MSc Economics

Revenue-Maximizing Combinatorial Auctions in a Simplified Setting

A Master's Thesis submitted for the degree of
"Master of Science"

supervised by

Martin Meier

Béla Szabadi

1026546

Vienna, 8 June 2012

MSc Economics

Affidavit

I, Béla Szabadi

hereby declare

that I am the sole author of the present Master's Thesis,
Revenue-Maximizing Combinatorial Auctions in a Simplified Setting

69 pages, bound, and that I have not used any source or tool other than those referenced or any other illicit aid or tool, and that I have not prior to this date submitted this Master's Thesis as an examination paper in any form in Austria or abroad.

Vienna, 8 June 2012

Signature

Contents

List of Figures

List of Tables

List of Appendices

Abstract

Combinatorial auctions are multi-object auctions where bidding on packages of objects is allowed. These auctions have attractive features and both the bidders and the auctioneer can benefit from the combinatorial bids. However, the rich structure could lead to serious issues. The computational complexity related to these auctions could make it impossible to implement them even with a moderate number of goods. Moreover, finding a revenue-maximizing auction is still an open problem. The usual approach of the literature is to make (sometimes too strict) assumptions (usually on the bidders' preferences) or to run simulations in very small environments where the computational difficulties are not too serious.

In this paper, besides presenting the issues and their treatments proposed by the literature, we also investigate an alternative approach. Instead of making assumptions on the preferences, we restrict the set of available packages in a reasonable way. We present the idea through a real-life example and use simulations to determine the implications of the imposed restriction. We find that the alternative approach makes the situation simple enough and we can run larger simulations without huge difficulties. With these simulations we obtain important insights into the behavior of the applied auction forms and their revenue properties. Since the imposed structure is reasonable in other relevant auction situations too, this work can serve as a ground for future research that generalizes these conclusions.

1 Introduction

Consider the following situation. We possess a series of rare postage stamps that we want to sell. Among the many ideas that could possibly come into our mind, organization of an auction is surely there. It seems to be a good and profitable way of selling the stamps to bring some potential buyers together and let them fight for the items with their bids. However, it is not straightforward how we should set the rules of such an auction. Should we sell every item individually on different smaller auctions at the same time? Maybe in a sequential order? Or would it be a better idea to try to sell everything in one huge package? Our stamps are special in the sense that they are collector's items. We have good reasons to assume that the collectors of these stamps might be willing to pay more for more stamps together than the sum of what they would pay for the individual items. There might be synergies between the stamps. From this point of view, first, we would prefer selling everything in one large bundle. However, we cannot be sure that our potential buyers do not possess some items from the series already. It might happen that one of them needs only half of the stamps, another buyer the other half and they do not want to pay much more for the whole package. But then why don't we let them bid on every combination they might want to have? In that way we may be able to utilize the synergies between the stamps and find an allocation which could maximize our income. This would be also beneficial on the bidders' side, since they could avoid very unfortunate outcomes by bidding on packages. For example, if a bidder needs only three stamps to complete his collection, he might want to pay more for a package which contains all of them than for smaller packages which contain only one or two of those desired stamps. He could express this in higher bids on the former package and lower bids on the latter ones. Whatever happens, he will not risk too much in the auction. However, in an auction where bidding on packages is not allowed, he should post bids exceeding his item-valuations to increase his chance of obtaining the three items together. In this case, losing one out of the three could lead to a disaster: He would pay more for the remaining two than what those are really worth to him. Thus in this example, bidding on packages seems to be highly beneficial for both sides.

Now we have the basic idea of how we want to auction off these items, but there are some other details left that need to be solved. First, after we collected the bids, how should we allocate the goods and determine the payments? We can choose the allocation which gives the largest overall bid combination and we can simply charge those bids like we pictured it before. But are we sure that there is no better way of doing this? Did we create the incentives for the bidders to tell the truth in our auction? Are there other rules which could result in a higher income? Secondly, even a small number of stamps could lead to a huge number of possible packages. Are we able to choose the winning packages if the number of possible package allocations is so huge? Unfortunately the answers to these are not straightforward and we need to use and develop economic models to be able to solve these questions.

Such procedures are called combinatorial auctions in the literature. One of the earliest appearances of such auctions is Rassenti et al. [1982]. They used combinatorial auctions for the allocation problem of airport time slots. In this context, the idea of having packages is quite natural since airlines want to have working flight schedules and thus are interested only in certain combinations of time slots. In the past decades, combinatorial auctions became very popular within the field of multi-item auction design. The survey paper of de Vries and Vohra [2003] presents the main contributions and the status of the combinatorial auction literature at the beginning of the 21st century. Vohra [2011], which is a recently published book on mechanism design, devotes an entire chapter to combinatorial auctions. Cramton et al. [2010] is a 1000-page book dedicated only to this type of auctions. The reason for this popularity is that there is a huge number of difficult and in some cases unresolved problems associated to this type of auctions.

First, there is a computational difficulty since the rich structure of packages becomes impossible to handle even with a reasonable number of items. Secondly, there are issues related to economic theory: the bidders' behavior and the incentives we need to create are challenging questions too. It is not clear either and still an unsolved problem how we should set rules to reach maximal revenue in the auction.

The computational difficulty can be reduced by imposing extremely strict restrictions on the preferences. The bidders' behavior, the appropriate incentives and the role of the rules can be modeled and investigated by the tools of game theory and mechanism design. Although there is no general solution to the maximal revenue problem, we can improve the properties of existing mechanisms by small modifications.

In this paper, we would like to discuss the aforementioned issues and their proposed solutions in detail and use an alternative approach to overcome the difficulties. In section 2 we give a formal description of the model. Section 2.1 presents the economic model we work with. Section 2.2 discusses the issues with combinatorial auctions and their proposed solutions in the literature. Section 2.3 presents particular auction mechanisms and discusses their properties. We depart from the usual approach of the literature in section 3 and instead of making hard assumptions on the bidders we restrict the sets of possible packages in a reasonable way. We discuss the implications of this and present them through simulations. Section 4 concludes the paper.

2 Combinatorial Auctions

In this part we describe the auction environment and the key notions behind combinatorial auctions. After this, we discuss the issues related to combinatorial auctions and their possible solutions in the literature. At the end of the section, a typical class of auctions and their properties are presented in detail.

2.1 The Model

In our world we have finitely many players called bidders, agents or potential buyers and one decision maker called the seller, the auctioneer or the designer of the auction. Let the set of bidders be $\mathcal{N} = \{1, \ldots, N\}$[1], the auctioneer be denoted as player 0 and the set of the bidders with the auctioneer be defined as $\mathcal{N}^* \doteq \mathcal{N} \cup \{0\}$.

The auctioneer possesses finitely many heterogeneous goods $\mathcal{G} = \{1, \ldots, G\}$ and wants to sell them to the bidders in a way that implements some goal like expected revenue maximization. For simplicity, we assume that the auctioneer gets no utility from possessing any of his goods. On the other side, the bidders assign non-negative valuations to each object. Moreover, goods consumed together might give a different utility to a bidder than the sum of utilities gained by consumption of the parts. There might be synergy effects between the goods. Because of these interdependencies, the bidders' valuations are defined on the set of possible packages $\mathcal{H} \subseteq 2^{\mathcal{G}}$ and not only on the set of goods $\mathcal{G}$. [2] We also assume that receiving nothing is possible, thus $\emptyset \in \mathcal{H}$, and the associated valuation is zero in every case. The valuations on the set $\mathcal{H}$ are represented by valuation functions $v_n \colon \mathcal{H} \to \mathbb{R}$. For every $n \in \mathcal{N}$, let the set of all possible valuation functions be denoted by $\mathcal{V}_n$. The elements of this set are also called types of the bidders. We assume that at the beginning of the game a type is drawn independently for every bidder according to probability density/mass functions $p_n \colon \mathcal{V}_n \to [0, 1]$. The distributions are common knowledge, but the actual realizations are private information. Since the draws are independent, no one gets additional information about the distribution of

[1] In our work we denote such index sets by calligraphic letters ($\mathcal{N}$) and their largest elements by the same non-calligraphic capital letter (N). We denote an arbitrary element by the lower-case equivalent ($n \in \mathcal{N}$). In this way it is much easier to follow the notation and talk about cardinality of sets (since $|\mathcal{N}| = N$).

[2] The set of possible packages might be restricted by the auctioneer for some reasons, that is why we use a subset $\mathcal{H}$ of all packages $2^{\mathcal{G}}$.

the others after learning his valuation. Let the set of all valuation profiles be denoted by $\mathcal{V} \doteq \times_{n\in\mathcal{N}}\mathcal{V}_n$.

The bidders might make transfers during the auction and we suppose that their utility function is linear in money: for each bidder $n \in \mathcal{N}$, $u_n \colon \mathcal{H} \times \mathbb{R} \to \mathbb{R}$ such that for all $h \in \mathcal{H}$ and $t \in \mathbb{R}$, $u_n(h,t) = v_n(h) + t$.

The auctioneer chooses the rules of the auction. First of all, he defines what the bidders are allowed to do during the process. This means that he assigns a strategy set $\mathcal{S}_n$ to each player $n \in \mathcal{N}$ which contains the player's available strategies. Let the set of all strategy profiles be denoted by $\mathcal{S} \doteq \times_{n\in\mathcal{N}}\mathcal{S}_n$. Second of all, he can define the consequences of the joint behavior of the players. This means that given each strategy profile $s \in \mathcal{S}$, he chooses a possible distribution of the goods in $\mathcal{G}$ and charges payments for this allocation. Formally, the set $\mathcal{A}$ of all possible allocations of the objects is defined by

$$\mathcal{A} \doteq \left\{(h_0,\ldots,h_N) \in 2^{\mathcal{G}} \times \mathcal{H}^N : \bigcup_{n=0}^{N} h_n = \mathcal{G} \text{ and } h_n \cap h_m = \emptyset, \forall n,m \in \mathcal{N}^*, n \neq m\right\}.$$

The elements of $\mathcal{A}$ are basically partitions of the set of goods $\mathcal{G}$, such that the first element denotes the goods not sold at the end of the auction and every element h_n gives the package allocated to bidder n.[3] In the text, we will use the $a_n = h_n$ notation as well to refer to the package that is allocated to $n \in \mathcal{N}^*$ under allocation a. As we mentioned before, the fact that there are possible restrictions on the packages sold to the bidders is governed by the choice of the set $\mathcal{H}$. Now, the first rule, called an allocation rule, can be defined as a function from the space of strategy profiles into the set of possible allocations: $A \colon \mathcal{S} \to \mathcal{A}$. The second object, called the payment rule, is simply a function from the set of strategy profiles to the N-dimensional real vectors: $P \colon \mathcal{S} \to \mathbb{R}^N$.

Definition 1 The list describing the rules of the auction $(\mathcal{S}, A, P)$ is called a game form. A game form together with the list of possible preference profiles, $(\mathcal{S}, A, P; \mathcal{V})$, is called an auction mechanism.[4]

Notice that the game form $(\mathcal{S}, A, P)$ is different from the concept of a normal form game. The outcomes here are packages and payments, not utility

[3] We call everything that a bidder receives a package. In this context it makes no sense to talk about receiving more than one packages since the valuation of the obtained goods will come from the obtained goods together and not from the individual packages independently.

[4] Thus, an auction mechanism is just a special case of a general mechanism defined in mechanisms design. Throughout this paper, we use the expressions auction mechanism and auction as synonyms. They are the tools to achieve the goal of the auctioneer.

levels. Thus, for every valuation combination $v \in \mathcal{V}$, this structure defines different utility levels over the outcomes and thus a different normal form game. Therefore, the players' behavior might be also different for different valuation profiles. In this context, an auction mechanism is a family of normal form games.

Within this framework, an auction can be organized as follows:

1. The set of players $\mathcal{N}^*$, the set of goods $\mathcal{G}$, the set of available packages $\mathcal{H}$, the valuation profiles $\mathcal{V}$ and the pmfs/pdfs $(p_n)_{n\in\mathcal{N}}$ are given.
2. The valuation functions $(v_n)_{n\in\mathcal{N}}$ of the bidders $\mathcal{N}$ are drawn from the distributions described by the pmfs/pdfs. Everyone knows these distributions, but the actual realizations are kept private.
3. Given $\mathcal{N}$ and the beliefs of the auctioneer, he declares the rules of the auction. This means that he chooses the strategy sets, the allocation and the payment rules ($\mathcal{S}$, A and P respectively) and tells them to the bidders.
4. Given the objects defined so far, the bidders choose their strategies (make their bids) which results in a strategy profile $s \in \mathcal{S}$.
5. The auctioneer allocates the goods according to $A(s)$ and collects the payments $P(s)$ from the players.

2.1.1 The Bidders' Behavior

We will use game theoretical equilibrium concepts to model the behavior of the bidders in an auction mechanism. In a given auction mechanism $(\mathcal{S}, A, P; \mathcal{V})$, let $E(\mathcal{S}, A, P; \cdot)\colon \mathcal{V} \rightrightarrows \mathcal{S}$ denote the equilibrium correspondence of some equilibrium concept. Thus, every strategy combination $s \in E(\mathcal{S}, A, P; v)$ is an equilibrium of the game $(\mathcal{S}, A, P; v)$. We call this abstract construction the E equilibrium concept. We intend to design auctions such that for every $v \in \mathcal{V}$, at least one equilibrium exists.

In this work, we use only the dominant strategy equilibrium therefore we define this equilibrium concept here. For all $s \in \mathcal{S}$ and for every player $n \in \mathcal{N}$, let $a_n(s)$ denote the package assigned to n by the allocation rule A.

Definition 2 (Dominant strategy equilibrium) In a game $(\mathcal{S}, A, P; v)$, a strategy profile $s \in \mathcal{S}$ is a dominant strategy equilibrium if for all players $n \in \mathcal{N}$ the following holds:

$$v_n(a_n(s_n, \tilde{s}_{-n})) - P_n((s_n, \tilde{s}_{-n})) \geqq v_n(a_n(\tilde{s}_n, \tilde{s}_{-n})) - P_n((\tilde{s}_n, \tilde{s}_{-n})) \text{ for all } \tilde{s} \in \mathcal{S}.$$

In the definition, $\tilde{s}_{-n}$ denotes the strategies of the players except n in the strategy profile $\tilde{s}$. The definition simply says that in a dominant strategy equilibrium, every player plays a strategy which is better than all his other strategies irrespective of the other players' behavior. In other words, at every strategy profile every player has an incentive to deviate to his equilibrium strategy.

2.1.2 Objectives of the Auctioneer

The auctioneer has to design an auction which helps him to fulfill his goals. Of course these goals could be very different, but in this work we will stick to the two main concepts of the auction literature: efficiency and optimality.

The objectives can be conveniently defined by using the terminology of mechanism design. To be able to do this, first we need to introduce some concepts. A correspondence $A_d \colon \mathcal{V} \rightrightarrows \mathcal{A}$ is called a direct allocation rule. For the definition of the objectives, we will select direct allocation rules and construct mechanisms that have equilibrium outcomes which are consistent with these direct allocation rules. This is captured in the following definition.

Definition 3 (Weak implementation) Given the direct rule $A_d \colon \mathcal{V} \rightrightarrows \mathcal{A}$, we say that the auction mechanism $(\mathcal{S}, A, P; \mathcal{V})$ weakly implements A_d in the equilibrium concept E if for all possible valuation combination $v \in \mathcal{V}$, there is a strategy profile $s \in E(\mathcal{S}, A, P; v)$ such that $A(s) \in A_d(v)$.

Thus, for every possible valuation profile, there is an equilibrium outcome which is consistent with the direct allocation rule. Of course this does not imply that there are no other equilibrium outcomes that are not consistent with this rule and might be considered to be very bad. However, we are not concerned with this issue since the theory of combinatorial auctions usually sticks to this definition and the implications of changing to a stronger concept would lead us too far. Now we are ready to discuss the goals of the auctioneer.

Efficiency In the first case, the auctioneer wants to allocate the goods in an efficient way, which means that for every valuation profile $v \in \mathcal{V}$, he is willing to maximize the sum of the individual valuations of an allocation. This allocation can be found by solving the following problem:

$$\max_{a=(a_0,\ldots,a_N)} \sum_{n\in\mathcal{N}} v_n(a_n)$$
$$\text{s.t.} \quad a \in \mathcal{A}$$

This problem always has a solution since with finitely many bidders and goods, the number of all possible allocations is bounded from above by N^G. Moreover, the set of possible allocations is never empty by our assumptions since the auctioneer can always keep his goods. Thus, finding an efficient allocation can be done by calculating the overall utility level for all possible allocations and choosing the best one.

Although writing down all the possible allocations is an exciting combinatorial exercise, this may not be the simplest and most efficient way to solve this problem. Luckily, we can characterize the set $\mathcal{A}$ by linear inequalities and use the following integer programming problem to find the utility-maximizing allocation:[5] [6]

$$\begin{aligned} \max_{\boldsymbol{y}} \; & \sum_{n\in\mathcal{N}} \sum_{h\in\mathcal{H}} v_n(h)\, y(h,n) \\ \text{s.t.} \; & \sum_{n\in\mathcal{N}} \sum_{h\ni g} y(h,n) \leqq 1 \qquad \forall g \in \mathcal{G} \\ & \sum_{h\in\mathcal{H}} y(h,n) \leqq 1 \qquad \forall n \in \mathcal{N} \\ & y(h,n) \in \{0,1\} \qquad \forall h \in \mathcal{H}, n \in \mathcal{N} \end{aligned}$$

Here the 1 value of the $y(h,n)$ variable represents the fact that the package h is assigned to player n. The first group of constraints ensures that no assigned packages overlap, i.e. every good is assigned at most once. The second group guarantees that every player can get at most one package. Again, this is an important condition since otherwise one could end up with more than one package and the utility gained from these (the utility of the union) would not

[5] As defined in Vohra [2011].

