TECHNISCHE UNIVERSITÄT WIEN

DIPLOMARBEIT

Large Neighborhood Search for the Nurse Rostering Problem

Ausgeführt am Institut für
Stochastik und Wirtschaftsmathematik
der Technischen Universität Wien

unter der Anleitung von
o. Univ.-Prof. Dipl.-Ing. Dr. Richard F. Hartl

durch

Alexander Kiefer, MSc
Spreitzergasse 13
2130 Mistelbach

Wien, Oktober 2015

Acknowledgement

I would like to thank my colleagues at the chairs of Production and Operations Management and Production and Operations Management with International Focus of the University of Vienna for their support. In particular I want to thank my supervisor o. Univ.-Prof. Dipl.-Ing. Dr. Hartl and Univ.-Prof. Mag. Dr. Dörner. Last but not least I would like to thank my family and friends.

Abstract

Generating rosters for nurses in a hospital involves the assignment of nurses to shifts, taking into account their skills, preferences and contract types. This task is typically a very complex problem, not least because of the many restrictions that need to be taken into account. These constraints generally include hard constraints, such as forbidden shift successions and providing a minimum number of nurses for each shift. In addition, several soft constraints, representing a well-balanced schedule, should also be satisfied as far as possible.

The actual real-world problem might deviate from hospital to hospital. However, the *international nurse rostering competitions* in 2010 and 2015 generated a common basis for research. They enhanced academic interchange by stating problem formulations and benchmark instances, that have been widely accepted and used by the research community since then. While for the first international nurse rostering competition, a formulation with many constraints was proposed, the second competition used a smaller number of constraints but extended the problem to a sequence of periods for which a roster has to be created successively. The first contribution of this thesis refers to the development of a mixed integer programming model for the problem formulation of the second competition.

Due to the inherent complexity of the problem, heuristic approaches appear to be appropriate, if the problem has to be solved in reasonable time. Hence, I propose a metaheuristic based on *large neighborhood search* for the nurse rostering problem in this thesis. The algorithm repetitively destroys and re-builds relatively large parts of the incumbent solution by making use of several destroy and repair operators. Well-performing selection rates are precomputed by applying a modified version of the algorithm, in which selection probabilities are adjusted dynamically depending on the performance of the operators. These selection rates are then passed to the original algorithm. Within the proposed approach, a *simulated annealing* acceptance scheme is employed for deciding whether a generated solution is accepted as new incumbent.

The algorithm is applied to the benchmark instances of the second international nurse rostering competition. The generated results are then compared with those of the finalists of the competition.

Contents

1 Introduction

Staff scheduling and rostering are highly relevant issues in practice for several reasons. In the context of healthcare institutions, Petrovic and Vanden Berghe [2012] point out the benefits of optimized personnel schedules, including positive effects on the quality of care and the well-being of the workers. In general, personnel rostering seeks to construct a timetable for employees to meet particular requirements with regard to production or service levels, as described by Ernst et al. [2004].

Staff scheduling may be categorized in different ways. In the early survey by Baker [1976], the author distinguishes between two types of workforce allocation problems considering cyclic demand for staff, including *shift scheduling* and *day-off scheduling*. The former involves the allocation of a number of employees to work periods on a day, while the latter consists of scheduling the working days of the staff. Shift scheduling problems are typically encountered in situations when the personnel demand fluctuates over the day, for example in case of telephone operators. Day-off scheduling arises for instance, when the production goes on for seven days per week, but employees work for only five days per week. The combination of these two problem types is usually called *tour scheduling*.

Ernst et al. [2004] divides the process of personnel scheduling into modules, including *demand modelling*, *day-off scheduling*, *shift scheduling*, *line of work construction*, *task assignment*, and *staff assignment*. Demand modelling involves determining the number of workers required for each point in time in the planning horizon. In the context of nurse rostering, the authors refer to *shift based demand* that is directly derived from specifications of the required number of nurses for different shifts based on service measures like nurse-patient ratios. The line of work construction depends on the type of the basic building blocks, i.e., shifts, duties, or stints. In case of shifts, a working day of an employee may consist of any shift, as long as the shift patterns are feasible. Task assignment consists of the actual scheduling of tasks to lines of work, while the scheduling of individual employees to lines of work is done in the last module. Ernst et al. [2004] note, that for generating a roster only a subset of these modules may be required in some cases. Moreover, the solution processes might solve several modules at once, rather than in a step by step manner.

In the review by Ernst et al. [2004], several fields of applications of staff scheduling problems are described. These areas include *transportation systems*, *call centers*, *protection and emergency services*, *civic services and utilities*, *venue management*, *financial services*, *hospitality and tourism*, *retail*, *manufacturing*, and *health care systems*. Within the latter field, nurse rostering is the most prevalent problem, which is also the focus of this thesis.

In a hospital, Warner [1976] distinguishes between three nursing manpower decisions, including *staffing*, *scheduling*, and *allocation*. The staffing decision is made every year and determines the number of nurses for each skill. Every four to six weeks a schedule has to be generated, where each nurse is assigned to shifts and skills, while some coverage constraints have to hold. The allocation process refers to short term shifting of nurses from to different units to overcome unforeseen events, e.g., demand fluctuations and absentees. Warner [1976] note that the task assignment may also be considered as a fourth short term decision step.

Regarding the scheduling phase, Warner [1976] notes that different stakeholders may have conflicting interests. While hospitals have to provide a certain service level, nurses demand continuity, overtime minimization and the consideration of their preferences. Moreover, the author lists six criteria that may play a role in identifying high quality scheduling systems:

Coverage Number of nurses compared to the minimum number of required nurses, for each shift and each skill.

Quality A measure from the perspective of the nurses, to which extent are weekends off, preferences, etc. respected.

Flexibility How flexible is the scheduling system, when it comes to changes like vacations, different contracts, day off requests, etc.

Stability A measure of the consistency of the schedules.

Fairness A measure of an evenly balanced schedule between nurses.

Cost The costs for generating the schedule.

Administrative modes of operation are described by Burke et al. [2004]. In *centralised scheduling*, the personnel scheduling for the whole hospital is executed by one administrative department. Thereby, resources are utilized well and costs

are saved. However, there might be limitations regarding the consideration of local ward requirements, or unfair rosters may be generated. In case the scheduling is done by head nurses or unit managers, Burke et al. [2004] name this type *unit scheduling*. Finally, in *self-scheduling* the roster is generated by the nurses themselves. This process is very time-consuming. Silvestro and Silvestro [2000] highlight the risk of over- or under-staffing, as the nurses are focused on the staff convenience. On the other hand, this method may lead to a high staff satisfaction. A similar differentiation of administrative modes is stated by Silvestro and Silvestro [2000]. In addition to *departmental rostering* and *self-rostering*, they mention *team rostering*, where the rostering process is performed by a nominated member of a team of nurses.

The practical relevance of nurse rostering can particularly be seen from the fact, that several authors deal with practical nurse rostering cases, e.g., Bellanti et al. [2004], Hadwan et al. [2013]. Even benchmark instances are often based on real date, e.g., Brucker et al. [2010], Burke et al. [2008], Smet et al. [2014]. This practical aspect also motivates approaches based on case based reasoning, where previous decisions of the personnel manager may be used for solving future problems, e.g., Beddoe et al. [2009], Petrovic and Vanden Berghe [2012].

Petrovic and Vanden Berghe [2012] argue, that many hospitals still employ manual solution approaches. However, Burke et al. [2004] point out that automated approaches may improve the quality of the rosters and save time for the administrative staff. One reason for these prevalent manual approaches might be that the approaches are typically hard to transfer from one problem to a different one, as noted by Petrovic and Vanden Berghe [2012]. The problems faced by the hospitals often deviate significantly from each other. Objectives and restrictions of the institutions are based on legal issues, management perspectives, and staff requirements. As a consequence of the different focuses of the stakeholders, diverse objective functions and distinct sets of constraints are employed. According to Petrovic and Vanden Berghe [2012] these variations include different shift types, rules for substituting skills and qualifications, and quality measures for a well-balanced schedule.

Petrovic and Vanden Berghe [2012] criticize that complex real world problems are still not completely covered in the research papers. Moreover, these papers typi-

cally formulate their own objective function and set of constraints, making comparisons between existing approaches almost impossible. However, recently there is a trend to close these shortcomings. A collection of staff scheduling problems is provided by the Automated Scheduling, Optimisation and Planning research group at The University of Nottingham[1] derived from real world data. A nurse scheduling problem library, called NSPLib, is described by Vanhoucke and Maenhout [2007]. The library as well as a problem generator for nurse rostering problems (NRPs) are provided on the website by the Operations Research & Scheduling Research Group at the Ghent University[2].

Similarly, Smet et al. [2014] argue, that typically academic NRP models do not incorporate all the features of a complex real-world NRP. Therefore the authors propose a rich, generic model. Other trends to close the gap between research and practice include the two nurse rostering competitions. Within these competitions, new benchmark instances were released, that enable comparisons between different approaches. Among others, the First Nurse Rostering Competition (INRC-I) aimed at stimulating the debate within the scheduling community and bridging the gap between theory and practice, as stated by Haspeslagh et al. [2014]. The INRC-II, described by Ceschia et al. [2015b], basically differs from the first competition with regard to considered problem, as it used a multi-stage formulation. This thesis focuses on the INRC-II problem formulation.

Typically, NRPs incorporate many constraints, as noted by Burke et al. [2010]. For a review of staff scheduling papers with regard to personnel characteristics, flexibility measures, different constraints and other features I refer to Van den Bergh et al. [2013]. A categorization of NRPs analogue to the $\alpha|\beta|\gamma$ notation for scheduling is proposed by De Causmaecker and Vanden Berghe [2011]. The classification is based on several criteria, including general problem characteristics, objectives, constraints and problem dimensions. In the proposed $\alpha|\beta|\gamma$ notation, α refers to the *personnel environment*, β to *work characteristics*, and γ to the *optimization objective*. Besides classifying the diverse problems in this field, this work underlines the broad spectrum of NRP characteristics.

[1] `www.cs.nott.ac.uk/~tec/NRP/index.html`

[2] `www.projectmanagement.ugent.be/?q=research/personnel_scheduling/nsp`

De Causmaecker and Vanden Berghe [2011] referring to Osogami and Imai [2000] state that the NRP is NP-hard. The proof is based on a reduction of the NP-complete timetabling problem to a decision version of the NRP. For that purpose, only a subset of the constraints of a typical NRP is needed. However, according to De Causmaecker and Vanden Berghe [2011] it is still unclear in which way particular objectives or sets of constraints affect the complexity. Brucker et al. [2011] discuss polynomially solvable and NP-hard special cases of personnel scheduling problems. Complexity indicators, including measures regarding the problem size, the preference distribution, the coverage distribution, and time related constraints are analyzed by Vanhoucke and Maenhout [2009]. Other complexity indicators, such as the ward size, the predictability of demand (ratio of planned and emergency operations), the demand variability (based on the length of a patient stay) and the complexity of the skill mix, are discussed by Silvestro and Silvestro [2000].

In case of very complex problems, one has to resort to heuristic approaches. Therefore, a metaheuristic based on Adaptive Large Neighborhood Search (ALNS) is developed for this theses. ALNS has been proposed by Ropke and Pisinger [2006] for tackling vehicle routing problems. Its basic idea is to repetitively destroy and subsequently repair parts of the incumbent solution. It extends the Large Neighborhood Search (LNS) proposed by Shaw [1998] and is related to the Ruin and Recreate approach by Schrimpf et al. [2000]. The adaptive part refers to the dynamic adjustment mechanism for the operator selection during the search. The selection rates resulting from ALNS are then passed to a version of the algorithm without the adjustment scheme, which in turn is used to generate the final results. In the following, the version without the adjustment scheme will be referred to LNS. Ahuja and Orlin [2002] survey *Very Large-Scale Neighborhood Search* techniques, including approaches that employ neighborhoods that either grow exponentially in problem size or are too large to be searched explicitly in practice. With respect to the latter characteristic, LNS belongs to that class.

The outline of this thesis is as follows. A literature review about solution approaches for NRPs with a focus on recent methods is given in Section 2. Approaches that have been applied to the benchmark instances of the INRC-I are particularly highlighted. Background information about the nurse rostering competitions and a detailed problem formulation of the INRC-II are provided in Sec-

tion 3. A Mixed Integer Programming model (MIP) for this problem is stated in Section 4. The proposed solution approach is described in Section 5, while computational results and a sensitivity analysis are provided in Section 6. A conclusion is given in Section 7.

