boldsymbol{\gamma}^o(1 + \mathcal{O}(\rho_{\max})) \\
&\quad + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})] \\
&= \left[0 \pm \left(\sigma_{\max}\left(3P_R + 4(F+E) + 3B\right) + \frac{2\epsilon_{\max}}{P_R - B - \epsilon_{\max} - 4(F+E)\rho_{\max}}\right)\right] \\
&\quad + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}(\pi_{\max} + B\rho_{\max}))] \qquad (5.104)
\end{aligned}$$

and applying (4.48) again from Lemma 4.7 gives $\gamma_{\max}$ as stated in (5.100). This completes the proof about the consonance properties of algorithm OP-RATE. □

To assess the above results we calculate the *optimal* rate resynchronization period P_R^*, since each consonance expression is composed by a linear and hyperbolic dependence on P_R. Remember, this property was already announced in Chapter 4 on relative rate measurement. In particular, solving this problem for $\gamma_{\max}$ from (5.100), we get

$$P_R^* = \sqrt{\frac{2\epsilon_{\max}}{3\sigma_{\max}}} + B + \epsilon_{\max} + 4(F+E)\rho_{\max} \qquad (5.105)$$

and the associated overall consonance turns out to be

$$\gamma_{\max}^* = \sqrt{96\ \sigma_{\max}\epsilon_{\max}} + \mathcal{O}\left(\sigma_{\max}(F+E)\right). \qquad (5.106)$$

Considering our distributed system from Chapter 2 with transmission delay uncertainty $\epsilon_{\max} \approx 10^{-6}$ and oscillator stability $\sigma_{\max} \approx 10^{-9}$, we end up with $P_R^* \approx 26$ s and $\gamma_{\max}^* = 3 \cdot 10^{-7}$ s/s. Note again, that we are dealing with worst case results here, indeed making sense for our usage.

Theorem 5.4 (Drift of OP-RATE) *For the system model complying to Assumptions 4.1–4.4 and to the fault model in Assumption 5.1 (adapted to the rate framework), the rate interval* $\boldsymbol{R}_p^{(k)}$ *of a non-faulty node* p *at the beginning of round* $k+1$, $k \geq 0$, *as computed by the optimal precision clock rate algorithm* OP-RATE *with* $\boldsymbol{\gamma}^o$ *given by (5.96),* $\boldsymbol{\gamma}_H$ *given by (5.97), and* $\Theta_{\max}$ *chosen according to (5.98), obeys the bounds* $\mathrm{align}(\boldsymbol{R}_p^{(k)}) \subseteq \boldsymbol{V}_p^{(k)}$ *with*

$$\begin{aligned} \mathbf{V}_p^{(k)} &= \boldsymbol{V}_p^{(k-1)} + [0 \pm (\sigma_p P_R + \Theta_{\max})] \\ &\quad + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})] && (5.107) \\ \mathbf{V}_p^{(0)} &= [\rho_p/(1+\rho_p), 1, \rho_p/(1-\rho_p)] && (5.108) \end{aligned}$$

and error term $G_{\max} = \left(\rho_{\max} + \sigma_{\max} P_R + \frac{B+\epsilon_{\max}+(F+E)\rho_{\max}}{P_R}\right)^2$.

Proof. According to Theorem 4.2, the lengths of the bounding rate intervals $\boldsymbol{V}_p^{(k)}$ are just the drift preservation functions $\Phi_\delta^\pm(\cdot)$ applied to the bounding remote rate intervals $\boldsymbol{V}_{p,q}^{(k)}$ that are valid at the end of round k. The drift preservation functions $\Phi_\delta^\pm(\cdot)$ equal the accuracy preservation functions $\Phi_\alpha^\pm(\cdot)$ of $\boldsymbol{\mathcal{OP}}(\cdot)$ up to factor $(1+\mathcal{O}(\rho_{\max}))$ as certified by item (4) of Lemma 4.12, so we can begin with

$$\begin{aligned} \mathbf{V}_p^{(k)} &= \Big[\Phi_\alpha^-\Big(\{\mathbf{V}_{p,1}^{(k)}, \dots, \mathbf{V}_{p,n}^{(k)}\}, \{\boldsymbol{\gamma}_{p,1}, \dots, \boldsymbol{\gamma}_{p,n}\}, \boldsymbol{\gamma}_H, \boldsymbol{\gamma}^o, 0\Big), 0, \\ &\quad \Phi_\alpha^+\Big(\{\mathbf{V}_{p,1}^{(k)}, \dots, \mathbf{V}_{p,n}^{(k)}\}, \{\boldsymbol{\gamma}_{p,1}, \dots, \boldsymbol{\gamma}_{p,n}\}, \boldsymbol{\gamma}_H, \boldsymbol{\gamma}^o, 0\Big)\Big] \; (1+\mathcal{O}(\rho_{\max})) \\ &\quad + [0 \pm \mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})] && (5.109) \end{aligned}$$

for $k \geq 1$. From Lemma 5.8 we can extract that

$$\begin{aligned} &\Big[\Phi_\alpha^-\Big(\{\mathbf{V}_{p,1}^{(k)}, \dots, \mathbf{V}_{p,n}^{(k)}\}, \boldsymbol{\gamma}_H, \boldsymbol{\gamma}^o, 0\Big), 0, \\ &\quad \Phi_\alpha^+\Big(\{\mathbf{V}_{p,1}^{(k)}, \dots, \mathbf{V}_{p,n}^{(k)}\}, \boldsymbol{\gamma}_H, \boldsymbol{\gamma}^o, 0\Big)\Big] \subseteq \boldsymbol{V}_{p,p}^{(k)} + [0 \pm \Theta_{\max}], && (5.110) \end{aligned}$$

because $\Theta_{\max}$ from (5.98) provides a valid bound for the involved minimum expression in (5.53)/(5.54). Also the necessary multiplicativity of $\Phi_\alpha^\pm(\cdot)$ can be seen right away. Setting $\boldsymbol{V}_{p,p}^{(k)} = \boldsymbol{V}_p^{(k-1)} + [0 \pm \sigma_p P_R]$ gathered from the clock rate algorithm's Definition 4.8, we can combine (5.109) and (5.110) into the recursive statement (5.107).

For $k = 0$ we assumed a neutral coupling factor and set $\mathbf{V}_p^{(0)}$ according to (5.108) justified by Lemma 4.1. □

Making recursion (5.107) explicit, we simply get $\mathbf{V}_p^{(k)} = \boldsymbol{V}_p^{(0)} + [0 \pm k(\sigma_p P_R + \Theta_{\max})] + [0 \pm k\mathcal{O}(G_{\max} + \sigma_{\max}\pi_{\max})]$. As already mentioned at the end of Chapter 4, these bounds are rather weak and need to be strengthened by using advanced stability assumptions and/or improved drift preservation functions.

5.4 Summary and Future Research

In this chapter we introduced and rigorously analyzed a novel convergence function-based optimal precision clock state synchronization algorithm OP-STATE that provides on-line bounds of the accuracy w.r.t. external time. It utilizes the optimal precision convergence function $\boldsymbol{\mathcal{OP}}(\cdot)$, which is based on Marzullo's function $\boldsymbol{\mathcal{M}}(\cdot)$. Relying on the generic interval-based framework from Chapter 3 our comprehensive analysis reveals that OP-STATE has a similar worst case performance as the optimal algorithm of [10]: Maximum precision $(4\varepsilon + 4P_S\rho)$, maximum clock correction $(2P_S\rho)$, and global rate (ρ) match their provably necessary lower bounds, cf. [9]. In terms of accuracy intervals, OP-STATE's both positive and negative worst case accuracy can be as large as $3P_S\rho$.

Another important fact revealed by our analysis is that clock granularities (G) and, in particular, rate adjustment uncertainties (usually $u = G$) of discrete rate-adjustment techniques have a considerable impact (as much as $11u + 3G$) upon achievable worst case precision and accuracy. This makes clear that any attempt to approach 1 μs worst case precision —as targeted by our project SynUTC— must utilize clocks with $G, u \ll 1\mu$s.

Furthermore, we showed that the optimal precision convergence function $\boldsymbol{\mathcal{OP}}(\cdot)$, which was originally tailored to clock state synchronization, can be reused in the rate framework as well, resulting in algorithm OP-RATE. It is an interesting alternative to high-stability quartz oscillators when targeting clock synchronization with very high precision, where decreasing any clock's drift rate below 10^{-7} s/s is mandatory. Our analysis indicated that OP-RATE can achieve a consonance around $10\sqrt{\sigma\varepsilon}$, and the drifts grow roughly $3\sigma P_R$ per round when no external synchronization takes place.

An important part of our future research is hence devoted to the analysis of alternative algorithms, since OP-STATE's suboptimality w.r.t. accuracy intervals suggests that there might be room for improvement. In particular, a certain generalization of the fault-tolerant midpoint algorithm of [30] seems to be promising. Unlike OP-STATE, this algorithm does not deal with precision and accuracy orthogonally but rather in an integrated way, which hopefully leads to considerably improved accuracy bounds.

Finally, we are running simulations of our algorithms with the help of the elaborated C++ program [74]. It uses a discrete-event approach to simulate the environment of such algorithms, and allows the user to control this environment, thus exposing the clock synchronization algorithm to certain conditions. By carrying out well-devised experiments, we expect to gain further insights of the algorithm's (average case) behavior and clues for improvements.

◇

Chapter 6

COMPLETE SYNCHRONIZATION ALGORITHM

6.1 Introduction

Synchronizing clocks in a distributed system turns out to be a challenging problem since many subjects are involved, like the physics of clocks, networking technologies, software engineering or theoretical concepts. Consequently, it is not surprising that our approach for a 1 μs clock synchronization running on COTS components embraces several headings: hardware support in Chapter 2, interval-based state synchronization in Chapter 3, interval-based rate synchronization in Chapter 4, and interval-based convergence functions in Chapter 5. It is the purpose of this chapter to synthesize all pieces to a complete synchronization algorithm, supplement it with on-line measurements, and provide a comprehensive analysis.

First of all, let us take a look at Figure 6.1 to get an overview of the objects for clock synchronization residing at a single node. The hardware clock gets its adjustment data from the clock synchronization software (represented as rounded boxes) and in its turn needs to generate timestamps and to invoke duties. Our well designed UTCSU-ASIC accomplishes these functions to the full extent, see Section 2.4. The network interface is responsible to send/receive timestamped packets to/from other nodes including the supply of external time information from GPS receivers. The necessary hardware to support those features is lumped together in our NTI M-Module, see Section 2.3.

Turning our attention to the software, we can indicate three major algorithms: the clock state algorithm (CSA), the clock rate algorithm (CRA) and the transmission delay measurement algorithm (TDA). Chapter 3 and 4 cover the first two algorithms, but left open how they affect each other. This will be the focus of Section 6.3 by reviewing both frameworks and putting together the relevant algorithmic parts.

After all, interval-based clock state and rate algorithms have the inevitable shortcoming that they depend explicitly on certain system parameters. Most importantly, the

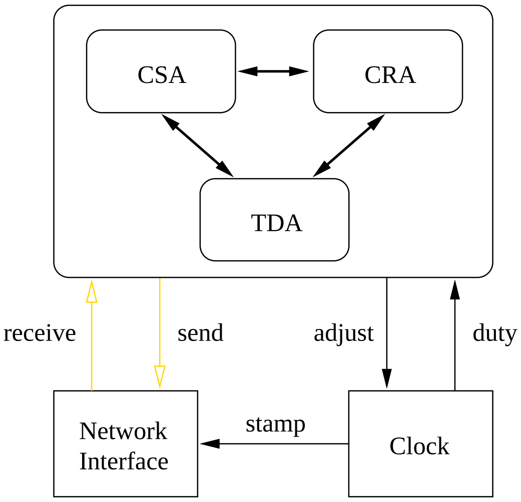


Figure 6.1: *Overview of Synchronization Algorithms*

transmission delay characteristics when a packet is sent from one node to another needs to be known. The characteristic parameters could be provided statically to the algorithm, but an on-line measurement is in fact the most appealing alternative. In Section 6.2 we devise a simple algorithm to measure these parameters during operation and quantify its efficiency. Extensions to our approach are listed in the final Section 6.4.

6.2 Transmission Delay Measurement

In Assumption 3.4 and 4.4, we characterized the transmission delay δ'_{pq} of a packet when sent from node p to q by the deterministic part δ_{pq} and the uncertainty $\boldsymbol{\varepsilon}_{pq} = [-\varepsilon^-_{pq}, \varepsilon^+_{pq}]$ such that $\delta'_{pq} \in [\delta_{pq} - \varepsilon^-_{pq}, \delta_{pq} + \varepsilon^+_{pq}]$. These parameters are not only important for the analysis, but are explicitly required by our clock state resp. clock rate algorithms from Definition 3.7 resp. 4.8 as well. As a consequence, the correctness and performance of our synchronization algorithms —especially the CSA— strongly depend on a good setting of these parameters: we say that $[\delta^I_{pq} \pm \boldsymbol{\varepsilon}^I_{pq}]$ is better than $[\delta^{II}_{pq} \pm \boldsymbol{\varepsilon}^{II}_{pq}]$ if both contain δ'_{pq} and $[\delta^I_{pq} \pm \boldsymbol{\varepsilon}^I_{pq}] \subset [\delta^{II}_{pq} \pm \boldsymbol{\varepsilon}^{II}_{pq}]$ holds. An underestimation of these values can lead to faulty intervals among the input of a particular convergence function; also the parametrization of the latter might be faulty.

Instead of working with a constant parameter setting, we propose an on-line measurement of them, since they are usually not at hand and may even change during operation. If fact, the transmission delay δ'_{pq} can be understood as a stochastic quantity obeying some continuous distribution. This conception has been used for other approaches as well, for instance the probabilistic clock synchronization [4] or the well-known NTP [38].

Our *transmission delay measurement algorithm* (TDA) is based on a conventional round-trip protocol to ascertain four timestamps:

1. Node p initiates a broadcast of timestamped packets $\langle p, T_p \rangle$
2. Node q receives $\langle p, T_p \rangle$ and timestamps it by T_q
3. Node q initiates a send to node p of timestamped packet $\langle q, T_p, T_q, T_q^{\prec} \rangle$
4. Node p receives $\langle q, T_p, T_q, T_q^{\prec} \rangle$ and timestamps it by $T_p^{\prec}$

If the corresponding real-times are t_p, t_q, $t_q^{\prec}$, and $t_p^{\prec}$, then we are interested to find out $\delta'_{pq} = t_q - t_p$ and $\delta'_{qp} = t_p^{\prec} - t_q^{\prec}$. Since each node hosts the same hardware, accesses the same (broadcast) channel and the load is supposed to be balanced, we make the assumption that $\delta'_{pq} = \delta'_{qp} = \delta^*_{pq}$. Furthermore, duration estimations can only be given locally, i.e., $\Delta t_p = t_p^{\prec} - t_p$ and $\Delta t_q = t_q^{\prec} - t_q$, hence in absence of adjustments during round-trip measurement, node p should be able to estimate

$$\delta^*_{pq} = \frac{\Delta t_p - \Delta t_q}{2}.$$

By applying Lemma 3.8 on the above equation and using the uniform bounds $\rho_{\max}$, $u_{\max}$, and G, we easily obtain that $\delta^*_{pq} \in [D^-_{pq}, D^+_{pq}]$, where

$$\begin{aligned}[D^-_{pq}, D^+_{pq}] &= \frac{(T_p^{\prec} - T_p) - (T_q^{\prec} - T_q)}{2}[1 \pm \rho_{\max}] \\ &\quad + \frac{1}{2}[-u_{\max}, u_{\max}] + (1 + \rho_{\max})[-G, G],\end{aligned}$$

so the uncertainty to measure δ^*_{pq} is roughly given by $e_{\max} = 2\rho_{\max}\delta_{\max} + u_{\max} + 2G$ unless the response of peer node q is delayed. Even in the ideal case of $\boldsymbol{\varepsilon}_{pq} = \emptyset$ we have to cope with $e_{\max}$, thus we can argue that this method cannot measure ε_{pq} better than $e_{\max}$. Plugging in reasonable numbers ($\delta_{\max} = 50\ \mu$s, $\rho_{\max} = 1$ ppm, $u_{\max} = 120$ ns, $G = 60$ ns) reveals that the granularities have a greater influence than the clock drifts. Therefore the TDA can be safely executed when the clocks are not (rate) synchronized or when a continuous amortization is going on.

