Elendner, Thomas; Femerling, R.

Working Paper — Digitized Version

Allocation of in-house services: Experimental comparison of allocation mechanisms

Manuskripte aus den Instituten für Betriebswirtschaftslehre der Universität Kiel, No. 577

Provided in Cooperation with:
Christian-Albrechts-University of Kiel, Institute of Business Administration

Manuskripte
aus den
Instituten für Betriebswirtschaftslehre
der Universität Kiel

No. 577

Allocation of In–house Services: Experimental Comparison of Allocation Mechanisms

T. Elendner[§]
R. Femerling[‡]

Abstract

In this paper we consider the problem of allocating scarce resources in a divisionalized company; the resources are made available by the headquarters and requested by profit centers (PCs). This problem was observed at a large german insurance company (LAGIC). Currently, the resources are allocated for free; for the future, LAGIC proposes to use an iterative allocation mechanism to solve the allocation problem. Alternatively, we present a combinatorial auction–based allocation mechanism. Within experimental tests the performance of these two mechanisms is analyzed. To this end, instances are generated motivated by the problem size and structure occurring at LAGIC.

Keywords: Resource allocation, in–house services, combinatorial auctions, efficient allocation, winner determination, multidimensional binary knapsack problem, instance generator

1 Introduction

In this paper we analyze an allocation problem arising at LAGIC. LAGIC is a divisionalized company with one service center providing $|K|$ types of resources that are demanded by $|I|$ divisions (or customers). Each division manager is responsible for the profit being made in his division, i.e., we have profit centers. Every profit center i has a set of jobs (or tasks) J_i, indexed j. A job causes costs amounting to $\gamma_{i,j} \geq 0$, requires $c_{i,j,k} \geq 0$ capacity units of resource $k \in K$, and creates revenue $r_{i,j} \geq 0$. The headquarters are not aware of the net profit $(r_{i,j} - \gamma_{i,j})$ for each job (private information).
We assume that the more resources a profit center gets the more profit it can achieve. In addition, the total demand for the resources $\sum_i \sum_j c_{i,j,k}$ exceeds supply A_k of resource k for at least one k, i.e., at least one service is scarce. This results in an allocation problem: how should the resources be distributed among the self–interested profit centers in order to achieve a firm-wide profit-maximizing job portfolio? For this we need an efficient allocation provided by an allocation mechanism which maximizes total net profit. Such a mechanism is applied to assign arbitrary items to entities. It consists of an allocation rule and incoming signals from the participants who demand the resources. The allocation rule determines, depending on the signals, who is awarded the resources, see, e.g., [18]. Signals, in this particular context, are parameters that indicate the net profit involved using the resources in the profit centers.
In [12] it is shown how to allocate resources when resource productivity differs between divisions; the productivity is assumed to be private information. The authors in [13] deal with the issue why auctions are used to allocate resources in certain environments and which type of auction is most efficient. For more literature about mechanisms for allocating resources in a divisionalized firm we refer to, e.g., [2], [14], and [31]. A nice literature survey on auction theory can be found in [17].
We analyze two different allocation mechanisms. The first, an iterative procedure proposed by LAGIC, is a mechanism based on priority rules. Alternatively, we present a combinatorial auction–based allocation mechanism. Within experimental studies these two algorithms are compared with respect to achieved overall net profit, computational time, and resource usage. Due to lack of real–world data, artificial instances for simulating the problem arising at LAGIC are generated. To get more experimental evidence, alternative problem sizes and structures are considered.

2 Priority Rule–based Allocation Mechanism

2.1 Basic Idea

The idea of this mechanism is to first serve those customers that are most "important" or "big", measured in terms of the cumulated costs

$$\Gamma_i = \sum_{j \in J_i} \gamma_{i,j}$$

for getting done all jobs within their profit center. The priority rule–based allocation mechanism (*PAM*) picks the participant i with the highest Γ_i who is first allowed to contribute to the overall job portfolio. He chooses the job $j \in J_i$ he prioritizes highest and is assigned the resources required to execute this job. Hence, the mechanism selects among the biggest customers the best jobs first and adds them to the job portfolio. Then Γ_i is reduced by the costs which will be necessary to realize the scheduled job, $\gamma_{i,j}$, and the capacities A_k are reduced by the resource usage of the job, $c_{i,j,k}$. Then, in the next iteration, *PAM* selects again the customer with the highest Γ_i and so on. The implementation details are presented below.

2.2 Implementation Details

In this section we first provide some assumptions about the behavior of the participants in order to automate *PAM*. Second, we present an algorithm that realizes these assumptions.

As already mentioned, it is assumed that each participant i knows his job costs $\gamma_{i,j}$. Furthermore we assume that he is aware of revenues $r_{i,j}$, $\forall j \in J_i$, which occur if the required resources are assigned to him; obviously, he makes a profit with a job if $r_{i,j} - \gamma_{i,j} > 0$. The revenues are presumed to be private information.

If at any stage of the process customer i has the highest cumulated costs, i.e.,

$$\Gamma_i = \max_{h \in I} \{\Gamma_h\},$$

he tries to get the required amount of resources for a single job j^*; we assume that job j^* maximizes his additional profit, that is,

$$j^* = \left\{ j \mid r_{i,j} - \gamma_{i,j} = \max_{h \in J_i} (r_{i,h} - \gamma_{i,h}) \right\}$$

with ties broken arbitrarily. If this job cannot be realized because there are not enough resources he tries to realize the second best job and so on.
Notice, that the customer does not act strategically, e.g., that he does not try to execute a job with low costs in order not to reduce his cumulated costs too much. This is a reasonable assumption since he cannot observe the behavior of the other participants.

Now we can define an algorithm since the actions of a participant are sufficiently determined. As a consequence, every participant can authorize a software agent, i.e., a piece of software that acts on his behalf.

The algorithm can be stated as given in Table 1.
The number of customers is reduced if no job of the customer with the currently highest cumulative costs can be scheduled. The procedure stops if there are no participants left and, thus, no schedulable job.

