STUDENT PAPER: Risk and Expectations in a-priori Time Allocation in Multi-Agent Contracting*

Alexander Babanov
Dept of Computer Science and Engineering
Dept of Economics
University of Minnesota
babanov@cs.umn.edu

John Collins
Dept of Computer Science and Engineering
University of Minnesota
jcollins@cs.umn.edu

Maria Gini
Dept of Computer Science and Engineering
University of Minnesota
gini@cs.umn.edu

ABSTRACT

In related research we have proposed a market architecture for multi-agent contracting and we have implemented prototypes of both the market architecture and the agents in a system called MAGNET. A customer agent in MAGNET solicits bids for the execution of multi-step plans, in which tasks have precedence and time constraints, by posting a Request for Quotes to the market. The Request for Quotes needs to include for each task its precedence constraints and a time window. In this paper, we study the problem of optimizing the time windows in the Requests for Quotes. Our approach is to use expected utility theory to reduce the likelihood of receiving unattractive bids, while maximizing the number of bids that are likely to be included in the winning bundle. We describe the model, illustrate its operation and properties, and discuss what assumptions on the market structure are required for its successful integration into MAGNET or other multi-agent contracting systems.

Categories and Subject Descriptors

K.4.4 [**Computers and Society**]: E-commerce; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence

General Terms

Algorithms, Economics, Theory

Keywords

Automated auctions, multi-agent contracting, expected utility, risk estimation

1. INTRODUCTION

*Tracking ID 249

Submitted to the First International Joint Conference on Autonomous Agents and Multi-Agent Systems, Bologna, Italy, July 2002

The MAGNET (Multi-AGent NEgotiation Testbed) [5] system is designed to support multiple agents in negotiating contracts for tasks with temporal and precedence constraints. We distinguish between two agent *roles*, the *Customer* and the *Supplier*. A Customer is an agent who needs resources outside its direct control in order to carry out his plans. A Supplier is an agent who, in response to a Request for Quotes (RFQ), may offer to provide the requested resources or services for specified prices, over specified time periods.

In this paper, we focus on the decision process a customer agent needs to go through in order to generate an RFQ. We study in particular the problem of how to specify in the RFQ the time windows for the different tasks. This decision determines an approximate schedule by setting limits on the start and end times for each individual task, since the RFQ includes early start and late finish times for each task.

Choosing appropriate time windows affects the number and price of the bids received, the ability to compose the bids into a feasible schedule, and the financial exposure of the customer agent. There are two major decisions here: the relative allocation of time among the different tasks in the plan, and the extent to which the time windows of adjacent tasks (connected by precedence relations) are allowed to overlap.

There is a tension between issuing an RFQ that will solicit the maximum number of bids and reduce costs, and one that will guarantee the feasibility of any plan constructed with the resulting bids [4]. We assume that suppliers will bid depending of their current resource commitments, and therefore larger time windows will result in more bids and better utilization of resources, which will result in lower prices. However, an RFQ with overlapping time windows will make the process of winner determination much more complex [3].

An additional factor to be considered is the financial exposure of the customer agent [2]. We assume non-refundable deposits are paid to secure awarded bids, and payments for each task are made as the tasks are completed. The payoff for the customer agent occurs only at the completion of the plan. Once a task starts and, in case it is successfully completed in the time period specified by the contract, the customer is liable for its full cost, regardless of whether in the meantime the plan as a whole has been abandoned due to a failure on some other branch of the plan.

We define successful plan execution as "completed by the deadline," and we define successful completion of a task as

"completed without violating temporal constraints in the plan." Note that a task can be completed successfully even if it is not finished within the duration promised by the bidder, as long as the schedule has sufficient slack to absorb the overrun. If a plan is completed after its deadline, it has failed, and we ignore any residual value to the customer of the work completed.

The uncertainty of whether the tasks will be completed on time as promised by suppliers further complicates the decision process. Because of the temporal constraints between tasks, failure to accomplish a task does not necessarily mean failure of the goal. Recovery might be possible, provided that whenever a supplier fails to perform or decommits there are other suppliers willing to do the task and there is sufficient time to recover without invalidating the rest of the schedule.

If a task is not completed by the supplier, the customer agent is not liable for its cost, but this failure can have a devastating effect of other parts of the plan. Having slack in the schedule increases the probability that tasks will be completed successfully or that there will be enough time to recover if one of the tasks fails. However, slack extends the completion time and so reduces the payoff. In made-to-order products speed is the essence and taking extra time might prevent a supplier from getting a contract. This complicates the selection of which bids to accept. The lowest cost combination of bids and the tightest schedule achievable is not necessarily the preferable schedule because it is more likely to be brittle.

Risk can also be reduced by consolidating tasks with fewer suppliers. Suppliers can bid on "packages" composed of subsets of tasks from the RFQ. In general, the customer is better off from a risk standpoint if it takes these packages, assuming that the supplier is willing to be paid for the whole package at the time of its completion. In some cases, the customer may be willing to pay a premium over the individual task prices in order to reduce risk. The advantage of doing this is greater toward the end of the plan than near the beginning, since at that point the customer has already paid a significant part of the tasks. Having a greater financial exposure provides an additional incentive to reduce risk.

