A Market-Based Approach to Multi-factory Scheduling

Perukrishnen Vytelingum[1], Alex Rogers[1], Douglas K. Macbeth[2], Partha Dutta[3], Armin Stranjak[3], and Nicholas R. Jennings[1]

[1] School of Electronics and Computer Science, University of Southampton, UK
{pv,acr,nrj}@ecs.soton.ac.uk
[2] School of Management, University of Southampton, UK
D.K.Macbeth@soton.ac.uk
[3] Strategic Research Centre, Rolls Royce, UK
{Partha.Dutta,Armin.Stranjak}@Rolls-Royce.com

Abstract. In this paper, we report on the design of a novel market-based approach for decentralised scheduling across multiple factories. Specifically, because of the limitations of scheduling in a centralised manner – which requires a center to have complete and perfect information for optimality and the truthful revelation of potentially commercially private preferences to that center – we advocate an informationally decentralised approach that is both agile and dynamic. In particular, this work adopts a market-based approach for decentralised scheduling by considering the different stakeholders representing different factories as self-interested, profit-motivated economic agents that trade resources for the scheduling of jobs. The overall schedule of these jobs is then an emergent behaviour of the strategic interaction of these trading agents bidding for resources in a market based on limited information and their own preferences. Using a simple (zero-intelligence) bidding strategy, we empirically demonstrate that our market-based approach achieves a lower bound efficiency of 84%. This represents a trade-off between a reasonable level of efficiency (compared to a centralised approach) and the desirable benefits of a decentralised solution.

1 Introduction

Job-shop scheduling is an important and challenging problem that has long been the subject of extensive research in Artificial Intelligence [2]. Specifically, it is the problem of allocating resources for the completion of customers' jobs within specific deadlines in a single or a number of factories. Such a problem is usually solved optimally as a combinatorial optimisation problem (that maximises the utility of all stakeholders, i.e. both customers and factories) by a center that has complete information of the system. However, this centralised approach can be problematic in a number of ways. First, a completely new solution often needs to be recomputed from scratch with every change in the system (e.g. new stakeholders entering the system or existing ones updating their job requirements). Second, stakeholders, which usually represent competing organisations in the real world, are often reluctant to share their private and sensitive information with a center that would then have complete knowledge of the whole system, since this may shade their competitive edge. Finally, and perhaps most importantly, the solution

S. Das et al. (Eds.): Amma 2009, LNICST 14, pp. 74–86, 2009.

quickly becomes intractable with increasing problem size, because of the combinatorial nature of the task at hand.

For these reasons, there is an increasing trend towards solving these scheduling problems in a decentralised manner. Such approaches are inherently more robust because they don't have a single point of failure and they also do not require the divulging of commercially valuable information to a third party. Moreover, the distributed nature of the computation also means a more scaleable solution is possible, with the system being more dynamic in adapting to changes in the system.

Against this background, this paper reports on the development of a market-based approach for scheduling across multiple factories. Our work is principally motivated by the need for more robust and scaleable solutions in the Aero Repair and Overhaul (AR&O) context where customers (typically airlines) in the system require that their engines be scheduled for routine maintenance in *overhaul bases* (OHBs). In addition, engines may require more urgent inspection due to some damage (e.g. through bird-strike or icing) or mechanical failure. These disruptions introduce uncertainties in the system which increase the complexity of the scheduling process since it requires us to dynamically construct schedules capable of effectively coping with such unforeseen events in real time. The challenge is to dynamically schedule repairs and routine maintenance subject to these constraints. To date, the state of the art consists of a pragmatic scheduling solution [8] that consists of a center that computes the engine repair schedules based on the capacity and capability of the multiple factories and the engine repair severity. However, this is a centralised solution that suffers from all the aforementioned problems.

In more detail, the AR&O scheduling problem, as described above, is an important class of problems in its own right (being worth over $25B worldwide in 2008). It also has characteristics that are found in many other applications as diverse as classic job-shop scheduling, production planning and manufacturing scheduling [2,6]. Thus, in addition to its immediate goals of addressing the AR&O problem, this research endeavour has further applications within many adaptive decision processes where there is an extended and dynamic network of interdependent customer and supplier business entities. As such, it has the potential to provide utility to many other supply network settings where again traditional approaches have tended to a static representation unable to respond quickly to the rate of change in the environment.