Algorithm 1 *(PAM)*

Input: I, J_i, $\gamma_{i,j}$, $r_{i,j}$, K, A_k
Output: ZF^{PAM}, $\mathbb{L}$;

$\bar{A}_k \leftarrow A_k$; *//set residual capacities*
$\Gamma_i = \sum\limits_{j \in J_i} \gamma_{i,j}$; *//set cumulative costs*
$\bar{\Gamma}_i \leftarrow \Gamma_i$; *//set residual cumulative costs*
$J_i' \leftarrow J_i$; *//set remaining jobs*
$I' \leftarrow I$; *//set active participants*
$j^* = 0$; *//job with highest priority*
$\mathbb{L} = \emptyset$; *//initial solution empty*
$ZF^{PAM} = 0$; *//current objective function value*

do{
$i^* = \left\{ i \mid \Gamma_i = \max\limits_{h \in I'} \{\Gamma_h\} \right\}$; *//choose bidder with highest cumulative cost*
$j^* = \left\{ j \mid r_{i^*,j} - \gamma_{i^*,j} = \max\limits_{h \in J'_{i^*}} (r_{i^*,h} - \gamma_{i^*,h}) \right\}$; *//choose job with highest priority*
if $(\bar{A}_k \geq c_{i^*,j^*,k} \; \forall k \in K)${ *//if job schedulable*
$\mathbb{L} = \mathbb{L} \cup \{j^*\}$; *//add job to solution*
$ZF^{PAM} = ZF^{PAM} + r_{i^*,j^*} - \gamma_{i^*,j^*}$; *//update objective function*
$J_i' = J_i' \setminus \{j^*\}$; *//update remaining jobs*
$\bar{\Gamma}_{i^*} = \bar{\Gamma}_{i^*} - \gamma_{i^*,j^*}$; *//update cumulative cost*
$\bar{A}_k = \bar{A}_k - c_{i^*,j^*,k} \; \forall k \in K$; *//update resource availability*
}
else { *//job not schedulable*
$J_i' = J_i' \setminus \{j^*\}$; *//update remaining jobs*
if $(J_i' = \emptyset)$ *//no job remains*
$I' \leftarrow I' \setminus \{i^*\}$; *//reduce participants*
}
if $(I' = \emptyset)${ *//no more participants*
return $\mathbb{L}$, ZF^{PAM}; *//return solution and objective function value*
stop;
}
}

Table 1: Algorithm *PAM*

3 Combinatorial Auctions as Mechanism

3.1 Basic Idea

A combinatorial auction is an auction where many items are sold simultaneously. Bidders are allowed to bid on arbitrary subsets of items. This is reasonable if there exist from the side of the participants nonadditive preferences for subsets of items. For example, we have superadditive preferences (complementarities) if a bundle of goods has a higher valuation than the sum of the valuations for the individual items. Complementarities can stem from economies of scale or economies of scope, for example. Whenever they occur it is not reasonable to employ multiple sequential single–item auctions, for instance, because then the bidders face a forecasting problem; they have to guess at an early stage of the auctions which items they will get later on. As a consequence, a participant has to estimate the private valuations of all other bidders. The forecasting problem can be illustrated using the data given in Table 2 with two items and two bidders.

	$\{A\}$	$\{B\}$	$\{A,B\}$
bidder 1	10	10	25
bidder 2	14	7	22

Table 2: Valuations of Two Bidders

The table says that bidder 1 values item $\{A\}$ at 10 currency units (cu), $\{B\}$ at 10 cu and receiving the bundle at 25 cu, for instance. Notice, that both bidders have superadditive preferences since the bundle is valued higher than the sum of the valuations for the single items. Assume that $\{A\}$ is auctioned off before $\{B\}$. To select the bidding price for $\{A\}$ bidder 1 has to forecast at what prize he could get $\{B\}$ in the later auction. If, for example, he can get item $\{A\}$ at 15 cu the bids for $\{B\}$ must not exceed 10 cu because, otherwise, he will be left with a loss.
Within a combinatorial auction bidder 1 can place bids on three subsets, on $\{A\}$, on $\{B\}$, and on $\{A,B\}$ with three bidding prices. Thus, combinatorial auctions enable the bidders to express their complementarities explicitly and the forecasting problem disappears.

Combinatorial auctions have been applied successfully in various ways. In [24] an application to airport time slot allocation is shown, where the bidders have complementarities about landing and starting slots for aircrafts. The authors in [4], [8], and [22] use the trucking services environment. Combinatorial auctions have also been considered for scheduling machines, cp. [7], [19], [30], disposing spectrum licenses ([21]), and purchasing airtime for advertising ([15]), to mention only a few.
The general winner determination problem is tackled in [1], [9], [26], and [27], for example. In [25] and [28] some special cases are presented that are solvable in polynomial time. A survey on combinatorial auctions is given in [6].

Next, we show how combinatorial auctions can be applied to our allocation problem.

3.2 Application

In our application the particular resources can be assumed to be the items that are auctioned off by the headquarters of LAGIC. Furthermore, each item comes along in multiple identical

copies (or units); therefore, we have the situation of a multi–item multi–unit allocation problem.
Every job requires a specific amount of resources in order to be executed. Note, that the jobs of a profit center are independent since it is possible for every PC to execute all jobs or none or anything in between. Therefore, the jobs can be interpreted to be the bidders. Subsequently, it is not decisive which PC provides which job. (This situation can change if the PCs are budget–restricted or if the bidding price is not the only decision criterion for assigning resources, cp. [3] and [18], for example.)

Furthermore, we have superadditive preferences whenever a job needs more than one unit of an arbitrary resource. To clarify that consider the example in Table 3.

	resource A	resource B	profit
job 1	5 units	2 units	25 cu
job 1	4 units	2 units	0 cu
job 1	5 units	1 units	0 cu

Table 3: Valuations of One Bidder

The first row states that job 1 needs 5 units of resource A and 2 of resource–type B to be realized; the incurring profit is 25. The next two rows imply that if too few units of resources are assigned to job 1 it cannot be executed and the profit is 0. (This situation changes if resources can be substituted by other resources. For example, the job could also be executed if it would get 4 units of A and 4 units of B. As a consequence, every job would place more than one bid.)

Each job is assumed to submit a bid that includes the amount of resources $c_{j,k}$ that have to be assigned to it in order to be processed and a bidding price b_j. Without loss of generality we assume that the bidding price is at most its incurring profit, i.e.,

$$b_j \leq p_j = r_j - \gamma_j, \ \forall j \in J. \tag{1}$$

Since the management's goal is to achieve an efficient allocation the social welfare has to be maximized. Presume for the moment that all bids have been submitted truthfully, i.e., $b_j = p_j$. Then, the efficient allocation can be determined solving a combinatorial optimization problem. The latter is frequently called the "Winner Determination Problem" and is formulated in the following.
Let the variable x_j equal 1, if the j–th bid is accepted and 0, otherwise. The objective is to maximize the social welfare. In our case, the social welfare can be stated as follows:

$$\sum_{j \in J} (r_j - \gamma_j)\, x_j = \sum_{j \in J} p_j\, x_j, \tag{2}$$

which adds up the profits p_j of the accepted jobs.
The winner determination problem can be stated as follows:

$$\max \quad \sum_{j \in J} b_j\, x_j \tag{3}$$

$$\text{s.t.} \quad \sum_{j \in J} c_{j,k}\, x_j \leq A_k \qquad \forall k \in K \tag{4}$$

$$x_j \in \{0, 1\} \qquad \forall j \in J \tag{5}$$

The objective function maximizes the sum of the bidding prices for the accepted bids. In the following, this value is denoted by ZF^{CA}. Restrictions (4) require that the claim of the accepted jobs on the resources does not exceed the respectively available capacity. (5) states the domain of the decision variables.