[6] In the paper, we denote vectors and matrices by bold letters. Obviously, some previously defined objects, like the valuation functions, can also be represented by vectors; v and $\boldsymbol{v}$ mean basically the same thing.

be the same as the utility represented in the objective function (sum of the utilities).

Note that maximizing the overall utility of the bidders from an allocation is the same as maximizing the social welfare since the payments are only transfers from the bidders to the auctioneer and the auctioneer values every package 0.

This problem defines a direct auction rule over $\mathcal{V}$. Call this the efficient allocation rule and denote it by A_e. An efficient mechanism is one which weakly implements this allocation rule in an equilibrium concept E. This means that in every valuation profile $v \in \mathcal{V}$, there is a strategy combination of the bidders which is consistent with our equilibrium concept and leads to a socially efficient outcome. We will show in section 2.3 that such a mechanism exists.

Optimality Revenue maximization is harder to achieve. In this case, we are interested in the maximization of our expected income from the auction. This is a non-trivial problem since a mechanism that weakly implements the efficient allocation rule does not give us the largest possible revenue in general. Thus, we might have to find a different direct allocation rule A^* that can be weakly implemented by a mechanism where the expected payments associated to the implementing strategies are maximal among all direct allocation rules and auctions weakly implementing these. Formally, we are searching for the solution of the following optimization problem:

$$\begin{aligned}
\sup_{\mathcal{S},A,P,A^*,s(v)} \quad & \mathbb{E}_v \sum_{n \in \mathcal{N}} P_n(s(v)) \\
\text{s.t.} \quad & s(v) \in E(\mathcal{S}, A, P, v) \qquad \forall v \in \mathcal{V} \\
& A(s(v)) \in A^*(v) \qquad \forall v \in \mathcal{V}.
\end{aligned}$$

We will discuss the goals and the difficulties in section 2.2 in detail. In the following we show how the difficulties can be reduced a bit by a famous result of mechanism design.

2.1.3 The Revelation Principle

Whatever our goal is, finding an implementing mechanism could be very difficult. We have to set free objects: the strategy sets $\mathcal{S}$, the allocation rule A and a payment rule P. Fortunately, the revelation principle of mechanism design can help us to simplify these problems by fixing the strategy sets without losing generality. First, we need to define a few things. We call an auction mechanism $(\mathcal{S}, A, P; \mathcal{V})$ a direct auction mechanism if $\mathcal{S} = \mathcal{V}$ holds. In a direct mechanism, every bidder announces his type, and for every possible type combination, an allocation and a payment scheme is assigned. In such auctions we can define a new notion of weak implementation.

Definition 4 (Truthful implementation) We say that the direct mechanism implements a direct allocation rule $A_d \colon \mathcal{V} \rightrightarrows \mathcal{A}$ truthfully in the equilibrium concept E, if $v \in E(\mathcal{V}, A, P; v)$ and $A(v) \in A_d(v)$ for all $v \in \mathcal{V}$.

Thus, telling the truth is an equilibrium strategy. If a mechanism satisfies this, we alternatively say that it is incentive compatible (IC). From now on, we completely restrict our attention to dominant strategy equilibria. The following result, in the form as stated in Myerson [1981], establishes a useful relation between weakly implementing auction forms and truthfully implementing direct auctions.

Theorem 5 (Revelation Principle) Let $A_d \colon \mathcal{V} \rightrightarrows \mathcal{A}$. If there exist an auction mechanism which weakly implements A_d in dominant strategy equilibrium then there exists a direct auction mechanism which implements it truthfully in dominant strategy equilibrium, giving the auctioneer the same expected revenue and the bidders the same expected utilities as the original mechanism.

Thus, we can focus our search on direct mechanisms. As we will see, while the first goal (efficiency) is straightforward to achieve, optimality is still an unsolved problem in the theory of multi-object auctions. Finally, we can impose a second requirement for a truthfully implementing direct auction mechanism. By incentive compatibility, we created the incentives to tell the truth but there is no guarantee that this strategy will give nonnegative expected payoff to the bidders. In most of the auction settings the bidders cannot be forced into participation. Individual rationality (IR) establishes the incentives for that. It requires that, under a given allocation rule A and payment rule P, the expected utility of telling the truth be non-negative: for all bidder n and $v_n \in \mathcal{V}_n$,

$$\mathbb{E}_{\tilde{v}_{-n}} \left(v_n(a_n(v_n, \tilde{v}_{-n})) - P_n((v_n, \tilde{v}_{-n})) \right) \geqq 0.$$

From the revelation principle we know that if the original mechanism resulted in a non-negative expected utility for a player, then the direct counterpart will do the same. Thus the IR property is also implied by the similar property of the original auction and the search among direct auction mechanisms is justified with this additional requirement too.

2.2 Issues with Combinatorial Auctions

The idea behind combinatorial auctions is fascinating and these mechanisms have several attractive features. However, these auctions are plagued with severe issues and the designer of the auction has to face a lot of difficulties. These will be discussed in this section of the paper.

2.2.1 Dimensions of the Problem

The first group of issues is related to the specific structure of the problem. First, if we consider direct auction mechanisms and allow participants to bid on packages of goods, the number of objects that can be bid on in the auction will significantly increase. If we auction off 20 goods (which, in real life auctions, is not an uncommon case) and bidding on every subset is possible, then every bidder has to make $2^{20} = 1048576$ bid decisions. This would be more than challenging in real-life situations.

The second problem is that after collecting the bids from every player, the auctioneer has to use this huge valuation profile to determine the allocation and the payments. Of course the process will depend on the defined A and P functions, but generally, it is not easy to determine the allocation and the payments and the process usually involves complicated mathematical optimizations. For example, if we want to implement efficient allocations in an incentive compatible auction (where we have reasons to believe that the bidders will tell the truth), we have to solve the problem defined in section 2.1.2 after we received the bids. With 20 goods and 5 players, this means $5 \cdot 2^{20} = 5242880$ variables which raises some questions regarding the possibility of solving these problems computationally.

2.2.2 Strategies

Obviously, given the vast number of possible bids, the strategical aspects become much more complicated too. How should a participant choose his bids to reach the best possible outcome? Can he use bids on less appreciated subsets to improve his chances at other packages? Unfortunately, the characterization of equilibrium strategies of different combinatorial auctions is still an unresolved problem in the literature.[7] This difficulty is also a reason why the literature mostly focuses on incentive compatible direct auctions and the weak notion of implementation.

2.2.3 Complexity Issues

We already mentioned computational difficulties. Now we go a bit deeper and discuss the properties of the optimization problem used to determine welfare-maximizing allocations. This mathematical programming problem is widely used in the theory of combinatorial auctions and has a central role in all of the auctions presented in this paper later. The problem is repeated here for convenience. After the bidders made their announcement $v \in \mathcal{V}$, the auctioneer has to choose the winning allocation in some way. This means that he has to solve the following problem:

$$\begin{aligned}
\max_{y} \; & \sum_{n \in \mathcal{N}} \sum_{h \in \mathcal{H}} v_n(h)\, y(h,n) \\
\text{s.t.} \; & \sum_{n \in \mathcal{N}} \sum_{h \ni g} y(h,n) \leqq 1 \qquad \forall g \in \mathcal{G} \\
& \sum_{h \in \mathcal{H}} y(h,n) \leqq 1 \qquad \forall n \in \mathcal{N} \\
& y(h,n) \in \{0,1\} \qquad \forall h \in \mathcal{H}, n \in \mathcal{N}
\end{aligned}$$

This is an instance of the famous Set Packing Problem (SPP). The SPP is a binary linear programming problem where given a family of subsets of an underlying set (here the sets $\mathcal{H}$ and $\mathcal{G}$ respectively) and a family of associated weights (here the v_n valuation functions), we want to select a non-overlapping collection which maximizes the sum of the weights of the selected subsets. As we discussed above, this problem always has a solution. Unfortunately finding it is not easy in the computational sense: if we increase the size of the problem, the required time for reaching the solution increases too fast (in a non-polynomial manner).[8] We will see in the next section that auction mechanisms usually use more than one SPP and we could easily end up in a computationally infeasible situation even with a relatively small number of goods.

[7] For a discussion on this topic, see Andersson and Wilenius [2009].

[8] For a formal discussion see Sandholm [2002].

In the following it will be convenient to use the matrix-vector formulation of the problem:

$$\begin{aligned} \max_{\boldsymbol{y}} \quad & \boldsymbol{v}\boldsymbol{y} \\ \text{s.t.} \quad & \boldsymbol{A}\boldsymbol{y} \leqq \mathbf{1} \\ & \boldsymbol{y} \in \{0,1\}^{HN}, \end{aligned}$$

where $\boldsymbol{A}$ is a 0-1 matrix representing the first two groups of constraints and $\mathbf{1}$ is a vector full of ones with the appropriate number of dimensions.

It is important to see that the reason for this computational difficulty is the fact that the variables are binary. The linear relaxation of the problem can be defined as the following linear programming problem:

$$\begin{aligned} \max_{\boldsymbol{y}} \quad & \boldsymbol{v}\boldsymbol{y} \\ \text{s.t.} \quad & \boldsymbol{A}\boldsymbol{y} \leqq \mathbf{1} \\ & \boldsymbol{y} \geqq \mathbf{0}. \end{aligned}$$

The only difference between the two problems is that we replaced the last constraint with a non-negativity constraint where $\mathbf{0}$ is a vector full of zeros. From the theory of linear programming we know that every such problem is solvable in polynomial time and thus an increase in the size of the problem does not lead to an explosion in the required time. If we could show that the two solution sets coincide then we could use the linear relaxation to weaken the computational difficulties. Unfortunately, this is not the case in general, but there are some conditions that can ensure the equivalence between the solution sets.

It can be immediately seen that this new constraint combined with the first two ensures that all the variables are between zero and one. From linear programming we also know that if an optimal solution exists, then there is an optimal solution which is an extreme point of the polyhedron defined by the inequalities of the problem.[9] Moreover, every extreme point can be represented as a solution of the problem with some nonzero $\boldsymbol{v}$ vector in the objective function. Thus, what we actually need is to find conditions on the problem that

[9] An extreme point is a vector which cannot be constructed as a convex combination of two other vectors from the solution set. In other words, it is a vertex of the polyhedron.

ensure that all of our extreme points will be binary.[10] A coefficient matrix $\boldsymbol{A}$ with this property is called perfect in the literature. It can be shown, that the next definition is a characterization of perfect matrices. [11]

Definition 6 (Perfection of a matrix) A 0-1 matrix $\boldsymbol{A}$ is called perfect if it contains no $m \times m$, $m > 3$ submatrix $\boldsymbol{B}$ that satisfies the following properties:

- the row and column sums of $\boldsymbol{B}$ are all equal to the same $k \geqq 2$;
- if we consider those columns in $\boldsymbol{A}$ where the elements of $\boldsymbol{B}$ were taken from, there is no row in this submatrix which has a row sum greater than the number k.

The set of perfect matrices is precisely the class of matrices where the polyhedron associated to the corresponding SPP has only binary extreme points. If we have such matrices, the linear relaxation of the SPP can be used and thus the original problem can be solved in polynomial time. [12]

Determining whether a matrix is perfect could be very difficult since we should prove somehow that every such submatrix violates at least on of the properties mentioned in Definition 6. Luckily, there are other notions that imply perfection and are much easier to identify. Such concepts are the consecutive ones property, total unimodularity and balancedness. Since we will use only the first one, only this is defined here. For a definition of total unimodularity and balancedness, see for example de Vries and Vohra [2003].

Definition 7 (consecutive ones property) We say that the 0-1 matrix $\boldsymbol{A}$ satisfies the consecutive ones property if the rows of $\boldsymbol{A}$ can be reordered in a way that in each column of the matrix, the 1 elements are consecutive.

It can be shown that the consecutive ones property implies total unimodularity which implies balancedness which leads to perfection. If we could show that our coefficient matrix satisfies any of these properties, we could use the linear relaxation and avoid much of the computational difficulties.

In general, the coefficient matrix $\boldsymbol{A}$ is not perfect. However, it can be shown that if we impose some (sometimes too strong) restrictions on the structure of

[10] Or that the non-integral extreme points can be optimal only with the valuation profiles outside $\mathcal{V}$, but we do not investigate this in this paper.

[11] See Padberg [1974] for more details.

[12] There are other special cases where there is a polynomial algorithm that can be used for the SPP, but we do not use these concepts in this paper. For a detailed account, see de Vries and Vohra [2003].

the packages and on the preferences of the bidders, we can reformulate the set packing problem in a way such that the new coefficient matrix fulfills at least one of these properties. For such restrictions, see Rothkopf et al. [1998].

2.2.4 Goals of the Auction

As we discussed before, the usual goals that an auction should achieve either is efficiency or revenue-maximization. Of course, the different aims need different approaches and different incentives for the players during the auction.

Efficiency is the simpler case. We will show in the next section that the problem of designing an efficient mechanism has been already solved in the literature and there are auction forms with attractive features that could implement the efficient allocation of the goods. Such an auction mechanism is the celebrated Vickrey–Clarke–Groves (VCG) mechanism which could internalize society's goals by charging the bidders for the damage they cause to other players by participating in the auction. Given these payments, this mechanism can be considered as the generalization of the second-price auctions to the multi-object case.

Unfortunately, optimality is much harder to achieve. Myerson [1981] solved the problem in the single good case but his results (construction of the optimal mechanism and revenue equivalence) cannot be generalized to combinatorial auctions in a straightforward way. The general solution to this problem has not been found yet. The existing results in the literature are either based on oversimplifications and too strong assumptions or simulations involving only a small number of goods given the computational complexity of the problem. We have seen that the problem can be formulated as an optimization problem: We should choose an allocation rule that can be implemented by an incentive compatible and individually rational auction with the largest expected payments.

This problem becomes "simple" if we have finitely many profiles in $\mathcal{V}$. In this case, there are only finitely many allocation rules ($\mathcal{V} \to \mathcal{A}$ functions) and finding a revenue-maximizing payment scheme for a given allocation rule such that IC and IR hold is just a linear programming exercise.[13] We could go over all the possible payment rules and compare the expected revenues at the end. However, this is impossible even in very small environments. Consider the case of 4 goods and 3 players where every player has binary valuations

[13]See de Vries and Vohra [2003] for more details.

for every good and the package valuations are calculated from these without any additional uncertainty. This means that every player has $2^4 = 16$ types, thus the number of type combinations is $16^3 = 4096$. There are no restrictions on the set of packages which means that we have $3^4 = 81$ possible allocations. However, the number of allocation rules (again, $\mathcal{V} \to \mathcal{A}$ functions) is $81^{4096} = 1.427701207 \cdot 10^{7817}$ which makes the above mentioned way impossible to use.

One thing is clear: We can give an upper bound of the revenue that can be achieved by such mechanisms. Since we assume that individual rationality holds, no one can have a larger expected payment than his total expected utility from the possible allocations. Thus, the expected revenue is bounded from above by the expected utility from the efficient allocation of goods which is simply the expected value of the value function of the SPP defined above. However, we have to give the right incentives to the bidders to tell the truth. Thus, there is no guarantee that there exist a mechanism which could extract all these expected valuations as an expected revenue.

There are only few analytical results in the literature. Levin [1997] used the assumption that the preferences of the bidders are known up to a single number, which collapsed the problem into a one-dimensional mechanism design problem he was able to solve. The optimal mechanism here had a similar second-price property as the VCG mechanisms. Armstrong [2000] derived the optimal auction in an other very simple environment with two bidders, two goods and no synergies. Monderer and Tennenholtz [2005] proved an interesting feature of the VCG mechanism: with symmetric bidders, it is asymptotically revenue maximizing and its expected revenue converges to the theoretical maximum.

Other studies in the literature, such as Krishna and Rosenthal [1996] and Andersson and Wilenius [2009], tried to compare the revenue properties of multi-object auctions of different types. Both considered environments where the bidders can be divided into two groups: local bidders who are interested in single objects and global ones who like multiple objects together and have some additional utility by consuming packages. The former investigated very small environments and reached the conclusion that combinatorial auctions are not revenue superior in these cases compared to simultaneous auctions while the second paper concluded the opposite in a slightly different setting by using combinatorial reasoning to construct lower and upper bounds on the revenues and separating the two possible revenue sets.

Some papers in the literature gave up the idea of finding an optimal auction and investigated how existing auction mechanisms can be modified in a way that they still satisfy incentive compatibility and individual rationality but they are able to reach higher revenues. Krishna and Perry [1998] generalized the VCG mechanism in a way that the new auction became revenue-maximizing among efficient auctions. Likhodedov and Sandholm [2004] and Likhodedov and Sandholm [2005] used a computational approach: they tried to mimic Myerson's idea and replaced the type announcements by modified "virtual" valuations in an incentive compatible way in the VCG mechanism. They introduced a larger class of mechanisms which could be described by additional parameters and contained the VCG as a special case. By a numerical optimization in this parameter space they were able to give the VCG large revenue boosts in most of the cases. We will describe these mechanisms in this paper in detail.

2.2.5 Other Common Problems in Auction Design

There are other issues that every auction form is vulnerable to and combinatorial auctions are not an exception either. Klemperer [2002] provides a good survey on the issues auction designers should care about when designing auctions in the real world. The main idea is that bidders basically behave like firms on the market. The stronger bidders make bid signals in order to deter weaker ones from entering the auction or to show their willingness to share the goods with other bidders and reduce competition. Sometimes they form coalitions, coordinate their actions and keep their bids down in the auction. This could lead to enormous inefficiencies or huge gaps in the revenue. However, there is no general recipe for auction design: the organizer of such an event must take the whole environment into account and create the rules of the auction correspondingly.

2.3 VCG Auctions

In this section, we present the combinatorial Vickrey–Clarke–Groves (VCG) auction mechanism and some of its extensions. The VCG auction is named after Vickrey [1961], Clarke [1971] and Groves [1973]. This mechanism can be used to implement efficient allocations and, as we will see, it has some really attractive features.

