Distributing complementary resources across multiple periods with stochastic demand: Hedging via time frame aggregation

Shih-Fen Cheng John Tajan Hoong Chuin Lau
School of Information Systems
Singapore Management University
{sfcheng, jtajan, hclau}@smu.edu.sg

Abstract

In this paper, we evaluate whether the robustness of a market mechanism that allocates complementary resources could be improved through the aggregation of time periods in which resources are consumed. In particular, we study a multi-round combinatorial auction that is built on a general equilibrium framework. We adopt the general equilibrium framework and the particular combinatorial auction design from the literature, and we investigate the benefits and the limitation of time-period aggregation when demand-side uncertainties are introduced. By using simulation experiments, we show that under stochastic conditions the performance variation of the process decreases as the time frame length (time frames are obtained by aggregating time periods) increases. This is achieved without causing deterioration in the mean performance.

1 Introduction

Allocating resources for problems with decentralized information and control is a challenge for both planners and agents. From the planner's perspective, s/he has to design protocols that would function well even with incomplete information from and limited control over these free-willed agents. From the agent's perspective, the challenge lies in the strategic nature of the resource allocation process; no matter what resource allocation protocol is chosen, every agent has to plan his/her acts considering the existence of other selfish and unpredictable agents. Adding in uncertainties that might come from either planners or agents, we have a problem domain that is well-sought-after but still a long way to go before being claimed well-understood.

The challenges posted by uncertainties can be elaborated more from both planner's and agent's perspectives. For planners, uncertainties might be introduced if managed resources are subject to unexpected breakdown or service disruptions. For agents, uncertainties might come from dynamic task arrivals or delayed (or canceled) task assignments. Assuming centralized control is possible, a wide array of methodologies (e.g., stochastic programming and Markov decision process) have been proposed and these uncertainties are addressed quite effectively. However, in the decentralized cases, the successes of these methodologies are much more restricted, mostly due to the fact that necessary information is often absent and the implementation of resources allocation plan is hard, both results of the existence of selfish agents.

Market is arguably the most well-studied mechanism in addressing issues related to decentralized information and control [9]. Agents' solvency is respected in market mechanisms and globally optimal allocation could be achieved with proper design [11]. A lot of factors are related to the success of using market mechanisms. For one, uncertainties could potentially cause a lot of issues for both planner and agents and several remedies have been proposed. Firstly, agents may be allowed to *decommit* at a penalty, meaning that they could forfeit their commitments if the supply or demand for the resources are disrupted unexpectedly. Although this design makes it much easier for agents to flexibly deal with uncertainties, the market contracts become less binding and it could be exploited by malicious agents which could compromise the fairness of the system and make the system less stable. Alternatively, the planner could establish a *secondary market* (or aftermarket) to grant second chances to agents affected by uncertainties.

Our approach differs from these earlier approaches in that we do not plan to design new mechanisms or rules; instead, we would like to explore the use of time-period aggregation in reducing impact created by uncertainties. This idea is fairly simple, and the intuition behind it is based on risk pooling. In our case, we aggregate multiple consecutive time periods so that variability resulted from demand or supply disruptions could be reduced. To be more specific, we are interested in finding a common aggregation factor (how many time periods should be included in

an aggregated *time frame*) so that performance variability measured by coefficient of variation is significantly reduced while trading off measures like absolute performance and allocation efficiency.

2 Review of Past Work

Past work on market-based approaches in resource allocation covers a wide range of applications. Sutherland [7] was one of the first to propose using markets in allocating computing resources. In cases where we can assume convex preferences and competitive economy, researchers have found promising results computationally [11, 1]. For scheduling problems, these assumptions do not hold in general, as it is quite common for scheduling problems to have discrete resources (time slots) and complementary preferences, and various special designs are proposed [3, 10].

Despite the success of using markets in various setups, a number of inefficiencies might occur when markets are used in allocating resources. One of the major source of inefficiencies is the complementarity among resources. That is, when multiple independently allocated resources are required in order to accomplish a single task, an agent might end up with only part of the required resources, and efficiency is lost as a result. This issue can be directly addressed by running a combinatorial auction [6]. Although we use a particular type of combinatorial auction in our research, the design of combinatorial auction is not our main focus and thus we will not go too deep into the vast amount of literature on it. Cramton et al. [2] provide a thorough treatment on the subject.