So far we have only considered a single round-trip measurement, but several of them are necessary to eventually obtain a proper interval $[\delta_{pq} \pm \boldsymbol{\varepsilon}_{pq}]$. More specifically, after sufficient many round-trips, we get empirical distributions for D^-_{pq} and D^+_{pq}. The left tail of D^-_{pq} resp. the right tail of D^+_{pq} determines $\delta_{pq} - \varepsilon^-_{pq}$ resp. $\delta_{pq} + \varepsilon^+_{pq}$. Taking the average over all measurements $(T_p^{\prec} - T_q^{\prec} + T_q - T_p)/2$ leads to the desired reference point δ_{pq}. For an efficient implementation of our round-trip protocol, it can be mingled with the periodic FMEs and by exploiting broadcasting features.

6.3 Interlocking Clock State and Rate Algorithm

Let us briefly review the principles of a clock state and rate synchronization algorithm. Referring to Figure 4.6, at state resynchronization instants the CSA achieves a worst case precision π_0, which is mainly determined by the message delivery uncertainties and clock granularities, see Chapter 3. Between these resynchronization instants, the clocks might drift apart by as much as $2\rho_{\max}P_S$ when no rate synchronization takes place. Consequently, the CRA aims to reduce this deviation by keeping the clock rates close together, which is primarily hampered by clock stabilities, see Chapter 4. A crisp comparison of these two algorithms is given in Table 6.1, recalling the most important parameters, intervals and functions.

item	*Clock State Algorithm*	*Clock Rate Algorithm*
external synch.	accuracy α	drift δ
internal synch.	precision π	consonance γ
intervals	accuracy $\mathbf{A}_p$, precision $\boldsymbol{\pi}$	rate $\mathbf{R}_p$, consonance $\boldsymbol{\gamma}$
resynch. period	P_S (short)	P_R (long)
adjustments	additive, instantaneous or cont. amort.	multiplicative, instantaneous
uncertainties	delivery $\epsilon_{\max}$, granularity G, rate adj. $u_{\max}$, drift $\rho_{\max}$	stability $\sigma_{\max}$, delivery $\epsilon_{\max}$, precision $\pi_{\max}$
characteristic funct. of $\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$	accuracy preservation $\Phi^{\pm}_{\alpha}(\cdot)$ precision preservation $\boldsymbol{\Phi}_{\pi}(\cdot)$ precision enhancement $\Psi_{\pi}(\cdot)$ intersection enhancement $\Psi^{\pm}_{\iota}(\cdot)$	drift preservation $\Phi^{\pm}_{\delta}(\cdot)$ consonance preservation $\boldsymbol{\Phi}_{\gamma}(\cdot)$ consonance enhancement $\Psi_{\gamma}(\cdot)$ —

Table 6.1: *Comparing Clock State and Rate Algorithms*

In the previous chapters the clock rate and state synchronization algorithms were studied in isolation, however, when they are running at the same time there are mutual dependencies between them. Informally, a CRA requires only a moderate state synchronization: Considering internal clock synchronization, the CRA guarantees a certain worst case consonance $\gamma_{\max}$ and the CSA a worst case precision $\pi_{\max}$. Section 6.3.3 reveals that $\gamma_{\max}$ depends only loosely on $\pi_{\max}$, which allows us to have a separated analysis, putting the CSA on top of the CRA. This is also true in terms of external clock synchronization, because the clocks' accuracies are not relevant for the CRA.

On the other hand, a CSA can take advantage of a good rate synchronization, hence we revisit thoroughly clock state synchronization in order to incorporate the merits of clock rate synchronization. In the following, we proposes a refined deterioration of accuracy/precision intervals, which is expedient to make improvements to a clock state algorithm followed by a joint analysis.

6.3.1 Accuracy/Precision Interval Deterioration

In Chapter 4 we have shown that clock rate synchronization is able to achieve a certain drift and consonance. Additionally, a maintained local rate interval $\boldsymbol{R}_p$ and consonance interval $\boldsymbol{\gamma}_p$ provides us with information about the current clock rate, which can be used to replace the "fixed" inverse rate deviation bounds $\boldsymbol{\rho}_p$ for clock state synchronization, see Chapter 3. In this section we give two lemmas about the deterioration of accuracy/precision intervals in the presence of rate/consonance intervals gained from clock rate synchronization.

Lemma 6.1 (Improved Deterioration of Accuracy Intervals) *Given a clock $\mathcal{C}_p$ at node p steered by an CRA according to Definition 4.8. Let t_0 resp. t_1 be arbitrary real-times and $T_0 = C_p(t_0)$ resp. $T_1 = C_p(t_1)$ the corresponding clock states, where $t_0 \leq t_1$ and no resynchronization occurred in between. If accuracy interval $\boldsymbol{A}_p = [T_0 \pm \boldsymbol{\alpha}_p]$ is accurate at t_0 and local rate interval $\boldsymbol{R}_p \subseteq \boldsymbol{R}_{\max}$ is correct during $[t_0, t_1]$, then*

$$\boldsymbol{A}_p' = \boldsymbol{A}_p + (T_1 - T_0)\boldsymbol{R}_p + \boldsymbol{u}_p + I_{t_0 \neq \theta_0}\overline{\boldsymbol{G}} + I_{t_1 \neq \theta_1}\boldsymbol{G}_{\boldsymbol{\rho}} \tag{6.1}$$

is accurate at t_1, with rate adjustment uncertainty $\boldsymbol{u}_p$, and granularity intervals $\boldsymbol{G}_{\boldsymbol{\rho}} = [0, G(1 + ||\boldsymbol{R}_{\max}||)]$ and $\overline{\boldsymbol{G}} = [-G, 0]$ from Definition 3.5.

Proof. Let us begin with the accuracy intervals by recalling Section 3.3.1 on local interval clocks. Obviously, if $t_0 \in \boldsymbol{A}_p$ holds then

$$t_1 \in \boldsymbol{A}_p + (t_1 - t_0), \tag{6.2}$$

which means that $\boldsymbol{A}_p + (t_1 - t_0)$ is correct at t_1. The real-time duration $t_1 - t_0$ cannot be observed directly, but we are able to get a handle on it by considering the instantaneous clock rate $v_p(t)$ during $[t_0, t_1]$. The latter is defined as the derivative of the time-dependable function $C_p(t)$ of the clock state, hence we know that

$$T_1 - T_0 = \int_{t_0}^{t_1} v_p(\xi) d\xi \tag{6.3}$$

in the absence of resynchronizations. Due to Definition 4.4 the correctness of the local rate interval $\boldsymbol{R}_p = [\vartheta_p^-, 1, \vartheta_p^+]$ asserts that

$$\frac{1}{1+\vartheta_p^+} \leq v_p(t) \leq \frac{1}{1-\vartheta_p^-}$$

for any $t \in [t_0, t_1]$. Therefore (6.3) becomes to

$$\frac{t_1 - t_0}{1+\vartheta_p^+} \leq T_1 - T_0 \leq \frac{t_1 - t_0}{1-\vartheta_p^-},$$

which delivers the desired bounds upon $t_1 - t_0$ as

$$(T_1 - T_0)(1 - \vartheta_p^-) \leq t_1 - t_0 \leq (T_1 - T_0)(1 + \vartheta_p^+). \tag{6.4}$$

Plugging (6.4) into (6.2) yields

$$\begin{aligned} t_1 \;\; \in \;\; & \boldsymbol{A}_p + \left[T_1 - T_0 \pm (T_1 - T_0)[\vartheta_p^-, \vartheta_p^+]\right] \\ & \boldsymbol{A}_p + (T_1 - T_0)[\vartheta_p^-, 1, \vartheta_p^+] \\ & \boldsymbol{A}_p + (T_1 - T_0)\boldsymbol{R}_p, \end{aligned}$$

where we made appropriate use of the interval notation both from the state and rate synchronization context. To wrap up the proof, we point out that the three remaining terms of $\boldsymbol{A}_p'$ in (6.1) are just taken over from Definition 3.5 to account for uncertainties stemming from rate adjustment and synchrony issues, where we bounded $\rho_{\max}^+$ by the length of the largest correct local rate interval $\boldsymbol{R}_{\max}$, see Lemma 4.1. □

Essentially we have shown that the inverse rate deviation bound ρ_p^- resp. ρ_p^+ from clock state synchronization can be substituted by the inverse rate drift ϑ_p^- resp. ϑ_p^+ drawn from an according correct local rate interval $\boldsymbol{R}_p = [\vartheta_p^-, 1, \vartheta_p^+]$. In case of consecutive local rate intervals $\boldsymbol{R}_p^{(j)}$ for $0 \leq j \leq k$, where each one is correct during $[t_{j+1}, t_j]$ and $T_j = C_p(t_j)$, we have to carry out a deterioration with $\sum_{j=0}^{k}(T_{j+1} - T_j)\boldsymbol{R}_p^{(j)}$.

Unfortunately, the correctness of a local rate interval could be violated during continuous amortization, see Section 3.5.3 and Definition 3.12. This has no impact on the CRA itself because rate intervals remain correct at the end of each round, however, applications may be in error. To account for the amortization rate deviation ψ during an amortization phase, it is sufficient to extend the local rate interval $\boldsymbol{R}_p$ with $[0 \pm \psi]$. The justification is easy, since the clock rate v_p gets multiplied by $1 - \psi$ for amortization, hence $1 \in v_p(1-\psi)\left(\boldsymbol{R}_p + [0 \pm \psi] + \mathcal{O}(\psi||\boldsymbol{R}_p||)\right)$. Anyway, we do not further investigate this issue and leave it as an extension.

To adapt the deterioration of precision intervals let us briefly recap the definition of internal global time $\tau(t)$ from Section 3.5.1. At the beginning $t_S^{(k_S)}$ of a particular state round k_S it is determined by the set $\hat{\mathcal{C}}^{(k_S)}$ of associated precision intervals. Of importance is that internal global time progresses as real-time does until the next round $k_S + 1$ commences, thus

$$\frac{\tau(t)}{dt} = 1 \qquad \forall\, t_S^{(k_S)} \leq t < t_S^{(k_S+1)}. \tag{6.5}$$

Based on this setting, it is necessary to deteriorate precision intervals the same way as accuracy intervals, which is convenient for the purpose of analysis, but does not reflect the fact when the consonance between clocks is known as well.

The theory about clock rate synchronization introduced internal global rate $\varphi(t)$ to capture the rate of the ensemble of clocks, see Section 4.5.1. The important property is that it stays constant for a particular rate round k_R, thus

$$\varphi(t) = \varphi^{(k_R)} \qquad \forall\, t_R^{(k_R)} \leq t < t_R^{(k_R+1)}. \tag{6.6}$$

For the sake of an efficient implementation a rate round encloses $m \geq 1$ state rounds, where m denotes the integer multiple between P_R and P_S, see Section 4.4.1. More formally, we can say that $t_R^{(k_R)} = t_S^{(k_S)} < t_S^{(k_S+1)} < t_S^{(k_S+2)} < \ldots < t_S^{(k_S+m-1)} < t_S^{(k_S+m)} = t_R^{(k_R+1)}$ for appropriate instances of round numbers. When clock state synchronization is improved with clock rate synchronization, we need to modify internal global time from (6.5) in such a way that it considers internal global rate from (6.6). The key idea is to advance $\tau(t)$ piecewise with $\varphi(t)$ instead of the ideal rate 1, what leads to the following definition.

Definition 6.1 (Improved Internal Global Time) *The starting values $\tau^{(k_S)}$ constituting internal global time $\tau(t)$ at their respective beginnings $t_S^{(k_S)}$ of state rounds $k_S \geq 0$ are identical with them from Definition 3.9. A co-running CRA according to Definition 4.8 induces an internal global rate $\varphi(t)$ as defined in Section 4.5.1. If any state round $k_S \geq 0$ is enclosed by a particular rate round $k_R = m(k_S)$ then the* improved internal global time *is defined by*

$$\tau^{(k_S)}(t) = \tau^{(k_S)}(t_S^{(k_S)}) + \varphi^{(m(k_S))}(t - t_S^{(k_S)}) \tag{6.7}$$

for all $t_S^{(k_S)} \leq t < t_S^{(k_S+1)}$.

For illustration, a short passage of an improved internal global time $\tau(t)$ is shown in Figure 6.2, where we have chosen $m = 2$. The above refinement upon internal global time may seem awkward at the first glance, but it allows us to solve elegantly the problem of deteriorating precision intervals in the case of a known (small) consonance.