2 Literature review

The nurse scheduling literature reaches back to the 1970s, as shown by the bibliography by Fries [1976]. Early papers in this field include the mathematical programming approaches, by Warner and Prawda [1972], Warner [1976] and Miller et al. [1976].

A review about personnel scheduling problems in general has been conducted by Ernst et al. [2004]. They describe various application areas and different solution methods. Another review dealing with personnel scheduling by Van den Bergh et al. [2013] gives references to previous surveys in this field. Furthermore, the authors categorize papers with respect to their considered problem characteristics. Surveys that are dedicated solely to NRPs are those by Cheang et al. [2003] and Burke et al. [2004], which are essentially structured with regard to solution methods. Burke et al. [2004] classify nurse rostering approaches into *mathematical programming*, *goal programming/multi-criteria approaches*, *artificial intelligence methods*, *heuristics* and *metaheuristic scheduling*.

Recently, MIP-based methods for the NRP have been proposed by Burke et al. [2010], He and Qu [2012], Valouxis et al. [2012], Santos et al. [2014] and Della Croce and Salassa [2014], which are often hybrid approaches, though. Burke et al. [2010] resort to MIP in order to solve a subproblem consisting of all the hard constraints but only a subset of the soft constraints. In a post-processing stage, a variable neighborhood search (VNS) method is employed to improve the solution of the MIP, particularly with regard to the previously excluded constraints.

Valouxis et al. [2012] propose a two phase MIP approach. In the first stage, the workload for each nurse and each day is determined. The daily shifts are then assigned in the second stage. Each phase is solved by a MIP. Additionally, the algorithm is enhanced by local search techniques. The approach is used to

solve the INRC-I benchmark instances. In particular, the authors submitted their algorithm to the competition and reached the first place in all tracks.

He and Qu [2012] propose a hybrid constraint programming based column generation method. Constraint programming is used to solve the pricing subproblem, while the master problem is modelled as a MIP. Two strategies are presented in order to produce high-quality diverse columns.

Santos et al. [2014] present a MIP formulation for the NRP of the INRC-I. In addition, the authors propose improved cut generation strategies. Furthermore, they present a two-stage approach, where an initial schedule is generated in a greedy way first. In the second phase a mathematical programming heuristic based on VNS is applied. For this purpose, sets of variables are fixed and the resulting neighborhoods are explored by the MIP. The authors apply their approach to the benchmark instances of the INRC-I.

A VNS-based matheuristic is proposed by Della Croce and Salassa [2014], where the neighborhoods are iteratively explored by a MIP solver. Among others, the authors apply their approach to the INRC-I benchmark instances.

Burke and Curtois [2014] present a branch and price algorithm and an ejection chain based method. Their approaches incorporate a dynamic programming method. The approaches have been applied to several benchmark instances, including those of the INRC-I. The algorithms performed well at the competition. However, for the final ranking the organizers used hidden instances, where the start date was different compared to the released instances. As a consequence, Burke and Curtois [2014] were not ranked first in end.

A hyperheuristic approach is suggested by Smet et al. [2014], where in each iteration a heuristic is selected from a set of various heuristics. The heuristic is then applied to the incumbent solution. The new candidate solution may be either accepted or rejected. The authors underline the advantage of hyperheuristics, as they are able to cope with a wide spectrum of problem characteristics, which is particularly relevant for the generic NRP model proposed by them.

Brucker et al. [2010] propose an adaptive decomposition approach. The first stage focuses on the generation of sequences of shifts for nurses by considering only a subset of the constraints, while complete rosters are produced in the second stage. Greedy local search is then employed to improve the solutions.

Methods based on case based reasoning have been proposed by Beddoe et al. [2009] and Petrovic and Vanden Berghe [2012]. In case based reasoning, knowledge about previous solutions, eventually reflecting managerial behaviour, is stored and used to solve future problems. Beddoe et al. [2009] employ a memetic algorithm to identify high quality sequences of reparations of constraint violations, while the case based reasoning system contains possible reparations. Petrovic and Vanden Berghe [2012] additionally propose a tabu search metaheuristic, which is then compared against the case based reasoning approach.

In the remainder of this section, recent metaheuristic approaches applied to the NRP are described. Bilgin et al. [2012] propose a VNS and an ALNS approach. Their algorithms seek to improve an initial solution, which fulfills the minimum coverage constraints. VNS makes use of a tabu list of performed moves and a token-ring-search to switch between neighborhoods, while ALNS uses a tabu list too.

An adaptive neighborhood search is described by Lü and Hao [2012]. For that purpose, two neighborhood moves and three intensification and diversification strategies are employed. The algorithm switches adaptively between these search strategies. The approach is applied to the INRC-I instances.

Hadwan et al. [2013] propose a harmony search algorithm. It is inspired by musical improvisation and belongs to the class of population-based metaheuristics.

Chiaramonte et al. [2015] employ an iterative local search method within an agent-based rostering system in order to improve rosters with regard to the preferences of the nurses. Therefore, the NRP is split into a cost minimization problem and a nurse preference rostering problem.

Tassopoulos et al. [2015] propose a two-phase VNS, where nurses are assigned to working days first and the shift assignment is done in the second stage. The algorithm is applied to several benchmark instances, including those of the INRC-I.

3 Problem description

Within this section, the nurse rostering competitions are described (Subsection 3.1) and the problem formulation of the INRC-II is explained in detail in Subsection 3.2. Finally, the constraints of the problem are illustrated in Subsection 3.3.

3.1 Nurse rostering competitions

Referring to Haspeslagh et al. [2010b], the INRC-I built upon the success of two timetabling competitions, that took place in 2002 and 2007. Similarly to nurse rostering, timetabling problems arise in practice at many different institution, particularly at universities, which frequently have distinct requirements regarding the schedule to generate. Hence, early research within this field focused on institution-specific problem variants, as pointed out by Schaerf [1999]. As a consequence, published results have often been only compared with previous - not seldom manually generated - solutions at the respective institution, but are rarely compared with different scientific approaches. Therefore, there was a need for a common definition and widely accepted benchmark instances in order to conclusively compare algorithms. The first international timetabling competition has generated this requested basis for the scientific community, as stated by McCollum et al. [2010]. The second timetabling competition considered three different problems that arise at universities and intended to close the gap between theory and practice. For a description of the second competition, I refer to McCollum et al. [2010]. Meanwhile, even a third timetabling competition with a focus on highschool timetabling was held, as reported by Post et al. [2013].

Similarly, the INRC-I provided a problem formulation and benchmark instances and wanted to stimulate interest in the field of rostering. Details about the competition, including its rules, instances and rankings, can be found on its website[1]. As stated by Haspeslagh et al. [2014], several goals were followed, including generating a common ground for comparing approaches, attracting researchers from different fields to develop new and eventually multi-disciplinary methods, bridging the gap between practice and academic approaches and promoting a debate within the community.

The competition itself consisted of three different tracks, i.e., *sprint*, *middle distance*, and *long distance*, differing in the maximum run times and the size of the instances. For the sprint track, a solution has to be found within seconds, which corresponds to an interactive use in practice. The middle distance track has the most practical relevance, as therefor a solution should be generated within a

[1] `http://www.kuleuven-kulak.be/nrpcompetition`

few minutes, referring to the case where a problem has to be solved a few times. Finally, for the long distance track, the solver is allowed to run for hours. The actual time granted for each track was determined by a benchmarking tool in order to balance different computational powers of the hardware of the participants. For each track, three types of benchmark instances were released, i.e., *early*, *late*, and *hidden* ones. While the early instances were available from the very beginning of the competition, the late ones were released shortly before the end, and the hidden ones were used solely by the organisers to rank the submitted algorithms. This scheme has been adopted from the second timetabling competition, as well as most of its rules.

The INRC-II described by Ceschia et al. [2015b] retained most of the framework of the INRC-I, including the provision of a benchmarking tool, the three different release dates of the benchmark instances, and basically the rules of the first competition. The two competitions manly differ with regard to the proposed problem formulation, i.e., a multi-stage formulation is considered for the INRC-II, compared to the single-stage problem of the INRC-I.

3.2 Problem formulation

The description of the INRC-II is based on the technical report by Ceschia et al. [2015b]. In general, the objective is to generate a roster for a number of days for which all stated hard constraints hold and soft constraint violations are minimized. The term hard constraint refers to a type of constraints that represent features of the roster that have to hold under any circumstances, while soft constraints indicate convenient characteristics. Hard constraints include a minimum coverage of all requested shifts, such as day and night shifts, and each nurse can only be assigned to one shift per day at most. Soft constraints are mainly based on the contracts of the nurses corresponding to work regulations and preferences by the nurses such as requesting a day off.

While the first competition, described by Haspeslagh et al. [2010b, 2014], considered a fixed planning horizon, where the requested coverage requirements for each day and each shift are known from the very beginning, a multi-stage formulation was proposed for the second competition. Moreover, a smaller set of

constraints is used for the INRC-II. In the considered problem, the planning horizon is fixed, as opposed to a rolling horizon, and is split into weeks. Not all of the coverage requirements are known in advance, but are rather revealed later week by week. The solver is then supposed to generate a weekly roster. Some *history* information is passed from each week to the next, as there are also constraints that refer to the whole planning horizon. These global constraints basically reflect the regulations specified by the contracts. The requested number of nurses per day and shift and the nurse preferences are given in the weekly data.

The data consists of a fixed number of nurses, with known skills, e.g., head nurse, regular nurse and trainee, where a nurse may have multiple skills. Moreover, the contract of each nurse is given, specifying

- the minimum and maximum number of shifts
- the minimum and maximum number of consecutive working days
- the minimum and maximum number of consecutive days off
- the minimum and maximum number of consecutive assignments per shift
- the maximum number of consecutive working weekends
- the request for complete weekends, i.e., work either both days or none

Forbidden shift successions such as no early shift after a night shift are valid independently of the contract. Weekly data includes the daily coverage requirements for each shift and each skill, consisting of a minimum number and an optimal number of nurses. In addition, each nurse may have requests for having a day off or not having a particular shift assigned on a day.

For each week, the solver is called and takes the scenario file, specifying work regulations and other settings regarding the whole planning horizon, the history file, containing data about previous assignments, and the requirements of the considered week as an input. The solution consists of the daily assignments of each nurse, specifying the shift and the skill. After a solver call, the new history file is computed on the basis of the old history file and the generated assignment of the week. Moreover, the solver may also pass another file to the next call, eventually containing information to ease the search in the next step.

In the INRC-II problem formulation, hard constraints include that the minimum requirements for each day, each shift and each skill have to be satisfied, each nurse has at most one assignment per day, feasible shift type successions have to hold, and the nurses need to have the skill required for the slot they are assigned to. The soft constraints are based on the contracts and the preferences of the nurses. The constraints are listed below in detail (hard constraints H, soft constraints S), while a formal description of the problem is provided in Section 4.

H1 *Single assignment per day* - for each nurse at most one assignment per day

H2 *Under-staffing* - minimum requirement for each day, each shift, and each skill

H3 *Shift type successions* - only feasible shift successions

H4 *Missing required skill* - nurses need the required skill for their assigned duty

S1 *Insufficient staffing for optimal coverage* - optimal requirement for each day, each shift and each skill; each nurse less than the optimal number is penalized by 30, additional nurses are not penalized

S2 *Consecutive assignments* - minimum and maximum assignments per day and per shift should be met; each day below the minimum and each day beyond the maximum is penalized by 15 for consecutive shift requirements and by 30 for consecutive day requirements

S3 *Consecutive days off* - minimum and maximum consecutive days off; each excess or missing day is penalized by 30

S4 *Preferences* - assignments to unwanted shifts are penalized by 10

S5 *Complete weekend* - nurses requesting complete weekends have either work on both days at the weekend or none; each violation is penalized by 10

S6 *Total assignments* - the minimum and maximum number of assignments within the planning horizon have to be respected; each assignment below the minimum and beyond the maximum is penalized by 20 for each nurse

S7 *Total working weekends* - the maximum number of weekends with a working day must not be exceeded within the planning horizon; for each nurse the excess number of working weekends is penalized by 30

3.3 Illustration of the constraints

In this subsection, the constraints are illustrated, based on a partial weekly schedule given in Table 1. For this example, three days are considered, i.e., *Monday*, *Tuesday* and *Wednesday*, five nurses, i.e., *A*, *B*, *C*, *D* and *E*, three shift types, i.e., *early*, *day* and *night*, and a single skill. The schedule for *Monday* is represented in a different way in Table 2. In Table 1, the dark gray shaded cells indicate the minimum required number of assignments of the respective shifts. The optimal number of scheduled nurses might exceed the minimum requirement, indicated by the light gray shaded cells. Bold symbols refer to assignments where penalties might occur. In table 2, the total number of assigned nurses to a shift is given in column Σ, while the minimum number and the optimal number is shown in the columns *min* and *opt*, respectively. In this example, the hard constraints hold, as each nurse is assigned to one shift per day at most (*H1*), the minimum number of assignments is reached for each shift and each day (*H2*), only feasible shift successions occur (*H3*), e.g. an *early* shift after a *night* shift is typically forbidden, and the nurses are assumed to have the required skill (*H4*).