This problem is the well–known multidimensional binary knapsack–problem which is proven to be $\mathcal{NP}$–hard in the general case, cp. [16].
Various algorithms have been proposed in the literature. We refer to [10] and [23] and the references therein.

Recall that each bidder is assumed to bid truthfully. Note, that the efficient allocation is likely not to be achieved if the bidders do not tell the truth about their valuations; but the bidders can be incited to bid truthfully using the Vickrey–Clarke–Groves pricing scheme (cp. [29], [5], and [11]). Similar to the Vickrey payments it is also a second–price scheme. The payments are determined as follows: Consider that (3)–(5) was solved and let W denote the set of winners, that is, the profit centers which were assigned at least one unit of any resource. Furthermore, let β_w state the sum of the bidding prices of the accepted bids of winner w, i.e.,

$$\beta_w = \sum_{j \in J_w} b_j \, x_j, \; \forall w \in W.$$

Then, for every $w \in W$, (3)–(5) is solved again – now without all jobs belonging to PC w, obtaining $ZF^{CA}_{\setminus w}$. The difference $ZF^{CA} - ZF^{CA}_{\setminus w}$ can be interpreted to be the contribution of winner w to the auction outcome. Following the idea of Vickrey payments, w has to pay

$$\mathcal{P}_w = \beta_w - \left(ZF^{CA} - ZF^{CA}_{\setminus w} \right).$$

This implies that he has to pay his bidding price reduced by his contribution to the auction. This pricing rule has the properties to be incentive compatible and individually rational and it is the profit maximizing mechanism of all mechanisms that yield an efficient allocation, cp. [18]. A drawback is that the optimization problem has to be solved $|W|+1$–times in order to calculate all payments. Therefore it is essential that the winner determination problem can be solved in reasonable time. In Section 4 we show that this is the case for the relevant instance sizes for LAGIC using a standard software package.

3.3 Special Cases

Next, we state some special cases such that the winner determination problem can be solved efficiently. In particular, special structures of the claims on the capacities are analyzed.

Theorem 1 *The special case* $\left\{ \exists \, k' \mid c_{j,k'} = \sum_{k \in K} c_{j,k}, \; \forall j \in J \right\}$ *can be solved in polynomial time.*

Proof. In this case every job requests units of only a particular resource. Therefore, the jobs can be partitioned by the resource they request. Then, the units of a resource can be auctioned off simultaneously and independently from other resources. Hence, this is the special case of combinatorial auctions with identical objects where the winner determination problem was proven to be polynomially solvable, cp., e.g., [28]. □

Theorem 2 *The special case* $\left\{\sum_{k\in K} c_{j,k} = 1, \forall j \in J\right\}$ *can be solved in polynomial time.*

Proof. In this case every job has only a claim on one unit of a particular resource. Observe, that this is a special case of Theorem 1. □

Theorem 3 *The special case* $\left\{\exists\, k' \mid \sum_{j\in J} c_{j,k'} > A_{k'} \text{ and } \sum_{j\in J} c_{j,k} \leq A_k, \forall k \in K\backslash\{k'\}\right\}$ *can be solved in pseudo–polynomial time.*

Proof. In this case only one resource is scarce. Therefore, the winner determination problem reduces to a binary knapsack problem that was shown to be solvable in pseudo–polynomial time, cp., e.g., [20]. □

Theorem 4 *The special case* $\{c_{j,k} = \bar{c}_k, \forall\, k \in K, \forall\, j \in J\}$ *can be solved in polynomial time.*

Proof. In this case every job requires the same number of units for each resource, i.e., we have standardized jobs. Obviously, the winner determination problem can be reduced to the case of a binary knapsack problem only considering the most scarce resource k', where

$$k' = \left\{k \mid \frac{A_k}{\bar{c}_k} = \min_{h\in K} \frac{A_h}{\bar{c}_h}\right\}$$

with ties broken arbitrarily. Observe that this is a polynomially solvable special case of the binary knapsack problem; an optimal solution can be derived by accepting the $\left\lfloor \frac{A_{k'}}{\bar{c}_k} \right\rfloor$ highest bids. Obviously, these jobs can be determined in polynomial time. □

4 Experimental Comparison

In order to test the performance of the two presented allocation mechanisms experimentally we implemented *PAM* in GNU C. The winner determination problem is solved to optimality using the standard software package CPLEX 7.0.
The tests have been performed on an AMD Athlon with a 1.8 GHz processor and 768 MB of RAM running a Linux operating system.

4.1 Test–bed

Since we are not aware of real–world data sets but only know the basic data for LAGIC we develop an instance generator so as to reflect the allocation problem at LAGIC. Its allocation problem has about 5 customers, 10 resources, and 500 jobs. In order to yield more experimental evidence we derive several parameters. In particular, Table 4 shows which parameters may have an impact on the performance of the mechanisms.
Without loss of generality the value of all parameters are chosen to be integer–valued.

Parameter $I\#$ denotes the number of participants of the allocation procedure. We choose $I1 = 5$ and $I2 = 10$. Furthermore, the number of jobs is $J1 = 250$ or $J2 = 500$. The number of resources is chosen to be $K1 = 10$ or $K2 = 15$.

Notation	Meaning
$I\#$	number of customers, with $\# \in \{1,2\}$
$J\#$	number of jobs, with $\# \in \{1,2\}$
$K\#$	number of resources, with $\# \in \{1,2\}$
$R\#$	magnitudes of resources, with $\# \in \{1,2\}$
$M\#$	magnitudes of jobs, with $\# \in \{1,2\}$
C	costs of jobs
$P\#$	revenues, with $\# \in \{1,2,3\}$
$D\#$	distribution of jobs, with $\# \in \{1,2,3,4\}$

Table 4: Control Parameters for Instance Generation

The resource availability $R\#$ is also selected in two ways. For $R1$, each capacity constraint is uniformly drawn from the interval $[9000; 11000]$. $R2$ chooses each availability with probability of $\frac{1}{3}$ from intervals $[6000; 8000]$, $[8000; 10000]$, and $[10000; 12000]$.