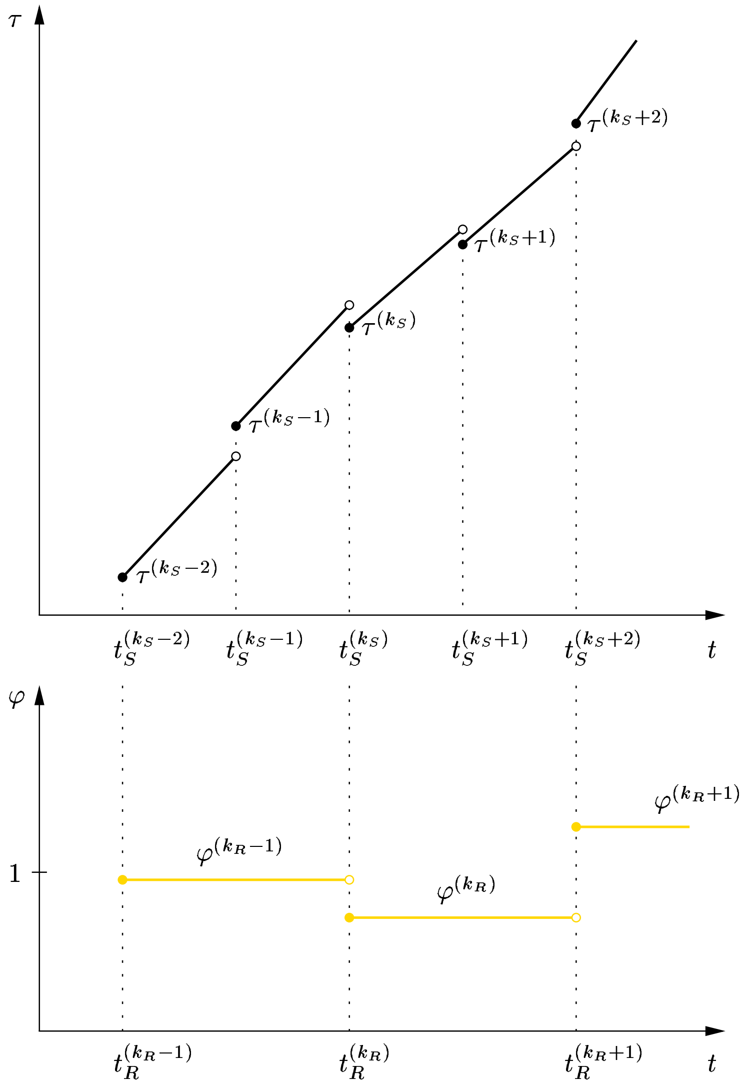


Figure 6.2: *Improved Internal Global Time*

Lemma 6.2 (Improved Deterioration of Precision Intervals) *Given a clock* $\mathcal{C}_p$ *at node* p *steered by an CRA according to Definition 4.8. Let* t_0 *resp.* t_1 *be arbitrary real-times and* $T_0 = C_p(t_0)$ *resp.* $T_1 = C_p(t_1)$ *the corresponding clock states, where* $t_0 \leq t_1$ *and no resynchronization occurred in between. If accuracy interval* $\boldsymbol{A}_p = [T_0 \pm \boldsymbol{\alpha}_p]$ *is* $\boldsymbol{\pi}_p$*-accurate w.r.t. internal global time* $\tau(t)$ *as given in Definition 6.1 at* t_0 *and local rate interval* $\boldsymbol{R}_p \subseteq \boldsymbol{R}_{\max}$ *is* $\boldsymbol{\gamma}_p$*-correct w.r.t. internal global rate* $\varphi(t)$ *during* $[t_0, t_1]$*, then* $\boldsymbol{A}'_p$ *given in (6.1) is* $\boldsymbol{\pi}'_p$*-accurate at* t_1*, where*

$$\boldsymbol{\pi}'_p = \boldsymbol{\pi}_p + (T_1 - T_0)\boldsymbol{\gamma}_p + \boldsymbol{u}_p + I_{t_0 \neq \theta_0}\overline{\boldsymbol{G}} + I_{t_1 \neq \theta_1}\boldsymbol{G}_{\boldsymbol{\rho}}, \tag{6.8}$$

with rate adjustment uncertainty $\boldsymbol{u}_p$*, and granularity intervals* $\boldsymbol{G}_{\boldsymbol{\rho}} = [0, G(1 + ||\boldsymbol{R}_{\max}||)]$ *and* $\overline{\boldsymbol{G}} = [-G, 0]$ *from Definition 3.5.*

Proof. Let us begin with the precision intervals by recalling Section 3.2.2. The given $\boldsymbol{\pi}_p$-accurateness of $\boldsymbol{A}_p$ at t_0 means that $\tau(t_0) \in [T_0 \pm \boldsymbol{\pi}_p]$. Since there are no resynchronizations until t_1, we know from Definition 6.1 that

$$\tau(t_1) \in [T_0 \pm \boldsymbol{\pi}_p] + \varphi^{(k_R)}(t_1 - t_0) \tag{6.9}$$

for a certain rate round k_R. Next we need to find bounds on the last term of (6.9) with the help of the available clock rate information. According to Definition 4.7, the $\boldsymbol{\gamma}_p$-correctness of $\boldsymbol{R}_p$ during $[t_0, t_1]$ from the CRA means that

$$v_p(t)(1 - \gamma_p^-) \leq \varphi^{(k_R)} \leq v_p(t)(1 + \gamma_p^+)$$

for any $t \in [t_0, t_1]$. Integrating the above inequality from t_0 to t_1 yields

$$(T_1 - T_0)(1 - \gamma_p^-) \leq \varphi^{(k_R)}(t_1 - t_0) \leq (T_1 - T_0)(1 + \gamma_p^+) \tag{6.10}$$

due to (6.3) and knowing that γ_p^-, γ_p^+ and $\varphi^{(k_R)}$ are constant here. Plugging (6.10) into (6.9) provides

$$\begin{aligned} \tau(t_1) \;\; \in \;\; & [T_0 \pm \boldsymbol{\pi}_p] + \Big[T_1 - T_0 \pm (T_1 - T_0)[\gamma_p^-, \gamma_p^+]\Big] \\ & \Big[T_1 \pm \boldsymbol{\pi}_p + (T_1 - T_0)\boldsymbol{\gamma}_p\Big], \end{aligned}$$

where the reference point T_1 happens to be the same as in (6.1), and the "deteriorated" precision interval $\boldsymbol{\pi}_p + (T_1 - T_0)\boldsymbol{\gamma}_p$ proves the claimed one of (6.8) apart from the rate adjustment uncertainty and granularity intervals. The usage of these intervals and the notational switch (rate/state context) is the same as in the proof of Lemma 6.1, hence $\boldsymbol{A}'_p$ is indeed $\boldsymbol{\pi}'_p$-accurate at t_1. □

We would like to emphasize again that w.r.t. improved internal global time, a precision interval can be deteriorated by the observable duration ΔT times the corresponding consonance interval $\boldsymbol{\gamma}_p$ rather than the rate interval $\boldsymbol{R}_p$ or even worse the inverse rate deviation bounds $\boldsymbol{\rho}_p$.

6.3.2 Improved Clock State Algorithm

In this section we make use of the new concepts from above and rebuild the clock state algorithm from Definition 3.7 insofar that it works together with the clock rate algorithm from Definition 4.8. Before we focus on the interplay of these two algorithms, let us reconsider the assumptions from both frameworks. For a graphical overview see again Figure 2.2 and 6.1.

Assumption 6.1 (System Model) *The distributed system consists of* $n \geq 2$ *nodes, which communicate with each other by message passing over a suitable communication network. Each node is equipped with a processor (with integer arithmetic only) that executes both the clock state and rate algorithm, an adjustable local interval clock, and a network interface. For the particular system parameters consult Table 3.1. Referring to the previously made assumptions, the complete system is characterized as follows:*

(1) execution times η_q*: As in Assumption 3.1 and 4.3, but the execution times refer in the sequel to a single computation of clock state and rate synchronization.*

(2) transmission characteristics $\delta_{pq}, \epsilon_{pq}^{\pm}, \lambda_{\max}, \omega_{\max}, G_A, b$*: As in Assumption 3.4 and 4.4, but with the relaxed technical condition* $\delta_{\min}\boldsymbol{\gamma}_{\max} \subseteq \boldsymbol{\varepsilon}_{\max}$ *(see Lemma 6.3).*

(3) local interval clocks *with* $G, G_S, u_q^{\pm}, \sigma_q, \rho_q$*: As in Assumption 3.2 and 3.3, where clocks are driven by oscillators according to Assumption 4.1.*

(4) abstract fault models $\mathcal{F}_S, \mathcal{F}_R$*: As presented in Section 3.3.3 and 4.4.6 independent from each other since they are attributed to different types of intervals.*

Based on these joined assumptions, we need to tailor the parameters required for the instance of the improved algorithm at node q. First of all, the *improved computation delay compensation* E_q is responsible to guarantee the maximum execution time η_q of node q. It has to satisfy

$$E_q \geq \frac{\eta_q + u_q^-}{1 - \vartheta_{\max}^-}, \tag{6.11}$$

where $\boldsymbol{R}_{\max} = [\vartheta_{\max}^-, 1, \vartheta_{\max}^+]$ is a stipulated maximal local rate interval such that $\boldsymbol{R}_q(t) \subseteq \boldsymbol{R}_{\max}$ for any non-faulty node q and any time $t \geq t^{(0)}$. The ensuing uniform bounds $E_{\max}$ and $E_{\min}$ are trivial. By the same token, the *improved broadcast delay compensation* $\Lambda + \Omega$ has to be chosen by

$$\Lambda + \Omega \geq \frac{\lambda_{\max} + \omega_{\max} + u_{\max}^-}{1 - \vartheta_{\max}^-}. \tag{6.12}$$

Together with the *improved transmission delay compensation* Δ, whose lower bound will be given in Lemma 6.3, this ensures that resynchronization starts only after all broadcast packets by non-faulty nodes have arrived during an FME. Finally, the state resynchronization period P_S has to be longer than $\Lambda + \Omega + \Delta + E_{\max}$.

The initial synchronization conditions remain unchanged, so at $t^{(0)}$ there are accuracy/precision intervals $\boldsymbol{A}_q, \boldsymbol{\pi}$ and $\boldsymbol{\pi}_0$ concerning the initial clock's state (see Definition 3.7) as well as rate/consonance intervals $\boldsymbol{R}_q$ and $\boldsymbol{\gamma}^{(0)}$ concerning the initial clock's rate (see Theorem 4.1 and 4.2). Again, they have to be achieved by some external means.

Definition 6.2 (Improved Clock State Algorithm) *Periodically, each node q in the system performs the following operations:*

- *(S)* *Send: At $k_S P_S$ calculate the reduced state round $\bar{k}_S = k_S \bmod m$. If $\bar{k}_S > 0$ then initiate broadcast of timestamped packets $\langle T_q, \boldsymbol{\alpha}_q \rangle$ else of $\langle T_q, \boldsymbol{\alpha}_q, \boldsymbol{R}_q, U_q \rangle$.*

- *(R)* *Reception and Preprocessing: Until $k_S P_S + \Lambda + \Omega + \Delta$ timestamp the arriving packets $\langle T_p, \boldsymbol{\alpha}_p \rangle$ or $\langle T_p, \boldsymbol{\alpha}_p, \boldsymbol{R}_p, U_p \rangle$ from nodes p with T_q^p. Then compute the accuracy interval*

$$\boldsymbol{A}_q^p = \begin{cases} \boldsymbol{A}_p + [\delta_{qp} \pm (\mathbf{2}\boldsymbol{G}_A + \boldsymbol{\varepsilon}_{qp})] & \\ \quad + (k_S P_S + \Lambda + \Omega + \Delta + E_q - T_q^p)\boldsymbol{R}_q + \boldsymbol{u}_q + \overline{\boldsymbol{G}} & \textit{if } p \neq q \\ \boldsymbol{A}_q + (k_S P_S + \Lambda + \Omega + \Delta + E_q - T_q^q)\boldsymbol{R}_q & \textit{otherwise} \end{cases}$$

and store it in an ordered set $\boldsymbol{\mathcal{A}}_q$. If $\bar{k}_S = 0$ then compute the quotient rate interval

$$\mathbf{Q}_{q,p} = \left[\frac{T_p - T_p^{\prec} + U_p}{T_q^p - T_q^{p,\prec} + U_q} \pm \left(\frac{(\sigma_q + \sigma_p)(T_p - T_p^{\prec} + U_p)}{2} + \frac{\varepsilon_{qp}}{T_q^p - T_q^{p,\prec} + U_q}\right)\right]$$

for $q \neq p$, preserving $T_p^{\prec} = T_p$ and $T_q^{p,\prec} = T_q^p$, the remote rate interval

$$\boldsymbol{R}_{q,p} = \begin{cases} \mathbf{Q}_{q,p} \cdot \mathbf{R}_p + [0 \pm (\sigma_p + \sigma_q)(\Lambda + \Omega + \Delta + E_q)] & \textit{if } p \neq q \\ \mathbf{R}_q & \textit{otherwise} \end{cases}$$

and store it in an ordered set $\boldsymbol{\mathcal{R}}_q$.

- *(C)* *Computation: At $k_S P_S + \Lambda + \Omega + \Delta$ apply the convergence function $\boldsymbol{\mathcal{CV}}_{\mathcal{F}_S}(\cdot)$ on $\boldsymbol{\mathcal{A}}_q$ resulting in the new accuracy interval $\boldsymbol{A}'_q$. If $\bar{k}_S = 0$ then apply the convergence function $\boldsymbol{\mathcal{CV}}_{\mathcal{F}_R}(\cdot)$ on $\boldsymbol{\mathcal{R}}_q$ resulting in the new rate interval $\boldsymbol{R}'_q$. The new local rate interval $\boldsymbol{R}_q = \mathrm{norm}(\boldsymbol{R}'_q) + \left[0 \pm (\bar{k}_S + 1)\sigma_q P_S\right]$. Reset $\boldsymbol{\mathcal{A}}_q$ and $\boldsymbol{\mathcal{R}}_q$ to $\emptyset$.*

- *(T)* *Termination and Resynchronization: At $k_S P_S + \Lambda + \Omega + \Delta + E_q$ set the local interval clock of node q to $\boldsymbol{A}'_q$ and deteriorate it with $\boldsymbol{R}_q$. Let the state adjustment $\Upsilon_q = \mathrm{ref}(\boldsymbol{A}'_q) - (k_S P_S + \Lambda + \Omega + \Delta + E_q)$. If $\bar{k}_S = 0$ then adjust the clock rate by setting the coupling factor $S_q = S_q \mathrm{ref}(\boldsymbol{R}'_p)$ and $U_q = \Upsilon_q$ else $U_q = U_q + \Upsilon_q$. Finally, increment state round k_S.*

Periodically, at $k_S P_S$ each node q initiates a broadcast of packet(s), which contain the accuracy interval $[T_q \pm \boldsymbol{\alpha}_q]$ at the moment of transmission as well as the local rate interval $\boldsymbol{R}_q$ and sum U_q of applied state adjustments when the reduced state round $\bar{k}_S = 0$.

The latter can be interpreted as the $\bar{k}_S$-th state round within the enclosing rate round, therefore if $\bar{k}_S > 0$ just a state resychnronization instant lies ahead, see Figure 4.6.

Until $k_S P_S + \Lambda + \Omega + \Delta$ each node q receives such packets from other nodes p and timestamp them on arrival with T_q^p. The accuracy interval $\boldsymbol{A}_q^p$ is computed as in the CSA of Definition 3.7, but improved by Lemma 6.1. If the next resychnronization instant is also regarding the clock rate (i.e., $\bar{k}_S = 0$) then the quotient rate interval $\boldsymbol{Q}_{q,p}$ and remote rate intervals $\boldsymbol{R}_{q,p}$ is computed as in the CRA of Definition 4.8. All these intervals are stored in their appropriate ordered set.

Similar to the preceding algorithms, the accuracy intervals $\boldsymbol{A}_q^p$ resp. rate intervals $\boldsymbol{R}_{q,p}$ (only when $\bar{k}_S = 0$) are fed into an interval-based convergence function $\mathcal{CV}_{\mathcal{F}_S}(\cdot)$ resp. $\mathcal{CV}_{\mathcal{F}_R}(\cdot)$ to output a new one $\boldsymbol{A}_q'$ resp. $\boldsymbol{R}_q'$. Note that the convergence functions do not need to be the same for clock state and rate synchronization due to the assumed independence between the underlying abstract fault models $\mathcal{F}_S$ and $\mathcal{F}_R$. Interesting is the calculation of the local rate interval $\boldsymbol{R}_q$ for the next state round $k_S + 1$, since a rate round encloses m state rounds it is sufficient to deteriorate the initial local rate interval $\mathrm{norm}(\boldsymbol{R}_q')$ gradually with $(\bar{k}_S + 1)\sigma_q P_S$ instead of $\sigma_q P_R$.

The state adjustment at $k_S P_S + \Lambda + \Omega + \Delta + E_q$ is as usual either applied instantaneously or by continuous amortization. The essential improvement to our algorithm comes into play by deteriorating the accuracy interval with the computed $\boldsymbol{R}_q = [\vartheta_q^-, 1, \vartheta_q^+]$. For example, in case of our UTCSU-ASIC the registers `LAMDBA±` are loaded by $\vartheta_q^\pm$ respectively, see Section 2.4. Finally, rate adjustment is carried out when due and the running sum U_q of state adjustments Υ_q is updated.