To address the challenge of developing an agile and decentralised scheduling system, a number of researchers have advocated the use of economic metaphors by adopting a market-based approach where customers and factories are self-interested agents[1] that compete, through offers to buy and sell resources, in order to maximise their utility. Markets are a particularly suitable approach in this context, because of their ability to facilitate resource allocation often with public information exchange and their distributed nature with the resource allocation emergent from the competition among buyers and sellers. Because of their distributed nature, no single agent computes the resource allocation, markets are robust against failure and, furthermore, markets have been shown to dynamically and efficiently react to changes in the system [3]. Now, there has been

[1] Self-interested agents are reluctant to share private information and are driven by the objective of maximising their own profit.

previous work that adopted a market-based approach in this domain. Specifically, Baker's work looked at a market-driven contract net mechanism to schedule a factory [1], and Rassenti *et al.* developed a sealed-bid combinatorial auction mechanism for scheduling flight landing and take-off [7]. However, while these mechanisms reduce the complexity of the particular problems they are tackling, they still need a center that collects all offers (which are not made public) to match them for scheduling. In contrast, Wellman *et al.* examine a number of auction mechanisms for decentralised scheduling [12], including multiple simultaneous ascending auctions. Here, while the latter approach circumvents the need for an auctioneer (as all offers are made public in the market), it considers only scheduling for a single factory, and this solution does not easily generalise to the multi-factory case considered here because they consider fundamentally different auction mechanisms, namely single-sided ones where only buyers compete for resources.

As a consequence of the fundamental issues discussed above, the aforementioned market mechanisms fail to solve our motivating problem for decentralised AR&O scheduling. Thus, we address these issues by proposing a variant of one of today's most prominent auction formats, namely the Continuous Double Auction (CDA) [3], which allows multiple customers and multiple factories to compete in a market. In so doing, we extend the single-factory job-shop scheduling problem proposed by Wellman *et al.* to a more general one that allows multiple factories (OHBs in the AR&O context) to compete. Second, we design a variant of the traditional CDA that also considers the time factor of the scheduling problem. Thus, in more detail, this work extends the state of the art in the following ways:

- First, we develop a variant of the Continuous Double Auction for multi-factory scheduling. Our market-based approach is novel in being the first auction mechanism that allows multiple customers and multiple factories to compete in a market for scheduling jobs without the need for a center and without the need to reveal private and often sensitive information to a center.
- Second, we demonstrate the effectiveness of our approach by evaluating our market mechanism. Specifically, we provide a lower bound efficiency of 84%, with our market-based approach sacrificing at most 16% for the added benefits of a more robust, dynamic and transparent solution.

The remainder of this paper is structured as follows. We begin in Section 2 by formalising the multi-factory scheduling problem. In Section 3, we describe our mechanism which we empirically evaluate in Section 4. Section 5 concludes.

2 The Multi-factory Scheduling Problem

In this section, we describe the general scheduling problem that this work focuses on. Although motivated by our specific AR&O problem, we believe this applies to a broad class of domains. We first extend Wellman *et al.*'s scheduling model to deal with the problem of multi-factory scheduling, rather than the restrictive single-factory case that they consider. To this end, we consider several factories, potentially owned by different

organisations, with the same number of one-hour unscheduled time-slots[2], denoted as $T = \{T_{start}, ..., T_{end}\}$. These time-slots can be allocated for customers' jobs, with each one having a limit price[3] representing the minimum price the factory will accept in exchange for that time-slot.

Next, we assume that each customer i, requires a single job of a certain *length* q^i (i.e. a number of time-slots), *value* ℓ^i (i.e the utility for all the q^i time-slots required) and *deadline* $t^i_{deadline}$ (i.e. completion is no later than a time-slot at $t^i_{deadline}$) scheduled in a single factory to follow up from Wellman *et al.*'s model. The customer is willing to spend no more than its value to have its job scheduled within its deadline. Furthermore, we assume that the customer has an inelastic demand[4]. That is, its utility is 0 if it cannot acquire sufficient time-slots to complete its job within its deadline.