The jobs are classified into three groups: large, medium, and small. For each large job the claim on the capacity was drawn uniformly from $[1\%; 5\%] \cdot A_k$, for each medium job from $[0.5\%; 1\%] \cdot A_k$, and for each small job from $[0.01\%; 0.5\%] \cdot A_k$. Thus, we get, e.g., for $R1$ the interval of $[90; 450]$ units for large jobs.

For $M1$, 2% $(18\%, 80\%)$ of the jobs are large (medium, small); for $M2$, 10% $(20\%, 70\%)$ are large (medium, small). For example, choosing $\{J1; M1\}$ yields 5 large, 45 medium sized, and 200 small jobs.

The costs for a job are assumed to be dependent on the sum of it's claim on capacities, which corresponds to parameter C. For each job a number is uniformly drawn from $[500; 750]$ and multiplied by $\sum_k c_{j,k}$. Hence, we do not check the influence of C

The profits are determined in three different ways. For each job size, a value is drawn uniformly from a specific interval. Table 5 summarizes the intervals for alternative job sizes and settings. We get the revenue by adding up the profit and the costs.

	large	medium	small
$P1$	$[25000; 50000]$	$[25000; 50000]$	$[25000; 50000]$
$P2$	$[15000; 20000]$	$[17500; 22500]$	$[20000; 25000]$
$P3$	$[20000; 25000]$	$[17500; 22500]$	$[15000; 20000]$

Table 5: Intervals for Profits

After having characterized and generated the jobs the instance generator "builds" the customers by distributing the jobs to them. We select four kinds of distributions.
$D1$ and $D3$ imply that the size of all customers is similar in terms of Γ_i. For $D1$, every participant approximately gets the same number of large, medium and small jobs.
For $D3$, $\rho_1 = \frac{\sum_j \gamma_j}{I\#}$ is calculated. Then, the largest jobs are assigned to the first customer until $\rho_1 \leq \sum_j \gamma_{1,j}$. Then, the rest of the large, medium, and small jobs are assigned uniformly to the other participants. Thus, we yield customers of equal size but with different "job–

structure".
$D2$ and $D4$ imply that the customers differ with respect to Γ_i. To be more precise, we assume that we have for $I1$ ($I2$) one (two) big, two (four) middle, and two (four) small customers in terms of Γ_i. Again, these two parameter settings differ in the "job–structure"; for $D2$ and $I1$, 40% of the large, the medium, and the small jobs are assigned to the first participant, 20% each to the second and third, and 10% each to the fourth and fifth. Consequently, we have one big, two medium–sized, and two small customers with the same job structure. For $I2 = 10$, the number of large, medium, and small customers are doubled.
$D4$ differs from $D2$ respecting the "job–structure"; for this, $\rho_2 = \sum_j \gamma_j$ is calculated. For $I1$, customer 1 gets the largest jobs until $0.4 \cdot \rho_2 \leq \sum_j \gamma_{1,j}$. Furthermore, two and three get the next largest jobs until $0.2 \cdot \rho_2 \leq \sum_j \gamma_{2,j}$ and $0.2 \cdot \rho_2 \leq \sum_j \gamma_{3,j}$, respectively. Then, the rest is assigned equally to the fourth and fifth participant.

We generated 5 instances for each possible parameter combination. Hence, in total we have

$$2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 3 \cdot 4 \cdot 5 = 1920$$

instances.

4.2 Results

In this section we study the experimental behavior of the two proposed allocation mechanisms. In order to analyze the impact of the parameter settings given in the instance generator we solved all instances for *PAM* and the combinatorial auction (*CA*).

For each setting we are interested in the following values:

- objective function
- runtime
- average resource usage

First, we give experimental evidence of how the objective function values for the two mechanisms and the chosen parameter settings behave.

For the objective function value we compared the values ZF^{PAM} and ZF^{CA}.
The value ZF^{CA} is a reasonable choice since a second–price auction is considered. Therefore, the participants are incited to bid truthfully. Consequently, (3) gives the social welfare of the allocation. Note, that for our considerations we do not need to calculate the second prices since these only state how the wealth derived in (3) is distributed among the PCs and the headquarters.

Notice, that *PAM* is a heuristic for the exact solution of (3)–(5) and, thus, yields a lower bound to *CA*. Hence, the following expression holds:

$$ZF^{CA} \geq ZF^{PAM} \tag{6}$$

As performance criterion we use the following formula:

$$\xi^{C|P} = \left(\frac{ZF^{CA}}{ZF^{PAM}} - 1 \right) \cdot 100\% \geq 0 \tag{7}$$

		R1				R2			
		$D1$	$D2$	$D3$	$D4$	$D1$	$D2$	$D3$	$D4$
$M1$	$P1$	0.56	0.27	1.30	1.06	0.00	1.97	1.40	1.32
		2.93	2.38	2.16	3.31	1.38	3.70	2.79	3.77
		7.19	3.93	2.92	8.38	3.13	5.90	3.85	6.66
	$P2$	0.00	0.09	0.04	1.16	0.01	0.01	0.05	1.20
		1.09	0.99	1.41	3.95	0.91	1.04	1.11	2.79
		2.04	2.09	2.85	7.21	2.10	1.98	1.72	4.63
	$P3$	0.88	1.24	3.01	0.00	3.05	0.00	0.00	0.88
		3.08	4.04	4.45	0.72	5.98	4.98	1.01	3.28
		5.14	7.99	6.38	2.00	9.52	8.11	2.28	5.31
$M2$	$P1$	27.09	36.11	25.07	80.97	33.53	29.46	26.67	71.27
		40.17	45.22	34.05	101.15	36.40	35.07	37.10	96.16
		47.02	52.66	47.60	121.90	40.19	37.63	44.13	114.78
	$P2$	2.73	9.45	4.99	88.33	1.82	6.59	4.64	80.91
		3.23	12.98	6.76	96.06	2.90	9.14	6.52	91.85
		4.08	18.11	7.67	108.23	4.02	11.96	9.18	112.88
	$P3$	247.44	129.01	145.26	144.63	221.24	145.64	147.18	129.81
		270.70	164.33	162.95	174.69	245.21	163.34	166.05	155.85
		303.85	193.72	178.42	239.37	265.75	183.82	186.76	184.23

Table 6: Performance for 5 Bidders, 10 Resources, and 250 Jobs

This can be interpreted as the average improvement of *CA* over *PAM* in percent. Observe that the values of expression (7) are always larger or equal to zero. The results are stated in Tables 6–13; each table is based on a combination of $[|I|; |K|; |J|]$ and the three entries for each parameter setting denote the minimal, average, and maximal value for $\xi^{C|P}$.
The following observations can be made:

- Auctions yield a significant higher result if we have a tendency towards large jobs. For $M2$ one can observe an average improvement of 192% and for $M1$ a value of 36% on average.