6.3.3 Closing Analysis

To evaluate the algorithm from Definition 6.2, we follow the analysis of Section 3.5 and make improvements to the lemmas and theorems. Our first one regards Lemma 3.11 about the evolution of accuracy/precision intervals during a state round. By presuming that rate synchronization enforces a certain drift and consonance of the clocks, we are able to determine a new transmission delay compensation as well as new bounds for the intervals of accuracies and precision intervals, see forthcoming Lemma 6.3. These bounds are the ingredients of the application of a certain convergence function, in order to re-establish the conditions for the next state round. Since most of these results are just simple substitutions, we do not provide an extra lemma, nevertheless, all improvements onto state synchronization by means of instantaneous correction are contained in Theorem 6.1. In Section 6.3.4, the optimal precision convergence function $\mathcal{OP}(\cdot)$ from Chapter 5 will be applied to gain some numerical results.

Lemma 6.3 (Improved FME Dissemination) *Given the algorithm from Definition 6.2 under Assumption 6.1 with the compensations* E_p *and* $\Lambda + \Omega$ *as specified by (6.11) and (6.12), respectively. Using the same notation and preconditions as in Lemma 3.11, and if for any non-faulty node p the local rate interval* $\boldsymbol{R}_p = [\vartheta_p^-, 1, \vartheta_p^+]$ *is*

[1'] correct during state round k_S *with* $\boldsymbol{R}_p \subseteq \boldsymbol{R}_{\max}$*,*

[2'] $\boldsymbol{\gamma}_p$*-correct w.r.t. internal global rate* $\varphi(t)$ *during state round* k_S *with* $\boldsymbol{\gamma}_p \subseteq \boldsymbol{\gamma}_{\max}$*,*

we can make the following improvements to the former results:

(1) The transmission delay compensation Δ *has to satisfy*

$$\Delta \geq \frac{\delta_{\max} + \varepsilon_{\max}^+}{1 - R_{\max}} + \frac{\pi_0 + u_{\max} + (P_S - E_{\min} + \pi^-)\gamma_{\max}}{1 + \gamma_{\max}^+} \tag{6.13}$$

instead of (3.26).

(2) The intervals of accuracies obey

$$\begin{aligned} \boldsymbol{\alpha}_q^{p,R} \subseteq\; & \boldsymbol{\alpha}_p + \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \boldsymbol{2G}_A + \boldsymbol{\varepsilon}_{pq} \\ & + (P_S - \Delta - E_p)\,\mathrm{align}(\boldsymbol{R}_p) + (E_q + \Delta - \delta_{pq})\,\mathrm{align}(\boldsymbol{R}_q) \\ & + (\Lambda + \Omega)\Big[-\max\{\vartheta_q^- - \vartheta_p^-, 0\}, \max\{\vartheta_q^+ - \vartheta_p^+, 0\}\Big] \\ & + \mathcal{O}(\pi + P_S R_{\max} + G + \varepsilon_{\max})\,\mathrm{align}(\boldsymbol{R}_q) \end{aligned} \tag{6.14}$$

for $p \neq q$ *instead of (3.29), and*

$$\boldsymbol{\alpha}_q^{q,R} = \boldsymbol{\alpha}_q + \boldsymbol{u}_q + P_S\,\mathrm{align}(\boldsymbol{R}_q) + \mathcal{O}(\pi)\,\mathrm{align}(\boldsymbol{R}_q) \tag{6.15}$$

instead of (3.31).

(3) The precision intervals comply with

$$\begin{aligned} \boldsymbol{\pi}_q^p \subseteq\; & \boldsymbol{\pi}_0 + \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{pq} + (P_S - \Delta - E_p)\boldsymbol{\gamma}_p + (E_q + \Delta - \delta_{pq})\boldsymbol{\gamma}_q \\ & + (\Lambda + \Omega)\Big[-\max\{\gamma_q^- - \gamma_p^-, 0\}, \max\{\gamma_q^+ - \gamma_p^+, 0\}\Big] \\ & + \mathcal{O}(\pi + P_S R_{\max} + G + \varepsilon_{\max})\boldsymbol{\gamma}_{\max} \end{aligned} \tag{6.16}$$

for $p \neq q$*,*

$$\boldsymbol{\pi}_q^q = \boldsymbol{\pi}_0 + \boldsymbol{u}_q + P_S\boldsymbol{\gamma}_q + \mathcal{O}(\pi)\boldsymbol{\gamma}_q, \tag{6.17}$$

$$\begin{aligned}\boldsymbol{\pi}_q^H \;\subseteq\; & \boldsymbol{\pi}_0 + \boldsymbol{u}_{\max} + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{\max} \\ & + (P_S - \Delta - E_{\min})\boldsymbol{\gamma}_{\max} + (E_q + \Delta - \delta_{\min})\boldsymbol{\gamma}_q \\ & + \mathcal{O}(\pi + P_S R_{\max} + G + \varepsilon_{\max})\boldsymbol{\gamma}_{\max}, \qquad (6.18)\end{aligned}$$

$$\begin{aligned}\boldsymbol{\pi}^H \;\subseteq\; & \boldsymbol{\pi}_0 + 2\boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{\max} + (P_S + E_{\max} - E_{\min} - \delta_{\min})\boldsymbol{\gamma}_{\max} \\ & + \mathcal{O}(\pi + P_S R_{\max} + G + \varepsilon_{\max})\boldsymbol{\gamma}_{\max}, \qquad (6.19)\end{aligned}$$

$$\begin{aligned}\boldsymbol{\pi}^p \;\subseteq\; & \boldsymbol{\pi}_0 + \boldsymbol{u}_p + \boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{\max} \\ & + (P_S - \Delta - E_p)\boldsymbol{\gamma}_p + (E_{\max} + \Delta - \delta_{\min})\boldsymbol{\gamma}_{\max} \\ & + (\Lambda + \Omega)\Big[-(\gamma_{\max}^- - \gamma_p^-), \gamma_{\max}^+ - \gamma_p^+\Big] \\ & + \mathcal{O}(\pi + P_S R_{\max} + G + \varepsilon_{\max})\boldsymbol{\gamma}_{\max}, \qquad (6.20)\end{aligned}$$

and

$$\begin{aligned}\boldsymbol{\pi}_I \;\subseteq\; & \boldsymbol{\varepsilon}_{\max} + H\boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + (\Lambda + \Omega + \Delta + E_{\max} - \delta_{\min})\boldsymbol{\gamma}_{\max} \\ & + \mathcal{O}(\pi_0 + R_{\max} + G + \varepsilon_{\max})\boldsymbol{\gamma}_{\max} \qquad (6.21)\end{aligned}$$

instead of (3.32), (3.33), (3.34), (3.35), (3.36), and (3.37), respectively.

Proof. As in the proof of Lemma 3.11, the real-times involved in the evolution of a state round k_S are related by $(t_q^p - t_q) = (t_{pq}^A - t_p) + \delta'_{pq} - (t_q - t_p)$, copied from (3.41), where t_p and t_q denote the respective round beginnings, t_{pq}^A the transmission and t_q^p the reception time of a packet when sent from node p to node q, and δ'_{pq} the transmission delay. To make the above equation amenable to our new concepts developed in Section 6.3.1, we map it to the corresponding points of internal global time, i.e., $\tau_p = \tau(t_p)$, $\tau_q = \tau(t_q)$, $\tau_{pq}^A = \tau(t_{pq}^A)$, and $\tau_q^p = \tau(t_q^p)$. To carry over δ'_{pq}, we just multiply it with the encompassing internal global rate $\varphi^{(k_R)}$, since $\tau(t_{pq}^A + \delta'_{pq}) = \tau_{pq}^A + \varphi^{(k_R)}\delta'_{pq}$ by virtue of Definition 6.1. Eventually, we arrive at

$$(\tau_q^p - \tau_q) = (\tau_{pq}^A - \tau_p) + \varphi^{(k_R)}\delta'_{pq} - (\tau_q - \tau_p). \qquad (6.22)$$

In the gist of (6.10) and (3.10) the l.h.s. is not smaller than $(T_q^p - T_q)(1 - \gamma_q^-) - u_q^-$ and the first expression in on the r.h.s. is bounded by $(T_{pq}^A - T_p)(1 + \gamma_p^+) + u_p^+$, where γ_q^- and γ_p^+ are the respective consonance values as presumed in [2']. Therefore (6.22) turns into inequalities

$$\begin{aligned}
(T_q^p - T_{pq}^A)(1-\gamma_q^-) &\leq (T_{pq}^A - T_p)(1+\gamma_p^+) + (T_q - T_{pq}^A)(1-\gamma_q^-) + u_q^- + u_p^+ \\
&\quad + \varphi^{(k_R)}\delta'_{pq} - (\tau_q - \tau_p) \\
&\leq (T_q - \tau_q) - (T_p - \tau_p) \tag{6.23} \\
&\quad + (T_{pq}^A - T_p)\gamma_p^+ + (T_{pq}^A - T_q)\gamma_q^- \tag{6.24} \\
&\quad + \varphi^{(k_R)}\delta'_{pq} + u_q^- + u_p^+. \tag{6.25}
\end{aligned}$$

Next we are applying a number of simplifications to the above terms. Let us begin with (6.23), which does not exceed π_0 since precondition [2] ensures $|(T_q - \tau_q) - (T_p - \tau_p)| \leq \pi_0$. Next, (6.24) can be reduced to $(P_S - \Delta - E_{\min} + \pi^-)(\gamma_q^- + \gamma_p^+)$ by crediting (3.39)/(3.40) and knowing that $E_p, E_q \geq E_{\min}$. Because $\delta'_{pq} \leq \delta_{pq} + \varepsilon_{pq}^+ \leq \delta_{\max} + \varepsilon_{\max}^+$ and $u_q^- + u_p^+ \leq u_{\max}^- + u_{\max}^+ \leq u_{\max}$, we bound (6.25) by $\varphi^{(k_R)}(\delta_{\max} + \varepsilon_{\max}^+) + u_{\max}$. It remains to find an estimation on the internal global rate $\varphi^{(k_R)}$, which can be done by applying Definition 4.4 and 4.7 jointly on precondition [1'] and [2'], thus

$$\varphi^{(k_R)} \leq \frac{1+\gamma_{\max}^+}{1-R_{\max}} = 1 + \gamma_{\max}^+ + R_{\max} + \mathcal{O}((\gamma_{\max}^+ + R_{\max})^2).$$

with $R_{\max} = ||\mathbf{R}_{\max}||$. When assembling all pieces (6.23)/(6.24)/(6.25) becomes to

$$\begin{aligned}
T_q^p - T_{pq}^A &\leq \frac{\pi_0 + u_{\max} + (P_S - \Delta - E_{\min} + \pi^-)\gamma_{\max}}{1-\gamma_{\max}^-} \\
&\quad + \frac{(\delta_{\max} + \varepsilon_{\max}^+)(1+\gamma_{\max}^+)}{(1-R_{\max})(1-\gamma_{\max}^-)}, \tag{6.26}
\end{aligned}$$

which allows us to to figure out Δ with the same kind of technique as in the proof of Lemma 3.11. As a result $T_q^p \leq (k+1)P_S + \Lambda + \Omega + \Delta$ if

$$\Delta\left(\frac{\gamma_{\max}}{1-\gamma_{\max}^-} + 1\right) \geq \frac{\pi_0 + u_{\max} + (P_S - E_{\min} + \pi^-)\gamma_{\max}}{1-\gamma_{\max}^-} + \frac{(\delta_{\max} + \varepsilon_{\max}^+)(1+\gamma_{\max}^+)}{(1-R_{\max})(1-\gamma_{\max}^-)},$$

which confirms item (1) of our lemma. Beyond that, (6.26) can be condensed to

$$T_q^p - T_{pq}^A = \delta_{pq} + \mathcal{O}(\pi_0 + P_S R_{\max} + G + \varepsilon_{\max}). \tag{6.27}$$

Turning our attention to the accuracy intervals, we know from the algorithm given in Definition 6.2 that each node q computes the accuracy interval

$$\boldsymbol{A}_q^p = \boldsymbol{A}_{pq}^A + [\delta_{pq} \pm (\boldsymbol{\varepsilon}_{pq} + \mathbf{2}\boldsymbol{G}_A)] + (T_q^R - T_q^p)\boldsymbol{R}_q + \boldsymbol{u}_q + \overline{\boldsymbol{G}}$$

upon reception of $\boldsymbol{A}_{pq}^A$ from node $p \neq q$, where the resynchronization time $T_q^R = k_S P_S + \Lambda + \Omega + \Delta + E_q$. If $\boldsymbol{A}_p = [T_p \pm \boldsymbol{\alpha}_p])$ is the accuracy interval on node p when the state round k_S started at local time T_p, then according to Lemma 6.1 the transmitted interval $\boldsymbol{A}_{pq}^A = \boldsymbol{A}_p + (T_{pq}^A - T_p)\boldsymbol{R}_p + \boldsymbol{u}_p$. Extracting the intervals of accuracies, we get

$$\begin{aligned}
\boldsymbol{\alpha}_q^{p,R} &= \boldsymbol{\alpha}_p + (T_{pq}^A - T_p)\,\mathrm{align}(\boldsymbol{R}_p) + \boldsymbol{u}_p \\
&\quad + \boldsymbol{\varepsilon}_{pq} + \boldsymbol{2G}_A + (T_q^R - T_q^p)\,\mathrm{align}(\boldsymbol{R}_q) + \boldsymbol{u}_q + \overline{\boldsymbol{G}} \\
&= \boldsymbol{\alpha}_p + \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \boldsymbol{2G}_A + \boldsymbol{\varepsilon}_{pq} \\
&\quad + (T_{pq}^A - T_p)[\vartheta_p^-, 0, \vartheta_p^+] + (T_q^R - T_{pq}^A)[\vartheta_q^-, 0, \vartheta_q^+] - (T_q^p - T_{pq}^A)\,\mathrm{align}(\boldsymbol{R}_q)
\end{aligned}$$

by remembering that $\mathrm{align}(\boldsymbol{R}_p) = \mathrm{align}([\vartheta_p^-, 1, \vartheta_p^+]) = [\vartheta_p^-, 0, \vartheta_p^+]$, which is good for dragging an accuracy interval without shifting it. When proceeding as in the proof of Lemma 3.11 we end up with

$$\begin{aligned}
\boldsymbol{\alpha}_q^{p,R} &\subseteq \boldsymbol{\alpha}_p + \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \boldsymbol{2G}_A + \boldsymbol{\varepsilon}_{pq} \\
&\quad + (P_S - \Delta - E_p + \pi^-)\,\mathrm{align}(\boldsymbol{R}_p) + (E_q + \Delta)\,\mathrm{align}(\boldsymbol{R}_q) \\
&\quad + (\Lambda + \Omega)\big[-\max\{\vartheta_q^- - \vartheta_p^-, 0\}, \max\{\vartheta_q^+ - \vartheta_p^+, 0\}\big] - \delta_{pq}\,\mathrm{align}(\boldsymbol{R}_q) \\
&\quad + \mathcal{O}(\pi_0 + P_S R_{\max} + G + \varepsilon_{\max})\,\mathrm{align}(\boldsymbol{R}_q)
\end{aligned}$$

and noting that (6.27) was used to capture $(T_q^p - T_{pq}^A)$. Case $p = q$ is a straightforward modification, which finishes item (2) of our lemma.