The problem at hand is then to allocate a set of jobs in the available time-slots of a set of factories, subject to the length, value and deadline constraints of the jobs and the limit price constraint of the time-slots. In this context, given a set S_{cus} of customers and a set S_{fac} of factories, we have the demand, $demand^i \ \forall i \in S_{cus}$ and the supply, $supply^{j,a} \ \forall j \in S_{fac}, \ \forall a \in A^j$, where A^j groups time-slots with same limit prices given by:

$$demand^i = (id^i_{cus}, \ell^i, q^i, t^i_{deadline}) \ \forall i \in S_{cus}, \ t^i_{deadline} \in T \tag{1}$$

$$supply^{j,a} = (id^j_{fac}, c^{j,a}, f^{j,a}_{T_{start}}, ..., f^{j,a}_{T_{end}}) \tag{2}$$

where

$$f^{j,a}_t \in \{0,1\} \ \forall a \in A^j, \ \forall j \in S_{fac}, \ c^j_t = \sum_{a \in A^j} c^{j,a} f^{j,a}_t$$

and id_{cus} and id_{fac} are unique identifiers for customers and factories respectively, $\sum_{a \in A^j} f^{j,a}_t$ defines if time-slot t in factory j is allocated or not, $c^{j,a}$ is the limit price for a group of time-slots and c^j_t is the limit price of a time-slot t in factory j. Note that A^j can be defined over the space between a single set with all time-slots having the same limit prices and $|T|$ sets of single-time-slots in the case of different limit prices for all the time-slots of the factory[5].

We can now formalise the scheduling problem as a maximisation of profits of all stakeholders (to conform to the literature on the classic job-shop scheduling). We first define the following terms:

[2] Our choice of one-hour time-slots is not crucial to this work and, indeed, can be changed to represent time at arbitrary levels of granularities. In addition, there is no requirement that each factory has the same number of time-slots. We simply make these choices to simplify the simulations that follow later.

[3] The limit price corresponds to the production cost within that time-slot. A factory would allow usage of its time-slot only if a customer pays more than the associated production cost such that it does not make a loss.

[4] Note that the assumption of inelastic demand does not constrain our work. A customer with elastic demand would simply split its job into a set of q^i single-time-slot jobs.

[5] Here, $\sum_{a \in A^j} \sum_{t \in T} f^{j,a}_t = |T|$ and $\sum_{a \in A^j} f^{j,a}_t \leq 1 \ \forall t \in T, \ \forall j \in S_{fac}$. See Figure 1 for an example, with $|A^1| = 2$, $|A^2| = 4$ and $|A^3| = 2$).

- $\mathcal{A}(i,j) \in \{0,1\}\ \forall j \in S_{fac}, i \in S_{cus}$ specifies which customer is allocated to which factory.
- $\mathcal{TS}(i,j,t) \in \{0,1\}\ \forall j \in S_{fac}, i \in S_{cus}$ specifies which factory's time-slot is allocated to which customer.

The system is optimally scheduled when the following objective (profits of all stakeholders) is maximised, as we assume that stakeholders are self-interested, profit-motivated economic agents in the system. To this end, we must find:

$$\max \sum_{j \in S_{fac}} \sum_{i \in S_{cus}} \left[\mathcal{A}(i,j)\ell^i - \sum_{t \in T} (\mathcal{TS}(i,j,t)c_t^j) \right] \tag{3}$$

subject to the following constraints:

1. *Job deadline constraint (i.e. jobs are allocated within their deadline):*

$$\sum_{j \in S_{fac}} \sum_{t=T_{start}}^{t^i_{deadline}} \mathcal{TS}(i,j,t) = \sum_{j \in S_{fac}} \mathcal{A}(i,j)q^i,\ \ \forall i \in S_{cus}$$

2. *Factory's limit price constraint (i.e. all accepted jobs have value equal to at least the limit price for all time-slots):*

$$\sum_{t=T_{start}}^{t^i_{deadline}} \mathcal{TS}(i,j,t)c_t^j \leq \mathcal{A}(i,j)\ell^i,\ \ \forall i \in S_{cus},\ \forall j \in S_{fac}$$

3. *Factory's time-slot scheduled to a single customer (i.e. no time-slots can be shared for more than one job):*

$$\sum_{i \in S_{cus}} \mathcal{TS}(i,j,t) \leq 1,\ \ \forall t \in T,\ \forall j \in S_{fac}$$

4. *Job scheduled to a single factory (i.e. factories cannot share jobs[6]):*

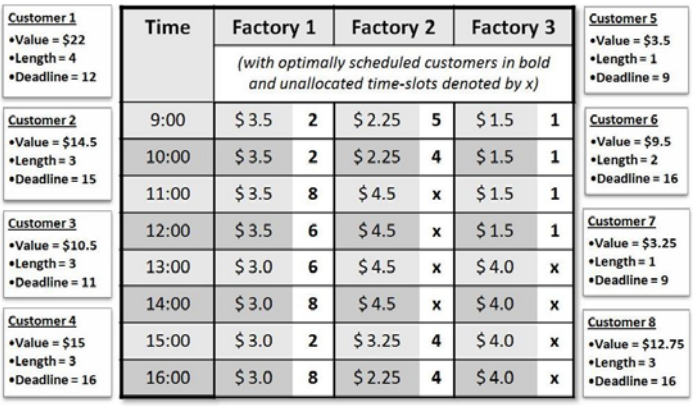
$$\sum_{j \in S_{fac}} \mathcal{A}(i,j) \leq 1,\ \ \forall i \in S_{cus}$$