For every valuation profile $(v_1, \ldots, v_M)$ of a subset of the players $\mathcal{M} \subseteq \mathcal{N}$, let $a(v_1, \ldots, v_M)$ denote an efficient allocation among the bidders in $\mathcal{M}$.[14] Now, the VCG mechanism can be defined as follows:

Definition 8 (VCG auction mechanism) Consider a situation described by the tuple $(\mathcal{N}^*, \mathcal{G}, \mathcal{H}, \mathcal{V}, p)$. A Vickrey–Clarke–Groves mechanism can be defined as a triple $(\mathcal{S}, A, P)$, where

- Strategies: $\mathcal{S} = \mathcal{V}$, thus we have a direct mechanism.
- Allocation rule: For every announcement profile $\tilde{v} \in \mathcal{V}$, the auction allocates the goods in an efficient way assuming that $\tilde{v}$ is the true profile, $A(\tilde{v}) = a(\tilde{v})$.
- Payment rule: every player $n \in \mathcal{N}$ pays the damage caused to the other players by his presence in the auction:

$$P_n(\tilde{v}) = \sum_{m \neq n} \tilde{v}_m(a_m(\tilde{v}_{-n})) - \sum_{m \neq n} \tilde{v}_m(a_m(\tilde{v}))$$

Thus, the mechanism naïvely assumes that everyone tells the truth and makes the bidders pay the utility loss caused to other players from the allocation difference.

It is easy to see that in the single-unit case, this auction is just the sealed-bid second price auction. In this setting, the winner of the object is the bidder with the highest announcement. He causes damage only to the bidder with the second-highest bid who would obtain the item if the winner was excluded. Thus, the winner will pay the second-highest announcement at the end of the auction.

2.3.1 Properties of the VCG Auctions

Now we discuss the properties of the VCG auction. First, we show that it is capable of creating the incentives for an efficient distribution of the goods. As a proof for this result we gave the usual proof of the literature but with more details. For a typical proof see Krishna and Perry [1998].

Theorem 9 The VCG mechanism is an incentive compatible, individually rational and efficient direct mechanism.

[14] As we have seen, such an allocation can be found by solving the set packing problem of section 2.2.3.

PROOF First, we show that truth-telling is a dominant strategy equilibrium.

Suppose that the true valuation profile is $v \in \mathcal{V}$ and consider an arbitrary announcement profile $\tilde{v} \in \mathcal{V}$. In this case, the mechanism selects $a(\tilde{v})$, a socially optimal allocation under $\tilde{v}$:

$$a(\tilde{v}) \in \arg\max_{a \in \mathcal{A}} \sum_{n \in \mathcal{N}} \tilde{v}_n(a_n).$$

Consider an arbitrary bidder $n \in \mathcal{N}$. This announcement with the associated payment gives him the utility

$$\begin{aligned} u_n(a_n(\tilde{v}), P_n(\tilde{v})) &= v_n(a_n(\tilde{v})) - P_n(\tilde{v}) \\ &= v_n(a_n(\tilde{v})) + \sum_{m \neq n} \tilde{v}_m(a_m(\tilde{v})) - \sum_{m \neq n} \tilde{v}_m(a_m(\tilde{v}_{-n})). \end{aligned}$$

Note that by modifying his announcement $\tilde{v}_n$, bidder n can influence only the first two terms in his payoff. If he announced his true valuation function v_n, the auctioneer would choose the allocation $a((v_n, \tilde{v}_{-n}))$, maximizing the objective function $v_n(a) + \sum_{m \neq n} \tilde{v}_m(a)$, which means that

$$v_n(a((v_n, \tilde{v}_{-n}))) + \sum_{m \neq n} \tilde{v}_m(a((v_n, \tilde{v}_{-n}))) \geqq v_n(a) + \sum_{m \neq n} \tilde{v}_m(a), \quad \forall a \in \mathcal{A}.$$

Since this holds for all allocations, it also holds for $a(\tilde{v})$ implying that

$$u_n(a_n((v_n, \tilde{v}_{-n})), P_n(v_n, \tilde{v}_{-n})) \geqq u(a_n(\tilde{v}), P_n(\tilde{v})).$$

Thus, by telling the truth, the bidder cannot get a payoff which is worse than that of his original announcement $\tilde{v}_n \in \mathcal{V}_n$. Since the announcement profile $\tilde{v}$ and the player $n \in \mathcal{N}$ were chosen arbitrarily, we just got back the definition of the dominant strategy equilibrium. Thus, telling the truth is a dominant strategy for every player.

For individual rationality, observe that the payoff of bidder $n \in \mathcal{N}$ if everyone tells the truth is

$$u_n(a_n(v), P_n(v)) = \sum_{m \in \mathcal{N}} v_m(a_m(v)) - \sum_{m \neq n} v_m(a_m(v_{-n})).$$

This is just the difference between the maximum social welfare from the auctions with and without n. If n is participating, the auctioneer can still choose the same allocation that he did in the other case. However, it is not necessarily welfare maximizing in this setting and he might be able to choose a better one. Thus the difference between the two welfare levels must be non-negative. We concluded that the utility levels are always non-negative under truth telling, consequently, the expected value of truth telling must be non-negative too.

Efficiency is trivially satisfied since telling the truth is an equilibrium strategy and given the defined allocation rule it results in a socially optimal distribution of goods.

■

It is important to emphasize two implications of the rules defining the VCG auction. The first one is that we did not use the information on the distribution of the types, the construction does not depend on this. Secondly, we have to use N additional set packing problems to determine the payments which makes the computational difficulties more severe in this case.

Before we continue with the description of the properties, we need some new notions. First, the agents are called symmetric if they have the same set of possible valuation functions and the same probability distributions defined on them. A second notion we need is monotonicity of valuation functions. A valuation function $v\colon \mathcal{H} \to \mathbb{R}_+$ of an agent is called monotone, if for all $h_1, h_2 \in \mathcal{H}$, $h_1 \subseteq h_2$, it satisfies that $v(h_1) \leqq v(h_2)$. Thus, the agents are never worse off by having additional items too (without considering the payments of course). Now we are ready to state the next property which is a result of Monderer and Tennenholtz [2005].

Theorem 10 (Asymptotic optimality) With symmetric, weakly risk averse bidders, and monotone valuation functions, the Vickrey–Clarke–Groves mechanism is asymptotically revenue maximizing. In other words, the expected revenue of the auction almost surely converges to the theoretical maximum as the number of bidders goes to infinity.

PROOF See Monderer and Tennenholtz [2005] for the formal proof. ■

Thus we have seen that the VCG mechanism has good revenue properties if the number of agents is large and the agents are symmetric. To see what happens in other cases, we need to define a generalized version of the VCG auction.

2.3.2 Generalized VCG Auctions

The idea behind this generalization is to modify the payment scheme of the VCG auction in a way that the mechanism still preserves its basic properties (incentive compatibility, individual rationality and efficiency) but it leads to higher revenues. This is done by choosing a "basis strategy" for every bidder and making them pay for the utility difference caused to other bidders by not playing their basis strategy plus their valuation under this strategy. This idea is introduced by Krishna and Perry [1998] and this section is based on their results. We can use this generalized mechanism to describe the revenue properties of the VCG auction in a general setting where the number of agents can be small too.

Definition 11 (GVCG mechanism) Consider an auction situation described by the tuple $(\mathcal{N}^*, \mathcal{G}, \mathcal{H}, \mathcal{V}, p)$. Let $\bar{v} \in \mathcal{V}$ be an arbitrary valuation combination. A Generalized Vickrey–Clarke–Groves (GVCG) mechanism with basis $\bar{v}$ can be defined as a triple $(\mathcal{S}, A, P)$, where

- Strategies: $\mathcal{S} = \mathcal{V}$.
- Allocation rule: For every announcement profile $\tilde{v} \in \mathcal{V}$, the auction allocates the goods in an efficient way assuming that $\tilde{v}$ is the true profile, $A(\tilde{v}) = a(\tilde{v})$.
- Payment rule: every player $n \in \mathcal{N}$ pays the following:

$$P_n(\tilde{v}) = \bar{v}_n(a_n(\bar{v}_n, \tilde{v}_{-n})) + \sum_{m \neq n} \tilde{v}_m(a_m(\bar{v}_n, \tilde{v}_{-n})) - \sum_{m \neq n} \tilde{v}_m(a_m(\tilde{v})).$$

Thus, the GVCG auction modifies the payment scheme of the VCG auction. Instead of considering the case where n is excluded, the mechanism lets n participate with his basis strategy $\bar{v}_n$ and adds his valuation to the damage caused to the other agents by not bidding his basis strategy. This construction also internalizes the overall welfare and gives incentives to tell the truth. This auction also involves $N + 1$ set packing problems and all of them include every bidder; thus, computationally it is a bit harder than the original VCG auction. It can be shown that this mechanism is revenue-maximizing under the efficient, incentive compatible and individually rational mechanisms. With some additional assumptions the two auctions coincide and in these cases the original VCG mechanism shares this great property too. But for this, first we need to see whether the generalization still satisfies incentive compatibility, efficiency and individual rationality. The first two properties can be immediately shown by the same way as we did in the VCG case.

Theorem 12 The GVCG mechanism is an efficient and incentive compatible direct auction.

PROOF Same as the proof of the same properties of the VCG auction. ■

Thus, the GVCG mechanism is efficient and incentive compatible. However, it is not always individually rational, but by choosing the right basis, it satisfies this property too. Moreover, with an appropriately chosen basis, it generates the largest revenue among all efficient IC-IR mechanisms. These findings are stated in the next theorem.

Theorem 13 Let the basis valuations be defined as follows. For all $n \in \mathcal{N}$,

$$\begin{aligned}\bar{v}_n &\in \arg\min_{v_n \in \mathcal{V}_n} \mathbb{E}_{v_{-n}} u_n(a_n(v), P_n^{(\mathrm{VCG})}(v)) \\ &= \arg\min_{v_n \in \mathcal{V}_n} \mathbb{E}_{v_{-n}} \left(\sum_{n \in \mathcal{N}} v_n(a_n(v)) - \sum_{m \neq n} v_m(a_m(v_{-n})) \right).\end{aligned}$$

Then, the Generalized VCG auction with this basis $\bar{v}$ is individually rational. Moreover, it maximizes the expected revenue of the auctioneer under all efficient, incentive compatible and individually rational auctions.

PROOF See Krishna and Perry [1998]. ■

Thus, for every player, the basis strategy is the player's "worst type", i.e. the valuation function that gives him the worst expected utility in the VCG auction.[15] The reason behind the larger revenue is that in the VCG auction, playing the worst type can still result in a positive after-payment utility. The GVCG scheme does not leave this utility to the worst-type bidders, it increases their payments by this amount instead. Thus, in contrast to the VCG auction, we used the extra information carried by the valuation distributions.

However, in some cases, the GVCG and the VCG auctions are the same. This finding is formalized in the following corollary:

[15] In some cases it could be hard to compute these worst types analytically. However, in some special settings like our example in section 3, they can be given immediately.

Corollary 14 If for every player $n \in \mathcal{N}$, the above defined basis strategy $\bar{v}_n$ is such that either the player never wins with that strategy or causes no harm to the other bidders by winning, then the VCG mechanism and the GVCG mechanism with basis $\bar{v}$ coincide. In such cases, the VCG mechanism is optimal among all efficient IC-IR auction mechanisms. The condition is automatically satisfied if $\mathbf{0} \in \mathcal{V}$.

2.3.3 Other Generalization: The VCG-μ Auction

In the previous section we have seen that the GVCG auction represents the limit case among efficient mechanisms in the expected revenue sense. If we want to reach a higher revenue than that of the GVCG mechanism, we must give up efficiency and find other allocation rules that can be implemented by an auction with higher expected payments. But how should we find such an allocation rule? This question is difficult to answer and, as we discussed in sections 2.1.2 and 2.2.4, no one has found a general solution to this problem. In this part of the paper, we present the results of Likhodedov and Sandholm [2004] and Likhodedov and Sandholm [2005]. They modified the allocation rule and the payment rules of the VCG auction such that the new mechanism still satisfies incentive compatibility and individual rationality but is able to reach higher revenues by giving up efficiency. They introduced a new class of IC-IR auctions where each auction can be described by a set of parameters. Given the bidders and their valuation function distributions, numerical optimization in the parameter space can be used to find the auction with the largest expected revenue within this class. Since the VCG is also included here, we can be sure that the new mechanism generates at least as large revenue as the VCG.

The introduced parameters are of two types. The first group consists of "bidder weights"; positive numbers by which the announced value functions are multiplied in the auction. The elements of the second group are "allocation boosters": additive modifications of the bidder's valuation function conditional on whether a particular bundle is received during the auction. Since the second group would introduce too many additional parameters which would make the auctions of the later sections computationally intractable, we describe only the situation of bidder weights here.

Let $\boldsymbol{\mu} \in \mathbb{R}^N_{++}$ denote a strictly positive vector of weights. Given the valuation profile $(v_1, \ldots, v_M)$ of a subset of the players $\mathcal{M} \subseteq \mathcal{N}$, let $a^\mu(v_1, \ldots, v_M)$ define an allocation with the following property:

$$a^{\mu}(v_1, \ldots, v_M) \in \arg\max_{a \in \mathcal{A}} \sum_{m \in \mathcal{M}} \mu_m v_m(a).$$

Thus, $\tilde{a}(v_1, \ldots, v_M)$ is the allocation maximizing the social welfare if we rescale the bidders' valuations by the $\boldsymbol{\mu}$ parameters. Now the VCG-μ auction can be defined by a VCG auction over these "virtual valuations":

Definition 15 (VCG-$\boldsymbol{\mu}$ auction) Consider a situation described by the tuple $(\mathcal{N}^*, \mathcal{G}, \mathcal{H}, \mathcal{V}, p)$. A Vickrey–Clarke–Groves-μ (VCG-μ) mechanism can be defined as a triple $(\mathcal{S}, A, P)$, where

- Strategies: $\mathcal{S} = \mathcal{V}$.
- Allocation rule: For every announcement profile $\tilde{v} \in \mathcal{V}$, the auction allocates the goods in an efficient way assuming that for each player $n \in \mathcal{N}$ $\mu_n \tilde{v}_n$ is the true profile. $A(\tilde{v}) = a^{\mu}(\tilde{v})$.
- Payment rule: every player $n \in \mathcal{N}$ pays the damage caused to the others by his presence in the auction given the virtual valuations, rescaled by his weight μ_n:

$$P_n(\tilde{v}) = \frac{1}{\mu_n} \left(\sum_{m \neq n} \mu_m \tilde{v}_m(a^{\mu}_m(\tilde{v}_{-n})) - \sum_{m \neq n} \mu_m \tilde{v}_m(a^{\mu}_m(\tilde{v})) \right).$$

Thus, a VCG-μ mechanism replaces the actual announcements with the scaled virtual valuations, executes a usual VCG mechanism and rescale the payments by the individual weights at the end. If $\boldsymbol{\mu} = \mathbf{1}$, we get back the original VCG mechanism.

We can see that both the chosen allocation a^{μ} and the payments depend only on the ratios of the scaling factors, thus we can set one of them to be equal to 1. This will be important for the simulations since it reduces the number of dimensions by 1.

Incentive compatibility and individual rationality can be proven in the same way as at the VCG mechanism. Therefore we just summarize these properties here.

Theorem 16 Let $\boldsymbol{\mu} > \mathbf{0}$ be arbitrary. Then, the defined VCG-μ auction is incentive compatible and individually rational.

PROOF Immediate from the proofs of the similar properties of the VCG auction. ■

We can see that in the optimization we use the probability distributions again (we maximize the expected revenue). Although with given $\boldsymbol{\mu}$, the overall number of set packing problems is still $N + 1$, the optimization requires multiple executions with different weight vectors in a possibly large-dimensional space. Thus, the computational difficulties can become really serious here. However, as we will see in the next chapter, this generalization is able to give significant boosts to the VCG/GVCG revenues.

3 The Office Complex Example

In this section, we present a simplified setting to create an environment where larger combinatorial auctions can be solved without much of the difficulties mentioned in the previous section.

Consider the following situation. Our company has recently built a high-tech office complex in the downtown of a huge city. The interior of the building is not fixed; each floor consists of a couple of "office blocks" that can be merged into larger offices later. There is a number of firms willing to buy offices in this building. Because of the flexibility of the interior structure, these firms can consider bundles of office blocks as potential offices. This structure suggests non-additive preferences: a bidder can utilize some neighboring blocks better, there can be synergies between these blocks. In other words, the bidders might want to pay more than the sum of their individual reservation prices for having a couple of blocks together. However, these preferences are restricted in some sense. Every firm wants to have blocks very close to one another; they cannot use a 200 m^2 office space such that 100 m^2 is located on the first and the other 100 m^2 on the 30th floor. They might not be indifferent regarding the location of their future office either: some might want to have a place at the bottom of the building, others could prefer blocks at the top. The set of possible packages might be also restricted on the auctioneer's side to reach computational manageability in this large problem.

There are several reasons why we should organize a combinatorial auction in this example. First of all, there is a synergy effect that we want to exploit, and making the bidders be able to bid on packages might lead to a better result in that case. The second reason is that both the auctioneer and the bidders might want to exclude some inconvenient outcomes that would be possible in simultaneous auctions. For example, suppose that a bidder is interested only in offices with neighboring blocks and he is willing to pay more for the package than the sum of the individual block valuations. In a simultaneous auction where a bidder submits bids on every item simultaneously, he has to overbid his individual valuations to increase his chances to win all the blocks. If he does so, too large bids are compensated by the synergy at the end. However, if he fails at obtaining one of the blocks, it might happen that he pays a too high price for the others. In a combinatorial auction, where he can make a bid on the whole package and different bids on its subsets, he can be sure that such a situation will never happen. For the auctioneer, allowing for combinational bids and restricting the set of possible allocations could result in higher revenues. Moreover, the auctioneer could exclude not desired outcomes (like fragmented offices) by making restrictions on the set of packages in the combinatorial auction.

