UNIVERSITY OF MINNESOTA

This is to certify that I have examined this bound copy of a master thesis by

Alexander Babanov

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Maria Gini

Name of Faculty Advisor

Signature of Faculty Advisor

Date

GRADUATE SCHOOL

Design and Testing of Methods for Scheduling Tasks with Precedence Constraints to Solicit Desirable Bid Combinations

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Alexander Babanov

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

Maria Gini, Advisor

July 2003

Dedication

To the memory of my father:

wish you were here

Acknowledgments

I owe this work to my teachers, advisors and friends.

First and foremost my greatest debt of gratitude is to my advisors. Professor Andy McLennan introduced me to the beautiful science of Game Theory, and offered his support and understanding when they mattered most. Professor Maria Gini generously shared her trust and knowledge, and devoted an unparalleled amount of time to guiding and encouraging me. She has been the best advisor one could have wished for, masterfully finding a balance between the greatness of vision and the perils of implementation.

I'm grateful to all former and current members of the MAGNET research team. Although I never met some of them, their contribution to the project mattered a lot, and I wish to thank them for that. My special thanks goes to John Collins, who helped me in more ways than he can possibly guess. His expert supervision of the project was indispensable, and his insight in problems was greatly inspirational. I would also like to thank Wolf Ketter for always being open for discussion and willing to help. I wish that he gets well and does not miss my Ph.D. defense, whenever it is to come. Maria, John, Wolf and I co-authored papers that paved the foundation of this thesis, so it is a product of their efforts as well as mine.

The National Science Foundation has supported our work in part under grants NSF/IIS-0084202 and NSF/EIA-9986042. Without NSF generous funding, the progress of this project would be slowed down significantly.

I would like to extend my gratitude to the faculty of Economics and Computer Science and Engineering Departments for their hard work and great patience. Their knowledge and the scientific rigor of the many anonymous reviewers of our papers were crucial in building and shaping this work. Thanks to Professor Mats Heimdahl for reviewing a draft of the thesis and partaking in my committee.

Finally, and most importantly, I'm obliged to my wife Ulyana and my son Nikita for their ever lasting love and understanding.

Abstract

We consider issues arising in applications of networks of intelligent software agents to electronic markets. Our domain of interest are problems involving making decisions on plans that consist of multiple tasks, with task ordering and temporal constraints. Our goal is to provide intelligent agents with the ability to reason about risks and gains in the uncertain and ever changing market environment. To support our theoretic derivations, we suggest building a large-scale evolutionary experimentation framework for experimenting with agents' strategies, and obtaining data for testing and improving the developed algorithms.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis concerns issues arising in applications of multi-agent systems to electronic markets. We consider problems involving making decisions on plans that consist of multiple tasks, with task ordering and temporal constraints. Our goal is to provide intelligent agents with the ability to reason about risks and gains in the uncertain and ever changing market environment.

1.1 State of the Market

Following the development of online services over the last few years, one may argue with certainty that internet-based tools facilitating customer-to-customer, business-to-customer, and business-to-business transactions are here to stay.

Online auctions, such as eBay, uBid and Amazon.com zShops, allow individuals and businesses to expose their goods, in however small quantities, to the immense audience of potential buyers. At the same time they provide buyers with a vast selection of items to choose from. For example, eBay lists several millions of auctioned items on a single day. Services, like eBay owned PayPal, InternetCash, and Escrow.com, help small-scale sellers with financial and security options.

The rise of online retail shops, marketplaces, and auctions is matched by the development of search engines, rating, and recommender systems. Big shops and auctions, like Amazon.com, Yahoo!, and eBay, offer on-site search and seller rating capabilities. Independent companies, such as mySimon, Epinions, PriceWatch and PriceGrabber,

serve as search engines, recommender, and rating systems both for businesses and traded goods. Closer to the business side, Net Perceptions offers support for collecting and converting transaction data into by-customer product recommendations.

Most importantly, some companies work on providing complex solutions, combining several services and products in one package or chain. An example of such approach is travel, hotel, and car rental matchmaking from Travelocity and Expedia. While planning a trip using their services, a customer has a choice of multiple hotels, carriers, and rental companies — bundling is unobtrusive, if existent. A more sophisticated auction-based solution is developed by CombineNet. Their approach is to create an auction mechanism supplied with an expressive language for describing multi-item offers and bids, as well as with a constrained optimization solver to match such language.