5. *Customer's inelastic demand (i.e. jobs cannot be partially allocated):*

$$\sum_{j \in S_{fac}} \sum_{t \in T} \mathcal{TS}(i,j,t) = \sum_{j \in S_{fac}} \mathcal{A}(i,j)q^i,\ \ \forall i \in S_{cus}$$

Given complete and perfect information (with all agents truthfully revealing their preferences to a center), the center can optimally compute the solution to this problem (e.g. using the ILOG CPLEX optimisation tool). To this end, Figure 1 gives the demand and supply in the system along with the optimal solution (i.e. the allocated customers in each factory) to an example of such a scheduling problem with 3 factories and 8 customers. In particular, Customers 2, 6 and 8 are scheduled in Factory 1, Customers 4 and 5 in Factory 2 and Customer 1 in Factory 3.

[6] Note that this is certainly the case within the AR&O domain since engines are always serviced solely at one overhaul base. However, this might not generally be true, and depending on the specific setting, it is possible to relax this constraint.



Time	Factory 1		Factory 2		Factory 3	
	(with optimally scheduled customers in bold and unallocated time-slots denoted by x)					
9:00	$3.5	**2**	$2.25	**5**	$1.5	**1**
10:00	$3.5	**2**	$2.25	**4**	$1.5	**1**
11:00	$3.5	**8**	$4.5	**x**	$1.5	**1**
12:00	$3.5	**6**	$4.5	**x**	$1.5	**1**
13:00	$3.0	**6**	$4.5	**x**	$4.0	**x**
14:00	$3.0	**8**	$4.5	**x**	$4.0	**x**
15:00	$3.0	**2**	$3.25	**4**	$4.0	**x**
16:00	$3.0	**8**	$2.25	**4**	$4.0	**x**

Fig. 1. Multi-factory Scheduling Problem with 8 customers (with their limit price ℓ^i, length q^i and deadline $t^i_{deadline}$ $\forall i \in S_{cus} = \{1, ..., 8\}$) and 3 factories (with their limit prices c^j_t $\forall t \in T = \{9, .., 16\}$, $\forall j \in S_{fac} = \{1, ..., 3\}$ for each time-slot) with the optimal solution (allocated customer for each time-slot). The maximum profit extracted here is $35.

Having formally described the problem, we now reconsider the original contex context of AR&O scheduling discussed earlier on, and note that within this domain, Stranjak *et al.* have previously provided a greedy, non-optimal solution to a similar scheduling problem across multiple factories [8]. As with the ILOG CPLEX solution described above, their solution is centralised in nature and, thus, its computationally complexity increasing exponentially, making the solution intractable for large problems. In the next section, we propose a market-based approach to solving such a problem in a decentralised manner with a linearly increasing computational complexity. We evaluate the *efficiency* of such a mechanism as the ratio of profit of all stakeholders extracted in the mechanism to the profit extracted in the optimal allocation given in Equation 3.

3 The Market-Based Solution

We now detail our market-based solution to the multi-factory scheduling problem detailed in Section 2. Specifically, in this section, we describe our approach; an auction mechanism that allows self-interested, profit-motivated buyers (bidding for customers) and sellers (bidding for factories) to compete for time-slots. The scheduling is then determined by transactions (when a set of bids match with a set of asks) among the buyers and sellers which allocate time-slots to jobs subject to the constraints outlined in Section 2 (and specifically those described by Equation 3). We now describe our market protocol that determines how the agents strategically interact in the market.

3.1 The Market Protocol

The protocol we have developed is a variant of the CDA [9,3], designed to maximise profits in the system. In particular, trading agents are allowed to submit multi-unit bids

BID ORDERBOOK			
ID	**Total Price**	**Quantity**	**Deadline**
Buyer8	$12	3	16
Buyer1	$14.5	4	12
Buyer3	$9.5	3	11
...	...	...	...