Another major source of inefficiency is caused by uncertainties. Just as complementarities might cause an agent to unexpectedly miss some critical resources required for accomplishing its tasks, an agent's preference might be altered unexpectedly either due to changes to its current tasks or dynamic arrivals of new tasks. These changes would result in different resource requirements and an agent's currently committed resources might not satisfy the latest requirement. Some authors suggest the use of finance-inspired mechanisms like options and futures in dealing with these uncertainties [7]. In our work we focus on how to use time period aggregation in dealing with uncertainties.

3 Problem Statement and Solution Methodology

The problems we are interested in studying are stochastic, decentralized, and require complementary resources. In particular, we would investigate scheduling problems with these properties. Applications with these properties could be easily found in many important domains.

We would like to emphasize that it is not our intent to justify whether markets are the allocation mechanism to use given some specific scenario; instead, we are interested in improving the performance of a particular market mechanism in responding to uncertainties if it is indeed adopted. More importantly, we would like to achieve this performance improvement without modifying the structure of the market mechanism.

When we look at the characteristics of the problem again, one would notice that resource complementarities and uncertainties are the major issues we need to take care of. Both issues could be resolved by introducing some structural changes to the mechanism. For example, aftermarkets and decommitment protocols could both solve these issues to some degree. However, just as argued earlier by Sandholm [6], these methods are all not as direct as addressing these issues directly within the market mechanism. On resource complementarities, combinatorial auctions are the most well-studied mechanism that by definition guarantees only desired bundles are allocated. Of course, this doesn't come without a price, and added computational complexities are what one has to pay for the services of combinatorial auctions. In our study, we choose a special combinatorial auction setup in order to balance the benefit and the computational burden. The detail of this will be deferred to subsections right after this.

The main contribution of our paper is the proposal of using time-period aggregation as a way in handling of uncertainties. Similar to the resource complementarities, although uncertainties could be handled by various additions to the mechanisms (like aftermarkets or decommitment protocols), it is not as direct as addressing uncertainties directly within the mechanism. We will defer the discussion on the time-period aggregation methodology to later sections.

3.1 Problem Statement

For illustration purpose, we choose to study a particular job shop scheduling problem inspired by the operations of a mega container terminal. In a typical container terminal (see Figure 1 for a schematic view on container port operations), there are *quay-side cranes* (QC) that load/unload containers to/from the ship, *prime movers* (PM) that carry containers around, and *yard cranes* (YC) that move containers between yard (storage space) and prime movers. Each QC operator will be given a job schedule (which contains both loading and unloading jobs), and the operations of QC can be considered as semi-independent: each operator is endowed with a small amount of guaranteed PM and YC resources but has to compete for the additional usage rights to PM and YC so that its jobs can be finished as soon as possible. To reflect the fact that these operators actually plan and act on their own behalves, we model the resulting

resource allocation problem in a decentralized manner, in which each operator is represented by an agent. From the operator's point of view, it is very important to ensure that operations at QC, PM, and YC are synchronized, i.e., there is no gap between operations. In practice, the processing times at both QC and YC are quite stable and thus can be viewed as almost deterministic; however, due to the congestion on the ground, the transport time by PM can be highly variable. The need for operation synchronization generates complementary resource needs, while the highly variable PM transport time makes the problem stochastic.

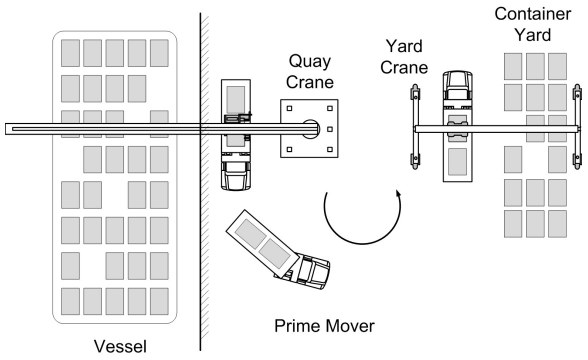

Figure 1. An overview of a typical container terminal operation.

To simplify the resource allocation process, we assume that the planning horizon is partitioned into discrete time periods with uniform length and the number of resources (PM and YC) is kept constant throughout the planning horizon. Agents will be competing for the *rights* to use resources within specified time periods. To address this decentralized problem with complementary resource requirements and stochastic job processing time (on PM only), we propose a special breed of combinatorial auction that is based on the general equilibrium framework.