- Auctions tend to work better the more jobs we have. The improvement with 250 jobs is on average 52%, in the 500 jobs case 176%.

- With respect to the distribution of jobs (D) we can state that the $D4$ case yields an average improvement of 170% which is much more than in the other cases ($D1$: 105%, $D3$: 90%, $D2$: 88%).

- Auctions work better in a "normal" correlation of job size with job revenues, that is, if big jobs create high revenues ($P3$); for this an average improvement of 225% can be observed. If the size of a job is independent from its revenue ($P1$) the average improvement is 76%. In the case of a negative correlation ($P2$) we yield a value of 41%.

- The number of resources (K) has only little influence on the performance of auctions. On average, auctions tend to perform better with a growing number of resources. With 10 different resources ($K1$) the improvement was 113%, with 15 resources ($K2$) 116%.

- The number of customers (I) has no significant influence on the performance of auctions in relation to *PAM*. The 5 customers case ($I1$) yielded 112% improvement, the 10 customers case ($I2$) 117%.

		R1				R2			
		D1	D2	D3	D4	D1	D2	D3	D4
M1	P1	26.42	32.96	33.77	75.49	28.98	38.95	31.93	86.59
		36.99	43.88	38.12	86.01	36.53	43.52	39.60	89.90
		50.08	56.53	41.91	94.67	43.37	48.56	45.18	93.54
	P2	10.16	17.58	26.63	78.17	11.50	19.59	26.79	81.16
		13.28	21.75	30.39	85.50	13.94	22.30	30.18	86.73
		15.82	23.48	33.51	94.03	17.51	24.22	33.20	92.08
	P3	122.45	85.65	75.55	113.25	123.40	86.70	78.12	104.41
		135.37	102.18	85.20	122.21	138.42	95.53	81.72	124.98
		159.42	120.46	89.57	132.10	152.13	112.57	88.22	131.51
M2	P1	82.42	77.14	80.37	372.58	88.97	85.01	84.68	380.75
		93.77	101.02	102.83	417.53	101.93	103.05	96.61	450.14
		101.70	133.05	126.22	483.23	117.80	111.96	108.86	497.60
	P2	8.45	38.44	33.70	293.72	10.53	33.81	26.59	270.76
		10.44	40.15	35.54	308.86	12.01	40.92	30.18	310.14
		12.73	42.69	38.63	321.98	14.61	45.81	33.99	358.46
	P3	567.87	451.18	515.81	558.11	578.09	461.08	461.95	439.45
		641.21	477.79	548.05	594.64	606.80	511.73	523.88	604.53
		712.85	521.72	572.80	630.40	653.57	599.18	578.94	749.18

Table 7: Performance for 5 Bidders, 10 Resources, and 500 Jobs

		R1				R2			
		D1	D2	D3	D4	D1	D2	D3	D4
M1	P1	1.62	1.35	0.55	1.15	1.88	0.76	1.06	0.27
		2.87	3.48	3.19	3.11	3.07	1.99	2.84	2.49
		3.45	6.84	5.70	5.49	4.86	3.28	4.64	3.38
	P2	0.03	0.62	0.39	1.92	0.00	0.04	0.67	2.27
		0.29	1.67	1.19	2.90	0.63	0.64	1.11	4.31
		0.41	3.04	1.44	3.56	1.05	1.07	1.81	7.67
	P3	5.20	3.05	3.03	0.96	4.51	2.22	1.20	2.68
		8.06	5.78	5.19	4.76	7.15	4.99	3.08	4.16
		12.16	8.84	7.15	9.18	10.42	9.97	7.19	5.99
M2	P1	25.68	29.50	37.73	83.43	29.54	27.38	25.45	75.25
		36.76	47.87	47.06	96.98	43.30	47.37	39.86	105.63
		45.36	56.26	56.70	107.64	57.08	61.10	68.62	135.37
	P2	1.45	8.22	3.35	83.11	2.17	6.23	4.20	84.00
		3.58	13.11	4.75	99.15	3.28	12.12	6.08	101.31
		6.83	19.75	5.39	118.17	4.49	16.56	7.19	124.06
	P3	231.88	155.38	161.82	153.75	254.76	141.68	141.02	132.22
		262.65	168.32	186.51	183.19	284.24	178.26	166.44	176.02
		285.78	183.28	225.00	230.91	307.85	197.51	197.90	254.01

Table 8: Performance for 5 Bidders, 15 Resources, and 250 Jobs

		R1 D1	R1 D2	R1 D3	R1 D4	R2 D1	R2 D2	R2 D3	R2 D4
$M1$	$P1$	33.17	38.07	36.97	87.24	35.72	36.35	38.21	76.87
		41.81	46.83	41.55	92.49	44.15	43.52	40.06	94.05
		51.31	52.13	46.16	97.33	50.43	58.15	44.35	128.52
	$P2$	11.92	19.59	27.72	78.14	11.31	21.88	23.55	76.62
		13.38	21.69	30.35	84.77	13.46	24.36	30.48	80.72
		15.62	24.31	34.74	90.13	14.87	25.70	37.81	87.52
	$P3$	130.09	102.30	80.64	123.68	130.08	90.96	73.45	115.30
		141.37	106.46	86.36	132.40	145.29	99.25	83.77	127.02
		156.08	110.72	91.22	151.44	166.54	106.51	98.23	144.75
$M2$	$P1$	89.11	94.08	69.14	358.59	73.06	83.31	75.80	352.84
		111.58	107.95	86.28	429.63	108.09	105.18	99.31	399.51
		138.91	119.17	125.06	533.44	132.64	146.66	145.51	464.91
	$P2$	8.31	38.53	30.44	268.50	10.83	33.87	28.97	290.52
		10.93	41.93	33.51	296.45	11.13	39.87	32.00	312.15
		13.03	49.13	35.17	325.00	11.41	44.33	35.26	349.71
	$P3$	626.15	457.69	445.52	561.59	598.25	444.13	505.41	521.28
		676.69	506.28	530.29	613.13	656.41	508.51	545.69	594.63
		741.60	548.76	600.23	702.25	696.19	599.92	647.92	645.21