In our previous work on Expected Utility [2] we were mostly concerned with computing the marginal expected utility of completing successfully all the tasks within the duration promised. In this paper, we show how to use Expected Utility Theory to determine the time windows for tasks in the task network, so that bids that are close to these time windows form the most preferred risk-payoff combinations for the customer agent. We further examine to what extent the behavior of the model corresponds to our expectations, explain what market information needs to be collected in order to integrate the model in MAGNET system and, finally, discuss how to use the resulting time allocations to construct RFQs.

2. THE MAGNET FRAMEWORK

2.1 General Terms

The *customer* is a human or artificial agent who wants to achieve some goal and needs resources or services beyond her direct control.

The *supplier* is a human or artificial agent who has direct control over some resources or services and may offer to provide those in response to external request, i.e., may submit and commit to bids.

The *mediator* is a MAGNET-assisted human agent who meets the needs of a consumer by negotiating over multiple goods or services with one or more suppliers. We often refer to the artificial part of the duo as to the *customer agent*.

The *Request for Quotes* is a signal composed by the customer agent on the basic of the customer's needs and is sent to solicit suppliers' bids. MAGNET is a mixed initiative system, so between composing RFQs and sending them out, there is a stage where a human user can impose her preferences on the RFQ choices.

The *task network* (see Figure 1) represents the structure of the consumer's plan. In essence, it is a connected directed acyclic graph.

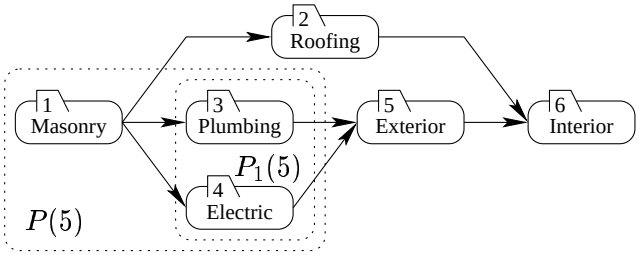

Figure 1: A task network example.

2.2 Task Network

Mathematically speaking, the task network is a tuple $\langle N, \prec \rangle$ of a set N of individual tasks and strict partial ordering on them. We conveniently abuse N to also denote the number of tasks.

We define $P(n) := \{m \in N | m \prec n\}$ to be a set of *predecessors* of $n \in N$, where every predecessor m should be completed before task n might start. Note that, in general, $P(n)$ is not completely ordered by $\prec$. We also define $P_1(n)$ to be a set of *immediate predecessors* of n.

Similarly, $S(n)$ and $S_1(n)$ are to denote a set of *successors* and a set of *immediate successors* of the task n respectively.

2.3 Time Allocation and Probabilities

The task network is characterized by a *start time* t^{s} and a *finish time* t^{f}. No individual task can be scheduled outside the interval $[t^{\mathrm{s}}, t^{\mathrm{f}}]$.

The placement of an individual task n on a schedule is, in turn, bounded by the following three parameters:

1. *Early start* time $t_n^{\mathrm{es}} \geq t^{\mathrm{s}}$, the task n cannot start earlier than that.
2. *Late start* time $t_n^{\mathrm{ls}} \geq t_n^{\mathrm{es}}$, the task n cannot start later than that.
3. *Late finish* time $t_n^{\mathrm{lf}} \geq t_n^{\mathrm{ls}}$, the task cannot finish later than that. Note, we assume no lower bound on the task finish time other than its actual start time, i.e. a supplier is free to work as fast as she wishes).

The actual placement of task n in the schedule is characterized by *start time* t_n^{s} and *finish time* t_n^{f}, where

$$t_n^{\mathrm{s}} \geq t_m^{\mathrm{s}} \geq t^{\mathrm{s}}, \qquad \forall m \in P_1(n)$$
$$t_n^{\mathrm{f}} \leq t_m^{\mathrm{s}} \leq t^{\mathrm{f}}, \qquad \forall m \in S_1(n)$$

The *probability of task n completion* by the time t, conditional on the successful task n completion, is distributed according to the cumulative distribution function (CDF) $\Phi_n = \Phi_n(t_n^{\mathrm{s}}; t)$, $\Phi_n(\cdot;\infty) = 1$. There is an associated unconditional *probability of success* $p_n \in [0,1]$ that characterizes the percentage of tasks that are successfully completed given infinite time.

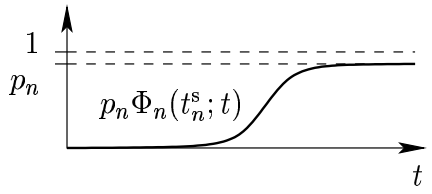

Figure 2: Unconditional distribution for successful completion probability.

2.4 Payoffs

Task n bears an associated *cost*. We split the total cost of task n in two parts: one is paid at time 0 and sums over n to c_0, the other is c_n and is due at some time after successful completion of n. We assume $c_0 = 0$ for the rest of the paper, since we never compare two plans with different c_0's.

There is a single *final payoff* V scheduled at the plan finish time t^{f} and paid conditional on all tasks in n being successfully completed by that time.