3.2 A Combinatorial Auction based on the General Equilibrium Framework

Following the definition by McAfee and McMillan [5], an auction is: "... a market institution with an explicit set of rules determining resource allocation and prices on the basis of bids from the market participants." In our study, resources are owned by a central owner and agents (QC operators) have to bid for the rights to use resources in the desired time periods. Due to the needs of having to synchronize operations, an agent has to secure a bundle of (resource, time period) tuples that satisfies its plan for job completions. Combinatorial auction is a special class of auction that accepts bids containing bundles. Unfortunately, combinatorial auctions come with hefty computational requirements, thus making its adoption hard. The reason for this is that in a combinatorial auction, the bid matching problem, or the *winner determination problem*, is usually modeled as an integer linear program and it is known to be $\mathcal{NP}$-hard. Although a number of efficient implementations are proposed (e.g., see Sandholm [6]), combinatorial auctions are still not very computational affordable.

To avoid solving winner determination problems, we assume that unless *clearing* is already feasible (implying that the aggregate demands for all (resource, time period) tuples could be met by supply), the auctioneer would not clear the auction and finalize the allocation. Instead, the auctioneer would adjust the current price for each tuple and announce the adjusted prices back to all agents. Hopefully, if the auctioneer adjust the price properly, after several iterations agents would come up with bids that are feasible in aggregate. To simplify the consideration, we would consider agents as price takers, meaning that they would view market prices as exogenous and not something that would change because of their own actions.

When a price vector induces agents to generates bids that have aggregate demands equal supplies for all tuples, we say that a *general equilibrium* is reached. General equilibrium is first proposed by Walras [8] and has since been a powerful concept in explaining a wide variety of economic phenomena. There are already a number of past work on applying the general equilibrium approach in solving distributed resource allocation problem. For example, the connection between Lagrangian relaxation and combinatorial auctions that are powered by the general equilibrium framework is established in Kutanoglu and Wu [3].

From the above discussion we can see that in developing a combinatorial auction that is based on the general equilibrium framework, there are two major problems that need to be addressed. The first is on how agents generate bids and the second is on how auctioneer adjusts prices. These details will be discussed in Section 4.

3.3 Time Period Aggregation

Combinatorial auctions can effectively handle complementarity in resource requirement. However, to address uncertainties that are associated with job processing times, we need additional device. In our study, we propose to keep the auction features unchanged and only tinker with the length of time periods. Assume that we have N time periods in our problem. By aggregating multiple time periods into *time frames*, we change the set of controls (the set of possible combinatorial bids) available to the agents. Given identical price rates (in dollars/time period) for resources, bidders would only bid for resources they plan to use. If the time frame length is one time period, each bid-

der can bid for the exact quantity of resources needed, and utilization will be unity (if the task schedule is deterministic). For simplicity, assume that there is only one resource type r allocated across N time periods, then $\mathbf{U}(1)$, the set of all possible bids (the strategy space) when the time frame length is one time period, is the set of all vectors of the form $< x_1^r, x_2^r, ..., x_N^r >$, where $0 \leq x_n^r \leq M$ is the quantity needed for time period n and M is the resource supply per time period.

When the time frame length is y time periods (with y integer), the new strategy space $\mathbf{U}(y)$ is a proper subset of $\mathbf{U}(1)$, as the elements of $\mathbf{U}(y)$ are vectors of the form: $< x_1^r, x_2^r, ..., x_N^r >$, where $0 \leq x_n^r \leq M$ and $x_{ky+1}^r = x_{ky+2}^r = \ldots = x_{(k+1)y}^r$, for $k = 0, 1, \ldots, \lfloor \frac{N}{y} \rfloor$. By increasing the time frame length from 1 to y time periods, the set of possible bids becomes smaller, from $\mathbf{U}(1)$ (with M^N elements) to $\mathbf{U}(y)$ (with $M^{\lceil \frac{N}{y} \rceil}$ elements). This rapid reduction in the strategy space means that it may be easier to obtain the optimal bid from $\mathbf{U}(y)$ compared to the optimal bid from $\mathbf{U}(1)$. However, it should be noted that since $\mathbf{U}(\mathrm{y}) \subset \mathbf{U}(1)$, the optimal bid found in $\mathbf{U}(y)$ might be suboptimal in $\mathbf{U}(1)$[1].