On *Monday*, the number of assigned nurses falls short of the optimal number. Consequently, the represented schedule is penalized by 30 with regard to constraint *S1 - insufficient staffing*. Note, that the optimal number of assignments might be exceeded without being penalized. Neglecting the history and following days, and given that the maximum number of consecutive working days of nurse *A* is two, the assignment of nurse *A* on *Wednesday* causes a penalty of 30 with regard to *S2 - consecutive working days*, as the maximum number of consecutive working days is exceeded by one day. If the maximum number of consecutive shifts of type *early* is two, then the assignment of nurse *A* on *Wednesday* causes an additional penalty of 15 with regard to *S2 - consecutive shifts*. In case, the contract of nurse *D* states, that the minimum number of consecutive days off is two, the assignment of nurse *D* on *Tuesday* violates this soft constraints as the concatenation of days

off falls short of the minimum number by one day, resulting in a penalty of 30 with regard to *S3*. If nurse E prefers not being assigned to the *day* shift on *Wednesday*, the assignment of E on *Wednesday* causes a penalty of 10 with regard to *S4*. The other constraints can hardly be illustrated within this example, as constraint *S5* refers to weekends, while *S6* and *S7* refer to the whole planning horizon. A more detailed explanation of the constraints is given by Ceschia et al. [2015b].

shift/day	Monday	Tuesday	Wednesday	
early	A	A C	**A**	
day	B E	B C	**E** D	
night	D	E	B	
off	C	**D**	C	

Table 1: Partial weekly schedule

	Monday							
nurse	A	B	C	D	E	Σ	min	opt
early	x					**1**	1	2
day		x			x	2	1	2
night				x		1	1	1
off			x			1	-	-

Table 2: Schedule on Monday

4 Model

In this section, a MIP model for the NRP of the INRC-II will be described. The presented model shares a few similarities with the MIP model proposed by Santos et al. [2014] for the INRC-I formulation, as some features of the problems coincide. Most of the constraints are modelled differently, though. The presented model refers to the complete planning horizon. Hence, it is not directly applicable to the weekly planning process, as therefor one would have to take into account, in which way the current stage would affect future weeks. Therefore, the model is mainly used to give a precise description of the considered problem. It might also be used

for evaluating the overall solution, when the data of the whole planning horizon has been revealed.

4.1 Notation

The notation used for modelling the NRP is shown in Table 3. The problem consists of a set of nurses N and a subset of nurses that request complete weekends N^*, i.e., working either on both days or none. Other sets needed in the model are a set of days D in the planning horizon, a set of Saturdays D^* being a subset of D, a set of shifts S for each day and a set of skills K. Whether nurse n posses skill k is indicated by σ_{nk}. For each day d, each shift s and each skill k, a minimum number of assigned nurses m_{dsk} is given, as well as an optimal level o_{dsk}, that is favored for a high quality service. For each shift type the forbidden successions are known. For example, a nurse being assigned to a night shift typically must not work in the morning of the next day. The preferences of nurse n are represented by the set P_n containing day-shift-pairs that are unwanted by the nurse.

There are several restrictions regarding consecutive assignments and assignments in total. The minimum and maximum number of consecutive working days of nurse n are denoted by a_n^{min} and a_n^{max}, respectively. The minimum and maximum number of consecutive shifts of type s is denoted by c_s^{min} and c_s^{max}, respectively. The minimum and maximum number of consecutive days off and the minimum and maximum number of total assignments of nurse n are denoted b_n^{min}, b_n^{max}, g_n^{min} and g_n^{max}, respectively. The maximum number of working weekends of nurse n is denoted by h_n^{max}.

4.2 Decision variables

The binary four-indexed decision variable x_{ndsk} indicates, whether nurse n is assigned to shift s on day d making use of skill k. Moreover, several auxiliary variables and slack variables are needed. The variables q_{dsk} count the number of nurses less than optimal number for each day d, each shift s, and each skill k. The number of consecutive working days, consecutive shifts of type s and consecutive days off are counted by the variables r_{nd}, u_{nds} and v_{nd}, respectively, for each nurse n and each day d. The binary variables ξ_{nd}, μ_{nds} and ϕ_{nd} indicate, whether

N	set of nurses
N^*	set of nurses that want complete weekends, $N^* \subseteq N$
D	set of days
D^*	set of Saturdays, $D^* = \{d \in D : d \text{ is Saturday}\}$
S	set of shifts
K	set of skills
σ_{nk}	indicator, if nurse n has skill k
m_{dsk}	minimum number of nurses on day d, shift s, skill k
o_{dsk}	optimal coverage on day d, shift s, skill k
F_s	set of forbidden successions after shift s
P_n	set of pairs (day, shift), when nurse n prefers not to work
a_n^{min}	minimum number of consecutive working days, nurse n
a_n^{max}	maximum number of consecutive working days, nurse n
b_n^{min}	minimum number of consecutive days off, nurse n
b_n^{max}	maximum number of consecutive days off, nurse n
c_s^{min}	minimum number of consecutive shifts of type s
c_s^{max}	maximum number of consecutive shifts of type s
g_n^{min}	minimum number of assignments of nurse n
g_n^{max}	maximum number of assignments of nurse n
h_n^{max}	maximum number of working weekends of nurse n

Table 3: Notation

the assignment of nurse n on day d exceeds the maximum number of consecutive working days, consecutive working shifts and consecutive days off, respectively. In either case, the concatenation of assignments refers to the consecutive events until the considered day. For each concatenation of assignments corresponding to day d and nurse n, the shortage with regard to the minimum number of consecutive working days, consecutive working shifts and consecutive days off is measured by the variables ζ_{nd}, ν_{nds} and ψ_{nd}, respectively. The gap may only be positive on a day d, where the concatenation of assignments of the same kind ends on the previous day. Thereby, there is at most one positive variable per concatenation.

Variable c_{nd} indicates, if nurse n has only one working day on the weekend specified by Saturday d. Note, that c_{nd} is only defined for Saturdays and nurses

requesting complete weekends. For each nurse n, the deviation of the actual number of assignments from the allowed number of assignments within the planning horizon is counted by the variable y_n. Thereby, y_n either represents the shortage compared to the minimum number of assignments or the excess number of assignments with respect to the upper bound. Variable w_{nd} indicates, whether nurse n works on the weekend with Saturday d and is only defined for Saturdays. Finally, the number of working weekends of nurse n exceeding the maximum number is counted by variable z_n. The variables of the model are listed as follows.

$$x_{ndsk} = \begin{cases} 1 & \text{if nurse } n \text{ does shift } s \text{ with skill } k \text{ on day } d \\ 0 & \text{otherwise} \end{cases}$$

q_{dsk} nurses less than optimal coverage on day d, shift s, skill k

r_{nd} number of consecutive working days until day d, nurse n

u_{nds} number of consecutive shifts of type s until day d, nurse n

v_{nd} number of consecutive days off until day d, nurse n

$$\xi_{nd} = \begin{cases} 1 & \text{if max. number of consecutive working days is exceeded} \\ 0 & \text{otherwise} \end{cases}$$

$$\mu_{nds} = \begin{cases} 1 & \text{if max. number of consecutive shifts is exceeded} \\ 0 & \text{otherwise} \end{cases}$$

$$\phi_{nd} = \begin{cases} 1 & \text{if max. number of consecutive days off is exceeded} \\ 0 & \text{otherwise} \end{cases}$$

ζ_{nd} days less than min. number of consecutive working days at day d, given that the considered concatenation ends on the previous day

ν_{nds} days less than min. number of consecutive working shifts at day d given that the considered concatenation ends on the previous day

ψ_{nd} days less than min. number of consecutive days off at day d given that the considered concatenation ends on the previous day

$$c_{nd} = \begin{cases} 1 & \text{if nurse } n \text{ has incomplete weekend, Saturday } d \\ 0 & \text{otherwise} \end{cases}$$

y_n distance to the allowed total number of assignments, nurse n

$$w_{nd} = \begin{cases} 1 & \text{if nurse } n \text{ works on weekend with Saturday } d \\ 0 & \text{otherwise} \end{cases}$$

z_n working weekends exceeding the maximum number, nurse n

$$x_{ndsk}, \xi_{nd}, \mu_{nds}, \phi_{nd}, c_{nd}, w_{nd} \in \mathbb{B} \quad q_{dsk}, r_{nd}, u_{nds}, v_{nd}, \zeta_{nd}, \nu_{nds}, \psi_{nd}, y_n, z_n \in \mathbb{N}_0$$

One has to note, that for the decision variables q_{dsk}, r_{nd}, ζ_{nd}, u_{nds}, ν_{nd}, v_{nd}, ψ_{nd}, y_n and z_n the integrality property hold automatically in the model, while the binary constraints have to be stated explicitly for all mentioned binary variables but c_{nd}. Note further, that alternatively one may define the decision variables x_{ndsk} only for skills possessed by the respective nurse. Consequently constraint *H4*, stating that only a nurse with the required skill can be assigned to a shift, would hold automatically. Moreover, as the penalty for deviating from the requested bounds of the consecutive working shifts may occur on a day only with regard to one shift type, the shift index may be omitted for the variables ν_{nds} and μ_{nds}. However, for the sake of readability, this more straightforward model is presented.

4.3 Objective function

Function (1) describes the weighted-sum objective function. The comments underneath the penalty terms indicate the respective soft constraints. The penalty terms referring to the constraints *S1*, *S6*, and *S7* are modelled by using the corresponding slack variables. Violations of the constraint *S4* can be derived directly from the roster. Violations of the different requirements regarding consecutive assignments are either identified by binary variables, indicating whether the assignment on the corresponding day exceeds the upper bound, or integer variables measuring the gap to the lower bound per concatenation.

$$\min \underbrace{\sum_{d\in D,s\in S,k\in K} 30\cdot q_{dsk}}_{\text{S1: Insufficient staffing}} + \underbrace{\sum_{n\in N} 20\cdot y_n}_{\text{S6: Total assignments}} + \underbrace{\sum_{n\in N,d\in D} 30\cdot(\xi_{nd}+\zeta_{nd})}_{\text{S2: Consecutive working days}} +$$
$$\underbrace{\sum_{n\in N,d\in D,s\in S} 15\cdot(\mu_{nds}+\nu_{nds})}_{\text{S2: Consecutive shifts}} + \underbrace{\sum_{n\in N,d\in D} 30\cdot(\phi_{nd}+\psi_{nd})}_{\text{S3: Consecutive days off}} +$$
$$\underbrace{\sum_{n\in N,p\in P_n,k\in K} 10\cdot x_{np_1p_2k}}_{\text{S4: Preferences}} + \underbrace{\sum_{n\in N^*,d\in D^*} 30\cdot c_{nd}}_{\text{S5: Complete weekend}} + \underbrace{\sum_{n\in N} 30\cdot z_n}_{\text{S7: Working weekends}} \quad (1)$$

4.4 Hard constraints

H1: Single assignment per day

$$\sum_{s\in S,k\in K} x_{ndsk} \leq 1 \qquad \forall\, n\in N, d\in D \quad (2)$$

H2: Under-staffing

$$\sum_{n\in N} x_{ndsk} \geq m_{dsk} \qquad \forall d\in D, s\in S, k\in K \quad (3)$$

H3: Shift type successions

$$\sum_{k\in K} x_{ndsk} + \sum_{k\in K, f\in F_s} x_{n,d+1,f,k} \leq 1 \qquad \forall\, n\in N, d\in D, s\in S \quad (4)$$

H4: Missing required skill

$$x_{ndsk} \leq \sigma_{nk} \qquad \forall\, n\in N, d\in D, s\in S, k\in K \quad (5)$$

The four hard constraints (2-5) are modelled straightforward. With regard to Constraints (4) one has to note, that on the first day of the planning horizon the history has to be taken into account. Moreover, the last day can be omitted as there is no following day that needs to be considered.