Table 9: Performance for 5 Bidders, 15 Resources, and 500 Jobs

		R1 D1	R1 D2	R1 D3	R1 D4	R2 D1	R2 D2	R2 D3	R2 D4
$M1$	$P1$	0.00	1.36	0.74	3.58	2.25	0.00	0.58	2.04
		2.44	4.41	2.77	5.23	3.48	3.22	2.61	3.09
		4.92	7.08	3.83	6.69	5.11	6.35	3.56	4.21
	$P2$	0.04	0.04	0.75	1.97	1.38	1.12	0.04	0.84
		2.14	0.89	1.40	4.47	1.57	1.89	1.51	4.49
		4.93	1.83	1.80	6.64	1.79	2.61	3.29	7.00
	$P3$	0.57	4.20	0.00	0.27	0.97	0.55	2.28	3.40
		4.51	6.28	2.31	4.79	3.67	4.26	4.00	4.95
		9.18	9.72	4.82	7.25	6.73	11.32	4.83	7.59
$M2$	$P1$	29.47	36.47	29.94	113.77	24.77	36.48	38.77	110.38
		48.19	41.80	40.40	133.94	39.83	43.20	44.75	124.44
		61.64	49.69	51.07	159.16	45.09	50.05	50.03	139.13
	$P2$	3.05	12.24	4.34	108.14	2.59	11.46	4.83	104.56
		4.48	14.69	5.00	116.78	3.85	15.29	5.59	114.48
		5.94	20.96	5.43	125.34	5.06	18.36	7.30	121.98
	$P3$	236.64	170.90	174.38	154.55	263.38	160.92	210.61	173.49
		288.11	184.78	215.65	180.07	275.98	180.86	233.21	190.44
		365.10	193.05	246.27	197.42	290.93	200.35	252.47	213.19

Table 10: Performance for 10 Bidders, 10 Resources, and 250 Jobs

		R1				R2			
		D1	D2	D3	D4	D1	D2	D3	D4
		26.54	41.88	42.19	74.79	36.16	40.60	39.24	78.39
	P1	40.38	45.17	44.95	89.36	42.70	48.44	44.69	89.55
		51.41	49.89	52.04	99.24	50.18	54.73	51.98	102.36
		9.50	20.49	22.37	78.39	9.85	19.54	19.79	79.50
M1	P2	13.83	22.63	26.96	86.25	12.36	21.53	24.70	86.14
		16.59	25.51	31.91	96.81	14.72	23.41	30.43	91.41
		134.27	94.11	91.58	104.60	108.58	100.69	87.36	110.34
	P3	141.89	106.24	103.09	122.18	130.37	113.44	96.95	120.86
		160.08	117.89	124.05	132.33	142.66	127.91	113.44	146.49
		89.04	89.84	97.79	357.39	75.31	92.55	77.91	322.30
	P1	99.67	108.39	107.63	435.85	104.64	104.74	99.02	374.80
		137.21	119.63	123.12	571.76	130.95	109.54	120.39	426.19
		9.44	37.31	19.70	272.45	9.58	37.42	20.98	287.08
M2	P2	11.30	40.80	22.91	296.06	11.10	40.15	22.61	307.48
		14.11	45.58	25.28	318.73	12.72	42.99	26.03	330.74
		585.08	452.94	475.55	577.12	511.66	423.52	432.50	549.92
	P3	646.80	475.50	538.01	616.40	605.76	498.45	559.03	660.95
		681.54	487.71	611.88	662.77	633.79	595.21	655.30	739.96

Table 11: Performance for 10 Bidders, 10 Resources, and 500 Jobs

		R1				R2			
		D1	D2	D3	D4	D1	D2	D3	D4
		2.41	0.32	1.90	0.71	1.17	3.92	2.28	1.40
	P1	3.19	3.16	3.89	3.14	3.46	4.48	3.17	4.08
		4.01	5.48	7.02	5.54	5.55	5.34	4.30	8.01
		0.78	1.09	1.02	4.01	0.00	1.45	0.39	1.96
M1	P2	2.18	2.25	1.52	5.32	0.74	2.04	1.40	4.97
		3.26	4.20	1.77	7.42	1.48	2.71	3.16	7.23
		2.28	1.97	4.10	4.06	2.30	1.61	1.56	2.36
	P3	4.48	3.98	5.53	6.10	6.68	3.80	3.53	5.95
		9.30	5.29	6.77	8.21	13.18	6.08	4.86	10.60
		30.84	47.11	35.30	117.65	37.89	38.91	36.09	107.28
	P1	43.18	53.08	49.38	150.51	52.60	50.53	45.88	123.93
		48.76	62.97	62.26	167.98	79.02	64.94	55.94	147.45
		2.30	12.84	3.04	99.72	2.11	13.86	0.73	96.56
M2	P2	3.26	17.06	4.39	114.29	3.20	15.43	4.39	108.00
		4.60	19.42	6.54	122.67	4.06	18.90	6.53	118.45
		284.50	166.90	190.19	166.21	234.42	150.25	206.45	180.24
	P3	306.25	177.79	231.19	194.61	280.22	173.42	219.39	192.96
		334.43	194.92	265.35	216.97	308.54	203.02	229.43	199.96

Table 12: Performance for 10 Bidders, 15 Resources, and 250 Jobs

		R1				R2			
		$D1$	$D2$	$D3$	$D4$	$D1$	$D2$	$D3$	$D4$
		31.23	37.58	41.36	86.87	43.84	38.77	34.16	86.17
	$P1$	44.32	43.51	44.16	95.67	46.13	49.61	41.82	90.83
		51.94	46.00	49.57	105.23	49.24	56.18	44.86	99.47
		11.79	17.74	22.87	86.11	11.43	19.33	20.61	75.99
$M1$	$P2$	13.87	20.79	27.87	88.09	14.08	22.62	25.26	86.14
		18.26	23.79	31.54	90.05	16.51	25.53	28.82	96.41
		116.43	110.31	93.04	124.91	122.81	96.63	95.47	122.26
	$P3$	128.83	115.69	107.05	135.26	136.13	103.55	105.37	127.74
		151.81	123.84	121.36	142.59	163.90	112.85	118.39	135.14
		94.96	110.61	75.01	363.93	65.21	97.15	105.53	373.71
	$P1$	108.66	119.15	93.84	430.96	114.09	115.85	115.62	430.89
		129.91	126.79	120.26	470.82	137.16	143.25	140.26	496.22
		10.53	36.35	20.00	288.70	9.28	37.97	23.81	292.23
$M2$	$P2$	12.14	43.41	24.50	319.44	10.67	47.00	25.09	309.57
		14.48	54.01	26.72	357.91	12.94	55.18	27.43	331.43
		539.84	470.87	476.65	571.70	507.26	445.24	508.99	550.10
	$P3$	572.24	510.03	568.36	674.64	592.55	476.96	575.49	614.41
		648.74	567.97	642.76	731.36	653.06	522.84	630.06	691.78

Table 13: Performance for 10 Bidders, 15 Resources, and 500 Jobs

- In the best case auctions yield a 749% higher objective function value, in the worst–case auctions do not improve the value of *PAM*.
- The resource availability has no influence on the performance of auctions in our setting. For $R1$ we have an improvement of 115%, for $R2$ of 113%.

