Robustness in Recurrent Auctions for Resource Allocation

Victor MUÑOZ and Dídac BUSQUETS
{vmunozs,busquets}@eia.udg.es
University of Girona

Abstract. Resource allocation problems where tasks have to be assigned to resources in such a way that no resource gets overused can be solved using recurrent auctions. In dynamic environments where unexpected changes may occur, searching the optimal solution may not be the best choice as it would be more likely to fail. In these cases a robust solution is preferable. In this paper we present a robustness mechanism for auctions, producing feasible and near optimal solutions even if non-planned events occur.

Keywords. Robustness, auctions, resource allocation

1. Introduction

Many real-world applications pose the problem of *resource allocation*, such as assigning memory and computing power to processes, distributing tasks to machines in a factory, or selecting the personnel to carry out certain jobs. This is an optimization problem where a set of resources is assigned to a set of agents (the entities needing the resources). The goal is to find a solution that maximizes or minimizes a given objective function (such as cost, revenue, makespan, etc.), while fulfilling a set of constraints (usually regarding the resources). However, finding the optimal solution is not always the best choice, since it could fail in case the environment changed (a machine breaking down, a process taking longer than expected, etc.). Therefore, it would be desirable to have a *robust solution* that could still be applicable even if unexpected events occurred. Obviously, the price of robustness is optimality [2], since usually a robust solution will be suboptimal. Therefore, there is a need of balancing the tradeoff between optimality and robustness.

In some scenarios where the resources can only be used temporally, the allocation process is continuously repeated. Moreover, in real environments (such as industrial ones), the access to resources is vital to perform tasks. Thus, it could be the case that an agent uses more resources than those it was allocated, or even use some resource without having been authorized to do so. The consequence of such behavior would be resource overuse, which would negatively affect the rest of the agents. One way of addressing this problem is by imposing fines or penalties to those agents not obeying the allocation [13]. However, even with that, an agent may still prefer to pay a penalty if it obtains a better profit by using the resources when not authorized. Therefore, in order to prevent such a situation, the allocation should incorporate some degree of robustness. Then, even if some agents disobeyed, the probability of having resource overuse would be lowered.

In this paper we focus on market-based mechanisms, namely auctions, for assigning resources to agents. Auction mechanisms have become a popular approach for dealing with resource allocation problems [4]. One of the advantages is that they provide privacy to the agents, since they do not need to disclose too much private information, and they also provide more autonomy in the decision-making, in comparison with purely centralized approaches. However, the problem of deciding the winners of an auction, known as the *Winner Determination Problem*, is a complex optimization problem, and most of the developed algorithms for solving it are focused on finding optimal solutions instead of robust ones.

Although robustness has already been studied in the field of mechanism design, it has been usually to tackle the problem of false-name bids [14] or to achieve mechanisms that are strategy-proof (that is, the agents' best strategy is truthful bidding) [5]. However, as mentioned before, we are dealing with robustness at the solution level, that is, solutions that are still valid even if the conditions for which they were computed change. This kind of robustness has been addressed in the planning and scheduling field [1,6,3], but, as far as we know, the only approach that deals with such robustness in auctions has been presented in [8]. This work uses the concept of super-solutions [7] to address the problem of bid withdrawal and generates solutions with a repair cost below a given threshold. However, it is not applicable to the disobedience problem presented above. Thus, we have focused our efforts in developing a mechanism for adding robustness to auctions for scenarios with potentially disobeying agents.

2. Auction-based Resource Allocation

Resource allocation problems can be solved using classical IA techniques, usually based on centralized approaches where a central element makes all the decisions. However, in recent years auctions are being increasingly used for these problems, as they are more indicated for distributed problems where the participants do not want to disclose private information related to their internal functioning upon which their decisions are based [4].