(a)

ASK ORDERBOOK		
ID	**Unit Price**	**Free Time Slots (9:00 to 16:00)**
Seller1	$4.25	[1, 1, 1, 1, 0, 0, 0, 0]
Seller2	$4.5	[1, 1, 0, 0, 0, 0, 0, 0]
Seller1	$5.25	[0, 0, 0, 0, 1, 1, 1, 1]
Seller3	$6	[1, 1, 1, 1, 0, 0, 0, 0]
Seller3	$6	[0, 0, 0, 0, 1, 1, 1, 1]
Seller2	$6.25	[0, 0, 0, 0, 0, 0, 1, 1]
Seller2	$6.75	[0, 0, 1, 1, 1, 1, 0, 0]
...	...	...

(b)

Fig. 2. (a) Orderbook with uncleared asks, first ordered by lowest bid price per unit and, second, by the earliest time-slots for similar prices. (b) Uncleared bids are queued in an orderbook, ordered by the highest bid price per unit.

and asks (i.e. offers to buy and sell a number of time-slots respectively) which are queued in a bid orderbook (see Figure 2(a)) and an ask orderbook (see Figure 2(b)) respectively. These offers indicate a commitment from the buyers and sellers and cannot be withdrawn. The multi-unit bids allows allow the customers to express the number of time-slots required, and the order books effectively provide the mechanism by which any matching bids and asks are cleared. In more detail, the protocol proceeds as follows:

1. ***Bid:*** Buyer i submits a multi-unit bid, $bid_i = (id^i_{cus}, p^i_{total}, q^i, t^i_{deadline})$, $t^i_{deadline} \in T$ to buy exactly q time-slots (given the inelastic demand) within its deadline $t_{deadline}$ for no more than a *total price* of p.
2. ***Ask:*** Seller j submits different multi-unit asks, $ask^a_j = (id^j_{fac}, p^{j,a}_{unit}, f^{j,a}_{T_{start}}, ..., f^{j,a}_{T_{end}})$, $\forall t \in T,\ a \in A^j_{offered} \subset A^j$ for (not necessarily all) the unscheduled time-slots in its factory, with $A^j_{offered}$ defining the set of multi-unit asks. Note that the ask is defined over all the different time-slots to allow multi-unit asks in the market, rather than single-unit asks over single unscheduled time-slots. This is to simplify the bidding process by grouping similar asks.
3. ***Bid orderbook:*** Bids are queued in a bid orderbook, ordered by the highest price *per unit*[7] (see Figure 2(a)). Bids cannot be retracted once queued in the order book. This is to ensure consistency in the orderbook such that a seller may accept a bid without the risk of that bid being retracted. Thus, a bid may only be replaced by improving the bid (i.e. submitting a higher price) which would allow buyers to compete by shading their bids.
4. ***Ask orderbook:*** Asks are queued in an ask orderbook, ordered first by the lowest unit price and second (given the same unit price) by the earliest time-slot (see Figure 2(b)). When we have asks with similar prices, our protocol prioritises the clearing

[7] Highest unit-price ordering is necessary because a job's value is defined for the whole job, rather than over the different time-slots required.

of the earlier ones. This is because the later time-slots have a higher probability of clearing future bids and, hence, a higher expected profit in the future than the earlier ones. Thus, our mechanism clears earlier asks with the same price first to maximise profit. As with buyers, sellers are not allowed to retract asks, but are allowed to improve on them by submitting a lower ask price.

5. ***Clearing a new bid:*** Whenever a new bid is added in the bid orderbook, the market attempts to clear by matching the new bid[8] with the ask orderbook. Our mechanism searches the ask orderbook for the set of lowest asks, $A^j_{matched} \subset A^j$ from each seller j that would completely clear the bid. The market then clears the matched asks $A^{j^*}_{matched}$ from the seller j^*, if any, with the lowest total price against the new bid. If the market clears, the newly matched bid is removed from the bid orderbook while the parameters of the matched asks, $f_t^{j^*}\ \forall t \in T$, are updated. If $\sum_{t \in T} f_t^{j^*,a} = 0,\ where\ a \in A^{j^*}_{matched}$, $ask^a_{j^*}$ is removed from the ask orderbook (because the ask has been completely cleared and all time-slots have been scheduled).
6. ***Clearing a new ask:*** When a new ask is received, the market attempts to match the seller (with now a better set of asks queued in the ask orderbook) which submitted that ask with the bid orderbook. In particular, the mechanism runs down the bid orderbook to find the highest bids that would be completely cleared by the seller's set of asks (stopping when all the asks from that seller are cleared or at the end of the bid orderbook). The market then clears these highest bids, if any, against the matched asks from the seller. All the cleared bids are then removed from the bid orderbook, while the cleared asks are updated and removed from the orderbook if completely allocated (as seen with the clearing of a new bid).