Finally, we reason about the precision intervals $\boldsymbol{\pi}_q^p$ that are associated with the $\boldsymbol{A}_q^p$'s. Relying on Lemma 6.2, the above derivations for the intervals of accuracies can be carried over by exchanging the rate intervals with suitable consonance intervals. Again, let $\boldsymbol{A}_p$ be $\boldsymbol{\pi}_0$-correct at the beginning of state round k_S. Hence $\boldsymbol{A}_q^p$ unfolds as being $\boldsymbol{\pi}_q^p$-correct with

$$\begin{aligned}
\boldsymbol{\pi}_q^p &\subseteq \boldsymbol{\pi}_0 + \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{pq} + (P_S - \Delta - E_p)\boldsymbol{\gamma}_p + (E_q + \Delta - \delta_{pq})\boldsymbol{\gamma}_q \\
&\quad + (\Lambda + \Omega)\big[-\max\{\gamma_q^- - \gamma_p^-, 0\}, \max\{\gamma_q^+ - \gamma_p^+, 0\}\big] \\
&\quad + \mathcal{O}(\pi + P_S R_{\max} + G + \varepsilon_{\max})\boldsymbol{\gamma}_{\max} \qquad (6.28)
\end{aligned}$$

for $p \neq q$, and $\boldsymbol{\pi}_q^q = \boldsymbol{\pi}_0 + \boldsymbol{u}_q + P_S\boldsymbol{\gamma}_q + \mathcal{O}(\pi)\boldsymbol{\gamma}_q$. The remaining precision intervals $\boldsymbol{\pi}_q^H, \boldsymbol{\pi}^H$, and $\boldsymbol{\pi}^p$ are just majorizations over (6.28), more specifically, $\boldsymbol{\pi}_q^H = \bigcup_p \boldsymbol{\pi}_q^p$, $\boldsymbol{\pi}^H = \bigcup_q \boldsymbol{\pi}_q^H$, and $\boldsymbol{\pi}^p = \bigcup_q \boldsymbol{\pi}_q^p$. For the inclusion of $\boldsymbol{\pi}_q^q$ the technical condition from (3.16) can be replaced by $\delta_{\min}\boldsymbol{\gamma}_{\max} \subseteq \boldsymbol{\varepsilon}_{\max}$, as provided in item (2) of Assumption 6.1.

Last but not least, the same line of reasoning holds for improving $\boldsymbol{\pi}_I$. By converting (3.49) we obtain

$$\begin{aligned}\boldsymbol{\pi}_I \;\subseteq\; & \bigcup_{q,p\neq q}\Big[(T_{pq}^A - \min_q\{T_{pq}^A\})\boldsymbol{\gamma}_p + (H-1)\boldsymbol{u}_p + \boldsymbol{\varepsilon}_{pq} \\ & \qquad + (T_q^R - T_{pq}^A)\boldsymbol{\gamma}_q - (T_q^p - T_{pq}^A)\boldsymbol{\gamma}_q + \boldsymbol{u}_q\Big] + \overline{\boldsymbol{G}},\end{aligned}$$

which eventually leads to (6.21). This completes the proof of lemma 6.3. □

Note that most improvements can be expressed by substitutions, since rate resp. consonance intervals are responsible for deteriorating accuracy resp. precision intervals. Of particular interest is the composition of Δ, because round beginnings and durations are tied with the maximum consonance, whereas transmission delays are connected with the maximum drift.

The major results for the improved state synchronization are obtained by applying the generic convergence function $\mathcal{CV}_{\mathcal{F}_S}(\cdot)$ from Definition 3.11 to the intervals from Lemma 6.3, which leads by using induction to the following theorem.

Theorem 6.1 (Improved Instantaneous Correction) *Given the algorithm from Definition 6.2 under Assumption 6.1 with the improved compensations* E_p, $\Lambda + \Omega$, *and* Δ *as specified by (6.11), (6.12), and (6.13), respectively. Using the same notation and as in Theorem 3.1, and if for any non-faulty node p the local rate interval* $\boldsymbol{R}_p^{(k_R)} = [\vartheta_p^{-,(k_R)}, 1, \vartheta_p^{+,(k_R)}]$ *is*

[1'] correct at the beginning of rate round $k_R \geq 0$ *with* $\operatorname{align}(\boldsymbol{R}_p^{(k_R)}) \subseteq \boldsymbol{V}_p^{(k_R)} \subseteq \boldsymbol{V}_{\max}$ *as given in Theorem 4.2,*

[2'] $\boldsymbol{\gamma}^{(k_R)}$*-correct w.r.t. internal global rate* $\varphi(t)$ *during rate round* $k_R \geq 0$ *with* $\boldsymbol{\gamma}^{(k_R)} \subseteq \boldsymbol{\gamma}_{\max}$ *as given in Theorem 4.1,*

we can make the following improvements to the former results:

(1) The set $\boldsymbol{\mathcal{B}}_q^{(k_S+1)}$ *of node q's accuracy bounds at the beginning of state round* k_S+1, $k_S \geq 0$, *is defined by* $\boldsymbol{\beta}_q^{p,(k_S+1)} = \boldsymbol{\beta}_p^{(k_S)} + \boldsymbol{\zeta}_q^p$ *with*

$$\begin{aligned}\boldsymbol{\zeta}_q^p \;=\; & \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \mathbf{2}\boldsymbol{G}_A + \boldsymbol{\varepsilon}_{pq} \\ & + (P_S - \Delta - E_p)\left(\boldsymbol{V}_p^{(k_R)} + [0 \pm (\bar{k}_S+1)\sigma_p P_S]\right) \\ & + (E_q + \Delta - \delta_{pq})\left(\boldsymbol{V}_q^{(k_R)} + [0 \pm (\bar{k}_S+1)\sigma_q P_S]\right) \\ & + (\Lambda+\Omega)\Big[-\max\{\vartheta_q^{-,(k_R)} - \vartheta_p^{-,(k_R)} + (\bar{k}_S+1)(\sigma_q-\sigma_p)P_S, 0\}, \\ & \qquad \max\{\vartheta_q^{+,(k_R)} - \vartheta_p^{+,(k_R)} + (\bar{k}_S+1)(\sigma_q-\sigma_p)P_S, 0\}\Big] \\ & + \mathcal{O}(P_S V_{\max} + G + \varepsilon_{\max})\boldsymbol{V}_{\max} \qquad (6.29)\end{aligned}$$

for $p \neq q$ *instead of (3.71), and*

$$\begin{aligned}\boldsymbol{\zeta}_q^q &= \boldsymbol{u}_q + P_S\left(\boldsymbol{V}_q^{(k_R)} + [0 \pm (\bar{k}_S + 1)\sigma_q P_S]\right)\\ &\quad + \mathcal{O}(P_S V_{\max} + G + \varepsilon_{\max})\boldsymbol{V}_{\max} \qquad (6.30)\end{aligned}$$

instead of (3.72), where rate round $k_R = \lfloor k_S/m \rfloor$ *and reduced state round* $\bar{k}_S = k_S \bmod m$.

(2) The set $\boldsymbol{\mathcal{P}}_q$ *of node q's precision bounds at the beginning of each state round is defined by*

$$\begin{aligned}\boldsymbol{\pi}_q^p &= \boldsymbol{\pi}_0 + \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{pq} + (P_S + E_q - E_p - \delta_{pq})\boldsymbol{\gamma}_{\max}\\ &\quad + \mathcal{O}(P_S V_{\max} + G + \varepsilon_{\max})\boldsymbol{\gamma}_{\max} \qquad (6.31)\end{aligned}$$

for $p \neq q$ *instead of (3.73), and*

$$\boldsymbol{\pi}_q^q = \boldsymbol{\pi}_0 + \boldsymbol{u}_q + P_S\boldsymbol{\gamma}_{\max} + \mathcal{O}(P_S V_{\max} + G + \varepsilon_{\max})\boldsymbol{\gamma}_{\max} \qquad (6.32)$$

instead of (3.74).

(3) The set $\boldsymbol{\mathcal{P}}$ *of uniform precision bounds at the beginning of each state round is defined by*

$$\begin{aligned}\boldsymbol{\pi}^p &= \boldsymbol{\pi}_0 + \boldsymbol{u}_p + \boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{\max} + (P_S + E_{\max} - E_p - \delta_{\min})\boldsymbol{\gamma}_{\max}\\ &\quad + \mathcal{O}(P_S V_{\max} + G + \varepsilon_{\max})\boldsymbol{\gamma}_{\max}, \qquad (6.33)\\ \boldsymbol{\pi}^{\overline{q}} &= \boldsymbol{\pi}_0 + \boldsymbol{u}_q + P_S\boldsymbol{\gamma}_{\max} + \mathcal{O}(P_S V_{\max} + G + \varepsilon_{\max})\boldsymbol{\gamma}_{\max} \qquad (6.34)\end{aligned}$$

instead of (3.80) and (3.81), respectively. Moreover,

$$\begin{aligned}\boldsymbol{\pi}^H &= \boldsymbol{\pi}_0 + 2\boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + \boldsymbol{\varepsilon}_{\max} + (P_S + E_{\max} - E_{\min} - \delta_{\min})\boldsymbol{\gamma}_{\max}\\ &\quad + \mathcal{O}(P_S V_{\max} + G + \varepsilon_{\max})\boldsymbol{\gamma}_{\max}, \qquad (6.35)\\ \boldsymbol{\pi}_I &= \boldsymbol{\varepsilon}_{\max} + H\boldsymbol{u}_{\max} + \overline{\boldsymbol{G}} + (\Lambda + \Omega + \Delta + E_{\max} - \delta_{\min})\boldsymbol{\gamma}_{\max}\\ &\quad + \mathcal{O}(P_S V_{\max} + G + \varepsilon_{\max})\boldsymbol{\gamma}_{\max} \qquad (6.36)\end{aligned}$$

instead of (3.82) and (3.83), respectively.

(4) The initial worst case precision satisfies

$$\begin{aligned}\pi_{0,\max} &= ||\boldsymbol{\pi}_0|| + u_{\max} + G + (E_{\max} - E_{\min})\gamma_{\max}\\ &\quad + \mathcal{O}(P_S V_{\max}\gamma_{\max} + G\gamma_{\max} + \varepsilon_{\max}\gamma_{\max}) \qquad (6.37)\end{aligned}$$

instead of (3.78), with $\boldsymbol{\pi}_0$ *being a solution of the equation* $||\boldsymbol{\pi}_0|| = \Psi_\pi(\boldsymbol{\mathcal{P}}, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I)$*, and the worst case precision*

$$\begin{aligned} \pi_{\max} \;=\; & \max\Big\{\pi^- + u^+_{\max} + (E_{\max} - E_{\min})\gamma^+_{\max}, \\ & \qquad \pi^+ + u^-_{\max} + (E_{\max} - E_{\min})\gamma^-_{\max}, \\ & \qquad \pi_0 + u_{\max} + P_S\gamma_{\max}\Big\} \\ & + G + \mathcal{O}(P_S V_{\max}\gamma_{\max} + G\gamma_{\max} + \varepsilon_{\max}\gamma_{\max}) \end{aligned} \tag{6.38}$$

instead of (3.84), with

$$\begin{aligned} \boldsymbol{\pi} \;=\; & \overline{\boldsymbol{\Phi}_\pi}(\boldsymbol{\mathcal{P}}, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I) + \boldsymbol{\pi}_0 + \boldsymbol{u}_{\max} + P_S\boldsymbol{\gamma}_{\max} \\ & + \mathcal{O}(P_S V_{\max} + G + \varepsilon_{\max})\boldsymbol{\gamma}_{\max} \end{aligned} \tag{6.39}$$

instead of (3.58).

(5) The maximum clock state adjustment Υ_q *meets* $\Upsilon_q \in \boldsymbol{\pi}_q \subseteq \boldsymbol{\pi}$*, where*

$$\begin{aligned} \boldsymbol{\pi}_q \;=\; & \overline{\boldsymbol{\Phi}_\pi}(\boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I) + \boldsymbol{\pi}_0 + \boldsymbol{u}_q + P_S\boldsymbol{\gamma}_{\max} \\ & + \mathcal{O}(P_S V_{\max} + G + \varepsilon_{\max})\boldsymbol{\gamma}_{\max} \end{aligned} \tag{6.40}$$

instead of (3.87).

Proof. As in the proof of Lemma 3.12, suppose that at the beginning of state round k_S the set $\boldsymbol{\mathcal{B}}^{(k_S)} = \{\boldsymbol{\beta}_1^{(k_S)}, \ldots, \boldsymbol{\beta}_n^{(k_S)}\}$ contains the bounds for accuracies $\boldsymbol{\alpha}_p^{(k_S)} \subseteq \boldsymbol{\beta}_p^{(k_S)}$, see also (3.50). According to item (2) of Lemma 6.3, the accuracies $\boldsymbol{\alpha}_q^{p,(k_S)}$ of the intervals fed into the convergence function are bounded by

$$\begin{aligned} \boldsymbol{\beta}_q^{p,(k_S+1)} \;=\; & \boldsymbol{\beta}_p^{(k_S)} + \boldsymbol{u}_p + \boldsymbol{u}_q + \overline{\boldsymbol{G}} + \boldsymbol{2G}_A + \boldsymbol{\varepsilon}_{pq} \\ & + (P_S - \Delta - E_p)\,\mathrm{align}(\boldsymbol{R}_p^{(k_S)}) + (E_q + \Delta - \delta_{pq})\,\mathrm{align}(\boldsymbol{R}_q^{(k_S)}) \\ & + (\Lambda + \Omega)\big[-\max\{\vartheta_q^{-,(k_S)} - \vartheta_p^{-,(k_S)}, 0\}, \max\{\vartheta_q^{+,(k_S)} - \vartheta_p^{+,(k_S)}, 0\}\big] \\ & + \mathcal{O}(\pi + P_S R_{\max} + G + \varepsilon_{\max})\,\mathrm{align}(\boldsymbol{R}_q^{(k_S)}) \qquad \text{for } p \neq q, \\ \boldsymbol{\beta}_q^{q,(k_S+1)} \;=\; & \boldsymbol{\beta}_q^{(k_S)} + \boldsymbol{u}_q + P_S\,\mathrm{align}(\boldsymbol{R}_q^{(k_S)}) + \mathcal{O}(\pi)\,\mathrm{align}(\boldsymbol{R}_q^{(k_S)}) \end{aligned}$$

forming set $\boldsymbol{\mathcal{B}}_q^{(k_S+1)}$, where $\boldsymbol{R}_p^{(k_S)} = [\vartheta_p^{-,(k_S)}, 1, \vartheta_p^{+,(k_S)}]$ and $\boldsymbol{R}_q^{(k_S)} = [\vartheta_q^{-,(k_S)}, 1, \vartheta_q^{+,(k_S)}]$ have to be local rate intervals that are correct during state round k_S. Bounds on them can be calculated from precondition [1'], since the enclosing rate round k_R is given by $\lfloor k_S/m \rfloor$ and the reduced state round $\bar{k}_S = k_S \bmod m$ as set out in our algorithm, hence

$$\begin{aligned} \mathrm{align}(\boldsymbol{R}_p^{(k_S)}) \;&\subseteq\; \boldsymbol{V}_p^{(\lfloor k_S/m \rfloor)} + [0 \pm ((k_S \bmod m) + 1)\sigma_p P_S], \\ \mathrm{align}(\boldsymbol{R}_q^{(k_S)}) \;&\subseteq\; \boldsymbol{V}_q^{(\lfloor k_S/m \rfloor)} + [0 \pm ((k_S \bmod m) + 1)\sigma_q P_S]. \end{aligned}$$

This leads to (6.29) resp. (6.30) featuring item (1) of our theorem with the appropriate assistant interval $\boldsymbol{\zeta}_q^p$ resp. $\boldsymbol{\zeta}_q^q$. As in the proof of Lemma 3.12, we can argue here that $\mathcal{O}(\pi) = \mathcal{O}(P_S R_{\max} + G + \varepsilon_{\max})$ with $R_{\max} = V_{\max} = ||\boldsymbol{V}_{\max}||$. Remember that the successive set $\boldsymbol{\mathcal{B}}^{(k_S+1)}$ of accuracy bounds is being computed by means of the accuracy preservation functions $\Phi_\alpha^\pm(\cdot)$; look at (3.69) for the formula.