Our conjecture is that making bidders bid for resources in longer time frames could improve the robustness of the resulting resource allocation in face of uncertain job processing times. Furthermore, when the time frame length is much larger than the variability range of the resource demand from the bidders, the performance of the system under stochasticity should mimic the performance of the system under deterministic conditions fairly well.

Our idea is based on the following simple inventory model: each bidder has an expected order arrival schedule (in time periods), and wishes to meet all of its customer demands. However, each bidder is unaware of the underlying stochasticity of the order quantity per time period. Thus, the anticipated demand for period *n* is D_n, while the actual demand for period *n* is could be higher or lower than D_n. Any unmet demand for period *n* will carry over and needs to be serve later.

When the time frame length is one time period, an optimal ordering policy for the bidder is to order D_n resources for each period *n*. When the time frame length is *y* time periods, the bidder orders $\max_{n=(j-1)y+1,\ldots,jy} D_n$ for time frame *j*, which is composed of time periods ranging from $((j-1)y+1)$ to jy. From the definition, when the time frame length is increased from *y* to *2y* time periods, the ordering level for each time period when the time frame length is *2y* will be at least as high as the ordering level when the time frame length is *y*. This reduces the probability of lost demand. We interpret this reduction in the probability of lost demand as a reduction in the amount of jobs that are forced to remain idle due to resource unavailability.

[1] It might be tempting to argue that the utility one could get from $\mathbf{U}(y)$ would be monotonically non-increasing in y. However, for a pair y_1 and y_2 where $y_1 < y_2$ and their GCD = 1, no subset relationship could be established and thus we could not conclude which case would generate a better bid.

4 Setup of the Combinatorial Auction

Following the introduction given in 3.2, we will now formally state the combinatorial auction model with time period aggregation.

We use a multi-iteration combinatorial auction in which each agent is allowed to submit only one quantity bid. Resources are assumed to be centrally owned by the auctioneer, and the exclusive rights to use the resources are allocated to agents through the bidding process. The planning horizon is discretized into time periods with uniform length, however, the resource usage rights are allocated in time frames, which is the multiples of time periods. The length of time frame, unit price for each (resource, time frame) tuple (for the rest of the paper, we will just use "tuple" for brevity), and the total available resource supply are assumed to be common knowledge among agents. Agents receive updates on the unit price for every tuple at the beginning of each round, and generates a single bid containing quantity requested for every tuple. Agents will submit their bids simultaneously to the auctioneer and if the stopping criterion is not met, prices for all tuples will be adjusted according to the net demands (difference between aggregate demand and supply). In general, prices are lowered for tuples with negative net demands and prices are increased for tuples with positive net demands. After prices are adjusted, all agents are notified of the new prices for another round of bidding. A feasible solution occurs when the demand for each tuple does not exceed the total resource supply.

We will highlight two important classes of problems that need to be addressed in the above mentioned process, namely "bid generation" and "price adjustment".

4.1 Bid Generation

As described in Section 3, the bidders in the system are assumed to be price takers and thus their bid generation problems, given tuple prices, can be treated as local optimization problems. Following the problem statement in 3.1, we now formally provide an abstract model for the bid generation problem.

For every agent, a job list containing both unload and load jobs is assigned and all unload jobs come before load jobs. An unload job requires a sequence of resources R_1, R_2, and R_3 (representing QC, PM, and YC respectively). A load job, on the other hand, requires a sequence of resources R_3, R_2, and R_1. No job can queue for resources in between processes, and there is a strict precedence constraint for all

processes at resource type R_1. Jobs might have different (and stochastic) processing times at each resource. From the description of this bidder model, we can see that both complementarity and uncertainty are embedded.