4.5 Soft constraints

S1: Insufficient staffing for optimal coverage

$$\sum_{n \in N} x_{ndsk} + q_{dsk} \geq o_{dsk} \qquad \forall\ d \in D, s \in S, k \in K \tag{6}$$

Constraints (6) link the decision variables representing the roster with the slack variables. In case the optimal coverage is not reached, the respective slack variable takes the value of the difference to the actual number of assignments.

S2: Consecutive assignments

$$r_{nd} \geq r_{n,d-1} - M + (1+M) \cdot \sum_{s \in S, k \in K} x_{ndsk} \qquad \forall\ n \in N, d \in D \tag{7}$$

$$r_{nd} \leq r_{n,d-1} + 1 \qquad \forall\ n \in N, d \in D \tag{8}$$

$$r_{nd} \leq M \cdot \sum_{s \in S, k \in K} x_{ndsk} \qquad \forall\ n \in N, d \in D \tag{9}$$

$$r_{nd} - M \cdot \xi_{nd} \leq a_n^{max} \qquad \forall\ n \in N, d \in D \tag{10}$$

$$\zeta_{nd} \geq a_n^{min} - r_{n,d-1} - \underbrace{M \cdot \left(1 + \sum_{s \in S, k \in K} x_{ndsk} - \sum_{s \in S, k \in K} x_{n,d-1,s,k}\right)}_{\begin{array}{ll} \geq M & \text{if } \sum_{s \in S, k \in K} x_{n,d-1,s,k} \leq \sum_{s \in S, k \in K} x_{ndsk} \\ = 0 & \text{if } \sum_{s \in S, k \in K} x_{n,d-1,s,k} > \sum_{s \in S, k \in K} x_{ndsk} \end{array}}$$

$$\forall\ n \in N, d \in D \tag{11}$$

The number of consecutive working days of nurse n until day d is first determined by making use of the Constraints (7)-(9), whereby the variable r_{nd} takes the respective value. Therein, M refers to a large number. According to Constraints (7), the number of consecutive assignments is incremented by 1 with respect to the previous day, if the nurse works on the considered day. The increment must not exceed one, as stated by Constraints (8). As soon as nurse n has a day off, the respective counter is set to 0, guaranteed by Constraints (9). The

variables r_{nd} are then linked by Constraints (10) to the binary variables ξ_{nd}, in a way that ξ_{nd} will take the value 1, if the upper bound of consecutive working days is exceeded by the actual number. Constraints (11) are used to identify the gap between the consecutive working days and the lower bound. The slack value ζ_{nd} takes the value of the distance to the lower bound, in case the actual number of consecutive assignments is less the the requirement. However, the constraints are relaxed, except for the case when the considered nurse has to work on the previous day but not on the actual day. Thereby, ζ_{nd} takes the value 0 in all other cases, due to the penalty in the objective function.

Note, that the initial number of consecutive working days r_{n0} has to be set according to the history data. Hence, the inequalities of the Constraints (7), (8) and (11) that correspond to the first day of the planning horizon have to be adjusted accordingly. Moreover, for the first day the term $\sum_{s \in S, k \in K} x_{n,d-1,s,k}$ of the Constraints (11) has to be substituted by the respective history data.

Also note, that when it comes to the evaluation of a roster, violations regarding the maximum number of consecutive assignments that happened in the past, should not be taken into account. For example, if the history data shows that the number of assignments of a nurse has exceeded the upper bound by two in the end of the previously planned period and the same nurse has another assignment on the first day of the considered period, this assignment counts only as one violation. In turn, possible violations regarding not reaching the minimum number of consecutive assignments in the end of the considered period are also ignored, since one cannot predict assignments of the next planning period. For instance, if the consecutive assignments of the last days of the considered period fall short of one assignment, this will not be counted as a violation, as there could be an assignment on the first day of the next period.

$$u_{nds} \geq u_{n,d-1,s} - M + (1 + M) \cdot \sum_{k \in K} x_{ndsk} \qquad \forall\, n \in N, d \in D, s \in S \qquad (12)$$

$$u_{nds} \leq u_{n,d-1,s} + 1 \qquad \forall\, n \in N, d \in D, s \in S \qquad (13)$$

$$u_{nds} \leq M \cdot \sum_{k \in K} x_{ndsk} \qquad \forall\, n \in N, d \in D, s \in S \qquad (14)$$

$$u_{nds} - M \cdot \mu_{nds} \leq c_s^{max} \qquad \forall\, n \in N, d \in D, s \in S \qquad (15)$$

$$\nu_{nds} \geq c_s^{min} - u_{n,d-1,s} - M \cdot \left(1 + \sum_{k \in K} x_{ndsk} - \sum_{k \in K} x_{n,d-1,s,k}\right) \qquad \forall\, n \in N, d \in D, s \in S \qquad (16)$$

The Constraints (12)-(16) referring to the minimum and maximum number of consecutive working shifts are modelled analogously to the constraints for consecutive working days. The main difference is that these constraints have to hold for each shift type, while all adjustments are done straightforward.

S3: Consecutive days off

$$v_{nd} \geq v_{n,d-1} + 1 - M \cdot \sum_{s \in S, k \in K} x_{ndsk} \qquad \forall\, n \in N, d \in D \qquad (17)$$

$$v_{nd} \leq v_{n,d-1} + 1 \qquad \forall\, n \in N, d \in D \qquad (18)$$

$$v_{nd} \leq M \cdot \left(1 - \sum_{s \in S, k \in K} x_{ndsk}\right) \qquad \forall\, n \in N, d \in D \qquad (19)$$

$$v_{nd} - M \cdot \phi_{nd} \leq b_n^{max} \qquad \forall\, n \in N, d \in D \qquad (20)$$

$$\psi_{nd} \geq b_n^{min} - v_{n,d-1} - M \cdot \left(1 - \sum_{s \in S, k \in K} x_{ndsk} + \sum_{s \in S, k \in K} x_{n,d-1,s,k}\right) \qquad \forall\, n \in N, d \in D \qquad (21)$$

The Constraints (17)-(21) identify violations of the minimum and maximum numbers of consecutive days off and are very similar to those regarding consecutive working days. Again, all adjustments are done straightforward. Basically, whenever terms of scheduled duties were used, they are replaced by one minus the term in order to identify a day off. Note, that alternatively one could also introduce an artificial shift type referring to a day off and use the same formulation as for the Constraints (12)-(16) for the new shift type.

S4: Preferences

This group of soft constraints is incorporated into the model solely by extending the objective function.

S5: Complete weekend

$$\sum_{s \in S, k \in K} (x_{ndsk} - x_{n,d+1,s,k}) \leq c_{nd} \qquad \forall\ n \in N^*, d \in D^* \qquad (22)$$

$$\sum_{s \in S, k \in K} (x_{n,d+1,s,k} - x_{ndsk}) \leq c_{nd} \qquad \forall\ n \in N^*, d \in D^* \qquad (23)$$

Incomplete weekends are identified by the Constraints (22) and (23). The variable c_{nd} is to 1 either if the nurse works on Saturday but not on Sunday (Constraints 22) or vice versa (Constraints 23).

S6: Total assignments

$$\sum_{d \in D, s \in S, k \in K} x_{ndsk} + y_n \geq g_n^{min} \qquad \forall\ n \in N \qquad (24)$$

$$\sum_{d \in D, s \in S, k \in K} x_{ndsk} - y_n \leq g_n^{max} \qquad \forall\ n \in N \qquad (25)$$

Whenever the total number of assignments of a nurse deviates from the required level, the slack variable y_n is set to the respective difference. If the actual number of assignments is below the lower bound, the slack variable takes the value of the distance to the lower bound, ensured by Constraints (24), while in case the actual number is beyond the upper bound the distance to the upper bound is used, as guaranteed by Constraints (25).

Note, that only one type of variables is used to measure the deviation from the requested total number of assignments, instead of representing the number of assignments less than the minimum level and the number exceeding the maximum level by individual variables. Hence, as a prerequisite, it is necessary that the minimum number of assignments is less that the maximum number.

S7: Total working weekends

$$\sum_{s \in S, k \in K} (x_{ndsk} + x_{n,d+1,s,k}) \leq 2 \cdot w_{nd} \qquad \forall\, n \in N, d \in D^* \qquad (26)$$

$$\sum_{d \in D^*} w_{nd} - z_n \leq h_n^{max} \qquad \forall\, n \in N \qquad (27)$$

A weekend is identified as a working weekend, if the nurse has to work either on Saturday, Sunday or both days, as stated by Constraints (26). Then, in case the actual number of working weekends exceeds the allowed level, the slack variable z_n takes the value of the difference, ensured by Constraints (27).

5 Solution approach

The complexity results for the NRP suggest, that exact approaches might be inappropriate to tackle the NRP, particularly in case a good solution is needed within a decent amount of time. Hence, I decided to develop a metaheuristic approach to solve the NRP of the INRC-II. More precisely, an ALNS approach based on the papers by Ropke and Pisinger [2006] and Pisinger and Ropke [2007] is applied. Basically, the method extends LNS proposed by Shaw [1998] and seeks to gradually improve a solution by repetitively destroying and repairing relatively large parts of the incumbent solution. ALNS is used to generate well-performing operator selection probabilities, which are then passed to LNS to produce the final results. Details about the approach and how it is tailored to the NRP will be explained in this section. In the following, the term *assignment* refers to a *nurse-day-shift-skill quadruplet*, i.e., a nurse that is assigned to a shift on a day making use of a skill. A day off is also considered as an assignment.

5.1 Adaptive large neighborhood search

Basic elements of the algorithm will be described in Subsection 5.1.1. The other subsections deal with the destroy limit (Subsection 5.1.2), the operator selection scheme (Subsection 5.1.3), the acceptance scheme of new incumbent solutions (Subsection 5.1.4) and the treatment of infeasible solutions (Subsection 5.1.5).

5.1.1 General description

Referring to Pisinger and Ropke [2010], for LNS an initial solution has to be generated first, which is then passed to the algorithm as an input. Next, typically large fractions of the incumbent solution are destroyed and subsequently repaired, as long as the iteration limit is not reached. Generated solutions are accepted according to a particular acceptance criterion to become the new incumbent. The overall best solution found is tracked and returned, when the algorithm terminates.

ALNS, depicted in Algorithm 1, extends LNS by an adjustment scheme for the operator selection. Initially, each destroy and repair operator is selected with equal probability. However, as the search proceeds, the operators gain scores depending on their performance in previous calls. Whenever a number of iterations has been executed, the scores are used to update the weights of the operators, which affect their selection probabilities in future iterations. More precisely, in each iteration a roulette wheel mechanism is used to independently select a destroy and a repair operator based on their weights for being applied to the incumbent solution in the current iteration.

Algorithm 1 Adaptive Large Neighborhood Search

```
1: input: solution x
2: evaluation function f
3: segment size s
4: maximum # removals n^max
5: best solution x^b = x
6: operator weights w = 1
7: operator scores π = 0
8: iteration i = 0
9: while time limit not reached do
10:    roulette wheel selection of destroy and repair operators d and r by making use of w
11:    draw # removals n ∈ [1, n^max]
12:    x' = r(d(x, n))
13:    if x' accepted then
14:       x = x'
15:    end if
16:    if f(x') < f(x^b) then
17:       x^b = x'
18:    end if
19:    i = i + 1
20:    update π_d, π_r
21:    if i mod s = 0 then
22:       update w w.r.t. π
23:       π = 0
24:    end if
25: end while
26: return x^b
```

Several destroy and repair operators are implemented into the ALNS framework. In this setting, the destroy operators release certain regions of the search

space which are then explored by the repair operators. According to Pisinger and Ropke [2010], different combinations of destroy and repair operators result in different neighborhoods of the incumbent solution. These neighborhoods $\mathcal{N}_i$ may have different structures and do not necessarily overlap, as depicted in Figure 1. The entire neighborhood of solution x is denoted by $\mathcal{N}^*$.