The precisions bounds of set $\boldsymbol{\mathcal{P}}_q$ and $\boldsymbol{\mathcal{P}}$ carry immediately over from item (3) of Lemma 6.3 with the specialization $\boldsymbol{\gamma}_p = \boldsymbol{\gamma}_q = \boldsymbol{\gamma}_{\max}$. To provide the links, (6.31) comes from (6.16), (6.32) and (6.34) from (6.17), (6.33) from (6.20), (6.35) from (6.19), and (6.36) from (6.21). This finishes already item (2) and (3) of our theorem.

Before deriving the expression of the observable initial worst case precision $\pi_{0,\max}$, the first part of item (4) in our theorem, we look again at the difference of the resynchronization times between any two non-faulty nodes p and q. By recalling the proof of item (5) of Lemma 3.12 and knowing from Lemma 6.2 that the consonance interval $\boldsymbol{\gamma}_{\max}$ deteriorates any precision interval, we easily obtain

$$\begin{aligned} t_q^R - t_p^R &\in T_q^R + \boldsymbol{\pi}_0 + \boldsymbol{u}_q + (P_S + \pi^-)\boldsymbol{\gamma}_{\max} \\ &\quad - T_p^R + \overline{\boldsymbol{\pi}}_0 + \overline{\boldsymbol{u}}_p + (P_S + \pi^-)\overline{\boldsymbol{\gamma}}_{\max} \\ &\in E_q - E_p + [-\pi_0, \pi_0] + \boldsymbol{u}_q + \overline{\boldsymbol{u}}_p + P_S(\boldsymbol{\gamma}_{\max} + \overline{\boldsymbol{\gamma}}_{\max}) \\ &\quad + \mathcal{O}(P_S R_{\max} + G + \varepsilon_{\max})[-\gamma_{\max}, \gamma_{\max}] \qquad (6.41) \\ &\in E_{\max} - E_{\min} + \mathcal{O}(P_S R_{\max} + G + \varepsilon_{\max}). \end{aligned}$$

When grabbing (3.90) from the proof of Theorem 3.1 and using inverse the drift rates $\vartheta_p^{\pm,(k_R)}$ to map logical times to real-times as implicated by Lemma 6.1, then

$$\begin{aligned} \boldsymbol{\pi}_{0,\max} &\subseteq \boldsymbol{\pi}_0 + \bigcup_p \left(\frac{\max_q\{t_q^R\} - t_p^R + u_p^-}{1 - \vartheta_p^{-,(k_R)}} \boldsymbol{\gamma}_{\max} + \boldsymbol{u}_p \right) + \boldsymbol{G\rho} \\ &\subseteq \boldsymbol{\pi}_0 + (E_{\max} - E_{\min})\boldsymbol{\gamma}_{\max} + \boldsymbol{u}_{\max} + \boldsymbol{G} \\ &\quad + \mathcal{O}(P_S R_{\max} + G + \varepsilon_{\max})\boldsymbol{\gamma}_{\max}, \end{aligned}$$

which provides (6.37) by extracting the interval lengths. In order to show the expression of the observable worst case precision $\pi_{\max}$, it is sufficient to redo the proof of item (3) of Theorem 3.1 with the replacement of $\boldsymbol{\rho}_{\max}$ by $\boldsymbol{\gamma}_{\max}$. Without going into details, this leads straightaway to (6.38) when taking $\boldsymbol{\pi}$ for granted at the moment.

It remains to provide a limit for the maximum clock state adjustment Υ_q applied instantaneously at non-faulty node q. Since the expiring virtual clock is $(\boldsymbol{\pi}_0 + \boldsymbol{u}_q + P_S\boldsymbol{\gamma}_{\max} + \mathcal{O}(\pi)\boldsymbol{\gamma}_{\max})$-correct and the new one is $\boldsymbol{\Phi}_\pi(\boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I)$-correct, both w.r.t. the same internal global time, we can conclude that any state adjustment lies within the interval

$$\boldsymbol{\pi}_q = \boldsymbol{\pi}_0 + \boldsymbol{u}_q + P_S\boldsymbol{\gamma}_{\max} + \overline{\boldsymbol{\Phi}_\pi}(\boldsymbol{\mathcal{P}}_q, \boldsymbol{\pi}^H, \boldsymbol{\pi}_I) + \mathcal{O}(P_S R_{\max} + G + \varepsilon_{\max})\boldsymbol{\gamma}_{\max}$$

by recalling Lemma 3.7. A majorization over q leads to $\boldsymbol{\pi}$, which justifies (6.39). This finishes the proof of our theorem. □

In Section 3.5.2 we motivated traditional accuracy $\aleph$, but in the course of merging both frameworks, statements (3.88) and hence (3.89) become meaningless, because between resynchronization instants internal global time does not progress as real-time any more, see Definition 6.1. This deficiency is of no particular loss, since we have explicit representations about drift and consonance anyhow.

So far we have studied how rate synchronization supports state synchronization, but not the other way around. By construction of the CRA from Definition 4.8 and hence the embedded one of Definition 6.2, only a proper round structure is necessary. More specifically, near the end of each rate round an FME and all computations have to be carried out before the next resynchronization instant takes place. This can be easily achieved by replacing the rather cumbersome parameters F and E from Lemma 4.11 with our well defined compensations $\Lambda + \Omega$, Δ and $E_{\max}$. Also the considerations about the round beginnings in Section 4.5.1 are obsolete. As a consequence, the algorithm of Definition 4.8 has already implemented $F = \Lambda + \Omega + \Delta$ and $E = E_{\max}$.

A cycle of dependencies becomes evident by observing that in the view of the CRA, the "incoming" quantities $E_{\max}$, Λ, Ω, Δ, and $\pi_{\max}$ rely on the "outgoing" quantities $\gamma_{\max}$ and $R_{\max}$. For the respective equations consult (6.11), (6.12), (6.13), and (6.38). However, by inspecting the analysis in Section 4.5 once again, we find out that there influence is restricted in the following manner:

1. Since the (rate) resynchronization times at different non-faulty nodes are assumed to be very close to each other, the most dominant occurrence of $\pi_{\max}$ is only in conjunction with $\sigma_{\max}$, e.g., see (4.68) for rate intervals and (4.69) for consonance intervals. This can be neglected, because $\sigma_{\max}\pi_{\max}$ is supposed to be much smaller than $\gamma_{\max}$ and $R_{\max}$.
2. In our new setting the maximum logical broadcast delay $B = \Lambda + \Omega$, which appears importantly only in terms $\sigma_{\max}B$ and $\frac{\varepsilon_{\max}}{P_R - B}$, see Theorem 4.1 for example. Using (6.12) and applying asymptotic approximations reveals that Λ and Ω depend on $R_{\max}$ just by factor $\sigma_{\max}$ and $\varepsilon_{\max}P_R^{-2}$, which can be neglected as well.

3. The compensations $F+E=\Lambda+\Omega+\Delta+E_{\max}$ show up in terms $\sigma_{\max}(F+E)$ and $\frac{\varepsilon_{\max}}{P_R-(F+E)R_{\max}}$, see Theorem 4.2 for example. By an analogous derivation, factors $\sigma_{\max}$ and $\varepsilon_{\max}P_R^{-2}R_{\max}$ express the influence of $R_{\max}$ on Λ, Ω, Δ, and $E_{\max}$, whose contributions are also negligible.

In summary, the cycle of dependencies can be broken, when

$$\sigma_{\max}\pi_{\max} \ll \gamma_{\max} \tag{6.42}$$

and

$$\max\left\{\sigma_{\max}, \frac{\varepsilon_{\max}}{P_R^2}\right\}(\Lambda+\Omega+\Delta+E_{\max})R_{\max} \ll \gamma_{\max}. \tag{6.43}$$

The satisfaction of these conditions is also the reason, why we are allowed to separate the analysis of state and rate synchronization. Otherwise the inevitable cross connections force us to deal with recursions, which would render the analysis more complicated.

6.3.4 Numerical Example

In order to become more familiarized with the performance of our complete synchronization algorithm from Definition 6.2, we give an example with decently realistic numbers. Figure 6.3 shows a graphical summary of our assumptions (three rectangles) and calculations (two rounded boxes). In the course of doing this exercise, we try to discover which parameters are important and which one can be ignored.

The 16 nodes of our representative distributed system communicate via a broadcast channel (e.g., Ethernet) specified by a large broadcast latency $\lambda_{\max}$, an average transmission delay $\delta_{\max}$, and a small uncertainty $\boldsymbol{\varepsilon}_{\max}$ due to some sophisticated timestamping facilities (e.g., our NTI M-Module). The processors are supposed to have generous execution times $\eta_{\min}$ and $\eta_{\max}$, and the clocks (e.g., our UTCSU-ASIC) are driven by 2^{24} Hz oscillators with a "high" stability $\sigma_{\max}$. Since the presumed maximal oscillator drift $\rho_{\max}$ is insignificant, we obtain compensations Λ, Ω, Δ, $E_{\min}$ and $E_{\min}$ in a straightforward way as depicted in Figure 6.3.

For clock rate synchronization the convergence function $\boldsymbol{\mathcal{OP}}_{n-d-e}^{\boldsymbol{\gamma}^o,\boldsymbol{\gamma}_H,\Theta_{\max}}(\cdot)$ from Section 5.3.2 is used with 1 minute as resynchronization period, since (5.105) suggests 49.1 seconds. The initial worst case consonance $\gamma_{0,\max}$ at the beginning of each rate round can be calculated from (5.99), where $F\approx B\approx\Lambda$, $E=E_{\max}$, and $\epsilon_{max}+4(\Lambda+E_{\max})\rho_{max}\ll\Lambda$. With the same simplifications, the overall worst case consonance interval $\boldsymbol{\gamma}_{\max}$ falls out from (5.104), and the maximal rate adjustment $\boldsymbol{\Theta}_{\max}$ from (5.101). The bounds

Clocks

$\sigma_{\max} = 10^{-4}$ ppm/s

$G = 60$ ns

$\boldsymbol{u}_{\max} = [-60 \text{ ns}, 60 \text{ ns}]$

$\rho_{\max} = 1$ ppm

Network

$\varepsilon_{\max} = [-120 \text{ ns}, 240 \text{ ns}]$

$\lambda_{\max} \approx \Lambda = 100$ ms

$\omega_{\max} \approx \Omega = 0$ s

$\delta_{\max} \approx \Delta = 50$ μs

Processors

$n = 16$

$e = d = 2$

$\eta_{\max} \approx E_{\max} = 10$ ms

$\eta_{\min} \approx E_{\min} = 2$ ms

OP-RATE

$P_R = 60$ s

$$\gamma_{0,\max} \approx \sigma_{\max}\left(4P_R + 14\Lambda + 8E_{\max}\right) + \tfrac{4\epsilon_{\max}}{P_R - \Lambda} = 0.0482 \ \mu\text{s/s}$$

$$\boldsymbol{\gamma}_{\max} \approx \left[0 \pm \left(\sigma_{\max}\left(3P_R + 7\Lambda + 4E_{\max}\right) + \tfrac{2\epsilon_{\max}}{P_R - \Lambda}\right)\right] = [0 \pm 0.0301 \ \mu\text{s/s}]$$

$$\boldsymbol{\Theta}_{\max} = [1 \pm 2\sigma_{\max} P_R] = [1 \pm 0.012 \text{ ppm}]$$

$$\boldsymbol{V}_p^{(k_R)} \approx \boldsymbol{V}_p^{(k_R - 1)} + [0 \pm 3\sigma_{\max} P_R] = \boldsymbol{V}_p^{(0)} + k_R\,[0 \pm 0.018 \ \mu\text{s/s}]$$

OP-STATE

$P_S = 10$ s

$$\pi_{0,\max} \approx 2\varepsilon_{\max} + 4u_{\max} + 3G + \left(P_S + \Lambda + 3E_{\max} - 2E_{\min}\right)\gamma_{\max} = 1.99 \ \mu\text{s}$$

$$\pi_{\max} \approx 2\varepsilon_{\max} + \tfrac{9}{2}u_{\max} + 3G + \left(2P_S + \Lambda + \tfrac{5E_{\max} - 3E_{\min}}{2}\right)\gamma_{\max} = 2.65 \ \mu\text{s}$$

$$\Upsilon_{\max} = P_S \gamma_{\max} + u_{\max} = 0.72 \ \mu\text{s}$$

$$\boldsymbol{\beta}_p^{(k_S)} \approx \boldsymbol{\beta}_p^{(0)} + k_S\left(10\boldsymbol{V}_p^{(0)} + [0 \pm 0.842 \ \mu\text{s}]\right) + k_S \lfloor (k_S - 1)/6 \rfloor [0 \pm 0.09 \ \mu\text{s}]$$

Figure 6.3: *Numerical Example*

$\boldsymbol{V}_p^{(k_R)}$ for the rate interval at local node p at the beginning of rate round k_R evolve according to (5.107), and making the recursion explicit is an easy task. It should be obvious that our consonance results are at least one order of magnitude better than the commonly used drift $\rho_{\max}$. Recalling (5.106), we get the remarkable outcome that $\gamma^*_{\max} = \sqrt{96\ \sigma_{\max}\varepsilon_{\max}} = 0.0588\ \mu s/s$.