Every bidder i has an arrival time a_i, a due time d_i, a makespan cost rate per time period ϕ_i and a tardiness cost rate per time period γ_i. Let the time bidder i finishes its job list be l_i, then the makespan of bidder i is defined as $M_i = l_i - a_i$. Let the unit price of R_k in time frame t be P_{kt} and the bid $B(i)$ of bidder i be a vector of tuples (R_k, t). The cost function of bidder i is:

$$\sum_{(R_k,t)\in B(i)} P_{kt} + M_i\phi_i + \gamma_i \max(0, M_i - d_i). \quad (1)$$

(1) can be minimized by solving an integer programming model, however, for large-scale problems with hundreds of jobs (as we study in this paper), this approach is computationally prohibitive. Alternatively, we can apply a local search technique called *relax and repair*. In the relax phase, the multi-time-frame problem is relaxed into a single-time-frame problem and solved. In the repair phase, bidders perform their respective local search, using the solution from the relax phase as an initial solution. For each time frame, the cost function descent direction for a resource type is determined, and the quantity is adjusted until a local optimum is reached. This is repeated for all time frames and then for all resource types. This method is shown to be very efficient and effective [4], and the computational requirement grows only linearly with the number of resources and time frames.

4.2 Price Adjustment

Let C_{kt} be the available supply for resource type R_k in time frame *t*, D^r_{ikt} be the demand of bidder i for resource type R_k in time frame t for the r^{th} iteration, P^r_{kt} be the price for a single unit of R_k in time frame t for the r^{th} iteration. The price is adjusted as follows:

$$P^{r+1}_{kt} = \max\left(0, P^r_{kt} + \alpha \frac{\sum_i D^r_{ikt} - C_{kt}}{C_{kt}} \bar{P}\right),$$

$$\bar{P} = \frac{\sum_{k,t} P^r_{kt} \sum_i D^r_{ikt}}{\sum_{i,k,t} D^r_{ikt}},$$

where α stands for the step size of the adjustment. Interpreted intuitively, the price is adjusted in proportion to the ratio between net demand and total supply, as well as the average resource price. Larger (smaller) values of α denote larger (smaller) changes in resource prices for the same net demand. A more detailed discussion on the selection of α can be found in [4].

During our initial experiments we discovered that when the time frame length is long, demands in adjacent time frames may be negatively correlated, as demand moves from the more expensive time frame to the cheaper neighbors. And this could easily result in a cycle as demands bounce back and forth. To reduce the possibility that demands oscillate between adjacent time frames, we dampen the price adjustment for selected time frames.

Without going into too much technical details, our idea of dampening price adjustment is to first detect the existence of potential price oscillation and then identify time frames that cause this oscillation. To achieve this, we only need to keep track of the event of *sign switch*, i.e., when the net demand changes from negative to positive or vice versa. Whenever this is detected, we would check the sign switch status in the adjacent time frames and if they are negatively correlated to the current time frame, a dampening factor is applied. This is an easy addition of the original price adjustment process since we only need to keep track of the aggregate demand from the last iteration for this purpose.

Each auction is assumed to last for K rounds. All feasible allocations are stored, and the one with least expected makespan (which is the sum of expected makespan reported by individual agents) will be selected as the final allocation.

5 Numerical Experiments

5.1 Experiment Setup

In all instances studied, there are four agents, each representing a QC operator. The implication of this is that every agent will be endowed with exactly one unit of R_1 (QC), and no additional unit would be available. For R_2 (PM) and R_3 (YC), each agent is initially assigned 2 units and 1 unit respectively[2], and there are 10 units R_2 and 8 units R_3 available for bidding in the market. The initial prices for R_2 and R_3 are \$50/hour and \$30/hour respectively.

All bidders have identical relax-and-repair bid generation strategies, and all have makespan cost rate of \$100/hour and tardiness cost rate of \$500/hour. All bidders have 10 unload jobs followed by 10 load jobs. To test the scalability of our approach, we also test it on the cases with 50 unload jobs and 50 load jobs. As the results are qualitatively the same, we will only report results from the earlier case.

To provide necessary granularity, it is assumed that the length of each time period is 1 minute. The processing times at R_1 and R_3 are assumed to be deterministic, and are fixed at 3 and 6 minutes respectively. The processing time at R_2 is stochastic and only realized after the auction process, however, the mean processing time is known to the bidders as μ_j for job j. We assume that the processing time at R_2 for job j follows a discrete uniform distribution that has 20 possible values, $\{0.525\mu_j, 0.575\mu_j, \ldots, (0.525 +$

[2]Since each agent has initial endowment on R_2 and R_3, even it gets no additional units from the market, its job list can still be finished. However, the resulting makespan would be prohibitively long.