3.1 Setup of the Problem

The Auctioneer First of all, we assume that the building has K floors with L office blocks on each floor. Thus, the overall number of goods is $G = KL$. In each floor, the blocks have a "chain" structure: between the two "corner blocks", every block has exactly two neighbors. Thus, they can be indexed by the number of the floor, and their position within this chain:

$$\mathcal{G} \doteq \{g_{k,l} : k \in \mathcal{K}, l \in \mathcal{L}\}.$$

The auctioneer is interested in the maximization of his revenue. Since the overall number of blocks G can be large in this example, the auctioneer faces a computational issue. With 30 floors and 10 blocks on each floor, he would have 2^{300} possible packages which would simply make the above mentioned auctions impossible to implement without any further restriction. The usual approach in the literature is to make assumptions on the preferences to simplify this problem. Unfortunately, in problems like the office complex example where we have many goods and many bidders, this cannot be done without imposing too strict assumptions on the preferences. Therefore, we use a different approach and instead of restricting the preferences, we restrict the set of available packages $\mathcal{H}$. This causes no problems in the auction mechanisms discussed in section 2.3 since we did not need to specify this set in the definitions and in the proofs of the properties.

We assume that the auctioneer wants to allocate only neighboring blocks as offices at the end and every firm can be accommodated in only one floor. With these restrictions, the auctioneer can avoid the chaos that could be caused by commuting between separated parts of an office. In this case, the set of available packages is defined in the following way:

$$\mathcal{H} \doteq \left\{ \{g_{k,l_1}, \ldots, g_{k,l_2}\} : k \in \mathcal{K}, l_1, l_2 \in \mathcal{L}, l_1 \leqq l_2 \right\}.$$

Property 1 In the office complex example, the number of available packages is

$$H = \frac{KL(L+1)}{2}.$$

PROOF On each floor $k \in \mathcal{K}$, we have 1 package of size L, 2 packages of size $L - 1$, and so on with L packages of size 1 at the end. Thus, the number of packages on a floor is just the sum of the first L natural numbers and this makes the overall number $\frac{KL(L+1)}{2}$. ■

Now we have a significant drop in the number of possible packages compared to the unrestricted case. For example, in the 30-floor, 10-blocks-per-floor case, there are only 1560 possible packages instead of 2^{300}.

The Bidders We assume that we have N possibly different, risk-neutral bidders. Their valuations are determined in the same way: for each bidder $n \in \mathcal{N}$ and each single block $g \in \mathcal{G}$, $v_n(g)$ is drawn randomly from the same bidder-specific distribution. The effect of synergies in the package-valuations is captured by a bidder-specific parameter $\alpha_n > 0$: for every package $h \in \mathcal{H}$,

$$v_n(h) \doteq \left(\sum_{g \in h} v_n(g) \right) \cdot \left(1 + \frac{|h| - 1}{|h|} \alpha_n \right).$$

Thus, the valuation of every package is just the sum of the item-valuations multiplied by a size-dependent synergy term.

We use this particular form since it captures two effects which are reasonable in the office complex example. First, merging two packages should result in a higher utility level than the sum of the utilities of the two individual packages since we can utilize the available space better. Secondly, this extra synergy of merging should become smaller at larger package sizes. Merging small packages should give a significantly higher increase in synergy than merging huge ones. Here we have exactly these effects governed by the size-dependent synergy multiplier. It introduces a gradual increase in synergy as the package size grows, but this additional increase vanishes at large sizes. For instance, a 0.2 synergy parameter means that after two blocks the bidder gets an additional 10% utility from the synergy, three blocks give 15% (additional 5% points), four gives 17.5% (additional 2.5% points) and so on. In the limit we have a 20% increase in the overall utility. This structure also implies that even if we receive something which is completely worthless by itself, we can still utilize the extra space and gain something from the increased synergy of the former blocks. By following the previous argument, it is easy to see that the preference of the bidders satisfy monotonicity.

Since we have risk neutral bidders with monotone preferences, we can use every result of section 2.3 in this part of the paper.

3.2 The SPP

In our analysis, we will use larger simulations to determine the theoretical maximum of the revenues that an auction mechanism could achieve and the revenue properties of the VCG, the GVCG and the VCG-μ mechanisms in different cases. In all of these simulations, the set packing problem has a central role and we have to define and solve SPPs with different (N, K, L) triples through our analysis. Since the underlying package structure makes the programming part a non-straightforward exercise, we would like to explain how the coefficient matrix in the SPPs looks like and how it can be created in a simple way in the programming part.

Once again, the SPP is the following binary linear programming problem:

$$\begin{aligned} \max_{\boldsymbol{y}} \ & \boldsymbol{v} \cdot \boldsymbol{y} \\ \text{s.t.} \ & \boldsymbol{A}\boldsymbol{y} \leqq \mathbf{1} \\ & \boldsymbol{y} \in \{0,1\}^{NH}. \end{aligned}$$

In the office complex example, the coefficient matrix $\boldsymbol{A}$ must capture three requirements:

1. the special structure of the packages;
2. the fact that no block can be assigned more than once;
3. the restriction that everyone can get at most one package.

For these, note first that we can label the blocks and the packages in the same order in every floor. Now define a matrix $\boldsymbol{B} \in \mathbb{R}^{L \times L(L+1)/2}$ which captures the structure of the packages in a given floor. Consider an arbitrary floor $k \in \mathcal{K}$ and define the set $\mathcal{H}_k$ as the packages in floor k. Now, define the elements of $\boldsymbol{B}$ according to the following rule. For every $l \in \mathcal{L}$ and $h \in \mathcal{H}_k$,

$$b_{lh} \doteq \begin{cases} 1 & \text{if } \ g_{k,l} \in h \\ 0 & \text{otherwise,} \end{cases}$$

where b_{lh} denotes the h-th element in the l-th row of matrix B.

Thus, in the matrix $\boldsymbol{B}$, every row represents a block, every column a package, and the 1 values indicate that the corresponding block is included in a package. Table 1 shows an example in a 5-block-per-floor case.

$$
\left[\begin{array}{ccccc:cccc:ccc:cc:c}
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1
\end{array}\right]
$$

Table 1: The coefficient matrix of a single floor with five blocks per floor.

The next step is to build the coefficient matrix $\boldsymbol{A}$ that guarantees that the package structure imposed by the vector $\boldsymbol{y}$ contains only non-overlapping offices and that the bidders can get at most one package at the end. The construction of Table 2 can satisfy these requirements.

$$
\begin{array}{cc}
 & \begin{array}{ccc} \overbrace{\qquad\qquad\qquad}^{\text{bidder 1}} & \overbrace{\qquad\qquad\qquad}^{\text{bidder 2}} \qquad & \overbrace{\qquad\qquad\qquad}^{\text{bidder N}} \end{array} \\
\boldsymbol{A} \doteq &
\left[\begin{array}{cccc:cccc:c:cccc}
\boldsymbol{B} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{B} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \cdots & \boldsymbol{B} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\
\boldsymbol{0} & \boldsymbol{B} & \cdots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{B} & \cdots & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{B} & \boldsymbol{0} & \boldsymbol{0} \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \cdots & \vdots & \vdots & \ddots & \vdots \\
\boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{B} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{B} & \cdots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{B} \\
\hdashline
\boldsymbol{1} & \boldsymbol{1} & \cdots & \boldsymbol{1} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\
\boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{1} & \boldsymbol{1} & \cdots & \boldsymbol{1} & \cdots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\
\boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} & \cdots & \boldsymbol{1} & \boldsymbol{1} & \boldsymbol{1} & \boldsymbol{1}
\end{array}\right]
\begin{array}{l}
\} \text{ floor 1} \\
\} \text{ floor 2} \\
\\
\} \text{ floor K} \\
\} \text{ bidder 1} \\
\} \text{ bidder 2} \\
\\
\} \text{ bidder N}
\end{array}
\end{array}
$$

Table 2: The coefficient matrix of the set packing problem.

In the table, the submatrices are separated by horizontal and vertical dashed lines. In the upper submatrices, $\boldsymbol{0}$ denotes $L \times \frac{L(L+1)}{2}$ matrices full of zeros. In the lower submatrices, $\boldsymbol{0}$ and $\boldsymbol{1}$ are row vectors of length $\frac{L(L+1)}{2}$, full of zeros and ones, respectively. We have one upper and one lower submatrix for each bidder, every upper submatrix contains K columns and K rows and every lower submatrix has K columns and N rows of smaller matrices.

If the binary vector $\boldsymbol{y}$ is built up in the same way (blocks-floors-bidders), then the $\boldsymbol{A}\boldsymbol{y} \leqq \mathbf{1}$ linear inequality system can characterize our set of possible packages. The constraints corresponding to the upper blocks represent the requirement that each block can be assigned at most once, while the inequalities defined by the lower blocks impose the bidder-specific constraints that everyone can have at most one package at the end.

In the computer program, it could be challenging to define this $\boldsymbol{A}$ matrix for different (N, K, L) parameters. However, the special structure of this matrix lets us use the Kronecker product as a very powerful tool in the parameter-dependent definition.

Definition 17 (Kronecker product) For two matrices $\boldsymbol{C} \in \mathbb{R}^{n\times m}$ and $\boldsymbol{D} \in \mathbb{R}^{p\times q}$, the Kronecker product of $\boldsymbol{C}$ and $\boldsymbol{D}$ is defined as

$$\boldsymbol{C} \otimes \boldsymbol{D} \doteq \begin{bmatrix} c_{11}\boldsymbol{D} & c_{12}\boldsymbol{D} & \cdots & c_{1m}\boldsymbol{D} \\ c_{21}\boldsymbol{D} & c_{22}\boldsymbol{D} & \cdots & c_{2m}\boldsymbol{D} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n1}\boldsymbol{D} & c_{n2}\boldsymbol{D} & \cdots & c_{nm}\boldsymbol{D} \end{bmatrix}.$$

Using the Kronecker product, we can define the matrix $\boldsymbol{A}$ for different K, L and N parameters as

$$\boldsymbol{A} = \begin{bmatrix} \mathbf{1}_{1\times N} \otimes (\boldsymbol{I}_K \otimes \boldsymbol{B}) \\ \boldsymbol{I}_K \otimes \mathbf{1}_{1\times L(L+1)/2} \end{bmatrix},$$

where $\boldsymbol{I}_K$ denotes the identity matrix of dimension K.

Now our SPP can be easily defined and solved in our computer program. One part of the computational difficulty is solved by the simplification, since the number of variables fell dramatically. Unfortunately, the polyhedron defined by these inequalities admits fractional extreme points, thus the problem cannot be replaced by its polynomially solvable linear relaxation. Although the first part of the coefficient matrix $(\mathbf{1}_{1\times N} \otimes (\boldsymbol{I}_K \otimes \boldsymbol{B}))$ satisfies the consecutive ones property, and thus is totally unimodular, the lower blocks destroy this feature. This matrix is not even perfect as Table 3 shows.

Thus, the usual conditions for the integral extreme points are not fulfilled and it can be shown that the bidders' preferences do not exclude the non-integral extreme points as solutions. See Appendix A for an example. We cannot use the linear programming relaxation here. However, with the restriction

$$
\left[\begin{array}{cccc:ccc:cc:c|cccc:ccc:cc:c}
1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\
\hdashline
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{array}\right]
$$

Table 3: Counterexample for perfection of the matrix $\boldsymbol{A}$.

of $\mathcal{H}$ we already made a huge improvement regarding the complexity of the problem which allows us to consider and analyze larger models too.

3.3 Simulation Results

In this section, we describe four simulation scenarios. In the first two cases we took symmetric bidders and showed the good revenue properties of the original VCG auction. In the other two cases, the symmetry among the bidders was broken, which led to ill-performing VCG auctions. We used the GVCG and VCG-μ auctions to "restore the order" and get significantly higher expected revenues in these cases. In the VCG-μ auctions, we used the same scaling factors for bidders of the same type to avoid computational intractability. This can be justified by the reasoning that players of the same type should get the same conditions in the auction while this should not necessarily hold for different bidders. For example, the auctioneer might want to act as if he was benevolent and subsidize the weaker bidders with a higher weight in the auction. We will see that even selfish reasons can stay behind such a generous behavior.

All the simulations were done in R. For the set packing problem, the open source mixed linear integer programming solver SYMPHONY was used. This was developed by the Computational Infrastructure for Operations Research community and can be downloaded without any limitations from their homepage[16] or from the R repository. The whole source code is contained in Appendix A.

Every case assumed a medium-sized building with 4 floors and 4 blocks on each floor. This means that we had 16 blocks and 40 packages overall. This size was large enough to get an insight into the implications of the specific structure of this model. On the other hand it was small small enough to obtain

[16] `http://www.coin-or.org/SYMPHONY/index.htm`

the simulation results within reasonable time. We did not need large samples in the simulations; the means converged very fast to the expected revenues.

3.3.1 Symmetric Case, $[0, 1]$ Bidders

In the first scenario, we took symmetric bidders with item valuations independently and uniformly distributed over the $[0, 1]$ interval and a 0.2 synergy parameter.

The maximal possible expected revenue and the revenue of the VCG auction were calculated with different numbers of bidders, from 1 to 30. The outcome is presented in Figure 1. We can observe at least two things here. The first one is that for relatively small numbers of bidders, the revenues of the VCG mechanism were far from the optimal value. The reason behind this is the special payment scheme of the VCG mechanisms: everyone pays the damage caused to the others. If we have fewer bidders than the number of floors, everyone will occupy a whole floor at the end, causing only negligible damage to the others. Although some other bidders might prefer my blocks to what they got, the expected value of the difference cannot be too large since they received whole floors after the auction. This situation changed at the moment when the number of bidders exceeded the number of floors. From this point on, the winners really harmed one another since the building was not large enough anymore to accommodate all the bidders in whole floors. They had to share and squeeze out the others causing significantly larger damages which led to increased payments. Consequently, the revenue jumped to a value close to the theoretical maximum which means that the VCG has good revenue properties in this case.

The second observation is that we can see the asymptotic optimality of the VCG mechanism in effect: as the number of bidders grew, the gap between the expected revenue and the theoretical maximum shrank gradually. Even with 15 bidders we got great revenues from the auction.

It is important to see whether the synergy parameter has a reasonable value in the auction. A too high parameter would lead to the sale of whole floors since one particularly high item valuation would generate high valuations for packages consisting that block too. This result would lead to an allocation scheme for which a simpler auction form can be used without all the computational difficulties. A typical package-distribution of a bidder is shown in Figure 2. This figure depicts the numbers how often the different packages were received during 100 auctions with ten bidders. The first ten numbers represent the first floor, the second ten the second floor and so on. Within a floor, the bundles of different sizes are separated by dashed lines. For instance, the points between the red and green lines belong to packages of size three. We can see in this figure that the package structure at the end is not trivial; our bidder received packages of every possible size during the 100 auctions. Table reftab:sympack shows the average numbers of different packages allocated in one auction considering all the ten players. We can see in the table that usually smaller packages were assigned. The whole distribution of the ten bidders can be seen in Figure 3. Not surprisingly, we can see that the distribution is close to "symmetric"; every agent has the same power in the auction.

Since the worst possible valuation vector $\mathbf{0}$ is included in the set of possible valuations for every player, the VCG and the GVCG auctions coincide in this scenario. Consequently, the VCG mechanism is optimal among the efficient auctions in this case. The VCG-μ auction was not implemented here since we stuck to the philosophy that every player of the same type should have the same conditions, i.e. the same weight in the auction.