An auction-based distributed scenario for resource allocation is composed of a central element (coordinator) representing the shared resources of certain capacities, and a set of individuals (agents) competing for the resources. Agents that want to use the resources for a given period of time send requests to the coordinator composed by the resource/s that they want to use, and the required period of time. Formally, a request is defined as $\{s_i, d_i, \overline{q_i}\}$, where s_i and d_i are, respectively, the start time and duration of the use of the resources and $\overline{q_i}$ is the capacity requirements of the resources ($q_i = \{q_{i,1}, q_{i,2}, ..., q_{i,n}\}$ where n is the total number of resources). The resources are then assigned to the requesters using an auction. The goal of the auction is to avoid overuses of the resources by selecting a subset of the requests, which will gain the right to use some resources for a given time period, while the remaining should wait for another opportunity. The selection criteria is based on the bids submitted by the agents.

Formally, the problem to solve in an auction where there are multiple resources of different capacities is named the Winner Determination Problem (WDP) for multi-unit combinatorial auctions [9] (similar to the multi-dimensional knapsack problem):

$$\max \sum_{i=1}^{NR} x_i \cdot v_i \qquad \text{s.t.} \sum_{i=1}^{NR} x_i \cdot q_{i,j} \leq Q_j \quad \forall j \in C \tag{1}$$

where NR is the number of requests, $x_i \in \{0,1\}$ represents whether request i is denied or authorized, $v_i \in I\!R^+$ is the bid value for request i, $q_{i,j}$ is its capacity requirement for the resource j, Q_j is the resource j capacity, and C is the set of resources.

Note that in a resource allocation environment where the agents continuously need the use of the resources, the auction is repeated several times. Concretely, each time the requests overuse the resources an auction is executed to decide which of the requests to authorize. This scenario where the bidders are continuously competing for the resources is known as a *recurrent auction*, and gives rise to a new problem called the Bidder Drop Problem which happens when an agent participating in many auctions is always losing [11]. In such a case, the agent could decide to stop participating in the auction or stop obeying the outcome of the auction. This problem has been typically addressed using fairness mechanisms [10,12]. However, although fairness incentivizes agents to participate in auctions, it does not address the robustness of the system. Robustness is a desired feature in these situations, as it would produce solutions taking into account the agents which are most likely to disobey, thus preventing overuse of the resources.

3. Robustness in Auctions

As mentioned before, in some domains an interesting feature on auctions is to incorporate robustness representing the ability of a solution to overcome unexpected changes in the environment. In such situations we are willing to accept a suboptimal solution in order to ensure that it remains feasible and near optimal even when the environment changes. There are two general approaches for achieving robustness in uncertain environments:

- *Reactive robustness* addresses the problem of how to recover from a disruption once it has occurred, for example providing an alternative solution.
- *Proactive robustness* is concerned in finding a solution that takes into account the possible events in the environment and therefore, the solution is robust by itself.

While reactive robustness has been quite studied combinatorial auctions in the work of Alan Holland [8], proactive robustness is still relatively unexplored. In the following, we will design a proactive robustness mechanism for recurrent combinatorial auctions that considers nearly every possible change on the auction. The mechanism is based on a building a model of the participants in the auction that is learned in successive clearings of the auctions. The robustness mechanism consists in three main components:

- **Trust model** of the agents requesting the resources
- **Risk function** of the agent selling the resources (the *auctioneer*, or *coordinator*)
- **Robust solution generation**

The first component (the trust model) is concerned with the agents requesting resources. It is a main part of the mechanism as it models the behavior of the agents by learning from their actions their behavior and the circumstances in which an agent is most likely to disobey the decisions of the coordinator. The second component is related to the coordinator and its risk function, as the concept of a robust solution varies depending on the risk attitude of this concrete agent. Finally, with the inputs coming from all the agents, the robustness of the system is achieved by combining the risk of the coordinator with the trust on the agents requesting the resources to generate a solution that is robust, that is, it is able to absorb some level of changes in the environment.