$0.05i)\mu_j, \ldots, 1.475\mu_j\}$. The standard deviation can be easily computed as $\sqrt{\frac{\sum_{i=0}^{19}((0.525+0.05i)\mu_j-\mu_j)^2}{20}} = 0.288\mu_j$. We can easily change the variability while preserving the mean by changing the range of the distribution; however, the conclusion we obtained is qualitatively the same, and is not presented.

By summarizing real-world data, we have designed 11 problem sets, each with different R_2 mean processing times. To better approximate the real-world operational conditions, we test four different agent arrival and due time patterns for each problem set. We assume that first agent always arrives at the beginning of the first time frame and has a due time that is $f/2$ hours after arrival, where f is the number of unload jobs. In our experiments, there are 6 different time frame lengths: 15, 30, 45, 60, 90 and 120. Further increases in the time frame length might reduce the problems into single period time frame problems.

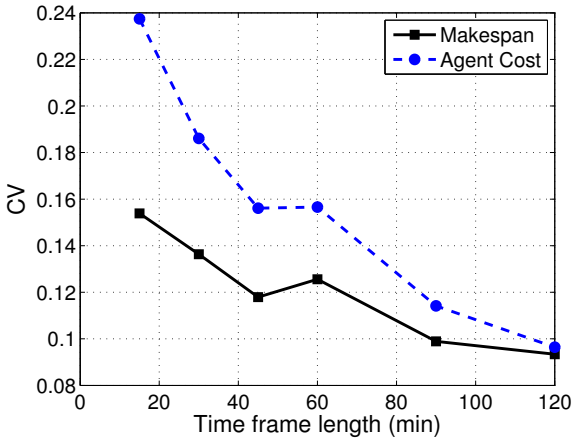

Figure 2. The average CV for makespans and agent costs.

5.2 Results and Discussions

The effectiveness of time period aggregation could be assessed by two measures: 1) the robustness of the resource allocation plan, and 2) the performance of the allocation plan under uncertainty (exact measures are to be defined later). In the following analysis, we repeatedly refer to the performance within a deterministic and a stochastic environment. We now formally define their definitions. For the performance in a deterministic environment, we refer to the performance achieved by the allocation plan generated with deterministic agents (agents have access only to the mean processing time, and bids are generated by solving deterministic problems). For the performance in a stochastic environment, we refer to the expected performance of the same allocation plan when the processing time of R_2 is a random variable. To estimate this stochastic performance, we perform 5000 Monte Carlo simulations on the R_2 processing times.

5.2.1 Impact of Time Period Aggregation on Robustness

The robustness of the resource allocation plan is measured by the *coefficient of variation* (CV) of makespans and agent costs. By definition CV is a normalized measure for the variability, and is computed by dividing the standard deviation by the mean. Since CV is dimensionless and scale invariant, it allows us to compare variability across scenarios consistently.

The CV of both makespans and agent costs are plotted in Figure 2, and the resource allocation plans are indeed more robust when the time frame lengths are longer. This is consistent with our conjecture that longer time frames could stabilize the unexpected fluctuations in the processing times. However, as the figure suggests, this benefit wanes as the time frames grow longer.

Another indication that longer time frames stabilize stochastic fluctuations is the difference between the average makespans obtained in a deterministic and a stochastic environment (please refer to the beginning of this section for their respective definitions). As shown in Figure 3, the gap between the deterministic case and the stochastic case narrows as time frame length increases.

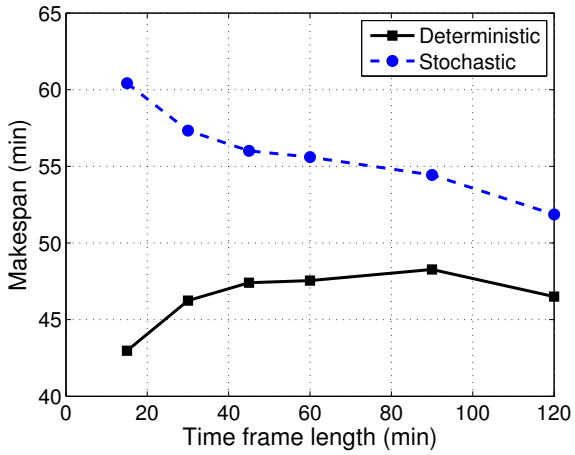

Figure 3. The average makespans in both deterministic and stochastic cases.