Now we turn our interest to the runtime behavior of *CA*. Table 14 states the absolute numbers of instances that were solved to optimality for a certain instance size and runtime bound.

$[\lvert I\rvert;\lvert K\rvert;\lvert J\rvert]$	$< 1s$	$< 5s$	$< 10s$	$< 50s$	$< 100s$	$< 500s$	$< 1000s$	$< 5000s$	$< 10000s$
$[5;10;250]$	231	9							
$[5;10;500]$	186	26	8	15	3	0	1	1	
$[5;15;250]$	221	19							
$[5;15;500]$	149	56	9	9	4	5	4	2	2
$[10;10;250]$	238	2							
$[10;10;500]$	173	31	11	9	6	7	2	1	
$[10;15;250]$	223	17							
$[10;15;500]$	147	60	9	10	5	5	2	1	1

Table 14: Runtime Performance

This table states that, e.g., 231 instances for $[5;10;250]$ could be solved within 1 second (s). One can observe that each instance was solved to optimality since the sum of each row is 240 which is the sum of instances for each instance size $[\lvert I\rvert;\lvert K\rvert;\lvert J\rvert]$.
For $[\cdot;\cdot;250]$ we observed a very good performance; the average runtime is below 1 second with a maximal computational time of $4.7s$.
The instances $[\cdot;\cdot;500]$ needed – as one may expect – more time; for the case of $[5;10;500]$, one instance needed 1100 seconds to be solved, all others were solved within $1000s$. For $[10;10;500]$ one instance was solved in $1300s$ and all others within 16 minutes. For 15 resources the running times went up. For $[10;15;500]$ CPLEX needed for two instances more

than 1000s (6600s and 2400s) to solve them; each of the other instances were solved within 16 minutes. The case $[5; 15; 500]$ resulted in four instances that could not be solved within 1000s (9700s, 5200s, 3300s, and 1200s). All other instances are solved within at most 16 minutes.

PAM solved all instances within 0.01 seconds. Therefore, *PAM* outperforms *CA* with respect to running time.

Let us take a look at a possible worst–case running time for calculating Vickrey payments; to this end, assume that the running time has to be multiplied by $|I| + 1$ in order to obtain all payments. Then, our worst–case gives $11 \cdot 6600s$ which is about 20 hours. That is, within one day starting at the end of the auction the allocation and all payments can be determined in the worst–case assumed. This is still a reasonable time since the runtime is not critical in the suggested setting. Therefore, combinatorial auctions have to be preferred to *PAM* because of extreme differences in quality of the achieved allocation.

Finally, we analyze the resource usage resulting from the two algorithms. The knowledge of the resource usage can contribute to long–term capacity planning.
We consider two average resource usage indices $\zeta^{C|P}$ and ζ^{CA}.

Let

$$\bar{A}_k^\delta = \sum_{j \in J} c_{j,k}\, x_j^\delta, \quad \text{for } \delta \in \{CA, PAM\},\ \forall k \in K, \tag{8}$$

denote the capacity of resource k used, then,

$$\zeta^{C|P} = \frac{1}{|K|} \left(\sum_{k \in K} \frac{\bar{A}_k^{CA}}{\bar{A}_k^{PAM}} - 1 \right) \cdot 100\% \tag{9}$$

can be interpreted as average improvement of resource usage from *CA* to *PAM*, expressed in percent. The results for $\zeta^{C|P}$ are given in Table 15.

	[5;10;250]	[5;10;500]	[5;15;250]	[5;15;500]	[10;10;250]	[10;10;500]	[10;15;250]	[10;15;500]
min	-2.70	-4.48	-3.14	-1.10	-2.98	-2.51	-2.95	-0.56
avg	2.45	4.09	2.70	4.44	2.29	3.94	2.84	4.63
max	19.39	17.32	12.77	15.40	13.24	18.95	13.82	16.18

Table 15: Results for $\zeta^{C|P}$

We can state that *CA* has a better resource usage than *PAM* for each parameter setting $[|I|; |K|; |J|]$ on average since all average values are positive. The overall worst–case is about -4.5%. Furthermore, one can observe that $\zeta^{C|P}$ is on average significantly higher for $[\cdot; \cdot; 500]$ than for $[\cdot; \cdot; 250]$.

In order to give a statement of overall average resource usage, let

$$\zeta^{CA} = \frac{1}{|K|} \sum_{k \in K} \frac{\bar{A}_k^{CA}}{A_k} \cdot 100\%, \tag{10}$$

that is, the average resource usage for *CA* over all resources expressed as the sum of the fraction of resource usage and capacity limit A_k for each $k \in K$. Table 16 states the results observed for ζ^{CA}.

	[5;10;250]	[5;10;500]	[5;15;250]	[5;15;500]	[10;10;250]	[10;10;500]	[10;15;250]	[10;15;500]
min	90.27	92.70	91.02	95.42	88.97	94.20	91.57	94.49
avg	96.07	97.79	95.58	97.43	96.05	97.85	95.62	97.37
max	98.95	99.50	98.55	99.00	98.76	99.65	98.39	98.89

Table 16: Results for ζ^{CA}

In the worst–case, the value for ζ^{CA} was 88.97%; in all other cases, ζ^{CA} was above 90%. Furthermore, one can observe a similar behavior as that of $\zeta^{C|P}$; for instances $[\cdot;\cdot;500]$ we yield a significantly higher resource usage than for the cases $[\cdot;\cdot;250]$. On average *CA* had a resource usage of 96.7%.

5 Conclusions

In this paper we considered an allocation problem arising in practice. Two allocation mechanisms were analyzed experimentally. Opposite to *PAM* combinatorial auctions lead to an efficient allocation. We observed that auctions almost always outperform *PAM* respecting objective function value and resource usage. On the other hand, *PAM* requires less computational time. However, we showed that running times are not critical for the instances generated. In fact, we can determine the allocation and all payments within one day in the worst–case. Note, that *PAM* provides no payments for the resources awarded.