3.3.2 Symmetric Case, $[2,3]$ Bidders

The only difference between this setting and the previous one is that we shifted the distributions of the item valuations to a uniform $[2,3]$ distribution. This implied that the VCG and GVCG auctions did not coincide anymore. We did the same analysis as before. We calculated the expected revenues for different numbers of agents between 1 and 30. The worst type of every player is when all of his item valuations are 2. Therefore, we used this as a basis in the GVCG auction. The expected revenues can be seen in Figure 4. Regarding the VCG auction, we can make the same observations as in the previous case: The VCG mechanism performed very badly if we had 4 or less bidders and produced high and asymptotically optimal expected revenues otherwise. On the other hand, the GVCG auction was able to produce significantly more in these cases. The reason behind the power of the GVCG auction was that it charged "reserve prices" from the winners. If the number of bidders was smaller than the number of floors, everyone occupied a whole floor and caused almost no damage to the others. Since the worst types received something too, they were charged by the first term in the payment scheme of the GCVG auction. This property disappeared when we reached the 5-bidder case: worst-type bidders

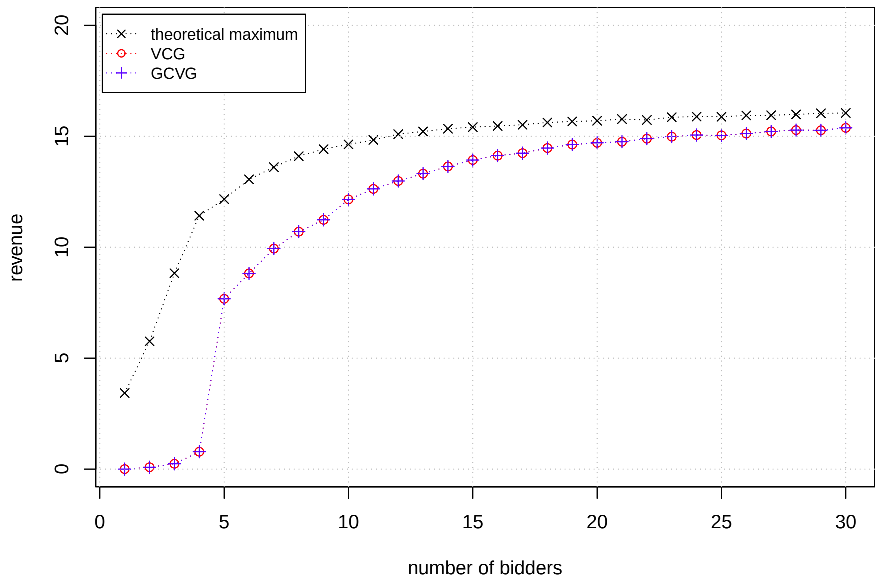

Figure 1: Expected revenues in the symmetric $[0, 1]$ case

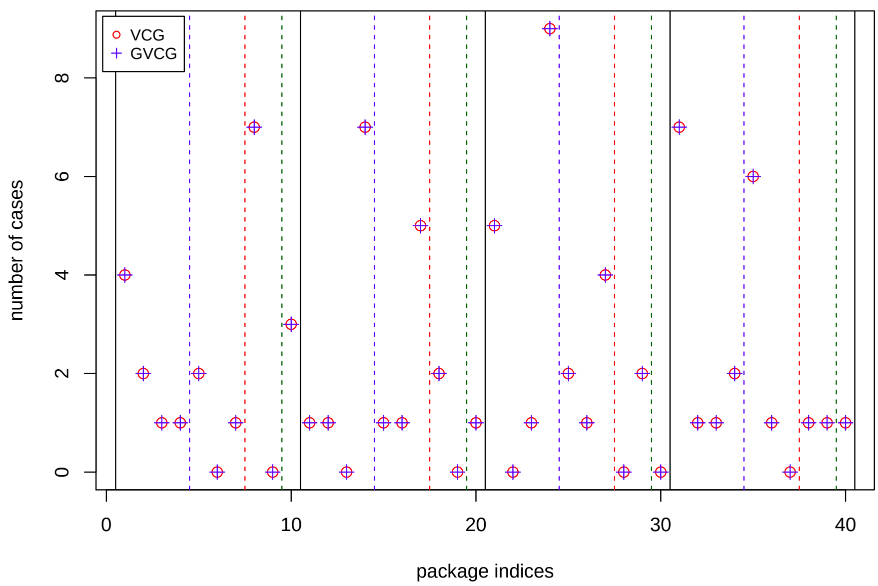

Figure 2: Package distribution of one bidder in the symmetric $[0, 1]$ case

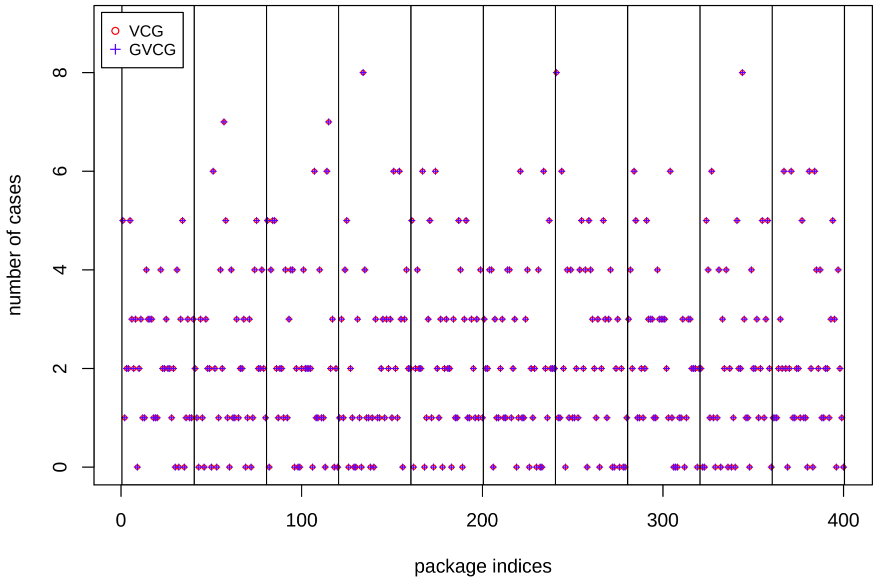

Figure 3: Package distribution of all bidders in the symmetric $[0, 1]$ case

Size	Case I	Case II
1	4.13	0.79
2	3.04	1.39
3	1.21	0.73
4	0.54	2.56

Table 4: Average numbers of packages in one round, symmetric cases

received nothing in general in the auction, thus the GVCG payments coincided with the VCG payments. Although the GVCG auction was able to increase the expected revenues in the cases with few bidders, this result is not particularly interesting in our context. In the office complex example we can assume that the number of bidders is large enough to have competition among them. However, the insight of this result will be important at the other scenarios where we have rare asymmetric types.

A typical package distribution of a bidder is shown in Figure 5. We still had non-trivial bundling and not surprisingly, the distribution under the VCG and GVCG auctions were the same. In Table 4, we can see the average numbers of different packages in one auction. Now the auction favors larger packages, thus the 0.2 synergy parameter has a stronger effect than it had in the previous scenario.

3.3.3 Asymmetric Case, Non-Overlapping Distributions

In this case, the symmetry between the bidders was broken. We assumed that there was a fixed, large number of low-valuation bidders (25 in the simulations) and that we had some bidders with dominating higher valuation functions. The item valuations were drawn uniformly from $[0, 1]$ for the first group and from $[2, 3]$ for the second. We used a synergy parameter of 0.2 for both types. The theoretical maximum and the expected revenues of the auctions are shown in Figure 6, as functions of the number of "strong" bidders in the auctions.

We can see that even though we had a large number of bidders, the VCG auction did not perform well if the number of the strong bidders was small. We can use the same reasoning as before: the few strong bidders occupied whole floors at the end causing almost no damage to one another and only small damages to the weak ones. This is a severe issue here since we cannot exclude this case in the office complex example as we did in the previous section. In the symmetric case it is not reasonable to assume that we have fewer bidders than floors, but here in the asymmetric scenario we cannot avoid the case where there is a small group of bidders with higher item-valuations.

Fortunately, the GVCG auction can be used here to set higher prices for stronger bidders. We can see that we were able to increase the revenue significantly by using the GVCG mechanism in the few-strong-bidder case since it acted in the same way as in the second scenario and imposed reserve prices

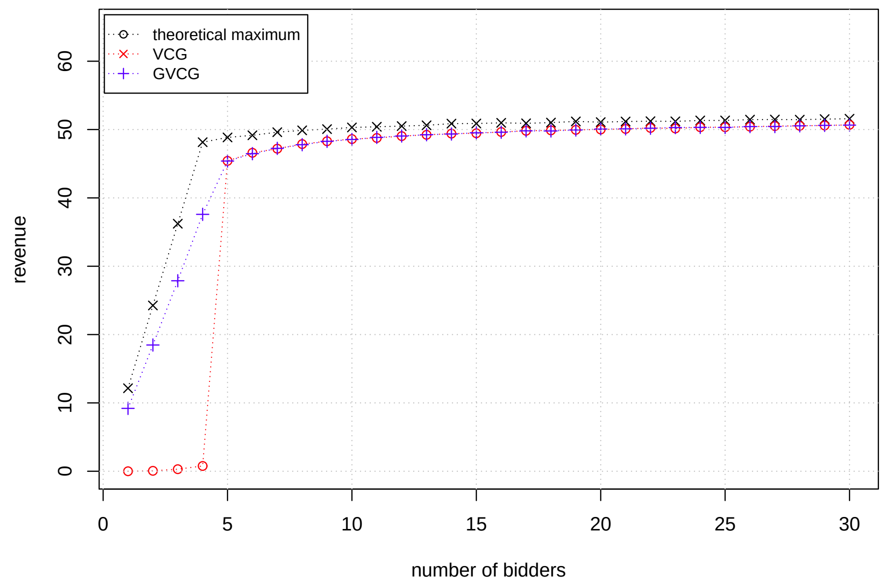

Figure 4: Expected revenues in the symmetric $[2,3]$ case

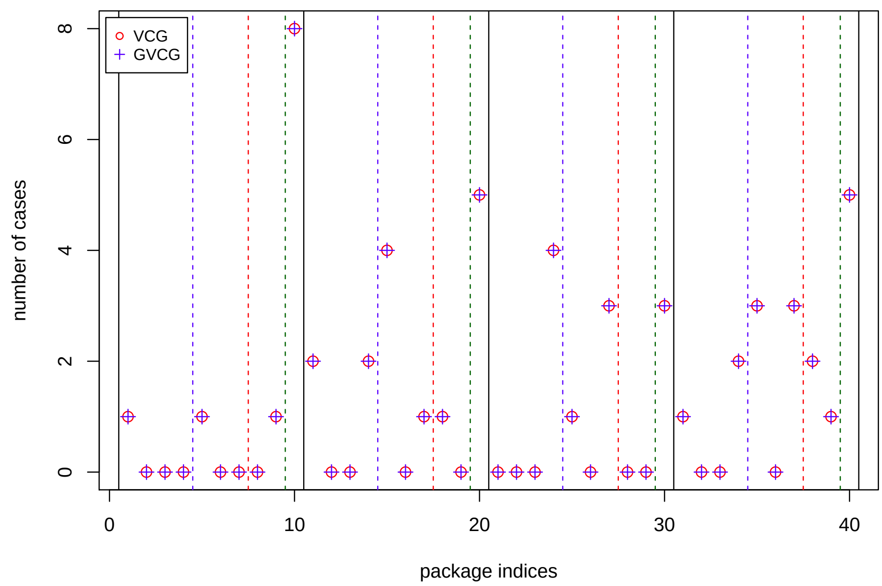

Figure 5: Package distribution of one bidder in the symmetric $[2,3]$ case

for the strong bidders. However, this asymmetry might allow us to do more. As a next step, we used the VCG-μ auction form, fixed the scaling factor of the first group at 1, and did a grid search over the $[0, 2]$ interval to find the μ_2 parameter which led to the maximal expected revenue. We can see in Figure 7 the expected revenue as a function of μ_2 in the case where we had only one strong bidder. As we decreased the weight of the strong bidder, the expected revenue started to increase immediately. This effect can be explained in the following way: as we decrease the weight of the second group, the weaker bidders start to become more important in the auction. With a small decrease in the weight, the optimal allocation does not change. However, the higher relative importance of weaker bidders increases the calculated and rescaled damage to them in the strong bidder's payment. On the other hand, the weak bidders' fee does not change at all since the allocation did not change and their weights cancel out in the payment scheme. Consequently, they cause no harm to the strong one and the harm caused to other weak ones is the same as before. This argument holds if the allocation is unchanged. However, after some point the weak bidders become rivals of the strong and efficiency breaks down. The payment of the strong bidder still increases but after a point he lose his power and the weaks share the whole building. Because of this, the whole situation crushes down to the symmetric $[0, 1]$ case which is responsible for the significant drop in the expected revenue. After this point, an increase in μ_2 does not change the result since the strong bidder will still not win anything and the weak bidders are equally important. Thus the weeks always get the same allocation and cause the same harm to one another.

In the other direction where we overweight the strong bidder, a similar argument holds. If we increase the importance of the strong bidder, the allocation does not change. However, the damage caused by him to the weak bidders will decrease with the modified valuation function, thus the revenue decreases.

The same reasoning holds in the cases where we have several strong bidders but their number is still below the number of floors.

However, as soon we have more strong bidders than floors, the situation radically changes (see Figure 8). The strong ones will share the blocks alone, thus the harm caused to the weak bidders will be zero. Since the relative weight within the strong group are the same, the allocation and the payments after decreasing μ_2 will not change before the weaks become equally important. If we reduce μ_2 further, we get back the symmetric $[0, 1]$ case again which results in a significant drop in the expected revenue. Thus, in these cases the VCG-μ auction leads to no improvement at all; all the three mechanisms give the same expected revenue as it can be seen in Figure 6.

It is interesting to see how far the optimal package distribution is from the efficient one. Figure 9 compares the distributions over the whole society in the case of 4 strong bidders. The bidders are separated by vertical lines. The package indices 1-1000 correspond to the weak bidders and the indices 1001-1160 to the strong ones. Figure 10 presents the distribution of the first strong bidder. We can see that in the VCG and GVCG cases, the weak bidders received nothing: Their distribution is completely flat. But we broke the efficiency in the VCG-μ auction and in few cases we gave packages to weak bidders too. However, the two distributions are still really "close" to each other and in most of the cases we gave the packages to the bidders who had the highest valuation for them. This small difference can be seen in Table 5 too: The average numbers of packages of different sizes are really close to each other in the two auctions.

	Case I				Case II			
	strong bidders		weak bidders		strong bidders		weak bidders	
Size	(G)VCG	VCG-μ	(G)VCG	VCG-μ	(G)VCG	VCG-μ	(G)VCG	VCG-μ
1	0	0.05	0	0.43	0.01	0.53	0.48	2.59
2	0	0.22	0	0.24	0.06	1.19	0.03	1.11
3	0	0.37	0	0.03	0.42	1.09	0.01	0.11
4	4	3.30	0	0.05	3.51	1.15	0.00	0.02

Table 5: Average numbers of packages in one round, asymmetric cases

3.3.4 Asymmetric Case, Overlapping Distributions

In this setting we created overlapping distributions by changing the item distribution of the strong group to $[0.5, 2.5]$. Thus, the worst type of the strong bidders dropped from **2** to **0.5**. The expected revenues are shown in Figure 11. Since we had 25 $[0, 1]$ bidders, their maximal valuation for every item was usually well above 0.5. Consequently, there was no significant difference between excluding stronger bidders and leaving them in the auction with their worst type. Thus, in general, the GVCG auction was not able to charge higher fees than the VCG auction. However, the VCG-μ auction performed well again, giving a significant increase in the expected revenue. Moreover, it was able to

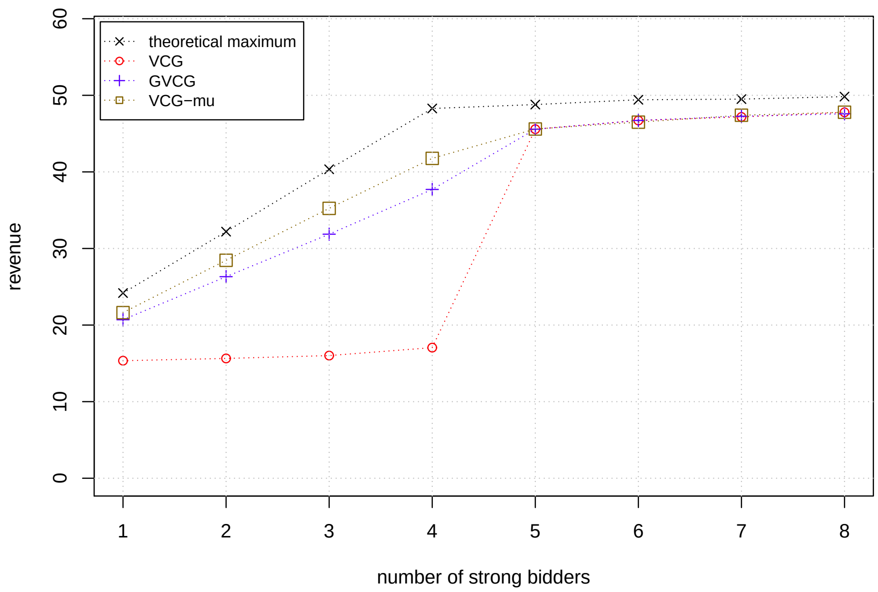

Figure 6: Expected revenues in the first asymmetric case

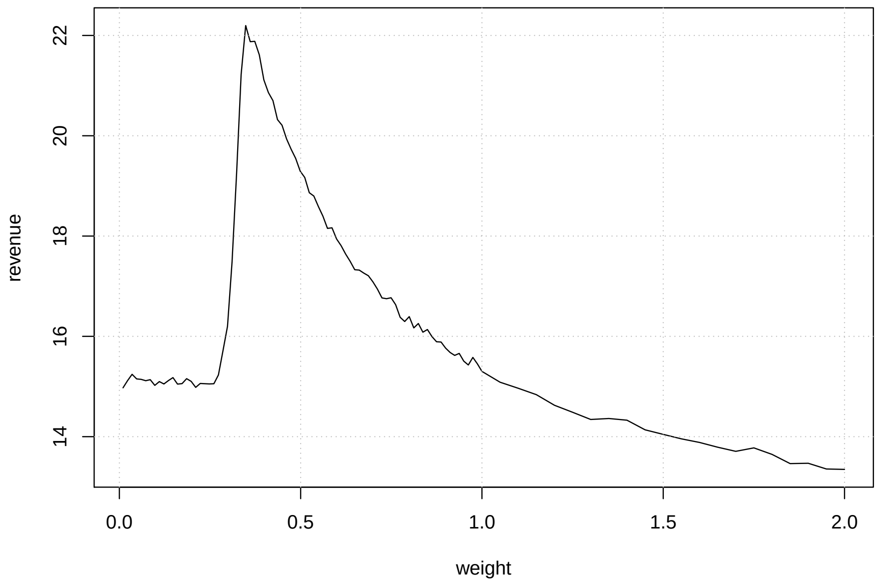

Figure 7: Expected revenues as a function of weights with one strong bidder, first asymmetric case