5.2.2 Impact of Time Period Aggregation on Performance

The performance of the resource allocation plan is measured by the average makespan of all agents and also the average agent cost (which includes tardiness cost and resource cost besides makespan cost). Makespans obtained under deterministic and stochastic cases can already been seen in Figure 3. Agent costs under deterministic and stochastic cases are plotted in Figure 4 and exhibit similar patterns.

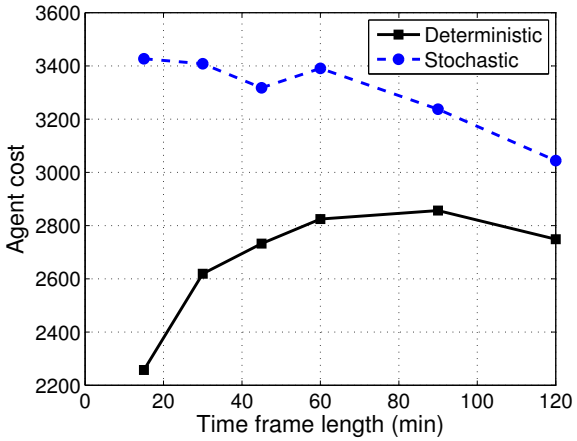

Figure 4. The average agent costs in both deterministic and stochastic cases.

As demonstrated in Figures 3 and 4, the performances of the allocation plans deteriorate as time frame length increases. This is expected because the deterministic models grow more constrained as time frame length increases. However, the improving trend of the performance in the stochastic cases suggests that for plans generated within the deterministic environment, longer time frames could indeed be helpful in stabilizing the fluctuations caused by unexpected disturbances. Although we are only showing the performance averages, this observation is generally true in all scenarios we tested (each scenario differs in agent's arrival times and deadlines).

6 Conclusion

An issue with allocating resources via auctions is the robustness of the resulting allocation under system stochasticity. By allowing bidders to purchases resources only when they are needed, the resulting allocations might be particularly sensitive to system stochasticity. We evaluate the premise that increasing the time frame length in which resources are auctioned off can accord stakeholders increased robustness in its solution quality.

To investigate this hypothesis, we assume bidders know only the mean resource requirements at the time of auction. Our numerical results show that increasing the time frame length when the time frame length is short (relative to system stochasticity) decreases the variability of both the mean cost and makespan, without increasing their mean values. Unfortunately, once the time frame length exceeds a certain threshold dependent on the variability in the individual processing times, further increases in the time frame length no longer bring the benefit of narrowing the performance gap between the deterministic and stochastic settings.

References

[1] J. Q. Cheng and M. P. Wellman. The WALRAS algorithm: A convergent distributed implementation of general-equilibrium outcomes. *Computational Economics*, 12:1–24, 1998.

[2] P. Cramton, Y. Shoham, and R. Steinberg, editors. *Combinatorial Auctions*. MIT Press, 2006.

[3] E. Kutanoglu and S. D. Wu. On combinatorial auction and lagrangean relaxation for distributed resource scheduling. *IIE Transactions*, 31(9):813–826, 1999.

[4] H. C. Lau, S.-F. Cheng, T. Y. Leong, J. H. Park, and Z. Zhao. Multi-period combinatorial auction mechanism for distributed resource allocation and scheduling. In *IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pages 407–411, 2007.

[5] R. P. McAfee and J. McMillan. Auctions and bidding. *Journal of Economic Literature*, 25(2):699–738, 1987.

[6] T. Sandholm. Algorithm for optimal winner determination in combinatorial auction. *Artificial Intelligence*, 135(1–2):1–54, 2002.

[7] I. E. Sutherland. A futures market in computer time. *Communications of the ACM*, 11(6):449–451, 1968.

[8] L. Walras. *Elements of Pure Economics*. 1874. English translation by William Jeffé, 1954.

[9] M. P. Wellman and P. R. Wurman. Market-aware agents for a multiagent world. *Robotics and Autonomous Systems*, 24:115–125, 1998.

[10] M. P. Wellman, W. E. Walsh, P. R. Wurman, and J. K. Mackie-Mason. Auction protocols for decentralized scheduling. *Games and Economic Behavior*, 35(1): 271–303, 2001.

[11] F. Ygge and H. Akkermans. Decentralized markets versus central control: A comparative study. *Journal of Artificial Intelligence Research*, 11:301–333, 1999.