3.1. Trust model

An agent requesting resources to perform tasks can disobey the decisions of the auctioneer for several reasons. It is not usually the case that an agent disobeys every decision of the auctioneer independently of the characteristics of the task to perform. Normally, an agent would disobey only the decisions that deny some tasks that it needs to perform for some reason. Therefore the trust model should not contain only a unique global value for the degree of trust of an agent, but the trust value should be related to a given task features. Possible task features to build the trust model with include the resources capacity requirements, the task duration, etc.

The trust model is learned during the recurrent auction storing not only the probability of disobeying of the agents, which happens when an agent uses the resource when it is not authorized to, but also its lying magnitude, representing the difference between the requested capacity of the resources and the real used capacity, as in some scenarios an agent may request to perform some tasks using a given capacity of resources and later use a higher capacity than requested. Consequently, the measures stored by the trust model are the following:

- **Probability of disobeying**. This value $\in [0..1]$ can be measured in many different ways, being the most obvious the average of disobediences in relation to the total number of auctions the agent has been involved in. However, it could be measured counting also the times where the agent has performed the authorized task but using a higher amount of capacity than requested.
- **Lie magnitude**. This value $\in [0..\infty]$ represents the degree of the disobedience. For example a value of 1 would represent that when the agent disobeys, it uses the quantity of resources requested for the task, while a value of 1.5 would represent that it uses 150% of the requested capacity.

A graphical representation of this trust model using only one characteristic of the task is shown in Figure 1 (to use more task characteristics, additional dimensions would be added). Note that this model is general enough to allow including even the case where an industry does never disobey the auctioneer, but it uses a higher amount of capacity than requested (having a lie magnitude greater than 1 at disobey probability of 0). This is particularly useful in problems where the resource capacity requirements of the agents are quite dynamic.

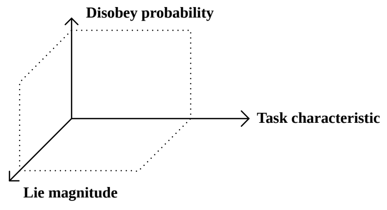

Figure 1. Trust model.

The trust model is learned by the auctioneer agent at execution time. Every time a task is performed the trust model of the respective agent is updated checking firstly if the task has been performed after the authorization of the auctioneer or not, that is, the agent has disobeyed the result of the coordination (the solution of the auction), and secondly if the resource capacity used is the same as what was requested.

3.2. *Risk function*

The risk attitude of the auctioneer characterizes the tradeoff between robustness and optimality that it wants, given that robustness and optimality are contradictory objectives. The risk function of the coordinator can be also seen as its willingness to face dangerous situations.

Risk attitudes are generally categorized in three distinct classes: risk averse, neutral and proclive. Risk aversion is a conservative attitude for individuals who do not want to be at stake. Risk neutral agents display an objective predilection for risk, whilst agents with a proclivity for risk are willing to engage in situations with a low probability of success. For example, a risk-averse auctioneer would consider that every request with a probability of disobeying greater than 0 is going to use the resources even if unauthorized, and thus it would auction only the remaining resources capacities over the rest of the requests. On the other hand a risk-proclive auctioneer would consider that if a request has a low probability of being disobeyed, it would not be the case at this time and hence the auctioneer would auction a bigger amount of resources capacities, although with a higher risk of being overused.

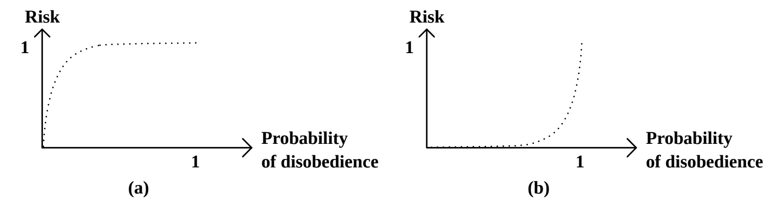

Figure 2. Risk attitude function: (a) averse, (b) proclive.