There is an associated *rate of return* q_n[1] that is used to calculate the *discounted present value* (PV) for payoff c_n due at time t as

$$\mathrm{PV}(c_n; t) := c_n (1+q_n)^{-t}.$$

We associate the return q with the final payoff V.

3. EXPECTED UTILITY

3.1 General Terms

The consumer agent's preferences are represented by a von Neumann-Morgenstern utility function u with constant absolute risk-aversion (CARA) coefficient r, where $r := u''/u'$. Thus, the *expected utility* [12] (EU) $Eu[\cdot]$ over the set of payoff-probability pairs, also referred as a *gamble*, $G = \{(x_i, p_i)_i\}$ s.t. $p_i > 0, \forall i$ and $\sum_i p_i = 1$ is

$$\begin{aligned} Eu[G] &:= \sum_{(x_i,p_i)\in G} p_i u(x_i) \\ &= -\sum_{(x_i,p_i)\in G} p_i \exp\{-r x_i\}. \end{aligned}$$

To make intuitive sense of the maximization criterion, we introduce the *certainty equivalent* (CE) of a gamble G as

$$\begin{aligned} \mathrm{CE}[G] &:= u^{-1} Eu[G] \\ &= \frac{-1}{r}\log Eu[G] \end{aligned}$$

One may think of a certainty equivalent as a payoff that is equally valuable as the corresponding gamble G for an agent with risk-aversion r. Naturally, the agent will not be willing to accept gambles with less than positive certainty equivalent and the higher values of the certainty equivalent will correspond to more attractive gambles.

To illustrate the concept, Figure 3 shows how the certainty equivalent depends on the risk-aversity of an agent. In this figure we consider a sample gamble that brings the agent either 100 or nothing with equal probabilities. Agents with positive r's are risk-averse, with negative — risk-loving. Note that agents with risk-aversity close to zero, i.e. almost risk-neutral[2], value G equal to its weighted mean 50,

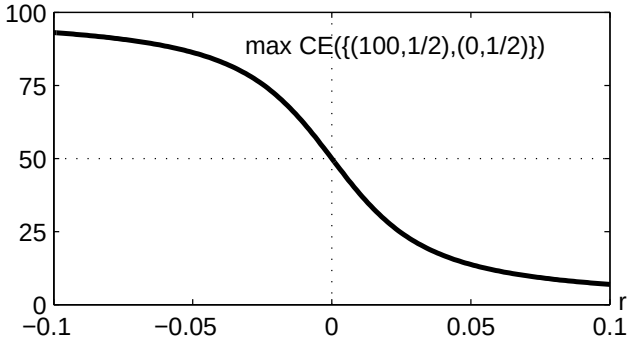

Figure 3: Certainty equivalent of a simple gamble as a function of the risk-aversity.

3.2 Cumulative Probabilities

To find the actual form of EU in our setup, we must be able to describe gambles that result from different task scheduling.

We assume that a payoff c_n is scheduled at t_n^{f}, so its present value $\tilde{c}_n$[3] is

$$\tilde{c}_n := c_n (1+q_n)^{-t_n^{\mathrm{f}}}$$

We define the conditional probability of task n success as

$$\tilde{p}_n := p_n \Phi_n\left(t_n^{\mathrm{s}}; t_n^{\mathrm{f}}\right).$$

We also define a set of *precursors* of task n as

$$\tilde{P}(n) := \left\{ m \in N | t_m^{\mathrm{f}} \le t_n^{\mathrm{s}} \right\}.$$

These are the tasks that were finished before task n had a chance to start.

The unconditional probability that the task n will be completed successfully is

$$\tilde{p}_n^{\mathrm{c}} = \tilde{p}_n \times \prod_{m\in\tilde{P}(n)} \tilde{p}_m.$$

That is, the probability of successful completion of every precursor and of the task n itself are considered independent events. The reason this is calculated in such form is because, if any task in $\tilde{P}(n)$ fails to be completed, there is no need to execute task n.

[1] The reason for having multiple q_n's is that individual tasks can be financed from different sources, thus affecting task scheduling.

[2] In the case of $r = 0$, the utility function takes the form of a straight line with a positive slope.

[3] Hereafter we "wiggle" variables that depend on the current task schedule, while omitting all corresponding indices for the sake of simplicity.

The probability of receiving the final payment V is therefore

$$\tilde{p} = \prod_{n \in N} \tilde{p}_n.$$

3.3 Example and Discussion

To illustrate the definitions and assumptions above, let's return to the task network in Figure 1 and consider a sample task schedule in Figure 4. In this figure the x-axis is the time in the plan, the y-axis shows both the task numbers and for each individual task it also shows the cumulative distribution of the unconditional probability of completion in the scale from 0 to 1 (compare to Figure 2). Circle markers show start times t^{s}_n, crosses — both finish times t^{f}_n and success probabilities $\tilde{p}_n$ (numbers next to each point). Square markers denote that the corresponding task cannot span past this point due to precedence constraints. Finally, the thick part of each CDF shows the time allocation for this task.