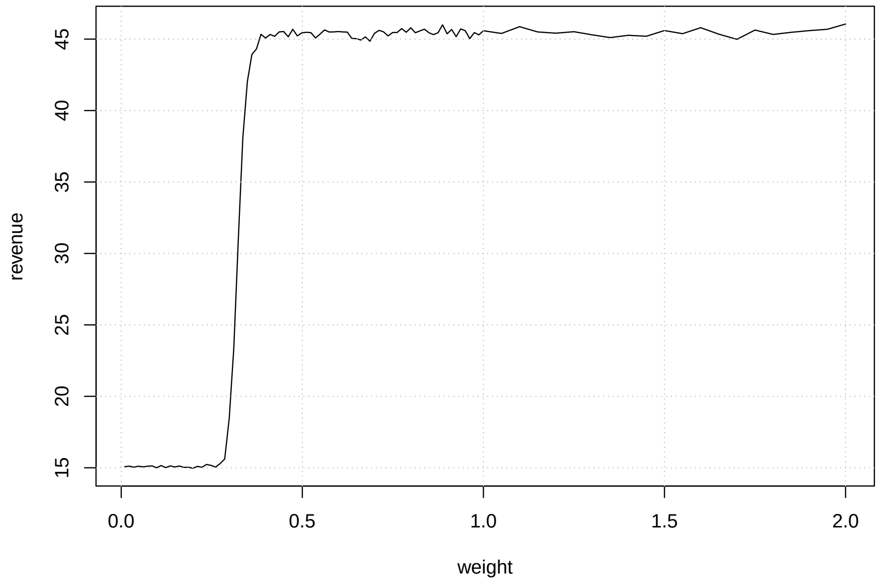

Figure 8: Expected revenues as a function of weights with five strong bidders, first asymmetric case

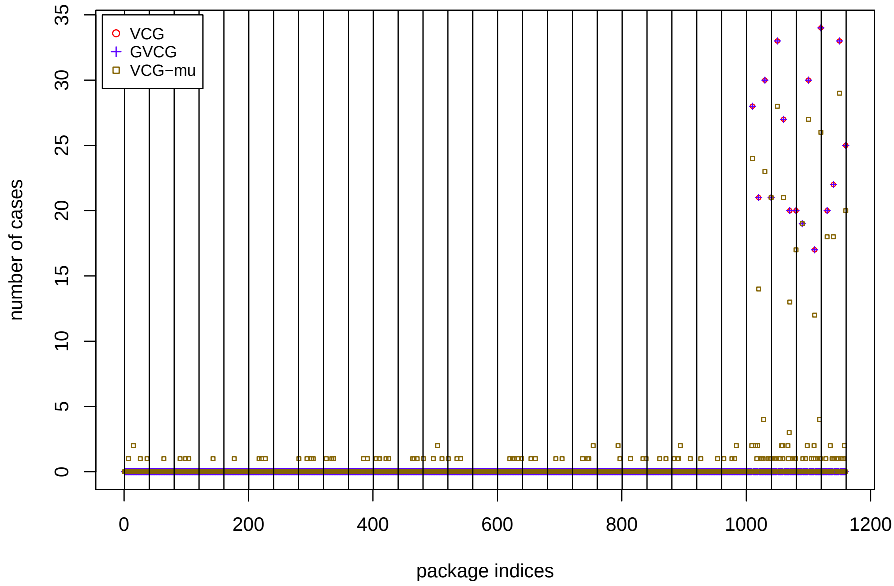

Figure 9: Package distribution of the bidders in the first asymmetric case

improve the expected revenue in cases where we had more than four bidders which is a different result than that we got in the non-overlapping case.

The reason behind this improvement is that the strong bidders do not always win: In few cases, the weaks can overbid the strong players. This means that our former argument does not hold in the situations where we have more strong bidders than floors. Now we cannot be sure that a strong bidder causes damage only to other strong bidders in the auction by his presence. It might happen that a weak bidder could benefit from the exclusion of the strong. This "inter-group" damage destroys the former neutrality of the weights in these cases. By lowering the weight of the strong bidders, we can make them pay more for the damage caused to weaker bidders. The weight-revenue function with 1 and 5 strong bidders are shown in Figures 12 and 13. Table 5 shows the average numbers of packages in rounds. We can see that the difference from the efficient allocation was larger in this case; the weak bidder won more in average in the VCG-μ auction.

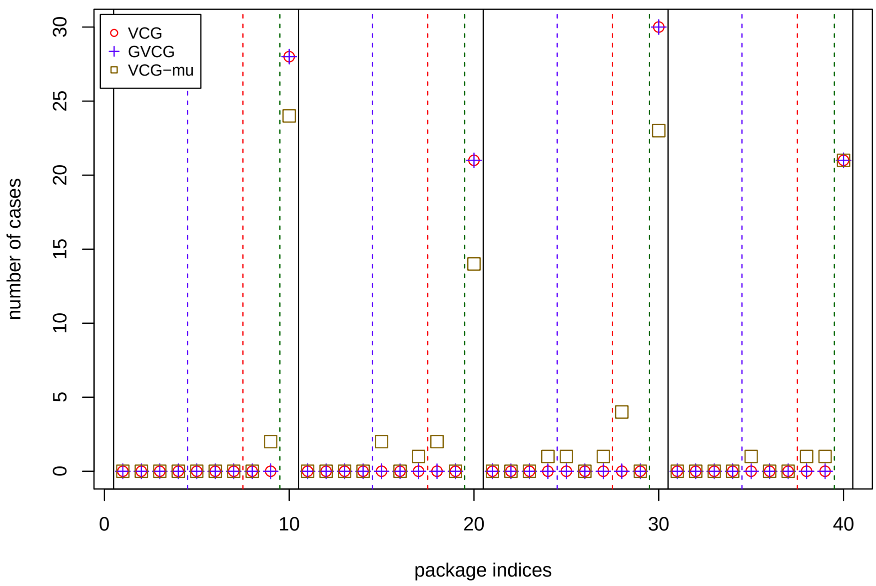

Figure 10: Package distribution of one bidder in the first asymmetric case

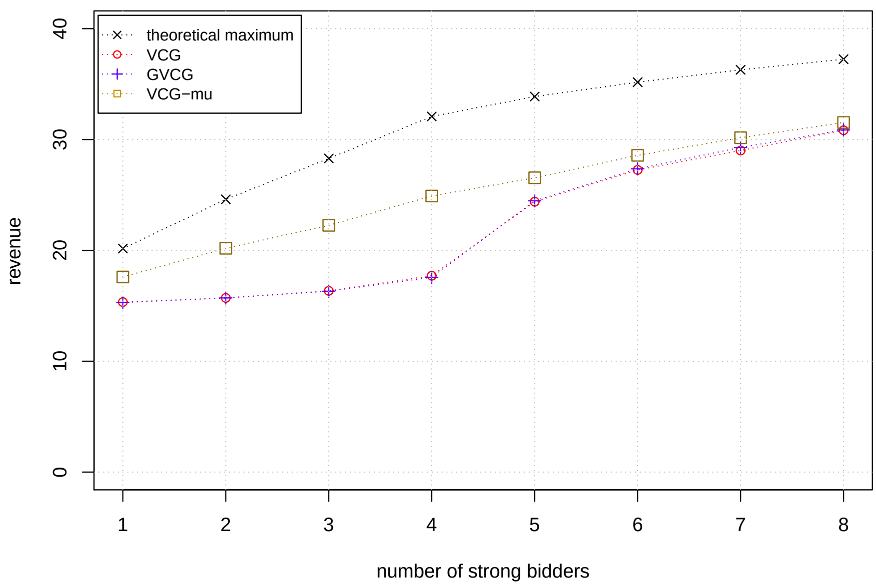

Figure 11: Expected revenues in the second asymmetric case

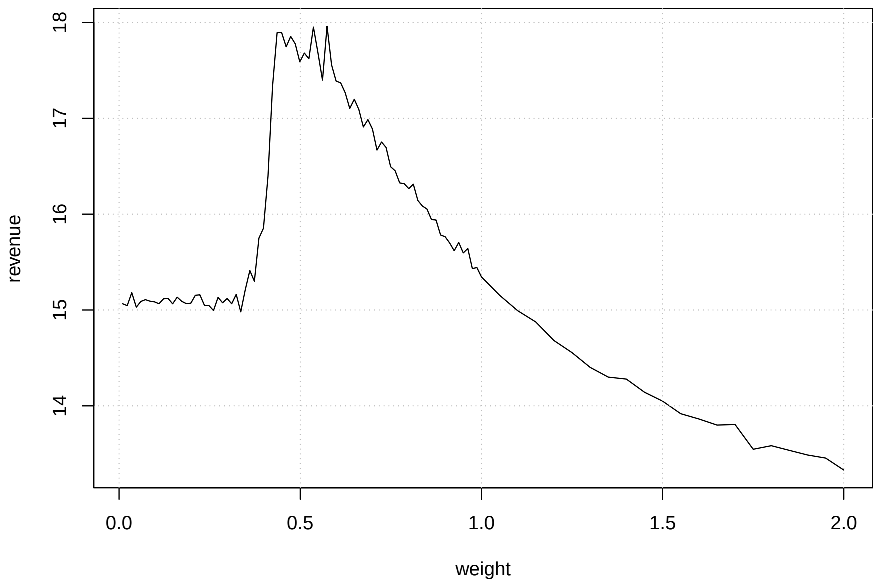

Figure 12: Expected revenues as a function of weights with one strong bidder, second asymmetric case

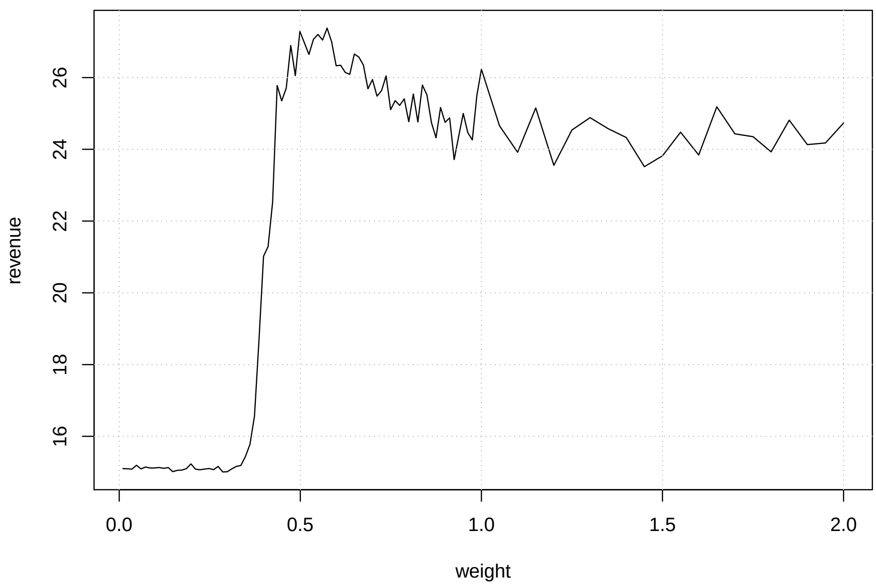

Figure 13: Expected revenues as a function of weights with five strong bidders, second asymmetric case

4 Concluding Remarks

The aim of this paper was to discuss the main issues of combinatorial auctions, to present some of the solutions proposed by the literature and to describe an alternative way to overcome some of the difficulties.

We have seen that combinatorial auctions have several attractive features which make them a very good alternative in multi-object auction design but the complex structure leads to computational and economic problems. The goals of the auctioneer are sometimes hard to realize. While with the Vickrey–Clarke–Groves auctions we possess a tool to implement efficient allocations, revenue-maximization is still an unsolved problem. We can investigate the revenue properties of different auctions but finding the optimal one is still an open question even in small environments. The issue of computational complexity makes the implementation of combinatorial auctions in larger settings difficult and after a point impossible. To avoid this problem, we have to impose assumptions on the primitives of the auction. The usual approach in the literature is to restrict the preferences of the players in a very strict way. The papers use binary valuations, distinguish between global and local players or impose other assumptions like partial additivity to make the underlying optimization problem simpler to solve. The simulations are usually done in very small settings, with only two or three players and few goods where every combination is allowed. The economic insight is sometimes hidden in such small environments.

This paper used a different approach. Through an example where we auction off a large office complex with a possibly large number of interior "blocks", we presented a different strategy. Instead of making too hard assumptions on the bidders' preferences, we restricted the set of possible packages. For several reasons (e.g. avoiding chaos from commuting) we used the restriction that the available offices (block packages) are constructed from neighboring connected blocks in the same floor. As a consequence of this structure, the number of possible packages fell dramatically and allowed us to investigate larger systems through simulations.

We considered four simulation scenarios in the paper. The first two assumed symmetric bidders with uniform $[0, 1]$ and $[2, 3]$ item valuations, respectively. We saw that in the first case, our auctions gave the same result: they did not perform well if the number of agents was smaller than the number of floors, but after this point the revenue jumped to a high level and as the number of bidders grew, it converged to the theoretical maximum. In the second case, we were able to improve the revenues of the VCG mechanism in the critical cases by using the GVCG auction. The reason behind this was that the GVCG auction charged reserve prices in these cases.

The second pair of simulations assumed asymmetric bidders. In both scenarios, we had 25 "weak" bidders and one to eight dominating "strong ones". In the first case, the item-valuations were not overlapping. The VCG auction had a poor performance if the number of strong bidders was smaller than the number of floors. In such cases, the strong bidders did not pay close to their real valuations and thus the VCG was not able to reach high revenues. However, we were able to use the "reserve price" property of the GVCG auction again. Moreover, by giving up efficiency and modifying the importance of the bidders in the auction, we could reach even higher revenues with the VCG-μ auction. After 5 strong bidders, the three mechanism coincided.

The overlapping case gave slightly different results. Because of the weaker "overlapping domination", the GVCG was not able to improve the bad revenues of the VCG auction. On the other hand, the non-efficient VCG-μ mechanism gave significant additional revenue even in the cases with more than five strong bidders. Thus, by either charging "reserve prices" or by artificially creating close rivalry between the bidders of different types we were able to reach revenues close to the theoretical maximum.

All in all, our simplification not only let us solve the auctioning problem in higher-dimensional environments but also gave us valuable economic insights into the way how these auctions work and what this particular restriction on the structure implies in the auction process. We think that this structure can be used in other allocation situations too where the winners cannot obtain goods from more than one subset at the same time for some reasons.[17] Thus, this work can be used as a preparation for a future research that might possibly use analytical tools to generalize the conclusions.

[17] Consider an auction of rights for some activity in a country where the different subsets are regions. In such a situation, it could happen that we do not want to allow a company to have a right in more than one region since this would possibly create an ill market structure.

References

A. Andersson and J. Wilenius. A new analysis of revenue in the combinatorial and simultaneous auction. Technical Report 2009-001, Department of Information Technology, Uppsala University, Jan. 2009. Updated May 2009.

M. Armstrong. Optimal multi-object auctions. *Review of Economic Studies*, 67 (3):455–81, July 2000.

E. H. Clarke. Multipart pricing of public goods. *Public Choice*, 11(1):17–33, Sept. 1971.

P. C. Cramton, Y. Shoham, and R. Steinberg. *Combinatorial auctions*. MIT Press, 2010.

S. de Vries and R. V. Vohra. Combinatorial auctions: A survey. *INFORMS Journal on Computing*, 15(3):284–309, 2003.

T. Groves. Incentives in Teams. *Econometrica*, 41(4):617–631, 1973.

P. Klemperer. What Really Matters in Auction Design. *Journal of Economic Perspectives*, 16(1):169–189, Jan. 2002.

V. Krishna and M. Perry. Efficient mechanism design. Technical report, Pennsylvania State University / The Hebrew University of Jerusalem, April 1998.

V. Krishna and R. W. Rosenthal. Simultaneous auctions with synergies. *Games and Economic Behavior*, 17(1):1–31, November 1996.

J. Levin. An Optimal Auction for Complements. *Games and Econ. Behavior*, 18 (2):176–92, Feb. 1997.

A. Likhodedov and T. W. Sandholm. Methods for boosting revenue in combinatorial auctions. Technical report, Computer Science Department, Carnegie Mellon University, 2004.

A. Likhodedov and T. W. Sandholm. Approximating revenue-maximizing combinatorial auctions. Technical report, Computer Science Department, Carnegie Mellon University, 2005.

D. Monderer and M. Tennenholtz. Asymptotically optimal multi-object auctions. Technical report, Faculty of Industrial Engineering and Management, TechnionIsrael Institute of Technology, 2005.

R. B. Myerson. Optimal auction design. *Mathematics of Operations Research*, 6 (1):58–73, 1981.

M. W. Padberg. Perfect zero-one matrices. *Mathematical Programming*, 6:180–196, 1974.

S. J. Rassenti, V. L. Smith, and R. L. Bulfin. A Combinatorial Auction Mechanism for Airport Time Slot Allocation. *The Bell Journal of Economics*, 13(2): 402–417, 1982.

M. H. Rothkopf, A. Pekeč, and R. M. Harstad. Computationally Manageable Combinational Auctions. *Management Science*, 44(8):1131–1147, 1998.

T. Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence*, 135(1-2):1–54, February 2002.

W. Vickrey. Counterspeculation, Auctions, and Competitive Sealed Tenders. *The Journal of Finance*, 16(1):8–37, 1961.

R. V. Vohra. *Mechanism Design: A Linear Programming Approach*. Econometric Society Monographs. Cambridge University Press, 2011.