The risk function f_{risk} defines the risk attitude of the auctioneer (between 0 and 1) as a function of the probability of disobeying of a given agent and a given request. An example of a risk function is shown in Figure 2(a). In this case it represents a risk-averse auctioneer, since the resulting *risk value* is almost always 1 (it considers risky requests as if they are going to surely use the resources even if unauthorized), regardless of the probability of disobeying. On the other hand, a risk-proclive auctioneer would have the final value almost always set to 0, as seen in Figure 2(b), and a risk-neutral one would have it set accordingly to the probability of disobeying.

3.3. *Robust solution generation*

The trust model and the risk function of the coordinator are used to generate the *robustness constraint* that will provide robust solutions according to the proactive approach. This constraint is added to the constraint optimization problem (previously formulated in Equation 1) related to the auction, in order to force the solution to be robust.

In the auction (executed each time a conflict is detected) the auctioneer is faced with a set of requests (the tasks involved in the conflict), each with trust features associated obtained from the trust model. Then the auctioneer decides which requests to authorize in function of its risk attitude.

The robustness constraint is formulated in a way that the solution finds a balance between the amount of resources required by the authorized requests and the assumed risk from the unauthorized requests (appropriately weighted by its probability of disobeying, lie magnitude and the risk function f_{risk} of the auctioneer). The objective is not to exceed the maximum capacities of the resources (Q_j). This constraint is defined as shown in Equation 2.

$$\sum_{i\in[1,n]} x_i{\cdot}c_i + \sum_{i\in[1,n]} (1-x_i){\cdot}c_i{\cdot}f_{risk}(P_i){\cdot}M_i \leq Q_j \quad \forall j \in C \tag{2}$$

The first summatory represents the resources used by the authorized requests, while the second summatory characterizes the resources potentially used by the unauthorized requests. Hence, the unauthorized requests are considered as if they were performed in the cases where the probability of disobeying of the associated agent (P_i) is higher than zero. However this value (appropriately weighted with its corresponding lie magnitude M_i) is considered as a function of the risk attitude of the auctioneer f_{risk}. In this case we have considered that the lie magnitude is directly multiplied by the *risk value*, but another function could be used as well.

Another way of understanding this equation is by moving the second summatory to the right side. Then it can be read as if a concrete capacity of the resource/s is reserved to be used by the unauthorized tasks that are likely to be disobeyed and performed anyway.

4. Experimentation

To test the robustness mechanism previously described, we have used a real-world problem: the Waste Water Treatment Plant Problem (WWTPP). The main components in this problem are the Waste Water Treatment Plant and the set of industries performing waste discharges to the sewage. The job of the treatment plant is to process the sewage coming from the industries, removing its contaminants in order to return a clean waterstream back to the river. If the discharges are done without any coordination, the amount of water arriving at the plant may exceed its capacity, which causes the overflow to go directly to the river without being treated and increasing its contamination. Thus, in order to prevent such dangerous environmental situation, the industrial discharges should be temporally distributed so that all of them can be fully treated.

We assume that each industry has a retention tank (of a given capacity) where it can store a discharge whenever it is not authorized, and empty it later on. In this scenario the recurrent auction mechanism will determine which discharges to authorize and which to be temporarily stored in the tank in order to not exceed the plant's capacity.

Regarding the robustness mechanism proposed in this paper, it is easier to understand more clearly with this concrete problem why it is useful. In this scenario it is conceivable that industries may sometimes disobey the decisions of the plant. The most obvious reason is when an industry has its retention tank completely full; in this case if the forthcoming discharge is not authorized, the industry will be forced to discharge it anyway, thus disobeying the plant. However, an industry could disobey the decisions of the plant for other uncontrolled and unpredictable reasons, for example when an industry cannot use its retention tank (for maintenance purposes, for instance), or when a concrete discharge cannot be stored in the tank because of its high level of contamination, etc. That is the reason why the robustness mechanism has been designed considering the characteristics of the task in the trust model.