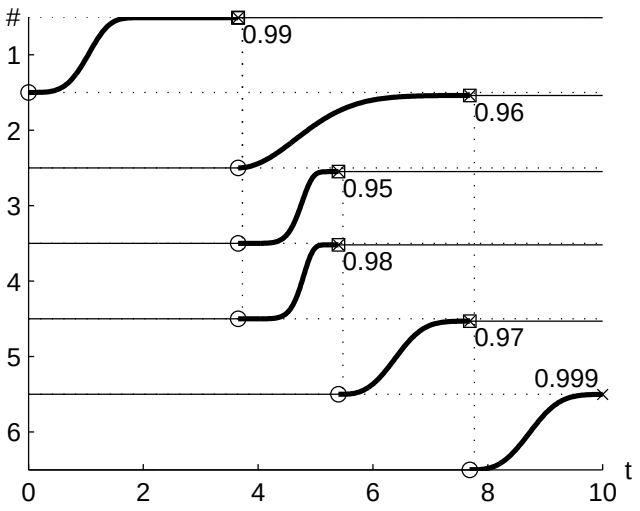

Figure 4: CE maximizing time allocations for the plan in Figure 1 for $r = -0.01$.

In practice, the consumer agent needs a way of collecting the market information necessary to build and use the model. The reason why we introduced the cumulative probability of success Φ_n and probability of success p_n, instead of the average project life span or probability of failure or alike, is because the probability of success is relatively easy to observe in the market.

To be specific, the information that the agent needs to collect is the empirical distribution of how long does it take from the point of starting some task to the point its completion is reported. This data, unlike the data on failures or actual positions in the supplier's schedule is less likely to be private or unobservable.

For example, it would be easy for a person sitting in a Chinese restaurant to determine p_n's and Φ_n's for various dishes. The time between ordering and receiving the dish is easily observable. The customer does not need to worry about obtaining detailed (and private) information such as the exact cooking time or the number of times the cook made a mistake and started over.

4. MAXIMIZATION

4.1 Gamble Calculation Algorithm

At this point we built up enough instruments to express the expected utility function form for a given task network and schedule. In the course of the research we found that writing a formal description of the expected utility as a function of gambles is overly complicated and relies on the order of task completions. At the same time, we found a simple recursive algorithm that creates these gambles. This algorithm is shown in Figure 5.

Procedure: $G \leftarrow$ **calcGamble**(T, D)
Require: T "tasks to process" $\vee$ D "processed tasks"
Ensure: G "subtree gamble"

for all $n \in T$ **do**
  if $\tilde{P}(n) \subset D$ **then**
    $G \leftarrow \varnothing$
    $T \leftarrow T \setminus \{n\}$
    $E \leftarrow$ **calcGamble**(T, D) "follow $\ldots \rightarrow \bar{n}$ path"
    for all $(x, p) \in E$ **do**
      $G \leftarrow G \cup \{(x, p \times (1 - \tilde{p}_n)\})$
    end for
    $I \leftarrow$ **calcGamble**$(T, D \cup \{n\})$ "follow $\ldots \rightarrow n$ path"
    for all $(x, p) \in I$ **do**
      $G \leftarrow G \cup \{(x + \tilde{c}_n, p \times \tilde{p}_n)\}$
    end for
    Return: G "subtree processed"
  end if
end for
"only reach to here if in the leaf"
if $N = D$ **then**
  Return: $\{(V, 1)\}$ "all done"
else
  Return: $\{(0, 1)\}$ "failure"
end if

Figure 5: The algorithm for gamble calculation.

In the first call the algorithm receives a "todo" task list $T = N$ and a "done" task list $D = \varnothing$, all the subsequent calls are recursive. To illustrate the idea behind this algorithm, let's refer to the payoff-probability tree in Figure 7. This tree was built for the time allocations in Figure 6 and reflects the precursor relations for this case.

Looking at the time allocation figure, we note that with probability $1 - \tilde{p}_1$ task 1 fails, the consumer agent does not pay or receive anything and stops the plan execution (path $\bar{1}$ in the tree). With probability $\tilde{p}^{\mathrm{c}}_1 = \tilde{p}_1$ the agent proceeds with task 3 (path 1 in the tree). In turn, task 3 either fails with probability $\tilde{p}_1 \times (1 - \tilde{p}_3)$, in which case the agent ends up stopping the plan and paying a total of c_1 (path $1 \rightarrow \bar{3}$), or it is completed with the corresponding probability $\tilde{p}^{\mathrm{c}}_3 = \tilde{p}_1 \times \tilde{p}_3$.

In the case where both 1 and 3 are completed, the agent starts both 2 and 4 in parallel and becomes liable for paying c_2 and c_4 respectively even if the other task fails (paths $1 \rightarrow 3 \rightarrow 2 \rightarrow \bar{4}$ and $1 \rightarrow 3 \rightarrow \bar{2} \rightarrow 4$). We will stop at this point with the final observation, that, if both 2 and 4 fail, the resulting path in the tree is $1 \rightarrow 3 \rightarrow \bar{2} \rightarrow \bar{4}$ and the corresponding payoff-probability pair is framed in the figure.