There are several shortcomings applying a combinatorial second–price auction. First, the net profit involved with each job has to be explicitly estimated. Second, for larger instances specialized algorithms for winner determination have to be employed. Furthermore – although endowed with nice theoretical properties – the acceptance among the participants may be affected by the complexity and lack of transparency of winner determination and the pricing scheme. Finally, we recommend to ensure that the private valuations are not made available to the headquarters and to the other profit centers. Therefore, a trustworthy third party should be employed as auctioneer.

Acknowledgement: We like to thank Frank Stege and his team for making us aware of this topic, Andreas Drexl and Thomas Andreßen for helpful comments, and Ingo Strunk, who took over parts of coding.

References

[1] A. Andersson, M. Tenhunen, and F. Ygge, *Integer Programming for Combinatorial Auction Winner Determination*, tech. report, Computer Science Department, Uppsala University, Uppsala, 2000.

[2] W. BAUMOL AND T. FABIAN, *Decomposition, Pricing for Decentralization and External Economics*, Management Science, 11 (1964), pp. 1–32.

[3] B. BURMEISTER, T. IHDE, T. KITTSTEINER, B. MOLDOVANU, AND J. NIKUTTA, *A Practical Approach to Multi–Attribute Auctions*, tech. report, DaimlerChrysler AG, 2002.

[4] C. G. CAPLICE, *An Optimization Based Bidding Process: A New Framework for Shipper-Carrier Relationships*, PhD thesis, Massachusetts Institute of Technology, 1996.

[5] E. CLARKE, *Multipart pricing of public goods*, Public Choice, 11 (1971), pp. 17–33.

[6] S. DE VRIES AND R. VOHRA, *Combinatorial Auctions: A Survey*, INFORMS Journal on Computing, 15 (2003). to appear.

[7] T. ELENDNER, *Scheduling and Combinatorial Auctions: Upper and Lower Bounds for the WJISP*, Manuskripte aus den Instituten für Betriebswirtschaftslehre 570, Universität zu Kiel, Kiel, Januar 2003.

[8] T. ELENDNER AND T. IHDE, *How Combinatorial Auctions work(ed) for DaimlerChrysler*, Manuskripte aus den Instituten für Betriebswirtschaftslehre, Universität zu Kiel, Kiel, 2003. in preparation.

[9] Y. FUJISHIMA, K. LEYTON-BROWN, AND Y. SHOHAM, *Taming the Computational Complexity of Combinatorial Auctions: Optimal and Approximate Approaches*, tech. report, Computer Science Department, Stanford University, Stanford, 1999.

[10] V. GABREL AND M. MINOUX, *A scheme for exact separation of extended cover inequalities and application to multidimensional knapsack problems*, Operations Research Letters, 30 (2002), pp. 252–264.

[11] T. GROVES, *Incentives in teams*, Econometrica, 41 (1973), pp. 617–631.

[12] M. HARRIS, C. H. KRIEBEL, AND A. RAVIV, *Asymmetric Information, Incentives and Intrafirm Resource Allocation*, Management Science, 28 (1982), pp. 604–620.

[13] M. HARRIS AND A. RAVIV, *Allocation Mechanisms and the Design of Auctions*, Econometrica, 49 (1981), pp. 1477–1499.

[14] P. JENNERGREN, *Studies in the Mathematical Theory of Decentralized Resource Allocations*, PhD thesis, Stanford University, 1971.

[15] J. L. JONES, *Incompletely Specified Combinatorial Auctions: An Alternative Allocation Mechanism for Business-to-Business Negotiations*, PhD thesis, University of Florida, 2000.

[16] R. M. KARP, *Reducibility among combinatorial problems*, in Complexity of Computer Computations, R. E. Miller and J. W. Thatcher, eds., Plenum Press, New York, 1972, pp. 85–103.

[17] P. KLEMPERER, *Auction Theory: A Guide to the Literature*, Journal of Economic Surveys, 13 (1999), pp. 227–286.

[18] V. KRISHNA, *Auction Theory*, Academic Press, San Diego, 2002.

[19] E. KUTANOGLU AND S. D. WU, *On Combinatorial Auction and Lagrangean Relaxation for Distributed Resource Scheduling*, IIE Transactions, 31 (1999), pp. 813–826.

[20] S. MARTELLO, D. PISINGER, AND P. TOTH, *Dynamic Programming and Strong Bounds for the 0–1 Knapsack Problem*, Management Science, 45 (1999), pp. 414–424.

[21] J. MCMILLAN, *Selling Spectrum Rights*, Journal of Economic Perspectives, 8 (1994), pp. 145–162.

[22] E. W. MOORE, J. M. WARMKE, AND L. R. GORBAN, *The Indispensable Role of Management Science in Centralizing Freight Operations at Reynolds Metals Company*, Interfaces, 21 (1991), pp. 107–129.

[23] M. A. OSORIO, F. GLOVER, AND P. HAMMER, *Cutting and Surrogate Constraint Analysis for Improved Multidimensional Knapsack Solutions*, Annals of Operations Research, 117 (2002), pp. 71–93.

[24] S. RASSENTI, V. SMITH, AND R. BULFIN, *A combinatorial auction mechanism for airport time slot allocation*, The Bell Journal of Economics, 13 (1982), pp. 402–417.

[25] M. H. ROTHKOPF, A. PEKEČ, AND R. M. HARSTAD, *Computationally Manageable Combinational Auctions*, Management Science, 44 (1998), pp. 1131–1147.

[26] T. SANDHOLM, *Algorithm for optimal winner determination in combinatorial auctions*, Artificial Intelligence, 135 (2002), pp. 1–54.

[27] T. SANDHOLM AND S. SURI, *Improved Algorithms for Optimal Winner Determination in Combinatorial Auctions and Generalizations*, in Proceedings of the National Conference on Artificial Intelligence (AAAI), 2000, pp. 90–97.

[28] S. VAN HOESEL AND R. MÜLLER, *Optimization in electronic markets: Examples in combinatorial auctions*, Netnomics, 3 (2001), pp. 23–33.

[29] W. VICKREY, *Counterspeculation, auctions, and competitive sealed tenders*, Journal of Finance, 16 (1961), pp. 8–37.

[30] M. P. WELLMAN, W. E. WALSH, P. R. WURMAN, AND J. K. MACKIE-MASON, *Auction Protocols for Decentralized Scheduling*, Games and Economic Behavior, 35 (2001), pp. 271–303.

[31] A. WHINSTON, *Price Guides in Decentralized Organizations*, in New Perspectives in Organization Research, W. W. Cooper, ed., 1964.