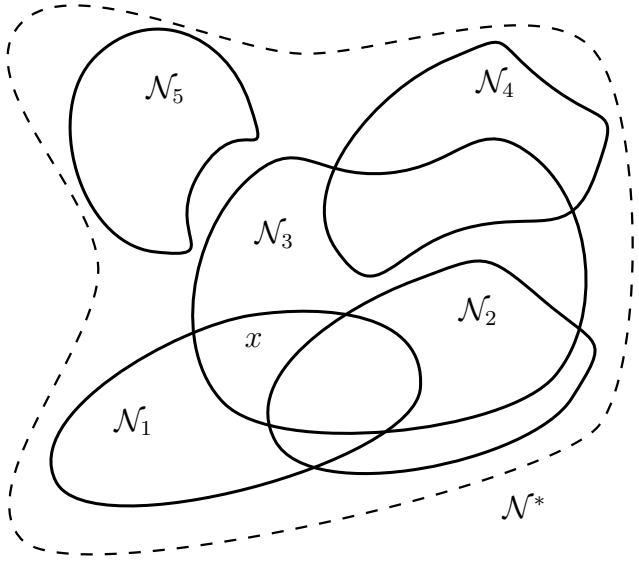


Figure 1: Structurally different neighborhoods (Pisinger and Ropke [2010], p. 412)

Referring to Ahuja and Orlin [2002], considering larger neighborhoods typically results in better local optima. However, the drawback of exploring larger neighborhoods is the probable increase in computation time. Thereby, fewer iterations may be executed within the given time limit.

5.1.2 Destroy limit

In each iteration, a number of assignments is removed from the schedule. This number $n \in \mathbb{N}$ is drawn randomly from $[1, n^{max}]$, where n^{max} denotes the destroy limit. The destroy limit is controlled by the parameter d, specifying the limit as a percentage of the total number of assignments. The schedule is then repaired, while those assignments that have not been removed are fixed.

5.1.3 Operator selection

The *adaptive* element of ALNS refers to the operator selection mechanism, that alters the selection probabilities of the operators dynamically. Operators are selected based on a roulette wheel principle, as suggested by Ropke and Pisinger [2006]. Thereby, the selection probability ϕ_j of operator j is computed as

$$\phi_j = \frac{w_j}{\sum_{i=1}^{k} w_i}$$

where k denotes the number of operators and w_i denotes the weight of operator i. The selection is done independently for destroy and repair operators.

Following Ropke and Pisinger [2006], in the end of each iteration the score π_j of the operator j that has been used is adjusted by adding σ. The actual value of σ depends on the quality of the generated solution.

$$\sigma = \begin{cases} \sigma_1 & \text{if the solution is a new global best} \\ \sigma_2 & \text{if the solution is better than the incumbent and not accepted before} \\ \sigma_3 & \text{if accepted, worse than the incumbent and not accepted before} \end{cases}$$

Obviously, operators that generate solution that have not been accepted before are encouraged. The rationale behind this is to try to direct the search towards yet unexplored regions of the search space.

As shown in Algorithm 1, the search is split into segments of size $s = 100$. Whenever s iterations have been executed, weights are updated based on the scores obtained within these iterations. The new weight w_j^{new} of operator j is computed as

$$w_j^{new} = w_j^{old} \cdot (1 - r) + r \cdot \frac{\pi_j}{\xi_j}$$

where w_j^{old} denotes the old weight of the operator, $r \in [0, 1]$ denotes a parameter controlling to what extent the weights are adapted and ξ_j denotes the number of calls of operator j in the segment.

5.1.4 Acceptance scheme

The criterion, whether a roster is accepted as new incumbent solution, is based on *Simulated Annealing*, proposed by Kirkpatrick et al. [1983]. The same acceptance scheme has been applied by Ropke and Pisinger [2006]. The probability of accepting a new solution x' is computed as $\min\left\{1, e^{-(f(x')-f(x))/T_i}\right\}$, where f denotes an evaluation function, x the incumbent solution and T_i the temperature in iteration i. Consequently, a roster with a smaller objective value is always accepted, while deteriorating solutions might be discarded.

The starting temperature is defined implicitly, in a way that a solution that is ψ-percent worse than the initial one is accepted with a probability of 50%, where ψ is a parameter. Hence, the starting temperature T_{start} is computed as $T_{\text{start}} = -(\psi \cdot f(x_0))/ln(0.5)$, where x_0 denotes the initial solution. In the end of each iteration i, the temperature is cooled down by being multiplied with a factor $\rho_i < 1$, i.e. the temperature for the subsequent iteration is calculated as $T_{i+1} = T_i \cdot \rho_i$. Following Lewis and Thompson [2015], ρ_i is computed for each iteration individually, in a way that in the end of the search a target temperature T_{end} is reached, which is passed to the algorithm as a parameter. The formula to compute the cooling factor is given as $\rho_i = (T_{\text{end}}/T_i)^{1/t_i}$, where t_i denotes the remaining computation time at iteration i before the time limit will be reached.

5.1.5 Infeasible solutions

Violations of the hard constraints *H1*, *H3* and *H4* are prohibited by the algorithm. However, the actual number of scheduled nurses might be below the minimum required number, which will be penalized by the evaluation function, though. The rationale behind allowing infeasible solutions is that thereby shortcuts through infeasible regions of the search space are possible, which might enhance the performance of the algorithm. In the context of the closely related timetabling problem, advantages and drawbacks of allowing infeasible solutions have been discussed by Lewis [2008].

When it comes to the evaluation of potential assignments or an entire roster, the penalty p for under-staffing is taken into account. Each nurse less than the minimum required number is penalized by p. Initially, p set to $p_{\max}$, refer-

ring to the upper bound of the hard constraint penalty. During the search, p is adjusted dynamically, as suggested by Gendreau et al. [1994]. In particular, whenever an infeasible solution is accepted, p is set to $\min(p^{old} \cdot \alpha, p_{\max})$, where α denotes a parameter and p^{old} denotes the previous value of p. Conversely, p is set to $\max(p_{\min}, p^{old}/\alpha)$, whenever a feasible solution is accepted, where $p_{\min}$ denotes the lower bound of the penalty p.

5.2 Destroy operators

The algorithm incorporates several destroy operators. Various other destroy operators have been tested but later excluded, as they were not able to improve the solution quality. Those operators, that are finally used, are described in the following subsections.

5.2.1 Related

The *related* destroy operator, originally proposed by Shaw [1998] and adopted by Ropke and Pisinger [2006], aims at removing similar assignments. Here, the relatedness between two assignments depends on the number of common skills of the two nurses and whether the two assignments are either on the same day or conflicting with regard to the forbidden shift successions. More precisely, the relatedness measure between two assignments i and j is computed as $v_{ij} + \beta \cdot \delta_{ij} \cdot |K| + \gamma \cdot \zeta_{ij} \cdot |K|$ where v_{ij} denotes the number skills that the nurses that correspond to assignment i and assignment j have in common, δ_{ij} indicates whether the two assignments are on the same day, while ζ_{ij} indicates whether assignment j is on an adjacent day of assignment i and would violate the shift succession constraint with respect to the shift of assignment i, $|K|$ denotes the total number of skills and β and γ are parameters. The rationale behind considering assignments of the same day and nurses with similar skills, is that these assignments might be easily swapped. However, as swapping them may requires to reschedule assignments of the same nurses on adjacent days as well, forbidden shift successions are also considered.

The *related* destroy operator is described in Algorithm 2. At first, an assignment is selected at random and added to the set B, representing the set of

assignments to remove from the roster. The set of assignments that are not in B is denoted by A. As long as the cardinality of B is less than the requested number of removals, an assignment b is randomly drawn from B and A is sorted in descending order with respect to the relatedness to b. An assignment is randomly drawn from A and added to B by computing its index $\lfloor |A| \cdot \upsilon^{\kappa_1} \rfloor$, where υ denotes a random number in $[0, 1)$ and the parameter $\kappa_1 \in \mathbb{R}$ controls the selection bias towards closely related assignments. If $\kappa_1 = 1$, assignments are selected with an uniform probability, while a large κ_1 results in a high probability of picking closely related assignments.

Algorithm 2 Related removal

1: input: roster R, parameter κ_1, number of requested removals n
2: set A of assignments in R
3: set of assignments to remove B
4: select random assignment $a \in A$
5: $B = \{a\}$
6: $A = A \setminus \{a\}$
7: **while** $|B| < n$ **do**
8: select random assignment $b \in B$
9: sort A in descending order w.r.t. relatedness to b
10: draw random number $\upsilon \in [0, 1)$
11: $a = A[\lfloor |A| \cdot \upsilon^{\kappa_1} \rfloor]$
12: $B = B \cup \{a\}$
13: $A = A \setminus \{a\}$
14: **end while**
15: remove all assignments in A from R

5.2.2 Penalty

Following Ropke and Pisinger [2006], the *penalty* destroy operator aims at removing penalized assignments. As most of the soft constraints refer to the rosters of nurses rather than being associated with single assignments, the penalties of the schedule are assigned to nurses. The list of nurses is then sorted in descending order with respect to these penalties. For that, all soft constraints are considered, except *S1 - insufficient staffing for optimal coverage*. While the requested number of removals is not reached, nurses are selected at random from the list with a bias towards those with the highest penalties. All the assignments of the selected nurses are then removed from the roster. The operator is depicted in Algorithm 3.

Algorithm 3 Penalty removal

```
1: input: roster R, parameter κ2,
   number of requested removals n
2: list of all nurses C
3: sort C by descending penalty
4: while n > 0 do
5:     draw random number υ ∈ [0, 1)
6:     c = C[⌊|C| · υ^κ2⌋]
7:     C = C \ {c}
8:     remove all x assignments of c from
       R
9:     n = n − x
10: end while
```

5.2.3 Understaffing

The *understaffing* destroy operator seeks to improve the objective with regard to the soft constraint *S1*. For each day-skill pair, the penalties for insufficient staffing regarding the optimal coverage that are associated to shifts on the considered day are determined. The list of the day-skill pairs A is then sorted according to the penalties in descending order. A day-skill pair is selected by computing its index $\lfloor |A| \cdot \upsilon^{\kappa_3} \rfloor$, where $\upsilon \in [0, 1)$ again denotes a random number and $\kappa_3 \in \mathbb{R}$ denotes a parameter. All assignments on the selected day of nurses with the required skill are removed from the schedule. The process is proceeded as long as the number of removals is less than the requested number.

5.3 Repair operators

Various repair operators are implemented. These operators are variants of a basic greedy operator, that essentially seeks to roster the currently best possible assignment without considering its consequences regarding subsequent assignments. Moreover, a regret operator has been tested but discarded, as it did not improve the solution quality. The finally employed repair operators are described in the following subsections.

5.3.1 Greedy

The basic *greedy* repair operator is described in Algorithm 4. A list of nurse-day pairs, indicating for which nurses a daily assignment has to be found, is passed to the operator, as well as a partial roster and a function for evaluating possible assignments. First, for each nurse-day pair in the list, possible assignments are

identified. The algorithm then evaluates each possible assignment for each entry in the list, and schedules the best assignment, i.e., the assignment that causes the least penalties or improves the partial roster the most. The process is repeated until an assignment has been found for each nurse-day pair in the list.

Algorithm 4 Greedy repair operator

```
1: input: list L of nurse-day pairs,
   roster R, evaluation function f
2: ∀l ∈ L: compute potential
   assignments A_l
3: while L ≠ ∅ do
4:    best assignment a_best
5:    penalty p_best = ∞
6:    nurse-day pair l_best
7:    for all l ∈ L do
8:       for all a ∈ A_l do
9:          if f(a) < p_best then
10:            a_best = a
11:            p_best = f(a)
12:            l_best = l
13:         end if
14:      end for
15:   end for
16:   R = R ∪ {a_best}
17:   L = L \ {l_best}
18:   update A_{l_best}
19: end while
20: return R
```

The evaluation of potential assignments with regard to the soft constraints is straightforward. If the optimal number of assigned nurses on a day, for a shift and a skill is not yet covered, assigning the considered nurse would improve the object value. Hence, the soft penalty of *S1* is subtracted. With regard to the constraints *S2* and *S3*, referring to consecutive assignments, the effect of the potential assignment on the respective concatenation is evaluated. If the minimum number of consecutive assignments is not reached on the adjacent days, performing the considered potential assignment would help to improve the objective value and the respective soft constraint penalty is subtracted accordingly. However, if there is no adjacent assignment of the same type, or if the maximum number of consecutive assignments would be exceeded, the respective penalty is added. If the considered assignment is one of those when the nurse prefers not to work, the penalty corresponding to *S4* is added. Finally, if a duty on the weekend is considered and the nurse does not work on the other day, the penalty of *S5* is added, given that the nurse prefers complete weekends. In turn, if the nurse already works on the other day of the weekend, an assignment to a working shift would improve the objective value with regard to *S5* and therefore the penalty is subtracted.