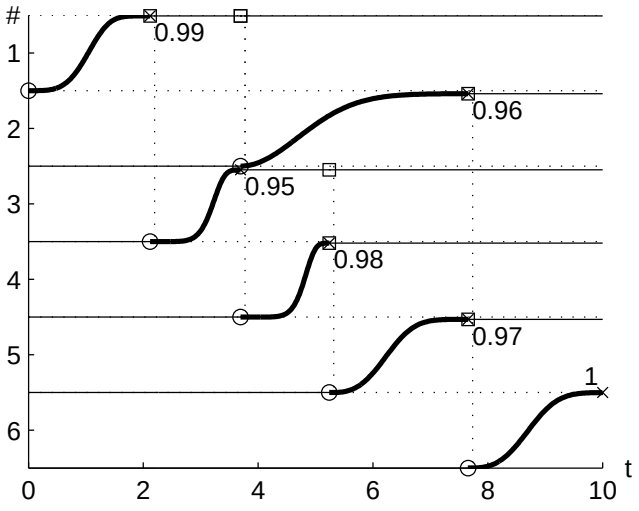

Figure 6: CE maximizing time allocations for the plan in Figure 1 for $r = 0.02$.

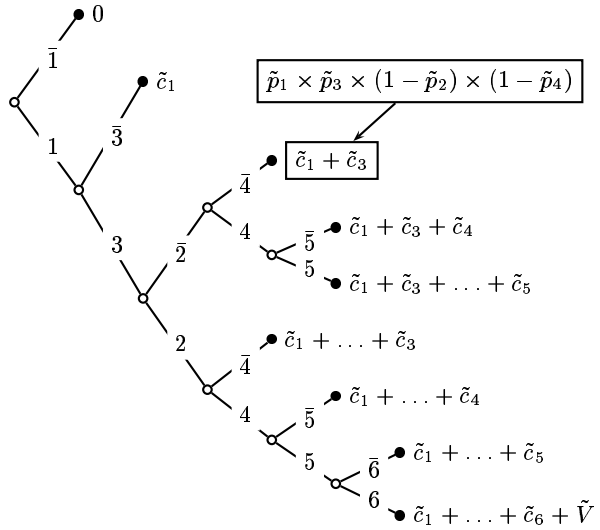

Figure 7: Payoff-probability tree for the time allocations in Figure 6.

4.2 Experimental Results

We have conducted a series of experiments on the CE maximization. Some of the results are summarized in Figure 8. In this figure, the y-axis shows 11 different risk-aversity r settings, the bottom x-axis — time t in the plan, and the top x-axis — maximum CE value for each r setting. The rounded horizontal bars in each of 11 sections denote time allocations for each of six tasks with task 1 being on top. Sections $r = -0.01$ and $r = 0.02$ correspond to Figure 4 and Figure 6 respectively. Finally, the vertical bars show the maximum CE values.

Let's examine the relative placement of time allocations as a function of r. For this purpose we highlighted task 3 (black bars) and task 4 (white bars). Here task 3 has higher variance of CDF and lower probability of success than task 4 (0.032 and 0.95 vs. 0.026 and 0.98), also task 3 is more expensive (-15 vs. -7). There are four different cases in the experimental data:

1. Risk-loving agents schedule tasks in parallel in order to maximize their final payoff. This confirms the intuition that we derive from Figure 3 – risk-lovers lean toward high risky payoffs rather than toward low certain payoffs.

2. Risk neutral and low risk-averse agents place risky task 3 first to make sure that the failure doesn't happen too far in the project. Note, that they still keep task 2 running in parallel, so, in case 2 fails, they are liable for paying the supplier of task 4 on success. One can consider those agents as somewhat optimistic.

3. Moderately risk-averse agents try to dodge the situation above by scheduling task 3 after task 2 is finished. These agents are willing to accept the plan, but their expectations are quite pessimistic.

4. Highly risk-averse agents shrink task 1 interval to zero, thus "cheating" to avoid covering any costs. One may interpret this as a way of signaling a refusal of accepting the plan.

5. ISSUES AND FUTURE RESEARCH

5.1 Multiple Local Maxima

One of the issues that arise from the CE maximization problem is the presence of multiple local maxima even in cases where task networks are fairly simple. The reason for this is that the relative task placement has two preferred configurations: independent individual tasks can be either performed in parallel (thus increasing probability of successful completion) or they can be scheduled in sequence to minimize overall payments, in case one of tasks fails.

To illustrate the issue, we constructed a sample task network with two parallel tasks. Task 1 has a higher variance of completion time probability and lower probability of success than task 2, everything else is the same. The resulting graph of CE is shown in Figure 9. There are 3 local maxima in this figure: one in the left side that corresponds to the task two being scheduled first in sequential order, another on the right side corresponding to the task one being first, and yet another one in the furthermost corner of the graph representing both tasks being scheduled at time 0 and executed in parallel.

In the course of the research, we were able to resolve this issue by running the maximization procedure from different start points. However, one may note that the number of possible start points grows considerably with the complexity of the task network and the algorithm that checks each and every one of them is not scalable. One of the solutions we are likely to pursue is to continuously run the maximization from different starting points and stop the procedure when the average marginal gain in CE equates to the marginal cost of the time. We will devote further efforts to the study of this and other possibilities in our future research.