For clock state synchronization the convergence function $\mathcal{OP}_{n-d-e}^{\boldsymbol{\pi}^o,\boldsymbol{\pi}^H,\Upsilon_{\max}}(\cdot)$ from Section 5.3.3 is used with a resynchronization period of 10 seconds, hence $m = 6$. The formula for the initial worst case precision $\pi_{0,\max}$ at the beginning of each state round has its origin in (5.66) and (5.69) enhanced by item (4) of Theorem 6.1 to insert $\gamma_{\max}$. By the same token, the overall worst case precision $\pi_{\max}$ can be derived from (5.70). Item (5) of Theorem 6.1 applied on (5.65) provides the maximum clock state adjustment $\Upsilon_{\max}$. Last but not least, the accuracy bounds $\boldsymbol{\beta}_p^{(k_S)}$ from (5.80)/(5.83) combined with item (1) of Theorem 6.1 yields the recursion

$$\boldsymbol{\beta}_p^{(k_S)} \subseteq \boldsymbol{\beta}_p^{(k_S-1)} + P_S\left(\boldsymbol{V}_p^{(\lfloor (k_S-1)/m \rfloor)} + [0 \pm \sigma_{\max} P_R]\right) + \boldsymbol{u}_{\max} + [0 \pm \Upsilon_{\max}].$$

To bring it in a closed form, we sum up the whole sequence, ending up with

$$\begin{aligned}
\boldsymbol{\beta}_p^{(k_S)} &\subseteq \boldsymbol{\beta}_p^{(0)} + \sum_{k=0}^{k_S-1}\left(P_S \boldsymbol{V}_p^{(\lfloor k/m \rfloor)} + [0 \pm \sigma_{\max} P_R P_S] + \boldsymbol{u}_{\max} + [0 \pm \Upsilon_{\max}]\right) \\
&\subseteq \boldsymbol{\beta}_p^{(0)} + k_S\left(\boldsymbol{u}_{\max} + [0 \pm (\sigma_{\max} P_R P_S + \Upsilon_{\max})]\right) \\
&\quad + P_S \sum_{k=0}^{k_S-1}\left(\boldsymbol{V}_p^{(0)} + \lfloor k/m \rfloor [0 \pm 3\sigma_{\max} P_R]\right) \\
&\subseteq \boldsymbol{\beta}_p^{(0)} + k_S\left(P_S \boldsymbol{V}_p^{(0)} + \boldsymbol{u}_{\max} + [0 \pm (\sigma_{\max} P_R P_S + \Upsilon_{\max})]\right) \\
&\quad + P_S[0 \pm 3\sigma_{\max} P_R] \sum_{k=0}^{k_S-1} \lfloor k/m \rfloor \\
&\subseteq \boldsymbol{\beta}_p^{(0)} + k_S\left(P_S \boldsymbol{V}_p^{(0)} + \boldsymbol{u}_{\max} + [0 \pm (\sigma_{\max} P_R P_S + \Upsilon_{\max})]\right) \\
&\quad + \frac{k_S}{2} \lfloor (k_S - 1)/m \rfloor \, [0 \pm 3\sigma_{\max} P_R P_S],
\end{aligned}$$

where in the last step we made use of the fact

$$\sum_{k=0}^{n} \lfloor k/m \rfloor = \frac{\lfloor n/m \rfloor (n - m + 2 + n \bmod m)}{2} \leq \frac{\lfloor n/m \rfloor (n+1)}{2}$$

by taking advise from [14]. As expected, the growth of $\boldsymbol{\beta}_p^{(k_S)}$ depends linearly and (weakly) quadratically on the state round k_S. All these results describe the worst case behavior.

6.4 Extensions

Despite the wealth of material presented in this thesis about interval-based clock state and rate synchronization, many issues still need to be explored. In the following we give a list of them and include a sketch of our ideas and further directions.

- *Hardware development:* For a specific application, we need to tailor the hardware support for clock synchronization, since our prototypes are only designed for research purposes. Depending on the requirments and the surrounding system, this touches the UTCSU (e.g., higher frequency, more timestamping features), the NTI (e.g., different interfaces, better oscillators), and the COMCO (e.g., other trigger mechanisms, drivers), see Chapter 2.

- *Validation:* An important open problem is the integration of accuracy and rate intervals from primary nodes to accomplish external clock synchronization, see Section 3.1 and 4.4.5. Our present research indicates that three major subproblems need to be addressed: a realistic assumption about the faultiness of GPS receivers, the rate/state coupling (including a history) between primary nodes and GPS receivers, and a definition of various operating modes.

- *Initial synchronization/joining:* Our algorithms work on a periodic basis and require some initial conditions, see Section 3.4 or 4.4. If primary nodes are working, the latter problem can be viewed as a special case of validation (infinitely long flywheel period), otherwise the system has to run through several phases to achieve at least an internal synchronization.

- *Multiple SSNs:* We just covered system architectures of one synchronization subnet, but large scale systems have several of them interconnected by gateway nodes, see Section 2.2. To provide a synchronization across all SSNs, gateway nodes act dynamically as primary nodes, hence this problem is related with clock validation.

- *Tradeoff between precision&accuracy:* There are several hints that precision and accuracy cannot be minimal at the same time, see Section 5.3.2. In a more general framework, it is challenging to prove either such an impossibility result or to present an optimal solution including its complexity.

- *Average case analysis:* The analysis of our algorithms were geared towards the worst case, see Section 3.5 or 4.5. However, we are convinced that average case results are significantly smaller than worst case one. To attack this problem, we need to set up assumptions with probabilistic elements that allow us to reason about the distribution or at least the expectation of precision and accuracy values.

- *Convergence functions:* We presented only one interval-based convergence function, namely $\boldsymbol{\mathcal{OP}}(\cdot)$ in Section 5.3. There are many alternatives around, in particular, we have a *fault-tolerant intersection function* $\boldsymbol{\mathcal{FTI}}(\cdot)$ in mind that computes its edges by making an "inside-out" sweep over the set of source intervals. Due to our generic framworks, only the associated characteristic functions have to be provided.

- *Continuous amortization:* A decrease/increase of the clock rate during some period is the idea behind continuous amortization to adjust the clock state, see Section 3.5.3. This technique should be studied in the rate framework as well, in order to get refinements.
- *Optimal consonance:* In Chapter 4 we introduced a framework for clock rate synchronization, but did not prove lower bounds on the achievable consonance in the sense of [29]. Nevertheless, we conjecture that they are of the form $\Omega(\sqrt{\sigma_{\max}\epsilon_{\max}})$.
- *Oscillator parameters:* The stability of oscillators has a large impact on the rate and hence state synchronization, see Section 6.3.4. Therfore it is important to measure the drift and stability as good as possible, since so far we just used some estimations from datasheets. For an improvement, it would be interesting to find out, if there is something like a "precision of stabilities", i.e., $|\sigma_p - \sigma_q| \leq \sigma^* \; \forall p, q$.
- *Simulation:* With the help of our simulation tools, we are about to make experiments with different system parameters, convergence functions, and fault scenarios. They are helpful to foster the understanding of the clock synchronization problem with all its ramifications.
- *Evaluation:* The only way to test if our algorithms are working properly or not, is to have them implemented and executed in a real distributed system, see Section 2.3.4. A careful specification and documentation of the runs can not be overestimated for a rigorous evaluation.

◇

List of Symbols

a, a'	number of unbounded accuracy faults
$\boldsymbol{A}, \boldsymbol{A}(t), \boldsymbol{A}_p(t)$	accuracy intervals
$\boldsymbol{\mathcal{A}}$	ordered set of accuracy intervals
$\boldsymbol{\alpha} = [-\alpha^-, \alpha^+]$	interval of accuracies
$\alpha = \alpha^- + \alpha^+$	length of $\boldsymbol{\alpha}$
$\boldsymbol{\alpha}_p(t) = [-\alpha_p^-(t), \alpha_p^+(t)]$	interval of accuracies at node p
$\boldsymbol{\alpha}_p^0 = [-\alpha_p^{0-}, \alpha_p^{0+}]$	initial accuracies at node p
$\aleph, \aleph^{(k)}$	traditional accuracy (at the beginning of round k)
b	data capacity of packets
B	maximum logical broadcast delay
$\boldsymbol{\beta}_p^{(k)}$	accuracy bound for node p at the beginning of round k
$\boldsymbol{\mathcal{B}}^{(k)}, \boldsymbol{\mathcal{B}}_p^{(k)}$	set of accuracy bounds at the beginning of round k
$C(t), C_p(t)$	function of clock (at node p)
$c_p(T)$	function of inverse clock at node p
$\boldsymbol{C}_p(t) = [C_p(t) \pm \boldsymbol{\alpha}_p(t)]$	function of local interval clock of node p
$\boldsymbol{\mathcal{C}}$	set of interval clocks
$\boldsymbol{\mathcal{CV}}_{\mathcal{F}}(\cdot)$	generic convergence function w.r.t. fault model $\mathcal{F}$
d, d'	number of restricted faults
δ_p	drift of clock $\mathcal{C}_p$
δ_{pq}	deterministic transmission delay from node p to q
$\delta_{\max}, \delta_{\min}$	maximum/minimum deterministic transmission delay
Δ	transmission delay compensation
e, e'	number of arbitrary faults
$\boldsymbol{\varepsilon}_{pq} = [-\varepsilon_{pq}^-, \varepsilon_{pq}^+]$	transmission delay uncertainty from node p to q
$\varepsilon_{pq} = \varepsilon_{pq}^- + \varepsilon_{pq}^+$	length of $\boldsymbol{\varepsilon}_{pq}$
$\boldsymbol{\varepsilon}_{\max}$	maximum transmission delay uncertainty
$\varepsilon_{\max}$	length of interval $\boldsymbol{\varepsilon}_{\max}$
η_p	execution time bound for node p
$\eta_{\max}, \eta_{\min}$	maximum/minimum execution time
E_p	logical execution time for node p
$E_{\max}, E_{\min}$	maximum/minimum logical execution time
f, f'	number of faults

f_p	nominal frequency of oscillator $\mathcal{O}_p$
$f_p(t)$	instantaneous frequency of oscillator $\mathcal{O}_p$
$\boldsymbol{f}(\cdot)$	generic interval-based function
F	FME duration
$\mathcal{F}, \mathcal{F}_R, \mathcal{F}_S$	abstract fault model for rate/state framework
G	clock granularity
$\boldsymbol{G} = [0, G]$	positive granularity interval
$\boldsymbol{G_\rho} = [0, G(1 + \rho^+_{\max})]$	positive granularity interval with drift
$\overline{\boldsymbol{G}} = [-G, 0]$	negative granularity interval
G_S	clock setting granularity
G_A	accuracy transmission loss
$\boldsymbol{2G}_A = [-G_A, G_A]$	accuracy transmission loss interval
$G_{\max}$	uniform error term for rate
$\boldsymbol{\gamma} = [\gamma^-, 0, \gamma^+]$	consonance interval
$\boldsymbol{\gamma}^{(k)}$	consonance interval at the beginning of round k
$\boldsymbol{\gamma}_H$	maximum consonance interval of exchanged intervals
$\boldsymbol{\gamma}_I$	maximum consonance interval of perceptions
$\boldsymbol{\gamma}_{p,q}$	consonance interval from node q received at p
$\gamma_{\max}$	consonance of an ensemble of clocks
$H \in \{1, 2\}$	indicator of pure ($H = 1$) broadcast network
$I_{t \neq \theta}$	indicator function of non-synchrony
$\boldsymbol{I} = [x, r, y]$	asymmetric interval
$\overline{\boldsymbol{I}} = [y, r, x]$	asymmetric interval with exchanged lengths
$\hat{\boldsymbol{I}}$	associated precision interval
$\vec{\boldsymbol{I}}$	mixed interval
$\|\|\boldsymbol{I}\|\| = x + y$	length of asymmetric interval
$\boldsymbol{I}^p_q$	interval from node p received at q
$\boldsymbol{\mathcal{I}}$	ordered set of intervals
ι^-_s, ι^+_s	common intersection of precision intervals
k, k_R, k_S	rate/state round number
$\lambda_{\max}$	maximum broadcast latency
Λ	logical broadcast latency
m	ratio between rate and state resynchronization period
$\boldsymbol{\mathcal{M}}(\cdot)$	Marzullo's function
n	total number of nodes
$\mathcal{N}', \mathcal{N}''$	set of primary/non-primary nodes

$\mathcal{N}^{(k)}$	set of non-faulty nodes during round k
o, o'	number of omission faults
$\mathcal{O}(\cdot)$	asymptotic upper bound function
$\boldsymbol{\mathcal{OP}}(\cdot)$	optimal precision convergence function
$\omega_{\max}$	maximum broadcast operation delay
Ω	logical broadcast operation delay
P, P_R, P_S	rate/state resynchronization period
$\boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{P}}_p$	set of precision bounds (at node p)
$\varphi(t)$, $\varphi^{(k)}$	internal global rate (of round k)
$\Phi^{\pm}_{\alpha}(\cdot)$	accuracy preservation function
$\Phi^{\pm}_{\delta}(\cdot)$	drift preservation functions
$\boldsymbol{\Phi}_{\gamma}(\cdot)$	consonance preservation function
$\boldsymbol{\Phi}_{\pi}(\cdot)$	precision preservation function
$\boldsymbol{\pi} = [-\pi^-, \pi^+]$	interval of precision
$\pi = \pi^- + \pi^+$	length of $\boldsymbol{\pi}$
$\pi_{\max}$	precision of an ensemble of clocks
$\boldsymbol{\pi}_0 = [-\pi_0^-, \pi_0^+]$	ideal initial precision interval
$\boldsymbol{\pi}^H = [-\pi^{H-}, \pi^{H+}]$	maximum precision interval of exchanged intervals
$\boldsymbol{\pi}_I$	maximum precision interval of perceptions
$\boldsymbol{\pi}^p$	precision interval of perceptions at node p
$\boldsymbol{\pi}^p_q$	precision interval from node p received at q
ψ	amortization rate deviation
$\Psi_{\gamma}(\cdot)$	consonance enhancement function
$\Psi^{\pm}_{\iota}(\cdot)$	intersection enhancement functions
$\Psi_{\pi}(\cdot)$	precision enhancement function
$\boldsymbol{Q}_{p,q}$	quotient rate interval of clock $\mathcal{C}_p$ against $\mathcal{C}_q$
r, r_p	reference point (at node p)
r_p^{-1}	inverse rate of clock $\mathcal{C}_p$
$\boldsymbol{R}_p$	local rate interval of clock $\mathcal{C}_p$
$\boldsymbol{R}_{\max}$	maximum local rate interval
$\boldsymbol{R}_{p,q}$	remote rate interval from node q received at p
$\boldsymbol{\mathcal{R}}$	set of rate intervals
$\boldsymbol{\rho}_p = [-\rho_p^-, \rho_p^+]$	inverse rate deviation at node p
$\rho_p = \rho_p^- + \rho_p^+$	length of $\boldsymbol{\rho}_p$
$\boldsymbol{\rho}_{\max} = [-\rho^-_{\max}, \rho^+_{\max}]$	maximum inverse rate deviation bound
$\rho_{\max} = \rho^-_{\max} + \rho^+_{\max}$	length of $\boldsymbol{\rho}_{\max}$

$\boldsymbol{\rho}_{osc}$	intrinsic inverse rate deviation of the oscillator $\mathcal{O}_p$
S_p	coupling factor between oscillator and clock at node p
σ_p	oscillator stability of oscillator $\mathcal{O}_p$
$\sigma_{\max}$	maximum oscillator stability
$t, \ldots$	real-time(s) in a Newtonian frame
t_0	real-time at initialization
$t^{(k)}$	begin of round k
Δt	real-time duration
$T, \ldots$	logical time(s)
T_{amort}	amortization duration
$\mathcal{T}$	non-empty real-time interval
ΔT	observable duration
$\tau(t)$, $\tau^{(k)}$	internal global time (of round k)
θ	real-time when oscillator tick occurs
Θ	logical time when oscillator tick occurs
$\vartheta_p^-, \vartheta_p^+$	inverse rate drift of clock $\mathcal{C}_p$
$\boldsymbol{\Theta}_p$	rate adjustment of clock $\mathcal{C}_p$
$\boldsymbol{\Theta}_{\max}$	maximum rate adjustment
U_p	accumulated state adjustments of clock $\mathcal{C}_p$
$\boldsymbol{u}_p = [-u_p^-, u_p^+]$	rate adjustment uncertainty at node p
$\boldsymbol{u}_{\max} = [-u_{\max}^-, u_{\max}^+]$	maximum rate adjustment uncertainty
$v_p(t)$	instantaneous rate of clock $\mathcal{C}_p$
$\boldsymbol{V}_p^{(k)}$	rate interval bound for node p at the beginning of round k
$\boldsymbol{\mathcal{VAL}}(\cdot)$	generic validation function w.r.t. fault model $\mathcal{F}$
Υ_p	state adjustment of clock $\mathcal{C}_p$
$\Upsilon_{\max}$	maximum state adjustment
$\mathbf{X}_{p,q}$	remote rate interval from primary node q received at p
$\boldsymbol{\mathcal{X}}_p^{(k)}$	set of remote rate intervals at node p for round k

Bibliography

[1] D. W. Allan. Time and frequency characterization, estimation, and prediction of precision clocks and oscillator. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control (UFFC)*, 34(6):295–327, November 1987.