The soft constraints referring to the whole planning horizon are only estimated, in a way that the threshold of the minimum and maximum number of working days and weekends is divided by the number of already planned weeks, including the one the is currently under consideration. If the adjusted maximum level has already been exceeded or would be exceeded by performing the considered working assignment, the respective penalty is added. In turn, if the adjusted minimum level is not reached, rostering a working shift would improve the objective value and the penalty is subtracted.

The schedule may violate the hard constraint *H2 - under-staffing*, however, the penalty for falling short of the minimum level is incorporated in the evaluation function. If the minimum level is not covered and a working assignment is considered, the under-staffing penalty of the current iteration is subtracted.

Following Ropke and Pisinger [2006], a random variable in $[-\mu, \mu]$ is added to the cost corresponding to a potential assignment, where μ denotes a parameter. This feature proved to be beneficial and is therefore added to all repair operators.

5.3.2 Adjusted

The basic *greedy* operator considers solely the assignment cost of the nurse on the respective day, while possible opportunity costs on adjacent days are neglected. Due to the hard constraint *H3 - shift type successions* some shifts on adjacent days might become infeasible. If an assignment has to be found for the same nurse on adjacent days and the respective best assignment would become infeasible, the distance to the best remaining feasible assignment is considered as opportunity costs. For the *adjusted* repair operator the evaluation function is extended by taking these opportunity costs into account, while all other parts of the algorithm coincide with the *greedy* algorithm.

5.3.3 Staffing

The *staffing* repair operator also builds upon the basic *greedy* repair operator, but tries to satisfy the hard constraint *H3* first. Therefore, the day-shift-skill triple g with the highest priority value is selected according to the formula $\Delta_g/A_g + \eta$, where Δ_g denotes the difference of the minimum number of assigned nurses to the

actual number of the considered slot g, A_g denotes the number of available nurses of slot g, and η denotes a random number in $[-\nu, \nu]$ controlled by the parameter ν. The nurse with the lowest penalty, computed in the same way as for the *greedy* operator, is then assigned to the selected slot. This procedure is repeated either as long as there are day-shift-skill triples that fall short of the minimum number of assigned nurses, or no nurses are available for these slots any more. The remainder of the roster is scheduled by applying the *greedy* operator.

The initial solution is generated by the *staffing* repair operator. Thereby, the initial solution is typically feasible.

6 Computational results

In this section the computational results are presented. The algorithm is tested on the *late* benchmark instances of the INRC-II that will be described in Subsection 6.1. In Subsection 6.2 the parameter setting is provided. The comparison of the proposed algorithm with the algorithms of the finalists of the competition is presented in Subsection 6.3. A sensitivity analysis of the employed operators is shown in Subsection 6.4.

6.1 Benchmark instances

For the INRC-II, various instance sets have been released. Referring to the website of the INRC-II[1], the organizers provided test datasets for debugging and testing purposes, competition datasets, which include the *late* instances for an initial ranking of the participants, and the *hidden* datasets to rank the finalists. In general, the competition datasets have been released in the beginning of the competition, while the *late* instances, i.e., the precise composition of scenarios, history files and week data, has been released shortly before the deadline of the competition. The comparison of the proposed algorithm to those of the competition finalists is based on the *late* instances. Their characteristics are presented in Table 4.

The first column of the table refers to the number of the instance, that will also be used in the tables of Subsection 6.3. The column *Name* states the name

[1] `http://mobiz.vives.be/inrc2/`[accessed 2015-09-23]

#	Name	Nurses	Weeks	Skills	Shifts	Forb.	Contr.
1	n030w4_1_6-2-9-1	30	4	4	4	3	3
2	n030w4_1_6-7-5-3	30	4	4	4	3	3
3	n030w8_1_2-7-0-9-3-6-0-6	30	8	4	4	3	3
4	n030w8_1_6-7-5-3-5-6-2-9	30	8	4	4	3	3
5	n040w4_0_2-0-6-1	40	4	4	4	2.5	3
6	n040w4_2_6-1-0-6	40	4	4	4	2.5	3
7	n040w8_0_0-6-8-9-2-6-6-4	40	8	4	4	2.5	3
8	n040w8_2_5-0-4-8-7-1-7-2	40	8	4	4	2.5	3
9	n050w4_0_0-4-8-7	50	4	4	4	2.5	3
10	n050w4_0_7-2-7-2	50	4	4	4	2.5	3
11	n050w8_1_1-7-8-5-7-4-1-8	50	8	4	4	2.5	3
12	n050w8_1_9-7-5-3-8-8-3-1	50	8	4	4	2.5	3
13	n060w4_1_6-1-1-5	60	4	4	4	3	4
14	n060w4_1_9-6-3-8	60	4	4	4	3	4
15	n060w8_0_6-2-9-9-0-8-1-3	60	8	4	4	2.5	3
16	n060w8_2_1-0-3-4-0-3-9-1	60	8	4	4	2.5	3
17	n080w4_2_4-3-3-3	80	4	4	4	2.5	4
18	n080w4_2_6-0-4-8	80	4	4	4	2.5	4
19	n080w8_1_4-4-9-9-3-6-0-5	80	8	4	4	2.5	4
20	n080w8_2_0-4-0-9-1-9-6-2	80	8	4	4	2.5	4
21	n100w4_0_1-1-0-8	100	4	4	4	2.5	4
22	n100w4_2_0-6-4-6	100	4	4	4	2.5	4
23	n100w8_0_0-1-7-8-9-1-5-4	100	8	4	4	2.5	4
24	n100w8_1_2-4-7-9-3-9-2-8	100	8	4	4	2.5	4
25	n120w4_1_4-6-2-6	120	4	4	4	2.5	3
26	n120w4_1_5-6-9-8	120	4	4	4	2.5	3
27	n120w8_0_0-9-9-4-5-1-0-3	120	8	4	4	2.5	3
28	n120w8_1_7-2-6-4-5-2-0-2	120	8	4	4	2.5	3

Table 4: Characteristics of the *late* instances

of the instance used at the competition indicating the composition of the files. The columns *Nurses*, *Weeks*, *Skills* and *Shifts* show the number of nurses, weeks, skills and shifts, respectively. *Forb.* refers to the average number forbidden shift successions per shift type, e.g., scheduling a late shift typically prohibits a night shift on the previous day as well as an early and a day shift on the next day resulting in three forbidden shifts in total. Finally, *Contr.* refers to the number of different contracts.

6.2 Parameter setting

Several parameters are used to control the algorithm. Their tuning is based on the average soft penalties of the solutions of the *late* instances with one run per instance. As the competition participants also had the possibility of tuning their algorithms on these instances, building decisions upon the *late* instances allows a fair comparison.

Initial parameter values of the adaptive mechanism are borrowed from Ropke and Pisinger [2006], while the others have been found during the implementation phase. The tuning is then performed by altering one parameter at a time, while keeping the other parameter values fixed. For each parameter, two different values are tested. After each parameter has been altered, their best setting is chosen, building the basis for another run. After the second run the procedure is stopped, as there are no significant deviations in quality observable. In general, the algorithm does not react very sensitive to slight changes of most of the parameters. A proper tuning seems to be important for the parameters controlling the temperature of the acceptance scheme and the destroy limit. The latter is particularly relevant, as the algorithm benefits from additional iteration granted by a smaller destroy limit. The final parameter setting is shown in Table 5.

6.3 Results

The organizers of the competition provide a benchmarking tool in order to determine the allowed runtime for generating a weekly roster for the competition dataset, particularly for the *late* instances, on a single core machine. The time limit is set according to this tool. The generated results for the *late* instances are

Parameter	Value	Description
ψ	70%	SA: Initially accept ψ-percent worse solution with 50%
T_{end}	6	SA: Target temperature
σ_1	30	ALNS: Score for new global best
σ_2	12	ALNS: Score for new, accepted, better than current
σ_3	13	ALNS: Score for new, accepted, worse than current
r	0.12	ALNS: Reaction factor for weight adjustment
α	1.01	Hard penalty: Adjustment factor
$p_{\min}$	20	Hard penalty: Lower bound
$p_{\max}$	400	Hard penalty: Upper bound
d	6%	Destroy limit: Maximum percentage of assignments
β	0.9	Relatedness measure: Influence of the same day
γ	0.25	Relatedness measure: Influence of adjacent day
κ_1	5	Related removal: Selection probability
κ_2	3	Worst removal: Selection probability
κ_3	7	Understaffing removal: Selection probability
μ	25	Noise: Scheduling cost, noise $\in [-\mu, \mu]$
ν	0.1	Noise: Priority for *staffing*, noise $\in [-\nu, \nu]$

Table 5: Parameter setting

then compared to those of the competition participants. The results are generated on a computer with 16 GB memory and an Intel Core i7-4770 CPU running at 3.4 GHz, where only one core is used, and an Ubuntu 14.04 operating system.

Among fifteen participants of the competition, seven have been selected as finalists. At the time, this thesis has been written, no details about their algorithms have been published. Starting with the best performing algorithms, the finalists are:

1. *NurseOptimizers*: Michael Römer, Taieb Mellouli, Institute of Business Information Systems and Operations Research, Martin Luther University Halle-Wittenberg

2. *Polytechnique Montreal*: Legrain Antoine, Omer Jérémy, Rosat Samuel

3. *SSHH*: Ahmed Kheiri, UK

4. *Hust.Smart*: Zhouxing Su, Zhuo Wang, Zhipeng Lü, Laboratory of Smart Computing and Optimization, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, P.R. China.

5. *ORTEC*: Hujin Jin, Gerhard Post, Egbert van der Veen, ORTEC

6. *LabGOL*: Federica Picca Nicolino, Francesco Bagattini, Luca Bravi, Niccolò Bulgarini, Alessandro Galligari, Fabio Schoen, Università degli Studi di Firenze, Italy

7. *ThreeJohns*: Tassopoulos X. Ioannis, Solos P. Ioannis, Beligiannis N. Grigorios, University of Patras, Greece

The final results are generated by determining well-performing selection rates by employing ALNS first, which are averages over all *late* instances. These rates are then used as an input for a variant of the algorithm (LNS), where the adaptive operator selection scheme is omitted. Thereby, additional iterations are performed within the same time limit, while eventually not the best instance-specific selection rates are used, though. The results of the proposed algorithm presented in Table 6 are averages over five runs with random seeds, while those of the competition finalists are based on single runs, taken from the website of the INRC-II[1]. The first column indicates the number of the instance. The column *LNS avg* refers to the average results of the proposed algorithm over five runs, while the best result out of these runs is reported in column *LNS best*. *Best* shows to the best known solution for the instances. Finally, in the remaining columns the results of the competition participants are stated, where the following abbreviations are used: *NO* (NurseOptimizers), *PM* (Polytechnique Montreal), *SH* (SSHH), *HS* (Hust.Smart), *OR* (ORTEC), *LG* (LabGOL), *TJ* (Three Johns). Bold numbers indicate the best result of the respective instance.

Compared to the finalists of the competition, the proposed algorithm performs worse. However, when looking at the submitted and validated results of all participants, available on the website[1], LNS outperforms six participants and performs approximately equally well as the ninth ranked participant.