5.2 Computational Complexity

The computational complexity of the maximization procedure is determined by two parameters: first, the procedure itself is a non-linear maximization over $2N$ choice variables with internal precedence constraints. Second, to calculate a

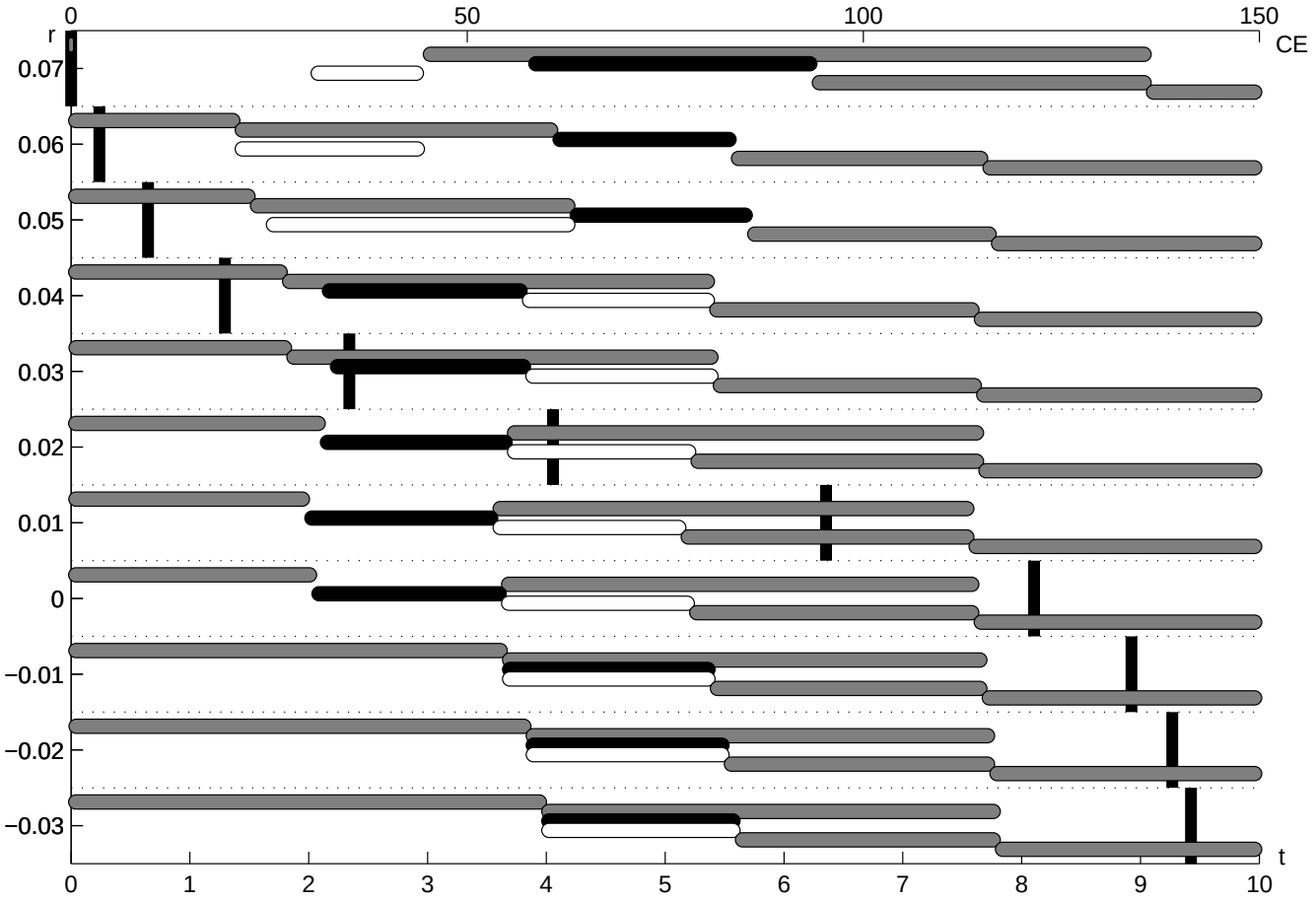

Figure 8: CE maximizing schedules and CE values for the plan in Figure 1 and $r \in [-0.03, 0.07]$.

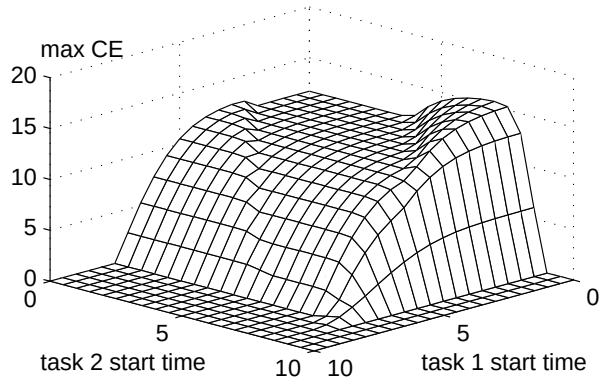

Figure 9: Local maxima for two parallel tasks.

certainty equivalent value for every time schedule, the maximization procedure should be able to build a corresponding gamble and find its expected utility.