Given the structure of our market mechanism, we now consider its behaviour. In particular, we use a simple bidding strategy for such a market protocol in order to provide a lower bound benchmark on the efficiency of the market mechanism.

3.2 The Bidding Strategy

One of the principal concerns in developing a market mechanism is to ensure that it is efficient and that the system does not break down even with comparatively simple bidding behaviour on behalf of the buyers and sellers. This is important because as designers, we cannot dictate the specific strategies of the buyers and sellers and, so, we want to ensure that the market performs well for whatever strategies are adopted. The underlying intuition here is that by considering this simple behaviour, we are able to establish a lower bound on the efficiency credited principally to the market structure rather than the behaviour (assuming that agents are motivated by profits and not malicious, e.g. sellers bidding less and buyers more than their limit price to break down the market). This approach has been advocated a number of researchers, most notably by Gode and Sunder [5], and thus, to this end, we adopt Gode and Sunder's Zero-Intelligence (ZI)

[8] Our mechanism attempts to clear only the new bid (ask) because we are continuously clearing the market which ensures that any queued bid (ask) cannot be cleared by the current ask (bid) orderbook.

bidding strategy in our work because it simply submits a random bid or ask based solely on its limit price, ignoring the state of the market or past market information [4]. The Zero-Intelligence strategy works as follows[9]:

1. For buyer $i \in S_{cus}$,

$$p^i \sim \mathcal{U}(0, \ell_i)$$
$$\text{bid}_i = (id^i_{cus}, p^i, q^i, t^i_{deadline})$$

2. For seller $j \in S_{fac}$,

$$p^{j,a} \sim \mathcal{U}(c^a_i, p_{max})$$
$$\text{ask}^a_j = (id^j_{fac}, p^{j,a}, f^{j,a}_{T_{start}}, ..., f^{j,a}_{T_{end}}) \; \forall a \in A^j$$

Thus, the buyer submits a multi-unit bid based on its limit price at random times. Conversely, the seller j submits a set of multi-unit asks, A^j, over all the unscheduled time-slots in its factory also at random times. Given our market protocol and the Zero-Intelligence strategy, we now empirically evaluate our market-based scheduling mechanism.

4 Empirical Analysis

In this section, we empirically compare the market-based solution (see Section 3) against the optimal solution (see Section 2). In our experimental setup, for every combination of buyers and sellers, we consider 50 different sets of demand and supply (see Equations 1 and 2). For each set of demand and supply, we consider a statistically significant number of runs[10], namely 100, and average the performance over these different runs and sets of demand and supply. Based on standard experimental setup of the CDA [9], we induce the demand and supply by drawing buyer i's and seller j's endowment (of time-slots) from random distributions[11] as follows:

$$\begin{aligned} T_{start} &= 9 \\ T_{end} &= 16 \\ q^i &\sim \mathcal{U}^{\mathcal{I}}(1, 4), \\ t^i_{deadline} &\sim \mathcal{U}^{\mathcal{I}}(q^i - 1, (T_{end} - T_{start})) + T_{start}, \\ \ell^i &\sim \mathcal{U}(1.5, 4.5) \times t^i_{deadline} \\ c^{j,a} &\sim \mathcal{U}(1.5, 4.5), \; \forall a \in A^j \end{aligned}$$

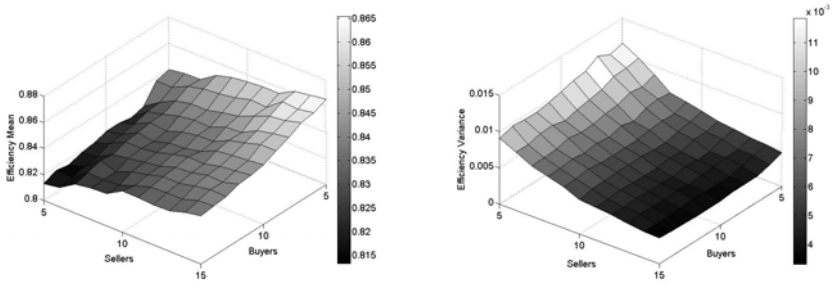
where A^j is a randomly generated set of sets grouping similar limit prices. Furthermore, because of the informationally decentralised nature of the mechanism, it is not possible to determine when the market reaches completion[12]. Thus, we impose a deadline to

[9] $X \sim \mathcal{U}(a, b)$ is a discrete uniform distribution between a and b with steps of 0.01.