Appendix

A Source Code

```
library(Rsymphony)

############################################################
# Definition of the auxiliary matrices
############################################################

## matrix B: package structure of a single floor ##########

B.mat <- function(L){
  #L: number of blocks in a floor
  B <- diag(L)
  for (l in 2:L){
    B.add <- matrix(0,L,L-l+1)
    for (k in 1:(L-l+1)){
      B.add[k:(k+l-1),k] <- t(rep(1,l))
    }
    B <- cbind(B,B.add)
  }
  B
}

## matrix A: the coefficient matrix of the SPP ############

A.mat <- function(N,K=2,L=2){
  #N: number of bidders
  #K: number of floors
  #number of blocks:
  H <- K*L*(L+1)/2
  A <- matrix(0,N+K*L,N*H) #the contraint matrix
  B <- B.mat(L)
  #non-overlapping packages:
  A[1:(K*L),] <- t(rep(1,N)) %x% diag(K) %x% B
  #everyone should get one package at the end:
  A[(K*L+1):(K*L+N),] <- diag(N) %x% t(rep(1,H))
  A
}

############################################################
# Definiton of the Set Packing Problems
############################################################

## The binary SPP ##########################################

spp <- function(N,v,A){
  #directions of the inequalities:
  dir <- rep("<=",dim(A)[1])
  #right hand side:
  b <- rep(1,dim(A)[1])
  #solution:
  x <- Rsymphony_solve_LP(v,A,dir,b,types="B",max=TRUE)
```

```
  list(x$objval,x$solution)
}

## The linear relaxation ##################################

spp.lp <- function(N,v,A){
  dir <- rep("<=",dim(A)[1])
  b <- rep(1,dim(A)[1])
  x <- Rsymphony_solve_LP(v,A,dir,b,types="C",max=TRUE)
  list(obj=x$objval,sol=x$solution)
}

############################################################
# Valuation function
############################################################

vals.unif.N <- function(N,K,L,vlim=c(0,1),alpha=.1,B){
  H <- K*L*(L+1)/2
  #item-valuations:
  v <- runif(N*K*L,vlim[1],vlim[2])
  #synergies:
  v.syn <- rep(1,L)
  for (l in 2:L){
    v.add <- t(rep(1+(l-1)/l*alpha,L-l+1))
    v.syn <- c(v.syn,v.add)
    }
  v.syn <- rep(v.syn,K*N)
  #package valuations:
  as.vector((t(v) %*% (diag(K*N) %x% B)) * v.syn)
}

## An example ##############################################

vals.unif.N(2,2,4,c(0,1),.1,B.mat(4))

############################################################
# Finding a non-integral extreme point
############################################################

# randomly generated valuation profiles to find an extreme
# point which is non-integral
vv <- matrix(0,20,18)
for (i in 1:20){
  vv[i,] <- vals.unif.N(3,1,3,v=c(0,1),alpha=.2,B.mat(3))
  x <- spp.lp(3,vv[i,],A=A.mat(3,1,3))$sol
  print(x)
}

#the result:

#the valuation profile
#[1]  0.3847793 0.9222561 0.5643365 1.4377389 1.6352518
#[6]  2.1208881 0.7114423 0.2962169 0.9145589 1.1084251
#[11] 1.3318534 2.1785139 0.0013681 0.9905686 0.1241367
#[16] 1.0911304 1.2261759 1.2648832
```

```
#
#gives the optimal solution
#[1] 0.0 0.0 0.0 0.0 0.0 0.5 0.5 0.0 0.5
#    0.0 0.0 0.0 0.0 0.5 0.0 0.0 0.0 0.0
#
#with objective function value
#[1] 2.368729
#
#which is different from the solution to
#the integer programming problem
#[1] 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0
#
#with objective function value
#[1] 2.352298

###########################################################
# SIMULATIONS
###########################################################

## THEORETICAL MAXIMUM ###################################

# asymmetric bidders:
# (symmetric bidders is a special case)

sim.max.N.asym <- function(N.vec,K,L,v.mat,alpha.vec,rept){
  #N.vec: vector with the number of people of different types
  #v.mat: Nx2 matrix with the intervals of the valuations for different types
  #alpha.vec: vector with the synergy parameters
  #rept: number of repetitions
  N <- sum(N.vec)
  A <- A.mat(N,K,L)
  B <- B.mat(L)
  r.max.N.s <- rep(0,rept)
  #generate valuation profiles and solve the auctions:
  for (i in 1:rept){
    vals <- rep(0,0)
    for (n in 1:length(N.vec)){
      vals <- c(vals,vals.unif.N(N.vec[n],K,L,vlim=as.vector(v.mat[n,]),alpha=alpha.
          vec[n],B))
    }
    r.max.N.s[i] <- as.real(spp(N,vals,A)[1])
  }
  #histogram and mean, minimum and maximum as outputs:
  hist(r.max.N.s)
  c(mean(r.max.N.s),min(r.max.N.s),max(r.max.N.s))
}

## VCG AUCTION  ##########################################

# VCG solver:

vcg <- function(N,K,L,v){
  H <- K*L*(L+1)/2
  #auxiliary matrices for the problems where a player is
  #not excluded and wher he is excluded:
```

```
  A.0 <- A.mat(N,K,L)
  A.1 <- A.mat(N-1,K,L)
  #valuation profiles excluding the nth player:
  v.excl <- matrix(0,N,(N-1)*H)
  #overall utility of the other players where n is included:
  val.0.excl <- rep(0,N)
  #overall utility of the other players where n is excluded:
  val.1.excl <- rep(0,N)
  #allocation and prices when n is excluded
  x.excl <- matrix(0,N,(N-1)*H)
  p.excl <- rep(0,N) #vickrey prices
  #SPP for N people:
  x <- spp(N,v,A.0)
  x.sol <- unlist(x[2])
  #SPP for the cases when a player is excluded:
  for (n in 1:N){
    v.excl[n,] <- v[-((1+(n-1)*H):(H+(n-1)*H))]
    x.excl[n,] <- x.sol[-((1+(n-1)*H):(H+(n-1)*H))]
    val.0.excl[n] <- x.excl[n,] %*% v.excl[n,]
    val.1.excl[n] <- as.real(spp(N-1,v.excl[n,],A.1)[1])
    p.excl[n] <- val.1.excl[n] - val.0.excl[n]
  }
  list(matrix=cbind(p.excl,val.0.excl,val.1.excl),sol=x.sol)
}

# VCG simulations with asymmetric bidders:
# (simmetric bidders is a special case)

sim.vcg.N.asym <- function(N.vec,K,L,v.mat,alpha.vec,rept){
  B <- B.mat(L)
  N <- sum(N.vec)
  r.vcg.N <- rep(0,rept)
  for (i in 1:rept){
    vals <- rep(0,0)
    for (n in 1:length(N.vec)){
      vals <- c(vals,vals.unif.N(N.vec[n],K,L,vlim=as.vector(v.mat[n,]),alpha=alpha.
          vec[n],B))
    }
    r.vcg.N[i] <- rep(1,N) %*% (vcg(N,K,L,vals)$matrix)[,1]
  }
  hist(r.vcg.N)
  c(mean(r.vcg.N))
}

## GVCG AUCTION ############################################

# GVCG solver:

gvcg <- function(N,K,L,v,vbar){
  H <- K*L*(L+1)/2
  A.0 <- A.mat(N,K,L)
  v.excl <- matrix(0,N,(N-1)*H)
  #"mixed" valuation profiles where one bidder plays his
  #basis strategy:
  v.mixed <- matrix(0,N,N*H)
```

```
  val.0.excl <- rep(0,N)
  #overall utility of players given the mixed profile:
  val.1.mixed <- rep(0,N)
  x.excl <- matrix(0,N,(N-1)*H)
  #generalized Vickrey prices:
  p.mixed <- rep(0,N)
  x <- spp(N,v,A.0)
  x.sol <- unlist(x[2])
  #calculation of the payments, case distinction is needed:
  v.excl[1,] <- v[-(1:H)]
  x.excl[1,] <- x.sol[-(1:H)]
  v.mixed[1,] <- c(vbar[1:H],v[(1+H):(N*H)])
  val.0.excl[1] <- x.excl[1,] %*% v.excl[1,]
  val.1.mixed[1] <- as.real(spp(N,v.mixed[1,],A.0)[1])
  p.mixed[1] <- val.1.mixed[1] - val.0.excl[1]
  if (N==2) {
    v.excl[2,] <- v[-((H+1):(N*H))]
    x.excl[2,] <- x.sol[-((H+1):(N*H))]
    v.mixed[2,] <- c(v[1:H],vbar[(1+H):(N*H)])
    val.0.excl[2] <- x.excl[2,] %*% v.excl[2,]
    val.1.mixed[2] <- as.real(spp(N,v.mixed[2,],A.0)[1])
    p.mixed[2] <- val.1.mixed[2] - val.0.excl[2]
  }
  else{
    for (n in 2:(N-1)){
      v.excl[n,] <- v[-((1+(n-1)*H):(H+(n-1)*H))]
      x.excl[n,] <- x.sol[-((1+(n-1)*H):(H+(n-1)*H))]
      v.mixed[n,] <- c(v[1:((n-1)*H)],vbar[(1+(n-1)*H):(H+(n-1)*H)],v[(1+H+(n-1)*H):(N*H)])
      val.0.excl[n] <- x.excl[n,] %*% v.excl[n,]
      val.1.mixed[n] <- as.real(spp(N,v.mixed[n,],A.0)[1])
      p.mixed[n] <- val.1.mixed[n] - val.0.excl[n]
    }
    v.excl[N,] <- v[-((1+(N-1)*H):(H+(N-1)*H))]
    x.excl[N,] <- x.sol[-((1+(N-1)*H):(H+(N-1)*H))]
    v.mixed[N,] <- c(v[1:((N-1)*H)],vbar[(1+(N-1)*H):(H+(N-1)*H)])
    val.0.excl[N] <- x.excl[N,] %*% v.excl[N,]
    val.1.mixed[N] <- as.real(spp(N,v.mixed[N,],A.0)[1])
    p.mixed[N] <- val.1.mixed[N] - val.0.excl[N]

  }
  list(matrix=cbind(p.mixed,val.0.excl,val.1.mixed),sol=x.sol)
}

# GVCG simulations with asymmetric bidders:
# (simmetric bidders is a special case)

sim.gvcg.N.asym <- function(N.vec,K,L,v.mat,vbar.mat,alpha.vec,rept){
  B <- B.mat(L)
  N <- sum(N.vec)
  r.vcg.N <- rep(0,rept)
  for (i in 1:rept){
    vals <- rep(0,0)
    vbars <- rep(0,0)
    for (n in 1:length(N.vec)){
```

```
      vals <- c(vals,vals.unif.N(N.vec[n],K,L,vlim=as.vector(v.mat[n,]),alpha=alpha.vec[n],B))
      vbars <- c(vbars,vals.unif.N(N.vec[n],K,L,vlim=as.vector(vbar.mat[n,]),alpha=alpha.vec[n],B))
    }
    r.vcg.N[i] <- rep(1,N) %*% (gvcg(N,K,L,vals,vbars)$matrix)[,1]
  }
  hist(r.vcg.N)
  c(mean(r.vcg.N))
}

## VCG-MU AUCTION ##########################################

# VCG-mu solver:

vcg.mu <- function(N,K,L,v,mu=rep(1,N)){
  H <- K*L*(L+1)/2
  A.0 <- A.mat(N,K,L)
  A.1 <- A.mat(N-1,K,L)
  #weighted valuation functions:
  v <- v * (mu %x% rep(1,H))
  v.excl <- matrix(0,N,(N-1)*H)
  val.0.excl <- rep(0,N)
  val.1.excl <- rep(0,N)
  x.excl <- matrix(0,N,(N-1)*H)
  p.excl <- rep(0,N)
  x <- spp(N,v,A.0)
  x.sol <- unlist(x[2])
  for (n in 1:N){
    v.excl[n,] <- v[-((1+(n-1)*H):(H+(n-1)*H))]
    x.excl[n,] <- x.sol[-((1+(n-1)*H):(H+(n-1)*H))]
    val.0.excl[n] <- x.excl[n,] %*% v.excl[n,]
    val.1.excl[n] <- as.real(spp(N-1,v.excl[n,],A.1)[1])
    #modified payments:
    p.excl[n] <- (val.1.excl[n] - val.0.excl[n]) / mu[n]
  }
  list(matrix=cbind(p.excl,val.0.excl,val.1.excl),sol=x.sol)
}

# VCG-mu simulations with asymmetric bidders:
# (simmetric bidders is a special case)

sim.vcg.N.asym.mu <- function(N.vec,K,L,v.mat,alpha.vec,mu.vec=rep(1,length(N.vec)),rept){
  B <- B.mat(L)
  N <- sum(N.vec)
  #mu vector of all the bidders:
  mu <- rep(0,0)
  for (n in 1:length(N.vec)){
    mu <- c(mu,rep(mu.vec[n],N.vec[n]))
  }
  r.vcg.N <- rep(0,rept)
  for (i in 1:rept){
    vals <- rep(0,0)
    for (n in 1:length(N.vec)){
```

```
      vals <- c(vals,vals.unif.N(N.vec[n],K,L,v=as.vector(v.mat[n,]),alpha=alpha.vec[
          n],B))
    }
    r.vcg.N[i] <- rep(1,N) %*% (vcg.mu(N,K,L,vals,mu)$matrix)[,1]
  }
  hist(r.vcg.N)
  c(mean(r.vcg.N))
}

##########################################################
# SIMULATIONS
##########################################################

# 4 blocks per floor and 4 floors in all cases
# item valuations are different
# the synergy parameter is always .2

## symmetric types, I (0 is included) ####################

# item valuations are i.i.d. uniform [0,1] draws.

# expected revenues:

revs.max.N <- rep(0,30)
revs.max.N.vcg <-rep(0,30)
revs.max.N.gvcg <-rep(0,30)

# all auctions give 0 in the one-bidder case

# other cases:

for (i in 2:30){
  print(i)
  revs.max.N[i] <- sim.max.N(i,4,4,c(0,1),.2,100)[1]
  revs.max.N.vcg[i] <- sim.vcg.N.asym(i,4,4,matrix(c(0,1),1,2),.2,100)[1]
  revs.max.N.gvcg[i] <- sim.gvcg.N.asym(i,4,4,matrix(c(0,1),1,2),matrix(c(0,0),1,2)
      ,.2,100)[1]
}

# figure of the expected revenues:

postscript(file="sim01-01.eps",paper="special",width=8, height=6,horizontal=FALSE)
plot(revs.max.N,ylim=c(0,20),xlab="number of bidders",ylab="revenue",type="o",lty=3,
    pch=4)
lines(revs.max.N.vcg,col="red",type="o",lty=3)
lines(revs.max.N.vcg,col="blue",type="o",lty=3,pch=3)
grid(col="grey")
legend(.1,20.5, c("theoretical maximum","VCG","GCVG"), bg="white",cex=0.85, col=c("
    black","red","blue"), pch=c(4,1,3),lty=3)
dev.off()

# package distribution with 100 draws:

vals.sym <- matrix(0,100,10*40)
vbars.sym <- matrix(0,100,10*40)
```

```
vcg.sol.sym <- matrix(0,100,10*40)
gvcg.sol.sym <- matrix(0,100,10*40)

# VCG and GVCG simulations:

for (i in 1:100){
  print(i)
  vals.sym[i,] <- vals.unif.N(10,4,4,c(0,1),.2,B.mat(4))
  vbars.sym[i,] <- vals.unif.N(10,4,4,c(0,0),.2,B.mat(4))
  vcg.sol.sym[i,] <- vcg(10,4,4,vals.sym[i,])$sol
  gvcg.sol.sym[i,] <- gvcg(10,4,4,vals.sym[i,],vbars.sym[i,])$sol
}

# summing up gives the distributions:

distr.vcg.sym <- rep(1,100) %*% vcg.sol.sym
distr.gvcg.sym <- rep(1,100) %*% gvcg.sol.sym

# figure for one player:

postscript(file="sim01-02.eps",paper="special",width=8, height=6,horizontal=FALSE)
plot(distr.vcg.sym[1:40],type="p",col="red",cex=1.2,ylab="number of cases",xlab="
    package indices")
lines(distr.gvcg.sym[1:40],type="p",pch=3,col="blue",cex=1.2)
abline(v=c(seq(.5,40.5,by=10)),lty=1)
abline(v=c(seq(4.5,34.5,by=10)),col="blue",lty=2)
abline(v=c(seq(7.5,37.5,by=10)),col="red",lty=2)
abline(v=c(seq(9.5,39.5,by=10)),col="dark green",lty=2)
legend(-0.2,9.25, c("VCG","GVCG"), cex=.8, bg="white",col=c("red","blue"),pch=c(1,3))
dev.off()

# figure for all the players:

postscript(file="sim01-03.eps",paper="special",width=8, height=6,horizontal=FALSE)
plot(distr.vcg.sym[1:400],type="p",col="red",cex=.5,ylab="number of cases",xlab="
    package indices")
lines(distr.gvcg.sym[1:400],type="p",pch=3,col="blue",cex=.5)
abline(v=c(seq(.5,400.5,by=40)),lty=1)
legend(-10.8,9.22, c("VCG","GVCG"), cex=.8, bg="white",col=c("red","blue"),pch=c(1,3)
    )
dev.off()

# average assignments of packages of different size:

distr.vcg.sym.tr <- matrix(distr.vcg.sym,ncol=10,byrow=TRUE)

sum(distr.vcg.sym.tr[,1:4])/100   #size 1
sum(distr.vcg.sym.tr[,5:7])/100   #size 2
sum(distr.vcg.sym.tr[,8:9])/100   #size 3
sum(distr.vcg.sym.tr[,10])/100    #size 4

## symmetric types, II (0 is not included) ################

# item valuations are i.i.d. uniform [2,3] draws.