4.1. Solving the WWTPP

The WWTPP can be modeled as a recurrent combinatorial auction, where the auctioneer is the treatment plant, the resource being sold is its capacity, and the agents using the resource are the industries that perform discharges. Here the resource consumption (as well as the individual discharges) does not have only a global capacity limit (hydraulic capacity), but it is extended with many thresholds, one for each contaminant type. The goal of the auctioneer is not to exceed any of its thresholds (hydraulic capacity and contaminant levels).

The coordinating scenario described in the previous sections can be easily adapted to be applied to this problem, so the robustness mechanism can also be used.

4.2. Implementation

To evaluate the coordination and robustness mechanisms we have implemented a prototype of the system reproducing the coordination and communication process between plant and industries. So far we have only considered the hydraulic capacity of the plant. Industry agents calculate their bids taking into account the urgency to perform a discharge, based on the percentage of occupation of the tank. In case an industry agent is denied to perform one of its discharges, it first tries to store the rejected discharge into the tank, scheduling the discharge of the tank as its first activity after the current conflict finishes. If the industry has its tank already full, the discharge is performed anyway.

The free linear programming kit GLPK (GNU Linear Programming Kit) has been used to solve the winner determination problem related to each (multi-unit) auction, modeling it as a mixed integer programming problem. The robustness constraint is added as an additional constraint.

The trust models of the industries have been implemented using only one characteristic of the discharges: the flow. The models of the industries are learned during the execution by storing the total number of lies and truths (that is, disobedient and obedient actions), together with a value to compute the lie magnitude. These values are updated after each performed discharge in the following way: if the industry was authorized then the number of truths of the corresponding flow is incremented; otherwise the number of lies is incremented. Independently, the lie magnitude is computed as the difference between the used capacity and the requested capacity.

4.3. Experimentation results

Results have been evaluated considering some quality measures based on different characteristics of the solution:

- **number of overflows (NO)** occurred during the simulation
- **maximum flow overflowed (MFO)**, measured in m^3/day
- **total volume overflowed (VO)**, in liters
- percentage of discharge denials **obeyed** by the industries **(%IO)**

The experiments consisted of simulations using a set of real data provided by the Laboratory of Chemical and Environmental Engineering (LEQUIA). This data is composed of the discharges of 5 industries in two weeks. The first one is a pharmaceutical industry; it is increasing its discharge flow during the week and does not discharge during the weekend. The second one is a slaughterhouse that discharges a constant flow, except at the end of the day when it increases. The third one is a paper industry that discharges a constant flow during the seven days of the week. The fourth one is a textile industry, whose discharges flow oscillates during the day. The fifth one is the waste water coming from the city, whose flow is fixed. The hydraulic capacity of the plant is 32000 m^3/day.

We have tested the mechanism in different scenarios and situations. In the first scenario there is no coordination among the industries (without coordination the industries perform its initial discharges plans, and the treatment plant does never unauthorise any discharge). The second scenario uses the coordination mechanism and assumes that the industries always obey the decisions of the plant, as long as they have enough tank capacity. In the third scenario we introduce a probability of disobeying the outcome of the coordination mechanism. This probability depends on the occupation of the tank (the higher the occupation, the higher the chances of disobeying); a graphical representation of this function is shown in Figure 3. Two variations of the disobeying probability of disobeying have been tested.

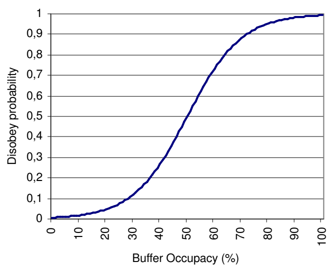

Figure 3. Disobey probability function.

Additionally, we have tested the system with another scenario where there is one single industry (the textile, chosen randomly), that will always disobey the decisions of the plant if any of its discharges is unauthorized. Every scenario has been tested activating and deactivating the robustness mechanism and with different risk attitudes of the coordinator (averse, neutral and proclive).