[10] We validated our results at the 95%-confidence interval by running the non-parametric Wilcoxon rank-sum test.

[11] $X \sim \mathcal{U}^{\mathcal{I}}(a, b)$ is a uniform distribution of integers between a and b.

[12] A market reaches completion when there can be no more transactions. This information is unknown unless all private information is available.



(a) Efficiency Mean (b) Efficiency Variance

Fig. 3. The efficiency of the market mechanism for different numbers of buyers and sellers in the market. Efficiency converges to an efficiency of 84%as the number of buyers and sellers increases, while the variance decreases.

limit the duration of the auction[13]. In our experiments, we set this deadline to 5000 rounds.

The mean and variance of the efficiency (defined in Section 2) of the market-based solution over different problem sizes is given in Figures 3(a) and 3(b) respectively. As we can observe, the market efficiency averages 84% (ranging between 81% and 87%) with efficiencies converging to 84% as the number of factories and consumers increases. We also observe that the variances of the efficiency decreases rapidly as the size of the scheduling problem increases. Thus, our mechanism becomes more effective in finding profitable allocations as the number of factories increases while its efficiency is unaffected by increasing demand.

Now, because we impose a deadline (as it is unknown as to when no more resources can be cleared in the market), it is insightful to analyse if we are closing the market too early or too late. To this effect, in an example of a large problem with 15 factories and 15 customers[14], we consider how the efficiency and the volume of allocation of time-slots vary over the rounds (see Figure 4). In particular, we observe that the bulk of the allocations are made within the first few hundreds rounds (with 85% allocated within the first 500 rounds) even though the market reaches completion after 3000 rounds. This validates our choice for a deadline at 5000 rounds as we effectively limit the duration of the auction without compromising on efficiency. Furthermore, because time-slots are gradually allocated, we can consider our market-based mechanism as an *any-time* approach (which can be halted at any time for a solution). This contrasts with the centralised approach where time-slots are only allocated once a solution is computed. An any-time solution would indeed be very useful in a problems with hard deadlines.

Furthermore, to examine the tractability of our market-based solution, we compare the computational time of our market mechanism against that of a centralised, optimal solution computed using ILOG CPLEX (as highlighted in Section 2). While the

[13] If we consider an environment where agents are allowed to enter or leave the system or can renew their endowment, we do not impose a deadline in our auction which possibly never runs out of transactions.

[14] Similar trends were observed for other numbers of factories and customers.

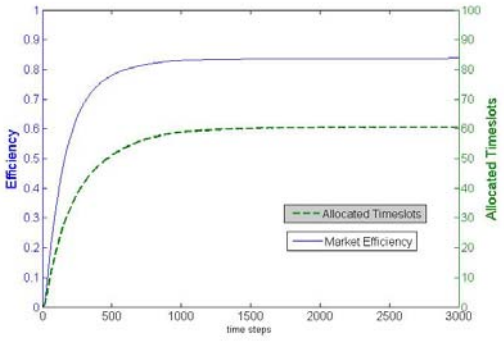
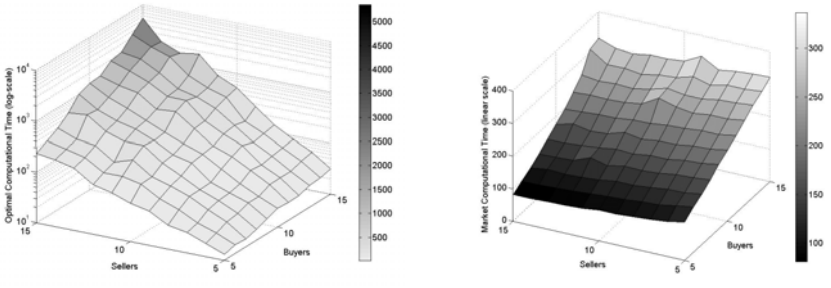


Fig. 4. For a problem of 15 factories and 15 customers, the market closes after 3000 rounds at an efficiency of 84%. Note that 84% of the volume of transactions are completed within the first 500 rounds and 97% within 1000 rounds. The efficiency after 500 rounds is within 7% of its maximum and within 0.9% after 1000 rounds.