```

```r
#expected revenues:

revs.max.N.2 <- rep(0,30)
revs.max.N.vcg.2 <-rep(0,30)
revs.max.N.gvcg.2 <-rep(0,30)

# one-bidder case:

revs.max.N.2[1] <- sim.max.N.asym(1,4,4,matrix(c(2,3),1,2),.2,100)[1]
revs.max.N.gvcg.2[1] <- vals.unif.N(1,4,4,c(2,2),.2,B.mat(4))[40]

# two or more bidders:

for (i in 2:30){
  print(i)
  revs.max.N.2[i] <- sim.max.N.asym(i,4,4,matrix(c(2,3),1,2),.2,100)[1]
  revs.max.N.vcg.2[i] <- sim.vcg.N.asym(i,4,4,matrix(c(2,3),1,2),.2,100)[1]
  revs.max.N.gvcg.2[i] <- sim.gvcg.N.asym(i,4,4,matrix(c(2,3),1,2),matrix(c(2,2),1,2)
      ,.2,100)[1]
}

# figure of the expected revenue:

postscript(file="sim02-01.eps",paper="special",width=8, height=6,horizontal=FALSE)
plot(revs.max.N.2,ylim=c(0,65),ylab="revenue",xlab="number of bidders",type="o",lty
    =3,pch=4)
lines(revs.max.N.vcg.2,col="red",lty=3,type="o")
lines(revs.max.N.gvcg.2,col="blue",lty=3,type="o",pch=3)
grid(col="grey")
legend(.05,66.8, c("theoretical maximum","VCG","GVCG"), bg="white", cex=0.85, col=c("
    black","red","blue"),pch=c(1,4,3),lty=3)
dev.off()

# package distributions:

vals.sym.2 <- matrix(0,100,10*40)
vbars.sym.2 <- matrix(0,100,10*40)
vcg.sol.sym.2 <- matrix(0,100,10*40)
gvcg.sol.sym.2 <- matrix(0,100,10*40)

for (i in 1:100){
  print(i)
  vals.sym.2[i,] <- vals.unif.N(10,4,4,c(2,3),.2,B.mat(4))
  vbars.sym.2[i,] <- vals.unif.N(10,4,4,c(2,2),.2,B.mat(4))
  vcg.sol.sym.2[i,] <- vcg(10,4,4,vals.sym.2[i,])$sol
  gvcg.sol.sym.2[i,] <- gvcg(10,4,4,vals.sym.2[i,],vbars.sym.2[i,])$sol
}

distr.vcg.sym.2 <-  rep(1,100) %*% vcg.sol.sym.2
distr.gvcg.sym.2 <- rep(1,100) %*% gvcg.sol.sym.2

# figure of one bidder:

postscript(file="sim02-02.eps",paper="special",width=8, height=6,horizontal=FALSE)
```

```
plot(distr.vcg.sym.2[1:40],type="p",col="red",cex=1.2,ylab="number of cases",xlab="
    package indices")
lines(distr.gvcg.sym.2[1:40],type="p",pch=3,col="blue",cex=1.2)
abline(v=c(seq(.5,40.5,by=10)),lty=1)
abline(v=c(seq(4.5,34.5,by=10)),col="blue",lty=2)
abline(v=c(seq(7.5,37.5,by=10)),col="red",lty=2)
abline(v=c(seq(9.5,39.5,by=10)),col="dark green",lty=2)
legend(-0.2,8.2, c("VCG","GVCG"), cex=.8, bg="white",col=c("red","blue"),pch=c(1,3))
dev.off()

# average assignments of packages of different size:

distr.vcg.sym.tr.2 <- matrix(distr.vcg.sym.2,ncol=10,byrow=TRUE)

sum(distr.vcg.sym.tr.2[,1:4])/100   #size 1
sum(distr.vcg.sym.tr.2[,5:7])/100   #size 2
sum(distr.vcg.sym.tr.2[,8:9])/100   #size 3
sum(distr.vcg.sym.tr.2[,10])/100    #size 4

## asymmetric types, I (non-overlapping case) #############

# 25 bidders with i.i.d. uniform [0,1] item valuations
# N bidder with i.i.d. uniform [2,3] item valuations

# optimal mu values by grid search:

opt.mus <- rep(0,8)

mu.search <- matrix(0,8,100)

mus <- c(seq(.01,1,length=80),seq(1.05,2,length=20))

for (n in 1:8){
  for (i in 1:100){
    print(c(n,i))
    mu.search[n,i] <- sim.vcg.N.asym.mu(c(25,n),4,4,matrix(c(0,1,2,3),2,2,byrow=TRUE)
        ,c(.2,.2),mu.vec=c(1,mus[i]),25)[1]
  }
  opt.mus[n] <- mus[which.max(mu.search[n,])]
}

# weight revenue function with one strong bidder:

postscript(file="sim03-02.eps",paper="special",width=8, height=6,horizontal=FALSE)
plot(mus,mu.search[1,],type="l",ylab="revenue",xlab="weight")
grid(col="grey")
dev.off()

# weight revenue function with five strong bidders:

postscript(file="sim03-03.eps",paper="special",width=8, height=6,horizontal=FALSE)
plot(mus,mu.search[5,],type="l",ylab="revenue",xlab="weight")
grid(col="grey")
dev.off()
```

```
# expected revenues:

revs.max.N.asym <- rep(0,8)
revs.vcg.N.asym <-rep(0,8)
revs.gvcg.N.asym <-rep(0,8)
revs.vcgmu.N.asym <-rep(0,8)

for (i in 1:8){
  print(i)
  revs.max.N.asym[i] <- sim.max.N.asym(c(25,i),4,4,matrix(c(0,1,2,3),2,2,byrow=TRUE),
      c(.2,.2),100)[1]
  revs.vcg.N.asym[i] <- sim.vcg.N.asym(c(25,i),4,4,matrix(c(0,1,2,3),2,2,byrow=TRUE),
      c(.2,.2),100)[1]
  revs.gvcg.N.asym[i] <- sim.gvcg.N.asym(c(25,i),4,4,matrix(c(0,1,2,3),2,2,byrow=TRUE
      ),matrix(c(0,0,2,2),2,2,byrow=TRUE),c(.2,.2),100)[1]
  revs.vcgmu.N.asym[i]  <- sim.vcg.N.asym.mu(c(25,i),4,4,matrix(c(0,1,2,3),2,2,byrow=
      TRUE),c(.2,.2),mu.vec=c(1,opt.mus[i]),100)[1]
}

# figure: expected revenues:

postscript(file="sim03-01.eps",paper="special",width=8, height=6,horizontal=FALSE)
plot(revs.max.N.asym,type="o",ylim=c(0,58),ylab="revenue",xlab="number of strong
     bidders",lty=3,pch=4)
lines(revs.vcg.N.asym,col="red",type="o",lty=3,pch=1)
lines(revs.gvcg.N.asym,col="blue",type="o",lty=3,pch=3)
lines(revs.vcgmu.N.asym,col="darkgoldenrod4",type="o",lty=3,pch=22,cex=1.6)
grid(col="grey")
legend(.78,59.6, c("theoretical maximum","VCG","GVCG","VCG-mu"), bg="white", cex
     =0.85, col=c("black","red","blue","darkgoldenrod4"), pch=c(4,1,3,22),lty=3)
dev.off()

# package distributions:

vals.asym <- matrix(0,100,29*40)
vbars.asym <- matrix(0,100,29*40)
vcg.sol.asym <- matrix(0,100,29*40)
gvcg.sol.asym <- matrix(0,100,29*40)
vcg.mu.sol.asym <- matrix(0,100,29*40)

for (i in 1:100){
  print(i)
  vals.asym[i,1:(40*25)] <- vals.unif.N(25,4,4,c(0,1),.2,B.mat(4))
  vbars.asym[i,1:(40*25)] <- vals.unif.N(25,4,4,c(0,0),.2,B.mat(4))
  vals.asym[i,(1+40*25):(40*29)] <- vals.unif.N(4,4,4,c(2,3),.2,B.mat(4))
  vbars.asym[i,(1+40*25):(40*29)] <- vals.unif.N(4,4,4,c(2,2),.2,B.mat(4))
  vcg.sol.asym[i,] <- vcg(29,4,4,vals.asym[i,])$sol
  gvcg.sol.asym[i,] <- gvcg(29,4,4,vals.asym[i,],vbars.asym[i,])$sol
  vcg.mu.sol.asym[i,] <- vcg.mu(29,4,4,vals.asym[i,],c(rep(1,25),rep(opt.mus[4],4)))$
      sol
}

distr.vcg.asym <- rep(1,100) %*% vcg.sol.asym
distr.gvcg.asym <- rep(1,100) %*% gvcg.sol.asym
distr.vcg.mu.asym <- rep(1,100) %*% vcg.mu.sol.asym
```

```
# figure: package distribution of a strong bidder:

postscript(file="sim03-05.eps",paper="special",width=8, height=6,horizontal=FALSE)
plot(distr.vcg.asym[1001:1040],type="p",col="red",cex=1.2,ylab="number of cases",xlab="package indices")
lines(distr.gvcg.asym[1001:1040],type="p",pch=3,col="blue",cex=1.2)
lines(distr.vcg.mu.asym[1001:1040],type="p",pch=22,col="darkgoldenrod4",cex=1.6)
abline(v=c(seq(.5,40.5,by=10)),lty=1)
abline(v=c(seq(4.5,34.5,by=10)),col="blue",lty=2)
abline(v=c(seq(7.5,37.5,by=10)),col="red",lty=2)
abline(v=c(seq(9.5,39.5,by=10)),col="dark green",lty=2)
legend(-0.22,30.85, c("VCG","GVCG","VCG-mu"), cex=.8, bg="white",col=c("red","blue","darkgoldenrod4"),pch=c(1,3,22))
dev.off()

# figure: package distribution of every bidder:

postscript(file="sim03-04.eps",paper="special",width=8, height=6,horizontal=FALSE)
plot(distr.vcg.asym[1:1160],type="p",col="red",cex=.5,ylab="number of cases",xlab="package indices")
lines(distr.gvcg.asym[1:1160],type="p",pch=3,col="blue",cex=.5)
lines(distr.vcg.mu.asym[1:1160],type="p",pch=22,col="darkgoldenrod4",cex=.5)
abline(v=c(seq(.5,1160.5,by=40)),lty=1)
legend(-35,35., c("VCG","GVCG","VCG-mu"), cex=.8, bg="white",col=c("red","blue","darkgoldenrod4"),pch=c(1,3,22))
dev.off()

# average assignments of packages of different size:

distr.vcg.asym.tr <- matrix(distr.vcg.asym,ncol=10,byrow=TRUE)
distr.vcg.mu.asym.tr <- matrix(distr.vcg.mu.asym,ncol=10,byrow=TRUE)

# weak bidders:

sum(distr.vcg.asym.tr[1:100,1:4])/100    #size 1
sum(distr.vcg.asym.tr[1:100,5:7])/100    #size 2
sum(distr.vcg.asym.tr[1:100,8:9])/100    #size 3
sum(distr.vcg.asym.tr[1:100,10])/100     #size 4

sum(distr.vcg.mu.asym.tr[1:100,1:4])/100    #size 1
sum(distr.vcg.mu.asym.tr[1:100,5:7])/100    #size 2
sum(distr.vcg.mu.asym.tr[1:100,8:9])/100    #size 3
sum(distr.vcg.mu.asym.tr[1:100,10])/100     #size 4

# strong bidders:

sum(distr.vcg.asym.tr[101:116,1:4])/100    #size 1
sum(distr.vcg.asym.tr[101:116,5:7])/100    #size 2
sum(distr.vcg.asym.tr[101:116,8:9])/100    #size 3
sum(distr.vcg.asym.tr[101:116,10])/100     #size 4

sum(distr.vcg.mu.asym.tr[101:116,1:4])/100    #size 1
sum(distr.vcg.mu.asym.tr[101:116,5:7])/100    #size 2
sum(distr.vcg.mu.asym.tr[101:116,8:9])/100    #size 3
```

```r
sum(distr.vcg.mu.asym.tr[101:116,10])/100     #size 4

## asymmetric types, II (overlapping case) ################

# 25 bidders with i.i.d. uniform [0,1] item valuations
# N bidder with i.i.d. uniform [.5,2.5] item valuations

# optimal mu values:

opt.mus.2 <- rep(0,8)

mu.search.2 <- matrix(0,8,100)

mus.2 <- c(seq(.01,1,length=80),seq(1.05,2,length=20))

for (n in 1:8){
  for (i in 1:100){
    print(c(n,i))
    mu.search.2[n,i] <- sim.vcg.N.asym.mu(c(25,n),4,4,matrix(c(0,1,.5,2.5),2,2,byrow=
        TRUE),c(.2,.2),mu.vec=c(1,mus.2[i]),25)[1]
  }
  opt.mus.2[n] <- mus.2[which.max(mu.search.2[n,])]
}

# figure: weight-revenue function with one strong bidder:

postscript(file="sim04-02.eps",paper="special",width=8, height=6,horizontal=FALSE)
plot(mus,mu.search.2[1,],type="l",ylab="revenue",xlab="weight")
grid(col="grey")
dev.off()

# figure: weight-revenue function with five strong bidders:

postscript(file="sim04-03.eps",paper="special",width=8, height=6,horizontal=FALSE)
plot(mus,mu.search.2[5,],type="l",ylab="revenue",xlab="weight")
grid(col="grey")
dev.off()

# expected revenues:

revs.max.N.asym.2 <- rep(0,8)
revs.vcg.N.asym.2 <-rep(0,8)
revs.gvcg.N.asym.2 <-rep(0,8)
revs.vcgmu.N.asym.2 <-rep(0,8)

for (i in 1:8){
  print(i)
  revs.max.N.asym.2[i] <- sim.max.N.asym(c(25,i),4,4,matrix(c(0,1,.5,2.5),2,2,byrow=
      TRUE),c(.2,.2),100)[1]
  revs.vcg.N.asym.2[i] <- sim.vcg.N.asym(c(25,i),4,4,matrix(c(0,1,.5,2.5),2,2,byrow=
      TRUE),c(.2,.2),100)[1]
  revs.gvcg.N.asym.2[i] <- sim.gvcg.N.asym(c(25,i),4,4,matrix(c(0,1,.5,2.5),2,2,byrow
      =TRUE),matrix(c(0,0,.5,.5),2,2,byrow=TRUE),c(.2,.2),100)[1]
  revs.vcgmu.N.asym.2[i]  <- sim.vcg.N.asym.mu(c(25,i),4,4,matrix(c(0,1,.5,2.5),2,2,
      byrow=TRUE),c(.2,.2),mu.vec=c(1,opt.mus.2[i]),100)[1]
```

```
}

# figure: expected revenues:

postscript(file="sim04-01.eps",paper="special",width=8, height=6,horizontal=FALSE)
plot(revs.max.N.asym.2,type="o",ylim=c(0,40),ylab="revenue",xlab="number of strong
    bidders",lty=3,pch=4)
lines(revs.vcg.N.asym.2,col="red",type="o",lty=3,pch=1)
lines(revs.gvcg.N.asym.2,col="blue",type="o",lty=3,pch=3)
lines(revs.vcgmu.N.asym.2,col="darkgoldenrod4",type="o",lty=3,pch=22,cex=1.5)
grid(col="grey")
legend(.76,41.2, c("theoretical maximum","VCG","GVCG","VCG-mu"), bg="white", cex
    =0.85, col=c("black","red","blue","darkgoldenrod3"), lty=3, pch=c(4,1,3,22))
dev.off()

# package distributions:

vals.asym.2 <- matrix(0,100,29*40)
vbars.asym.2 <- matrix(0,100,29*40)
vcg.sol.asym.2 <- matrix(0,100,29*40)
gvcg.sol.asym.2 <- matrix(0,100,29*40)
vcg.mu.sol.asym.2 <- matrix(0,100,29*40)

for (i in 1:100){
  print(i)
  vals.asym.2[i,1:(40*25)] <- vals.unif.N(25,4,4,c(0,1),.2,B.mat(4))
  vbars.asym.2[i,1:(40*25)] <- vals.unif.N(25,4,4,c(0,0),.2,B.mat(4))
  vals.asym.2[i,(1+40*25):(40*29)] <- vals.unif.N(4,4,4,c(.5,2.5),.2,B.mat(4))
  vbars.asym.2[i,(1+40*25):(40*29)] <- vals.unif.N(4,4,4,c(.5,.5),.2,B.mat(4))
  vcg.sol.asym.2[i,] <- vcg(29,4,4,vals.asym.2[i,])$sol
  gvcg.sol.asym.2[i,] <- gvcg(29,4,4,vals.asym.2[i,],vbars.asym.2[i,])$sol
  vcg.mu.sol.asym.2[i,] <- vcg.mu(29,4,4,vals.asym.2[i,],c(rep(1,25),rep(opt.mus
      .2[4],4)))$sol
}

distr.vcg.asym.2 <- rep(1,100) %*% vcg.sol.asym.2
distr.gvcg.asym.2 <- rep(1,100) %*% gvcg.sol.asym.2
distr.vcg.mu.asym.2 <- rep(1,100) %*% vcg.mu.sol.asym.2

# average assignments of packages of different size:

distr.vcg.asym.tr.2 <- matrix(distr.vcg.asym.2,ncol=10,byrow=TRUE)
distr.vcg.mu.asym.tr.2 <- matrix(distr.vcg.mu.asym.2,ncol=10,byrow=TRUE)

# weak bidders:

sum(distr.vcg.asym.tr.2[1:100,1:4])/100    #size 1
sum(distr.vcg.asym.tr.2[1:100,5:7])/100    #size 2
sum(distr.vcg.asym.tr.2[1:100,8:9])/100    #size 3
sum(distr.vcg.asym.tr.2[1:100,10])/100     #size 4

sum(distr.vcg.mu.asym.tr.2[1:100,1:4])/100    #size 1
sum(distr.vcg.mu.asym.tr.2[1:100,5:7])/100    #size 2
sum(distr.vcg.mu.asym.tr.2[1:100,8:9])/100    #size 3
sum(distr.vcg.mu.asym.tr.2[1:100,10])/100     #size 4
```

```
# strong bidders:

sum(distr.vcg.asym.tr.2[101:116,1:4])/100    #size 1
sum(distr.vcg.asym.tr.2[101:116,5:7])/100    #size 2
sum(distr.vcg.asym.tr.2[101:116,8:9])/100    #size 3
sum(distr.vcg.asym.tr.2[101:116,10])/100     #size 4

sum(distr.vcg.mu.asym.tr.2[101:116,1:4])/100    #size 1
sum(distr.vcg.mu.asym.tr.2[101:116,5:7])/100    #size 2
sum(distr.vcg.mu.asym.tr.2[101:116,8:9])/100    #size 3
sum(distr.vcg.mu.asym.tr.2[101:116,10])/100     #size 4

############################################################
```

code.r