The outcome of all the scenarios is shown in Table 1, with the average and deviation (in brackets) of 10 simulations performed for each scenario. Concretely, we can notice that the non-coordinating scenario produces the worst results regarding volume overflowed (which is the most important indicator), while the auction-based system improves the results, principally when all the industries obey (this reflects the best possible circumstances). With disobeying industries we can notice a subtle improvement when using the robustness mechanism in both the volume and maximum flow overflowed yet the difference is not much relevant, and the number of overflows is generally higher. Regarding the risk attitude of the coordinator we do not observe its effects in this scenario.

In the environment where there is one industry always disobeying, the robustness mechanism seems to mark differences given that all the indicators are significantly improved, specially regarding the volume overflowed and percentage of obedience. However, in this scenario, like in the previous, the different risk attitudes do not produce clear differences in the outcome.

			NO	**MFO**	**VO**	**%IO**
No coordination			80	9826	$15.21 \cdot 10^6$	-
Obey			28	4996	$3.74 \cdot 10^6$	98.95
Low Disobedience	No Robustness		77.60 (4.12)	14432 (865.93)	$11.5 \cdot 10^6$ (216866)	98.55 (0.12)
	Robustness	Averse	78.70 (7.15)	14360 (1522)	$11.3 \cdot 10^6$ (261362)	98.27 (1.57)
		Neutral	79 (7.83)	13531 (1396)	$11.4 \cdot 10^6$ (260669)	98.19 (0.24)
		Proclive	84.1 (5.16)	14052 (1006)	$11.3 \cdot 10^6$ (251712)	98.15 (0.17)
Medium Disobedience	No Robustness		126.60 (6.13)	14398 (1604)	$13.3 \cdot 10^6$ (363484)	96.48 (0.31)
	Robustness	Averse	126.60 (6.13)	14398 (1604)	$13.3 \cdot 10^6$ (363484)	96.48 (0.31)
		Neutral	122.9 (6.84)	13966 (803)	$13.2 \cdot 10^6$ (403934)	96.61 (0.32)
		Proclive	121.3 (7.94)	14233 (1358)	$13.2 \cdot 10^6$ (374673)	96.58 (0.41)
TEXTILE INDUSTRY ALWAYS DISOBEYING						
No coordination			80	9826	$15.21 \cdot 10^6$	-
Obey	No Robustness		112	6523	$6.89 \cdot 10^6$	90.84
	Robustness		58	6590	$5.47 \cdot 10^6$	96.77
Low Disobedience	No Robustness		112 (6.09)	14955 (1201.58)	$12.6 \cdot 10^6$ (233076)	90.98 (0.2)
	Robustness	Averse	77.70 (3.68)	14225 (1212)	$11.8 \cdot 10^6$ (205150)	96.69 (1.57)
		Neutral	82.5 (7.66)	15110 (997)	$11.9 \cdot 10^6$ (199074)	96.66 (0.16)
		Proclive	81.2 (4.44)	14018 (1596)	$11.8 \cdot 10^6$ (133988)	96.68 (0.18)
Medium Disobedience	No Robustness		119.70 (4.72)	14819 (1373.74)	$14.3 \cdot 10^6$ (263955)	89.96 (0.28)
	Robustness	Averse	109.50 (3.95)	14150 (1310)	$13.6 \cdot 10^6$ (242619)	95.19 (0.17)
		Neutral	113.5 (5.5)	13708 (1040)	$13.6 \cdot 10^6$ (445501)	95.16 (0.37)
		Proclive	110.9 (8.16)	14522 (1571)	$13.6 \cdot 10^6$ (338985)	95.31 (0.29)

Table 1. Simulation results.

5. Conclusions and Future Work

In this paper we have presented a proactive robustness mechanism for auction-based resource allocation problems. Through this mechanism, the system finds a solution that is robust, i.e. it is able to remain applicable even with changes in the environment. Changes involve both modifications on the resources capacities requests and using the resource when the user is not authorized to. The core of the robustness mechanism consists in a trust model that is learned during the execution and a risk function associated to the auctioneer of the resources, that are used together in order to produce a robust allocation.