While the realm of the non-linear maximization is well-studied and offers a fair selection of ready-to-use techniques[4], effective algorithms for the second part are not nearly as well researched. At this point, we know the direct approach in the form of the algorithm in Figure 5 and can estimate its complexity as at most $O\left(2^{K-1} \times N\right)$, where K is the maximum number of tasks that are scheduled to be executed in parallel.

The complexity estimate is based on the observation that the depth of the payoff-probability tree is N and that any subtree following an unsuccessful task execution has a depth of no more than $K-1$. The last statement follows from the assumption that there are no more than $K-1$ tasks running in parallel to the one that failed and therefore no other tasks will start after the failure was reported. Whether it is possible to create an algorithm with significantly lower computational costs is one of the questions we plan to address in future research.

Note that in the real commercial projects the ratio K/N is usually low, since not many of these exhibit the high degree of the parallelism. Also, the experimental results in Section 4.2 allow us to conclude that K/N ratio is likely to be lower for the risk-averse agents (presumably, businessmen) than for the risk-lovers (gamblers). These two considerations may relax the need for a faster algorithm and thus will be examined in the first place.

5.3 Slack Allocation in RFQ

The last issue we want to address in this paper is how to use the CE maximization-based time allocation procedure

[4]In particular, we use the Nelder-Mead simplex (direct search) method from the `Matlab` optimization toolbox.

to construct RFQs in the MAGNET framework. The CE maximizing schedule itself contains information on what is the most desirable task scheduling for the customer agent. However, it is hard to imagine that there will always be bids that cover exactly the same time intervals as in the maximizing schedule.

We suggest the following approach: first, let's specify what percentile α of the maximum CE value is considered acceptable by the agent. Define the start time interval for the task n as all values of t_n^{s}, such that for the schedule that differs from the maximizing one only in the start time of the task n the value of CE is no less than α of the maximum. Graphically, this process is represented by building the projection of the CE α-percentile graph (see Figure 10) on the task n time axis. Further assume there is only one continuous interval of t_n^{s} values for every $n \in N$ and denote it as $\left[t_n^{\mathrm{s}-}, t_n^{\mathrm{s}+}\right]$. Finally, submit the interval $\left[t_n^{\mathrm{s}-}, t_n^{\mathrm{f}} + \left(t_n^{\mathrm{s}+} - t_n^{\mathrm{s}}\right)\right]$, where t_n^{s} and t_n^{f} are times from the maximizing schedule, as a part of the RFQ.

The suggested way of composing the RFQ from the CE maximizing schedule is an intuitive consequence of the CE maximization idea. Nevertheless, there are many open questions concerning this approach, e.g., there could be more than one interval $\left[t_n^{\mathrm{s}-}, t_n^{\mathrm{s}+}\right]$ for some tasks, so we need to distinguish them in the RFQ composition. Also, it is possible that the acceptable CE percentile must be lowered for some tasks in the task network, due to the low bid submission rate. It is even possible, that the RFQ needs to be split in two or more parts, so that the requests for rare goods and services are submitted first and the rest of RFQ is composed after the bids for those rare products are received.

Although we do not specifically address the above mentioned and related issues in the current paper, the CE maximization approach promises to be powerful and flexible enough to help us resolve those in our future research.

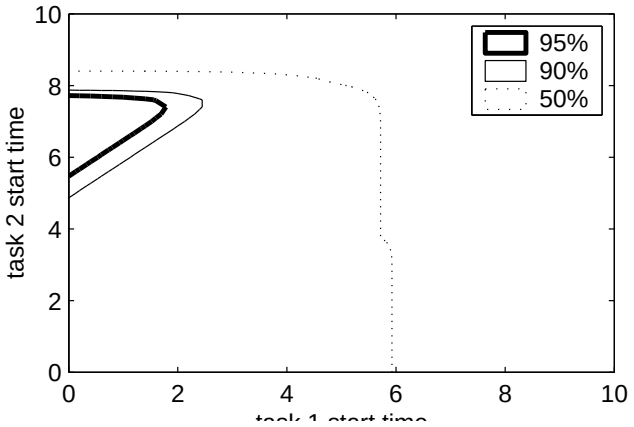

Figure 10: Contours of some α-percentile graphs for the CE graph in Figure 9.

6. RELATED WORK

Auctions are becoming the predominant mechanism for agent-mediated electronic commerce [7]. AuctionBot [20] and eMEDIATOR [17] are among the most well known examples of multi-agent auction systems. The determination of winners of combinatorial auctions [13] is hard. Dynamic programming [16] works for small sets of bids, but does not scale and imposes significant restrictions on the bids. Algorithms such as Bidtree [17] and CASS [6] reduce the search complexity, but their criterion to select bids is just price. Our bids include a time window for each task, and so bid selection cannot be separated from scheduling.