[1] `http://mobiz.vives.be/inrc2/`[accessed 2015-09-23]

#	NO	PM	SH	HS	OR	LG	TJ	LNS		Best
								avg	best	
1	**1755**	1790	1940	2010	2190	2065	2270	2378	2335	1755
2	**1935**	2040	2200	2135	2295	2230	2160	2512	2460	1935
3	**2340**	2365	2990	3135	3090	3125	3080	3479	3385	2340
4	**1900**	1990	2655	2725	2700	2630	2640	3097	2995	1900
5	**1730**	1765	1895	1910	2415	2275	2290	2601	2535	1730
6	**1880**	2010	2165	2080	2430	2365	2340	2872	2700	1880
7	**3310**	3375	3975	3855	4770	4505	4630	5883	5725	3310
8	3080	**3025**	3695	3625	4345	4140	4470	5483	5265	3025
9	**1490**	1630	1845	1775	2135	2135	2255	2344	2265	1490
10	**1480**	1645	1875	1790	2375	2185	2560	2449	2365	1480
11	**5410**	5790	6080	7135	7075	7225	7420	8095	7915	5410
12	**5435**	5710	5985	6775	7230	7105	7340	8233	8065	5435
13	**2815**	2830	3105	3140	3960	3440	3330	3889	3855	2815
14	3005	**2950**	3290	3425	4090	3715	3670	4112	4060	2950
15	**2765**	2925	3825	3970	4470	4725	4280	6015	5575	2765
16	**3065**	3800	3955	4550	5330	5230	4925	6113	6045	3065
17	**3535**	3615	3830	4210	4575	4485	4735	5351	5310	3535
18	**3570**	3840	3955	4015	4420	4530	4440	5057	4805	3570
19	**4995**	5200	5755	5665	7180	6840	6830	9723	9465	4995
20	**5030**	5645	6360	6390	7590	7480	7845	10275	9890	5030
21	1530	**1445**	1770	2000	2865	2550	2400	4192	4010	1445
22	2155	**2100**	2365	2520	3335	3090	2955	4708	4510	2100
23	3195	**3080**	3640	4145	5735	5290	4915	10331	9850	3080
24	**3055**	3630	4215	4345	6105	5595	5865	10537	10150	3055
25	**2435**	2515	2635	3100	4005	3650	3625	5012	4785	2435
26	**2485**	2575	2875	3345	3830	3700	3910	4975	4780	2485
27	**3615**	4145	4065	5295	6590	6315	6530	8884	8725	3615
28	**3510**	4385	4110	5460	7275	6040	7160	8433	8055	3510
avg	2947	3136	3466	3733	4443	4238	4317	5608	5424	2934

Table 6: Results for the *late* instances

Table 7 presents statistics of the generated results, where all numbers are averages over five runs. The column *Limit* shows the average number of iterations that have been executed within the given time limit. *Acc.* shows the percentage of solutions that have been accepted as new incumbent. In the column *Found*, the iteration, in which the best solution has been found, is given as a percentage of the total number of iterations. *Inf.* refers to the percentage of generated infeasible solutions with respect to the total number of iterations, while *InfAcc* shows the accepted infeasible solutions as a percentage of the total number infeasible solutions. The columns *Greedy*, *Staff*, and *Adjust* show the percentages of infeasible solutions generated by the *greedy*, *staffing*, and *adjusted* repair operators, respectively.

On average, the best solution is found after two thirds of the iterations. The *greedy* and the *adjusted* repair operators tend to produce infeasible solutions regularly, while the *staffing* operator generates mainly feasible solutions. In Subsection 6.4, it will be seen that the *staffing* operator is called most often. Hence, the total portion of infeasible solutions is relatively small. Infeasible solutions are barely accepted, though.

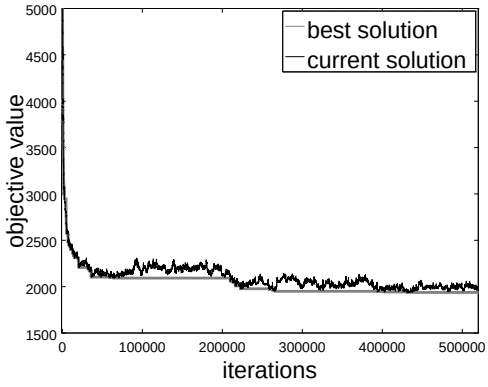
Figure 2 shows the convergence of the incumbent solution to the best solution for each week of instance 17. The gray line represents the best solution so far, while the black line shows the incumbent solution. In the beginning of the search, the best solution is gradually improved. However, as the search proceeds, the improvement stagnates, even though the gap to the best known solution is quite large. This comes despite the fact, that the temperature of the acceptance scheme is high enough to occasionally accept deteriorating solutions even in the end of the search, as indicated by the fluctuations of the black line.

6.4 Sensitivity analysis

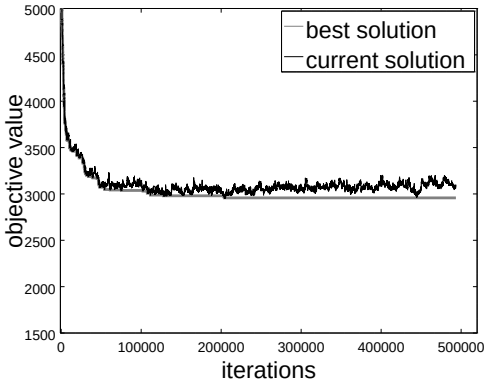
Table 8 shows the importance of the different destroy and repair operators. The results presented in the table are based on averages over two runs of the *late* instances and are generated with the ALNS algorithm. Column *Selection* shows the average selection rate, while *Deter.* refers to the deterioration in solution quality, when the operator is excluded. Effects of removing two operators at once have not been analyzed.

#	Limit	Acc.	Found	Inf.	InfAcc	Greedy	Staff	Adjust
1	329387	30.82%	67.65%	12.85%	6.33%	66.70%	5.28%	68.00%
2	360958	31.31%	51.22%	21.55%	13.39%	71.01%	14.58%	72.54%
3	270868	28.21%	67.51%	17.91%	10.50%	70.71%	10.50%	71.73%
4	220897	27.84%	59.06%	10.90%	3.41%	63.48%	3.51%	64.76%
5	293366	24.25%	60.70%	8.87%	0.87%	55.41%	2.30%	57.01%
6	362381	23.10%	69.46%	9.42%	0.31%	59.24%	2.43%	60.33%
7	513767	26.19%	67.60%	9.55%	1.85%	59.88%	2.49%	60.71%
8	513441	25.83%	68.51%	7.87%	0.63%	52.77%	1.58%	53.50%
9	665451	16.99%	68.72%	8.99%	0.15%	59.82%	1.83%	61.27%
10	661609	16.96%	62.54%	9.52%	1.12%	59.22%	2.52%	60.78%
11	664221	17.26%	61.03%	9.82%	0.45%	63.02%	2.37%	63.77%
12	434815	17.48%	61.58%	9.51%	0.45%	62.66%	2.05%	63.67%
13	401058	17.82%	69.20%	8.25%	0.60%	55.55%	1.46%	59.00%
14	533597	18.34%	70.77%	8.80%	1.58%	57.16%	1.87%	60.48%
15	449215	15.93%	69.84%	10.37%	0.36%	66.60%	2.46%	68.05%
16	546585	13.56%	62.89%	10.65%	0.30%	69.26%	2.46%	69.90%
17	654381	14.23%	77.82%	9.89%	0.34%	61.04%	2.45%	66.43%
18	493429	14.07%	73.96%	8.54%	0.28%	53.23%	1.94%	59.50%
19	719413	11.57%	79.90%	7.25%	0.16%	51.30%	0.87%	55.36%
20	573635	12.01%	81.21%	7.61%	0.21%	53.99%	0.92%	57.85%
21	790693	8.33%	79.87%	5.58%	0.14%	40.16%	0.67%	41.81%
22	821426	7.51%	78.43%	6.26%	0.07%	45.07%	0.77%	46.54%
23	824519	8.17%	80.03%	4.92%	0.15%	35.90%	0.55%	36.88%
24	822719	7.27%	85.03%	5.15%	0.09%	38.07%	0.51%	39.09%
25	568848	6.50%	83.06%	10.65%	0.11%	69.05%	2.45%	70.23%
26	716209	6.29%	80.29%	9.91%	0.10%	63.22%	2.42%	64.38%
27	840920	5.94%	79.61%	9.74%	0.08%	62.64%	2.33%	63.58%
28	848081	5.52%	78.67%	9.82%	0.07%	63.13%	2.34%	64.22%
avg	567710	16.40%	71.29%	9.65%	1.57%	58.19%	2.78%	60.05%

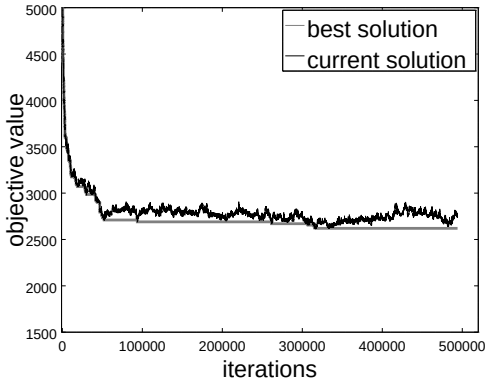
Table 7: Statistics of the intermediate solutions



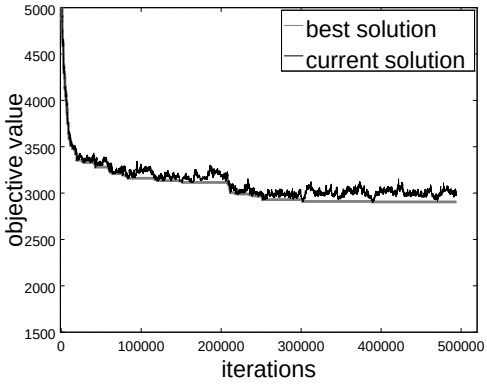
(a) Week 1



(c) Week 3



(b) Week 2



(d) Week 4

Figure 2: Solution convergence, instance 17

Operator	Selection	Deter.
Related	1.57%	0.02%
Penalty	96.58%	143.14%
Underst.	1.85%	0.4 %

Operator	Selection	Deter.
Greedy	8.03%	0.74%
Staffing	85.72%	33.96%
Adjusted	6.25%	0.79%

Table 8: Operator statistics

As the results are based on only two runs per instance, one may not derive meaningful propositions from small deviations. However, the *staffing* repair operator and the *penalty* destroy operator are clearly the most essential ones within their respective groups. An explanation for the good performance of the *penalty* destroy operator might be, that it removes all assignments of the selected nurses at once. Thereby the forbidden shift successions and concatenation constraints are less restrictive as for the other repair operators.

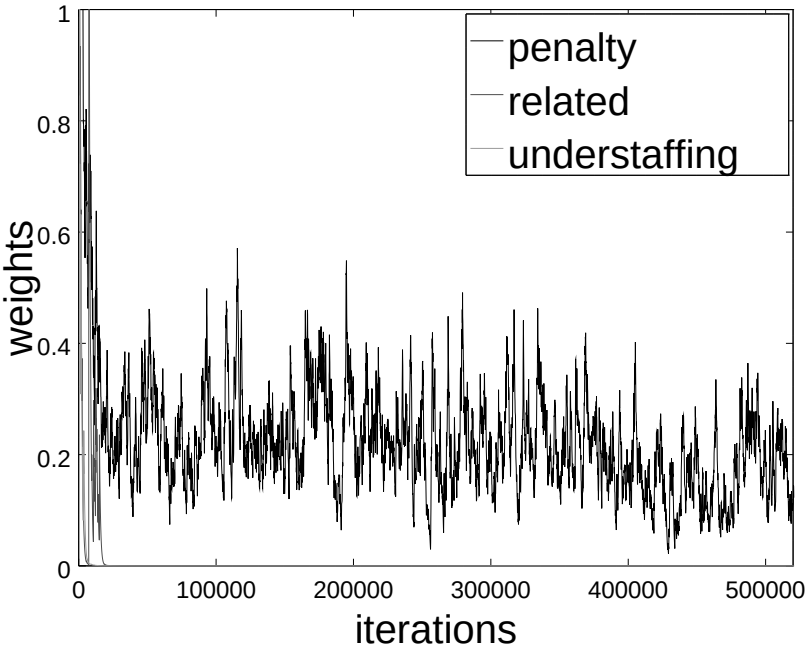
As pointed out by Ropke and Pisinger [2006], adding a noise term to the evaluation function of potential assignments is highly beneficial. Omitting this feature would lead to a deterioration of 34.94%, which is again based on two runs over each instance by applying the ALNS algorithm.

The development of the weights of the operators over iterations is depicted in Figure 3. The data is obtained by performing a single run of ALNS on instance 17 considering week 1. The series show that the weights of some operators converge to zero relatively fast, which coincides with the small selection rates of these operators stated in Table 8.