Results obtained through simulation using real data show that the robustness mechanism improves the results over the non-robust approach. However, further work has to be made in the risk attitudes of the auctioneer as we have not noticed significant changes when varying it. Also the trust model needs to be improved, as it considers tasks with different characteristics independently, yet in problems where the tasks characteristics were too dynamic it would be useless as there would not be two identical tasks.

It should be noted that the robustness mechanism may induce the agents to disobey. Different mechanisms to avoid this situation have already been studied, as for example the addition of *fines* (or *penalties*) to be paid whenever an agent does not obey; another method would be to stipulate a deposit to be paid for the participants before beginning the coordination, and returned later only to the obeying agents. However, the price of these fines or deposits should be studied in more detail in order to make it not too cheap so an agent would prefer to pay it instead of obeying the coordinator, neither too expensive so that a poor agent would have more problems than a rich one to pay it.

References

[1] A. Ben-Tal and A. Nemirovski,'Robust solutions of uncertain linear programs', *Operations Research Letters*, **25**, 1–13, (1999).

[2] D. Bertsimas and M. Sim, 'The price of robustness', *Operations Research*, **52**(1), 35–53, (2004).

[3] J. Branke and D.C. Mattfeld, 'Anticipation and flexibility in dynamic scheduling', *International Journal of Production Research*, **43**(15), 3103–3129, (2005).

[4] Y. Chevaleyre, P.E. Dunne, U. Endriss, J. Lang, M. Lemaître, N. Maudet, J. Padget, S. Phelps, J.A. Rodríguez, and P. Sousa, 'Issues in multiagent resource allocation', *Informatica*, **30**, 3–31, (2006).

[5] R.K. Dash, P. Vytelingum, A. Rogers, E.David and N.R. Jennings, 'Market-Based Task Allocation Mechanisms for Limited-Capacity Suppliers',*IEEE Transactions on Systems, MAN, and Cybernetics-Part A: Systems and Humans*, **37**(3), 391–405, (2007).

[6] A.J. Davenport, C. Gefflot, and J.C. Beck, 'Slack-based techniques for robust schedules', in *Proceedings of the Sixth European Conference on Planning (ECP-2001)*, pp. 7–18, (2001).

[7] E. Hebrard, B. Hnich, and T. Walsh, 'Super solutions in constraint programming', in *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, Lecture Notes in Computer Science, 157–172, Springer, (2004).

[8] A. Holland and B. O'Sullivan, 'Truthful risk-managed combinatorial auctions', in *Proceedings of IJ-CAI'07*, pp. 1315–1320, (2007).

[9] J. Kalagnanam and D. Parkes, *Handbook of Supply Chain Analysis in the E-Business Era*, chapter Auctions, bidding, and exchange design, Kluwer Academic Publishers, (2005).

[10] J.S. Lee and B.K. Szymanki, 'A novel auction mechanism for selling time-sensitive e-services', *Proc. 7th International IEEE Conference on E-Commerce Technology (CEC'05), pp. 75-82.*, (2005).

[11] J.S. Lee and B.K. Szymanki, 'Auctions as a Dynamic Pricing Mechanism for e-Services', *Service Enterprise Integration, pp. 131-156.*, (2006).

[12] V. Muñoz, J. Murillo, D. Busquets, and B. López, 'Improving water quality by coordinating industries schedules and treatment plants', in *Proceedings of the Workshop on Coordinating Agents' Plans and Schedules (CAPS)*, ed., Mathijs Michiel de Weerdt, pp. 1–8. IFAAMAS, (2007).

[13] T. Sandholm and V. Lesser, 'Leveled commitment contracting: A backtracking instrument for multiagent systems', *AI Magazine*, **23**(3), 89–100, (2002).

[14] M. Yokoo, Y. Sakurai and S. Matsuraba, 'Robust Combinatorial Auction Protocol against False-name Bids', *Artificial Intelligence Journal*, **130**(2), 167–181, (2001).