[2] M. H. Azadmanesh and R. M. Kieckhafer. New hybrid fault models for asynchronous approximate agreement. *IEEE Transactions on Computers*, 45(4), 1996.

[3] R. R. Brooks and S. S. Iyengar. Robust distributed computing and sensing algorithms. *Computer*, 29(6):53–60, June 1996.

[4] F. Cristian. Probabilistic clock synchronization. *Distributed Computing*, 3(3):146–158, 1989.

[5] P. H. Dana. Global Positioning System (GPS) time dissemination for real-time applications. *Real-Time Systems*, 12(1):9–40, January 1997.

[6] D. Dolev, J. Y. Halpern, and H. R. Strong. On the possibility and impossibility of achieving clock synchronization. *Journal of Computer and System Sciences*, 32(2):230–250, April 1986.

[7] D. Dolev, R. Reischuk, and H. R. Strong. Observable clock synchronization. In *Proceedings of the 14th Annual ACM Symposium on Principles on Distributed Computing (PODC)*, pages 284–293, New York, NY, 21–24 August 1994.

[8] R. Drummond and Ö. Babaoğlu. Low-cost clock synchronization. *Distributed Computing*, 6:193–203, 1993.

[9] C. Fetzer and F. Cristian. Lower bounds for function based clock synchronization. In *Proceedings of the 14th ACM Symposium on Principles of Distributed Computing (PODC)*, pages 137–143, Ottawa, Canada, August 1995.

[10] C. Fetzer and F. Cristian. An optimal internal clock synchronization algorithm. In *Proceedings of the 10th Annual IEEE Conference on Computer Assurance*, pages 187–196, Gaithersburg, MD, June 1995.

[11] C. Fetzer and F. Cristian. Integrating external and internal clock synchronization. *Real-Time Systems*, 12(2):123–171, March 1997.

[12] G. J. Geier, T. M. King, H. L. Kennedy, and R. D. Thomas. Prediction of the time accuracy and integrity of GPS timing. In *Proceeding of the 49th IEEE International Frequency Control Symposium*, pages 266–274, San Francisco, CA, 1995.

[13] M. Gergeleit and H. Streich. Implementing a distributed high-resolution real-time clock using the can-bus. In *Proceeding of the 1st International CAN-Conference*, September 1994.

[14] R. L. Graham, D. E. Knuth, and O. Patashnik. *Concrete Mathematics: A Foundation for Computer Science.* Addison-Wesley, Reading, MA, 1989.

[15] W.A. Halang and M. Wannemacher. High accuracy concurrent event processing in hard real-time systems. *Real-Time Systems*, 12(1):77–94, January 1997.

[16] D. Höchtl and U. Schmid. Long-term evaluation of GPS timing receiver failures. In *Proceedings of the 29th Precise Time and Time Interval (PTTI) Systems and Application Meeting*, Long Beach, CA, 2–4 December 1997.

[17] M. Horauer. *Entwicklung einer Network Timestamp Unit für einen Versatile Timing Analyzer zum Monitoring von verteilten Echtzeitsystemen.* Diploma thesis, Department of Computer Technology, Vienna University of Technology, September 1994.

[18] M. Horauer and D. Loy. Adder synthesis. In *Proceedings of Austrochip '95*, Graz, Austria, 1995.

[19] M. Horauer, D. Loy, and U. Schmid. NTI functional and architectural specification. Technical Report 183/1-69, Department of Automation, Vienna University of Technology, December 1996.

[20] M. Horauer, U. Schmid, and K. Schossmaier. NTI: A network time interface m-module for high-accuracy clock synchronization. Technical Report 183/1-76, Department of Automation, Vienna University of Technology, January 1997.

[21] M. Horauer, U. Schmid, and K. Schossmaier. NTI: A network time interface m-module for high-accuracy clock synchronization. In *Proceedings of the 6th International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS)*, Orlando, FL, March 30 – April 3 1998.

[22] H. Kopetz. Clock Synchronization Unit (CSU) datasheet. Technical Report 22/89, Vienna University of Technology, 1989.

[23] H. Kopetz, A. Krüger, D. Millinger, and A. Schedl. A synchronization strategy for a time-triggered multicluster real-time system. In *Proceeding of the 14th IEEE Symposium on Reliable Distributed Systems (SRDS)*, pages 154–161, Bad Neuenahr, Germany, 13–15 September 1995.

[24] H. Kopetz and W. Ochsenreiter. Clock synchronization in distributed real-time systems. *IEEE Transactions on Computers*, C-36(8):933–940, August 1987.

[25] L. Lamport. Synchronizing time servers. Technical Report 18, Digital System Research Center, June 1987.

[26] R. Lichtenecker. Terrestrial time signal dissemination. *Real-Time Systems*, 12(1):41–61, Jannuary 1997.

[27] B. Liskov. Practical uses of synchronized clocks in distributed systems. *Distributed Computing*, 6(4):211–219, 1993.

[28] D. Loy. *GPS-Linked High Accuracy NTP Time Processor for Distributed Fault-Tolerant Real-Time Systems.* PhD thesis, Faculty of Electrical Engineering, Vienna University of Technology, April 1996.

[29] J. Lundelius-Welch and N. Lynch. An upper and lower bound for clock synchronization. *Information and Control*, 62(2/3):190–204, 1984.

[30] J. Lundelius-Welch and N. Lynch. A new fault-tolerant algorithm for clock synchronization. *Information and Computation*, 77(1):1–36, 1988.

[31] N. A. Lynch. *Distributed Algorithms.* Morgan Kaufmann, San Mateo, CA, 1996.

[32] S. R. Mahaney and F. B. Schneider. Inexact agreement: Accuracy, precision, and graceful degradation. In *Proceedings of the 4th Annual ACM Symposium on Principles on Distributed Computing (PODC)*, pages 237–249, Minaki, Canada, 5–7 August 1985.

[33] T. Mandl. *Entwicklung und Integration des Network Time Interface MA-Moduls in VME-Bus basierende Prozessorboards und ins Echtzeit-Betriebssystem pSOS+m.* Diploma thesis, Department of Automation, Vienna University of Technology, to appear 1998.

[34] Manufacturers and Users of M-Modules (MUMM) e.V. *M-Module Specification*, April 1996.

[35] K. Marzullo. *Maintaining the time in a distributed system. An example of a loosely-coupled distributed service.* PhD thesis, Department of Electrical Engineering, Stanford University, February 1984.

[36] K. Marzullo. Tolerating failures of continuous-valued sensors. *ACM Transactions on Computer Systems*, 8(4):284–304, November 1990.

[37] K. Marzullo and S. Owicki. Maintaining the time in a distributed system. *ACM Operating Systems Review*, 19(3):44–54, July 1985.

[38] D. L. Mills. Internet time synchronization: The network time protocol. *IEEE Transactions on Communications*, 39(10):1482–1493, October 1991.

[39] D. L. Mills. Modelling and analysis of computer network clocks. Technical Report 92-5-2, Electrical Engineering Department, University of Delaware, May 1992.

[40] D. L. Mills. Improved algorithms for synchronizing computer network clocks. *IEEE Transactions on Networking*, 3:245–254, June 1995.

[41] MOTOROLA, Prentice Hall. *MC68030 Enhanced 32-Bit Microprocessor User's Manual*, prentice hall edition, 1990.

[42] W. Ochsenreiter. *Fehlertolerante Uhrensynchronisation in verteilten Realzeitsystemen.* PhD thesis, Faculty of Technical and Natural Sciences, Vienna University of Technology, 1987. in German.

[43] OSF. *Introduction to OSF DCE.* Prentice Hall, Englewood Cliffs, NJ, 1992.

[44] B. Patt-Shamir and S. Rajsbaum. A theory of clock synchronization. In *Proceeding of the 26th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 810–819, Montréal, Canada, 23–25 May 1994.

[45] P. Ramanathan, D. D. Kandlur, and K. G. Shin. Hardware-assisted software clock synchronization for homogeneous distributed systems. *IEEE Transactions on Computers*, 39(4):514–524, April 1990.

[46] P. Ramanathan, K. G. Shin, and R. W. Butler. Fault-tolerant clock synchronization in distributed systems. *IEEE Computer*, 23(10):33–42, October 1990.

[47] G. Richter. *Drivers for Real-Time Communications Coprocessors.* Diploma thesis, Department of Automation, Vienna University of Technology, June 1997.

[48] A. Schedl. *Design and Simulation of Clock Synchronization in Distributed Systems.* PhD thesis, Faculty of Technical and Natural Sciences, Vienna University of Technology, April 1996.

[49] U. Schmid. Monitoring distributed real-time systems. *Real-Time Systems*, 7:33–56, 1994.

[50] U. Schmid. Synchronized Universal Time Coordinated for distributed real-time systems. *Control Engineering Practice*, 3(6):877–884, 1995.

[51] U. Schmid. Interval-based clock synchronization with optimal precision. Technical Report 183/1-78, Department of Automation, Vienna University of Technology, December 1997.

[52] U. Schmid. Orthogonal accuracy clock synchronization. Technical Report 183/1-77, Department of Automation, Vienna University of Technology, March 1997.

[53] U. Schmid. Special issue on the challenge of global time in large-scale distributed real-time systems. *Real-Time Systems*, 12(1–3), 1997.

[54] U. Schmid and K. Schossmaier. Interval-based clock state synchronization revisited. Technical Report 183/1-80, Department of Automation, Vienna University of Technology, July 1997.

[55] U. Schmid and K. Schossmaier. Interval-based clock synchronization. *Real-Time Systems*, 12(2):173–228, March 1997.

[56] F. Schmuck and F. Christian. Continuous amortization need not affect the precision of a clock synchronization algorithm. In *Proceedings of the 9th Annual ACM Symposium on Principles on Distributed Computing (PODC)*, pages 133–144, Quebec City, Canada, August 1990.

[57] F. B. Schneider. A paradigm for reliable clock synchronization. In *Proceedings of Advanced Seminar of Local Area Networks*, pages 85–104, Bandol, France, April 1986.

[58] F. B. Schneider. Understanding protocols for byzantine clock synchronization. Technical Report 87-859, Department of Computer Science, Cornell University, August 1987.

[59] K. Schossmaier. Understanding interval-based clock rate synchronization algorithms. Technical Report 183/1-70, Department of Automation, Vienna University of Technology, December 1996.

[60] K. Schossmaier. An interval-based framework for clock rate synchronization. In *Proceedings of the 16th Annual ACM Symposium on Principles on Distributed Computing (PODC)*, pages 169–178, Santa Barbara, CA, 21–24 June 1997.

[61] K. Schossmaier and D. Loy. An ASIC supporting external clock synchronization for distributed real-time systems. In *Proceedings of the 8th Euromicro Workshop on Real-Time Systems*, pages 277–282, L'Aquila, Italy, 21–24 June 1996.

[62] K. Schossmaier and U. Schmid. UTCSU functional specification. Technical Report 183/1-56, Department of Automation, Vienna University of Technology, July 1995.

[63] K. Schossmaier, U. Schmid, M. Horauer, and D. Loy. Specification and implementation of the Universal Time Coordinated Synchronization Unit (UTCSU). *Real-Time Systems*, 12(3):295–327, May 1997.

[64] W. Schwabl. *Der Einfluß zufälliger und systematischer Fehler auf die Uhrensynchronisation in verteilten Echtzeitsystemen.* PhD thesis, Faculty of Technical and Natural Sciences, Vienna University of Technology, 1988. in German.

[65] B. Simons, J. Lundelius-Welch, and N. Lynch. An overview of clock synchronization. In B. Simons and A. Spector, editors, *Lecture Notes on Computer Science 448 (Fault-Tolerant Distributed Computing)*, pages 84–96. Springer, 1990.

[66] T. K. Srikanth and S. Toueg. Optimal clock synchronization. *Journal of the ACM*, 34(3):626–645, July 1987.

[67] J. A. Stankovic. Misconceptions about real-time computing. *IEEE Computer*, 21(10):10–19, October 1988.

[68] J. G. Steiner, B. C. Neuman, and J. I. Schiller. Kerberos: An authentication service for open network systems. In *USENIX Conference Proceedings*, pages 191–202, Berkeley, CA, 1988.

[69] D. B. Sullivan, D. W. Allan, D. A. Howe, and F.L. Walls. Characterization of clocks and oscillators. *National Institut of Standards and Technology (NIST) Technical Note*, 1337:1–342, March 1990.

[70] P. M. Thambidurai and Y. K. Park. Interactive consistency with multiple failure modes. In *Proceeding of the 7th IEEE Symposium on Reliable Distributed Systems (SRDS)*, pages 93–100, 1988.

[71] G. D. Troxel. *Time Surveying: Clock Synchronization over Packet Networks.* PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institut of Technology, May 1994.

[72] P. Veríssimo, L. Rodrigues, and A. Casimiro. Cesiumspray: a precise and accurate global clock service for large-scale systems. *Real-Time Systems*, 12(3):241–294, May 1997.

[73] J. R. Vig. Oscillator instabilities and specifications. In *Proceeding of the 13th Piezoelectric Devices Conference and Exhibition*, pages 120–130, Kansas City, MO, October 1991.

[74] B. Weiss. *Simulation Environment for Clock Synchronization.* Diploma thesis, Department of Automation, Vienna University of Technology, June 1997.

[75] Z. Yang and T. A. Marsland. Annotated bibliography on global states and times in distributed systems. *ACM SIGOPS Operating Systems Review*, pages 55–72, June 1993.

[76] V. N. Yarmolik and I. V. Kachan. *Self-Testing VLSI Design.* Elsevier Science Publisher, 1993.

Curriculum Vitae

Klaus Schossmaier

Finsterergasse 6/2/28
A-1220 Vienna, Austria

`http://www.auto.tuwien.ac.at/~kmschoss/`

Personal Data

Date of Birth:	January 13, 1967
Place of Birth:	Rottenmann, Styria
Citizenship:	Austria

Education

M.Sc.	February 1, 1994	UMass Amherst	Computer Science
Dipl.-Ing.	June 27, 1991	TU Vienna	Computer Science
	June 17, 1986	High-School Saalfelden	Electrical Engineering

Work Experience

July 1994 – present	Research Assistant in FWF-Project SynUTC (continued under START), Department of Automation, TU Vienna
January – July 1992	Medical Orderly, Military Hospital Innsbruck
Summer/Fall 1990	Teaching Associate for Embedded Systems, Department of Automation, TU Vienna

Awards

Fulbright Scholarship 92/93

Vienna, September 1998