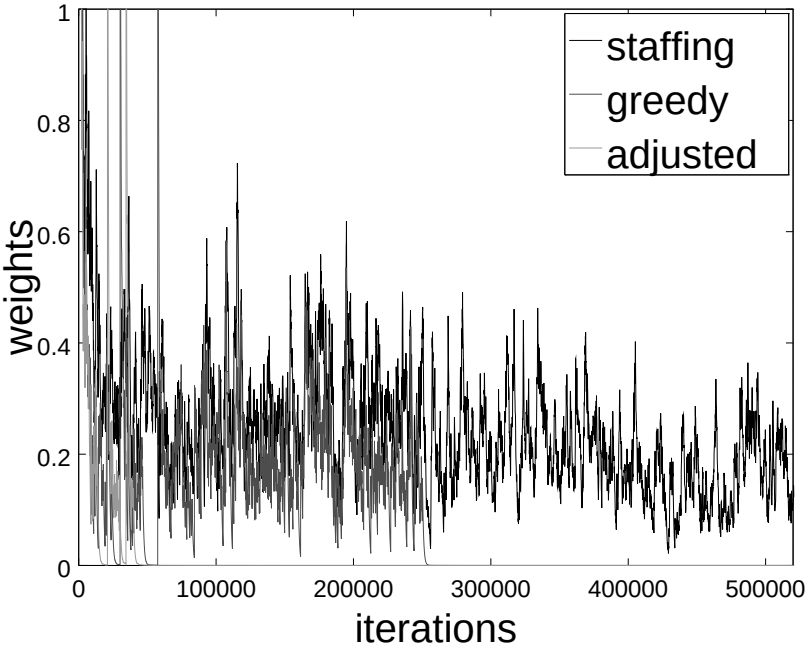
7 Conclusion

The thesis dealt with the nurse rostering problem with a special focus on the particular problem of the second nurse rostering competition. The competition was described in detail and background information about related competitions was provided. Moreover, several approaches that tackle nurse rostering problems were briefly surveyed. The main contributions of this thesis are the developed mixed integer programming model for the nurse rostering problem of the second competition and the metaheuristic solution approach.

The proposed approach is based on *Adaptive Large Neighborhood Search* by Ropke and Pisinger [2006], which is characterized by repetitive destructions and subsequent reparations of large parts of the roster. Furthermore, it incorporates an adaptive operator selection mechanism that encourages well-performing destroy and repair operators. Infeasible solutions with respect to one hard constraint are generally allowed but penalized in the objective function, whereby the penalty is adjusted dynamically. Generated solutions are accepted to become the new in-



(a) Destroy operators



(b) Repair operators

Figure 3: Progression of the weights, instance 17, week 1

cumbent solution according to a *Simulated Annealing* acceptance scheme. The penalty of potential assignments is perturbed by a noise term in the objective function, which proved to be an important feature. The final results were generated by a version of the algorithm that omits the adaptive mechanism but takes precomputed well-performing operator selection rates as an input.

Several destroy and repair operators were tailored to the considered problem. The most effective destroy operator focuses on removing the rosters of those nurses, whose assignments are responsible for most of the penalties of the incumbent solution. The best performing repair operator tries to reach a certain level of assignments for each day, each shift and each skill, first. Then, the operator schedules nurses by giving priority to those assignments that cause the least penalties.

The solution approach was compared to those of the 15 participants of the competition and performs approximately equally well as the ninth ranked method. With regard to an implementation in practice one has to note, that the proposed method incorporates several parameters that might be unattractive for practitioners. However, the algorithm seems to be very robust with respect to most of the parameters. Moreover, some operators may be removed without a substantial loss in performance, leading to a simpler and practically applicable approach.

References

K. Ahuja and J. B. Orlin. A survey of very large-scale neighborhood search techniques. *Discrete Applied Mathematics*, 123(1-3):75–102, 2002.

K. R. Baker. Workforce allocation in cyclical scheduling problems: A survey. *Operational Research Quarterly (1970-1977)*, 27(1):155–167, 1976.

G. Beddoe, S. Petrovic, and J. Li. A hybrid metaheuristic case-based reasoning system for nurse rostering. *Journal of Scheduling*, 12(2):99–119, 2009.

F. Bellanti, G. Carello, F. D. Croce, and R. Tadei. A greedy-based neighborhood search approach to a nurse rostering problem. *European Journal of Operational Research*, 153(1):28–40, 2004.

B. Bilgin, P. De Causmaecker, B. Rossie, and G. Vanden Berghe. Local search neighbourhoods for dealing with a novel nurse rostering model. *Annals of Operations Research*, 194(1):33–57, 2012.

P. Brucker, E. Burke, T. Curtois, R. Qu, and G. Vanden Berghe. A shift sequence based approach for nurse scheduling and a new benchmark dataset. *Journal of Heuristics*, 16(4):559–573, 2010.

P. Brucker, R. Qu, and E. Burke. Personnel scheduling: Models and complexity. *European Journal of Operational Research*, 210(3):467–473, 2011.

E. K. Burke and T. Curtois. New approaches to nurse rostering benchmark instances. *European Journal of Operational Research*, 237(1):71–81, 2014.

E. K. Burke, P. De Causmaecker, G. Vanden Berghe, and H. Van Landeghem. The state of the art of nurse rostering. *Journal of Scheduling*, 7(6):441–499, 2004.

E. K. Burke, T. Curtois, G. Post, R. Qu, and B. Veltman. A hybrid heuristic ordering and variable neighbourhood search for the nurse rostering problem. *European Journal of Operational Research*, 188(2):330–341, 2008.

E. K. Burke, J. Li, and R. Qu. A hybrid model of integer programming and variable neighbourhood search for highly-constrained nurse rostering problems. *European Journal of Operational Research*, 203(2):484–493, 2010.

S. Ceschia, N. T. T. Dang, P. D. Causmaecker, S. Haspeslagh, and A. Schaerf. The second international nurse rostering competition, 2015a. URL `http://mobiz.vives.be/inrc2/`. [accessed 2015-09-23].

S. Ceschia, N. D. T. Thanh, P. D. Causmaecker, S. Haspeslagh, and A. Schaerf. Second international nurse rostering competition (INRC-II) — problem description and rules —, 2015b. URL `http://arxiv.org/abs/1501.04177`.

B. Cheang, H. Li, A. Lim, and B. Rodrigues. Nurse rostering problems—a bibliographic survey. *European Journal of Operational Research*, 151(3):447–460, 2003.

M. Chiaramonte, J. Cochran, and D. Caswell. Nurse preference rostering using agents and iterated local search. *Annals of Operations Research*, 226(1):443–461, 2015.

P. De Causmaecker and G. Vanden Berghe. A categorisation of nurse rostering problems. *Journal of Scheduling*, 14(1):3–16, 2011.

F. Della Croce and F. Salassa. A variable neighborhood search based matheuristic for nurse rostering problems. *Annals of Operations Research*, 218(1):185–199, 2014.

A. T. Ernst, H. Jiang, M. Krishnamoorthy, and D. Sier. Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research*, 153(1):3–27, 2004.

B. E. Fries. Bibliography of operations research in health-care systems. *Operations Research*, 24(5):801–814, 1976.

M. Gendreau, A. Hertz, and G. Laporte. A tabu search heuristic for the vehicle routing problem. *Management Science*, 40(10):1276–1290, 1994.

M. Hadwan, M. Ayob, N. R. Sabar, and R. Qu. A harmony search algorithm for nurse rostering problems. *Information Sciences*, 233(0):126–140, 2013.

S. Haspeslagh, P. D. Causmaecker, M. Stølevik, and A. Schaerf. Nurse rostering competition, 2010a. URL `http://www.kuleuven-kulak.be/nrpcompetition`. [accessed 2015-09-23].

S. Haspeslagh, P. De Causmaecker, M. Stølevik, and A. Schaerf. The first international nurse rostering competition 2010. Technical report, KU Leuven Campus Kortrijk, 2010b.

S. Haspeslagh, P. De Causmaecker, M. Stølevik, and A. Schaerf. The first international nurse rostering competition 2010. *Annals of Operations Research*, 218 (1):221–236, 2014.

F. He and R. Qu. A constraint programming based column generation approach to nurse rostering problems. *Computers & Operations Research*, 39(12):3331–3343, 2012.

S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

R. Lewis. A survey of metaheuristic-based techniques for university timetabling problems. *OR Spectrum*, 30(1):167–190, 2008.

R. Lewis and J. Thompson. Analysing the effects of solution space connectivity with an effective metaheuristic for the course timetabling problem. *European Journal of Operational Research*, 240(3):637–648, 2015.

Z. Lü and J.-K. Hao. Adaptive neighborhood search for nurse rostering. *European Journal of Operational Research*, 218(3):865–876, 2012.

B. McCollum, A. Schaerf, B. Paechter, P. McMullan, R. Lewis, A. J. Parkes, L. Di Gaspero, R. Qu, and E. K. Burke. Setting the research agenda in automated timetabling: The second international timetabling competition. *INFORMS Journal on Computing*, 22(1):120–130, 2010.

H. E. Miller, W. P. Pierskalla, and G. J. Rath. Nurse scheduling using mathematical programming. *Operations Research*, 24(5):857–870, 1976.

T. Osogami and H. Imai. Classification of various neighborhood operations for the nurse scheduling problem. Technical Report 135, The Institute of Statistical Mathematics, 2000.

S. Petrovic and G. Vanden Berghe. A comparison of two approaches to nurse rostering problems. *Annals of Operations Research*, 194(1):365–384, 2012.

D. Pisinger and S. Ropke. A general heuristic for vehicle routing problems. *Computers & Operations Research*, 34(8):2403–2435, 2007.

D. Pisinger and S. Ropke. Large neighborhood search. In M. Gendreau and J.-Y. Potvin, editors, *Handbook of Metaheuristics*, volume 146 of *International Series in Operations Research & Management Science*, pages 399–419. Springer US, 2010.

G. Post, L. Di Gaspero, J. H. Kingston, B. McCollum, and A. Schaerf. The third international timetabling competition. *Annals of Operations Research*, pages 1–7, 2013.

S. Ropke and D. Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4): 455–472, 2006.

H. G. Santos, T. A. M. Toffolo, R. A. M. Gomes, and S. Ribas. Integer programming techniques for the nurse rostering problem. *Annals of Operations Research*, pages 1–27, 2014.

A. Schaerf. A survey of automated timetabling. *Artificial Intelligence Review*, 13 (2):87–127, 1999.

G. Schrimpf, J. Schneider, H. Stamm-Wilbrandt, and G. Dueck. Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, 159(2):139–171, 2000.

P. Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In M. Maher and J.-F. Puget, editors, *Principles and Practice of Constraint Programming - CP98*, volume 1520 of *Lecture Notes in Computer Science*, pages 417–431. Springer Berlin Heidelberg, 1998.

R. Silvestro and C. Silvestro. An evaluation of nurse rostering practices in the national health service. *Journal of Advanced Nursing*, 32(3):525–535, 2000.

P. Smet, B. Bilgin, P. De Causmaecker, and G. Vanden Berghe. Modelling and evaluation issues in nurse rostering. *Annals of Operations Research*, 218(1):303–326, 2014.

I. X. Tassopoulos, I. P. Solos, and G. N. Beligiannis. A two-phase adaptive variable neighborhood approach for nurse rostering. *Computers & Operations Research*, 60(0):150–169, 2015.

C. Valouxis, C. Gogos, G. Goulas, P. Alefragis, and E. Housos. A systematic two phase approach for the nurse rostering problem. *European Journal of Operational Research*, 219(2):425–433, 2012.

J. Van den Bergh, J. Beliën, P. D. Bruecker, E. Demeulemeester, and L. D. Boeck. Personnel scheduling: A literature review. *European Journal of Operational Research*, 226(3):367–385, 2013.

M. Vanhoucke and B. Maenhout. Nsplib: A nurse scheduling problem library: A tool to evaluate (meta-)heuristic procedures. In S. Brailsford and P. Harper, editors, *Operational research for health policy : making better decisions*, pages 151–165. Peter Lang, 2007.

M. Vanhoucke and B. Maenhout. On the characterization and generation of nurse scheduling problem instances. *European Journal of Operational Research*, 196 (2):457–467, 2009.

D. M. Warner. Scheduling nursing personnel according to nursing preference: A mathematical programming approach. *Operations Research*, 24(5):842–856, 1976.

D. M. Warner and J. Prawda. A mathematical programming model for scheduling nursing personnel in a hospital. *Management Science*, 19(4-part-1):411–422, 1972.