(a) Computational time for optimal solution (log-scale)

(b) Computational time for market-based solution

Fig. 5. Computational times of the centralised (a) and decentralised (b) approaches. Note that the former increases exponentially while the latter scales linearly. For a problem with 15 factories and 15 customers, the computational time for the optimal solution is around 5400ms compared to 320ms for the market-based one.

computational time of the centralised, optimal solution increases exponentially, that of the decentralised market-based approach increases linearly. This is because the computational complexity of the latter approach is due principally to the clearing process, with the size of the orderbooks to be cleared also increasing linearly. Thus, our market-based approach is indeed tractable as one of the desirable properties for a decentralised mechanism in the motivating AR&O setting.

5 Conclusions and Future Work

In this paper, we proposed a novel decentralised mechanism for multi-factory scheduling based on a variant of the Continuous Double Auction, that does not require the

revelation of private preferences to a third-party agent. We empirically demonstrated a lower bound efficiency of 84% in our auction mechanism using a Zero-Intelligence bidding strategy. We thus showed that we sacrifice a reasonably small level of efficiency for the benefits of a decentralised and transparent approach (through its public order-books and the fact that there is no center) and scaleability (given the linearly increasing complexity of our market-based solution).

Our future work in this area focuses on taking the solution that we have developed here and applying it within a fine grained industrial simulation of the Aero Repair and Overhaul domain (before ultimately deploying it within the operational system). As we envisage increasingly larger scheduling problems in this context, the need for robust and scaleable solutions of the kind that we have presented here, will become highly desirable and indeed essential in the future. To further improve the performance of our scheduling approach we would also like to explore more intelligent bidding strategies that can strategise effectively on the additional time factor that we have in this domain. In this respect, we believe that a considerably higher efficiency can be achieved, and this belief is supported by the observation that within the standard CDA, state of the art strategies can led to an improvement in efficiency from 97% (achieved with the Zero-Intelligence bidding strategy) to 99.9% [11,10]. In addition, we intend to analyse the efficiency of our system when agents enter and leave the system at any time. This is an important issue within the Aero Repair and Overhaul domain, since damaged engines must often be added to an existing schedule at short notice, and thus, we must evaluate how well our auction-based mechanism reacts to sudden change in the demand and supply.

References

1. Baker, A.D.: Metaphor or reality: A case study where agents bid with actual costs to schedule a factory. In: Market-based Control, ed. Clearwater. World Scientific, New York (1992)
2. Blazewicz, J., Ecker, K.H., Pesch, E., Weglarz, J.: Scheduling Computer and Manufacturing Processes. Springer, Berlin (1996)
3. Friedman, D., Rust, J.: The Double Auction Market: Institutions, Theories and Evidence. Addison-Wesley, New York (1992)
4. Gode, D.K., Sunder, S.: Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality. Journal of Political Economy 101(1), 119–137 (1993)
5. Gode, D.K., Sunder, S.: What Makes Markets Allocationally Efficient? The Quarterly Journal of Economics 35, 603–630 (1997)
6. Ichimi, N., Iima, H., Hara, T., Sannomiya, N.: Autonomous decentralized scheduling algorithm for a job-shop scheduling problem with complicated constraints. In: Proceedings the Fourth International Symposium on Autonomous Decentralized Systems, pp. 366–369 (1999)
7. Rassenti, S.J., Smith, V.L., Bulfin, R.L.: A combinatorial auction mechanism for airport time slot allocation. The Bell Journal of Economics 13(2), 402–417 (1982)
8. Stranjak, A., Dutta, P.S., Ebden, M., Rogers, A., Vytelingum, P.: A multi-agent simulation system for prediction and scheduling of aero engine overhaul. In: Proceedings of the seventh International Conference on Autonomous Agents and Multi-Agent Systems (Industrial Track), pp. 81–88 (2008)

9. Vytelingum, P.: The structure and behaviour of the Continuous Double Auction, Ph.D. dissertation, School of Electronics and Computer Science, University of Southampton (2006)
10. Vytelingum, P., Cliff, D., Jennings, N.R.: Strategic bidding in continuous double auctions. Artificial Intelligence Journal 172(14), 1700–1729 (2008)
11. Vytelingum, P., Dash, R.K., David, E., Jennings, N.R.: A risk-based bidding strategy for continuous double auctions. In: Proceedings of the Sixteenth European Conference on Artificial Intelligence, pp. 79–83 (2004)
12. Wellman, M., Walsh, W., Wurman, P., MacKie-Mason, J.: Auction protocols for decentralized scheduling. Games and Economic Behavior 35, 271–303 (2001)