A set of optimal and approximate methods, along with a test set for algorithm evaluation, was published by Fujishjima et al [6]. Hoos and Boutilier[8] describe a stochastic local search approach to solving combinatorial auctions, and characterize its performance with a focus on time-limited situations. A key element of their approach involves ranking bids according to expected revenue; it's hard to see how this could be adapted to the MAGNET domain with temporal and precedence constraints, and without free disposal. Andersson et al [1] describe an Integer Programming approach to the winner determination problem in combinatorial auctions. Nisan [14] extended this with an analysis of bidding languages for combinatorial auctions. More recently, Sandholm [18] has described an improved winner-determination algorithm called CABOB that uses a combination of linear programming and branch-and-bound techniques. It is not clear how this technique could be extended to deal with the temporal constraints in MAGNET, although the bid-graph structure may be of value.

Expected Utility Theory [15] is a mature, yet controversial, field of Economics, that attracted many supportive as well as critical studies, both theoretical [10, 11] and empirical [19, 9]. We believe that the expected utility and related concepts will attract more attention in the relation to automated auctions soon, in particular, because they suggest a practical way of describing risk estimations and temporal preferences.

7. ACKNOWLEDGMENTS

Partial support for this research is gratefully acknowledged from the National Science Foundation under award NSF/IIS-0084202.

8. REFERENCES

[1] A. Andersson, M. Tenhunen, and F. Ygge. Integer programming for combinatorial auction winner determination. In *Proc. of 4th Int'l Conf on Multi-Agent Systems*, pages 39–46. IEEE Computer Society Press, July 2000.

[2] J. Collins, C. Bilot, M. Gini, and B. Mobasher. Decision processes in agent-based automated contracting. *IEEE Internet Computing*, pages 61–72, March 2001.

[3] J. Collins and M. Gini. An integer programming formulation of the bid evaluation problem for coordinated tasks. In B. Dietrich and R. V. Vohra, editors, *Mathematics of the Internet: E-Auction and Markets*, volume 127 of *IMA Volumes in Mathematics and its Applications*, pages 59–74. Springer-Verlag, New York, 2001.

[4] J. Collins, M. Tsvetovat, R. Sundareswara, J. V. Tonder, M. Gini, and B. Mobasher. Evaluating risk: Flexibility and feasibility in multi-agent contracting. In *Proc. of the Third Int'l Conf. on Autonomous Agents*, May 1999.

[5] J. Collins, B. Youngdahl, S. Jamison, B. Mobasher, and M. Gini. A market architecture for multi-agent contracting. In *Proc. of the Second Int'l Conf. on Autonomous Agents*, pages 285–292, May 1998.

[6] Y. Fujishjima, K. Leyton-Brown, and Y. Shoham. Taming the computational complexity of combinatorial auctions: Optimal and approximate approaches. In *Proc. of the 16th Joint Conf. on Artificial Intelligence*, 1999.

[7] R. H. Guttman, A. G. Moukas, and P. Maes. Agent-mediated electronic commerce: a survey. *Knowledge Engineering Review*, 13(2):143–152, June 1998.

[8] H. H. Hoos and C. Boutilier. Solving combinatorial auctions using stochastic local search. In *Proc. of the Seventeen Nat'l Conf. on Artificial Intelligence*, pages 22–29, 2000.

[9] B. Jullien and B. Salanie. Estimating preferences under risk: The case of racetrack bettors. *The Journal of Political Economy*, 108(3):503–530, June 2000.

[10] M. J. Machina. Choice under uncertainty: Problems solved and unsolved. *The Journal of Economic Perspectives*, 1(1):121–154, 1987.

[11] M. J. Machina. Dynamic consistency and non-expected utility models of choice under uncertainty. *The Journal of Economic Literature*, 27(4):1622–1668, December 1989.

[12] A. Mas-Colell, M. D. Whinston, and J. R. Green. *Microeconomic Theory*. Oxford University Press, January 1995. ISBN: 0195073401.

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

[14] N. Nisan. Bidding and allocation in combinatorial auctions. In *Proc. of ACM Conf on Electronic Commerce (EC'00)*, pages 1–12, Minneapolis, Minnesota, October 2000. ACM SIGecom, ACM Press.

[15] J. W. Pratt. Risk aversion in the small and in the large. *Econometrica*, 32:122–136, 1964.

[16] M. H. Rothkopf, A. Pekeč, and R. M. Harstad. Computationally manageable combinatorial auctions. *Management Science*, 44(8):1131–1147, 1998.

[17] T. Sandholm. Approaches to winner determination in combinatorial auctions. *Decision Support Systems*, 28(1-2):165–176, 2000.

[18] T. Sandholm, S. Suri, A. Gilpin, and D. Levine. Cabob: A fast optimal algorithm for combinatorial auctions. In *Proc. of the 17th Joint Conf. on Artificial Intelligence*, Seattle, WA, USA, August 2001.

[19] V. K. Smith and W. H. Desvousges. An empirical analysis of the economic value of risk changes. *The Journal of Political Economy*, 95(1):89–114, February 1987.

[20] P. R. Wurman, M. P. Wellman, and W. E. Walsh. The Michigan Internet AuctionBot: A configurable auction server for human and software agents. In *Second Int'l Conf. on Autonomous Agents*, pages 301–308, May 1998.