1.2 Impeding Issues

As well-received and convenient as they seem to be, the existing electronic markets are limited by a handful of specific efficiency and scalability issues. To consider one example, participating in online auctions takes a fair amount of buyer's time. Searching for the right match is time-consuming, since most details of each possible match, together with the rules of transaction, are hidden in a free-form description. Bidding often requires a close supervision, since a private value assigned to the item based on its description may vary greatly depending on the other bidders' behavior.

To summarize issues with the current online marketplaces:

1. In many cases market exploration is a burdensome task for a customer. This alone takes away much of the customer's benefit from being exposed to a large society of candidate suppliers.

2. Selling and buying of goods that complement or substitute each other are severely limited. In fact, on eBay many of such transactions are handled informally by studying seller's other auctions and directly negotiating new conditions. To arrange a transaction involving more than two parties with such means is virtually impossible.

3. Transactions involving scheduling of services are mostly limited to travel and entertainment industries, which enjoy a highly predictable nature of plane flights, concerts, and sport events. Yet, the scheduling of contingency plans is a rare (and expensive) feat even in these industries.

The successful operation of companies listed in the previous section suggests that these issues are not specially limiting for simple customer-to-customer and, in some cases, business-to-customer interactions. Our concern though is with the agile made-to-order businesses that strive to improve the efficiency by using internet technologies.

The made-to-order business model is not novel to the market. Some of examples are: car and computer customization at final assembly, subcontracting auctions held by construction companies, and previously mentioned trip planning by travel agencies. It is important to note that all the listed examples are highly domain specific: customization options are only possible for massively produced and standardized products; construction auctions involve a limited number of well-known bidders; travel agencies rely on the uniform and predictable nature of their inputs.

The restrictive property of the made-to-order model is its reliance on a tight coordination across multiple self-interested suppliers. In order to shorten the time to delivery, the involvement of warehouses is minimized, and supply chains become longer and more convoluted. Dynamic supply chain building, automated market exploration and preliminary risk assessment, logistic planning and sophisticated auction mechanisms are tools developed by the academic community that might help opening the current business model to a wide variety of markets. Our research, in particular, is focused on the involvement of the networks of intelligent agents in facilitating human decision processes.

1.3 State of the Art

Intelligent software agents can significantly ease customer's decision processes by performing preliminary market exploration, filtering away non-essential data, even making decisions on their own in selected cases. On the supplier's side agents might help by monitoring the demand and dynamically adjusting the prices of supplier's goods and services to market changes. In either case, agents need methods for making decisions under uncertain and changing market conditions (see [Kephart *et al.*, 2000,

Kephart and Greenwald, 2001] for an analysis of pricing strategies).

Automated auctions [Maes *et al.*, 1999], especially those allowing multiunit [Krishna, 2002] and combinatorial [de Vries and Vohra, 2001] bid formulations, can deliver large savings considering that logistics costs often account for as much as 30% of a total cost. The range of applications that employ auctions and intelligent agents spans from agent-mediated auctions [Wellman *et al.*, 2001], to learning of pricing and supply management strategies [Cliff, 2001, Stone *et al.*, 2002], to distributed train scheduling [Parkes and Ungar, 2001], even to guiding of cooperative multi-robot systems [Gerkey and Matarić, 2002, Zlot *et al.*, 2002].

The agent-based automation of the dynamic supply-chain management has been under development during recent years [Veeramani and Joshi, 1998, Sadeh *et al.*, 1999]. Our research group develops algorithms and technologies for supply-chain applications in the auction-based multi-agent framework MAGNET (Multi-AGent NEgotiation Testbed) [Collins *et al.*, 2001, Collins *et al.*, 2002]. We adopt the combinatorial auction formulation and extend it over existing approaches [Hunsberger and Grosz, 2000, Wellman *et al.*, 2003] to the problem of soliciting resources to fulfill composite plans with task interdependencies and scheduling constraints.

1.4 Research Framework

The research effort presented in this thesis is based on the assumptions and achievements of the MAGNET project. The MAGNET system, or testbed, is designed to support multiple intelligent software agents in negotiating contracts for tasks with complex temporal and precedence constraints (see Figure 1.1).

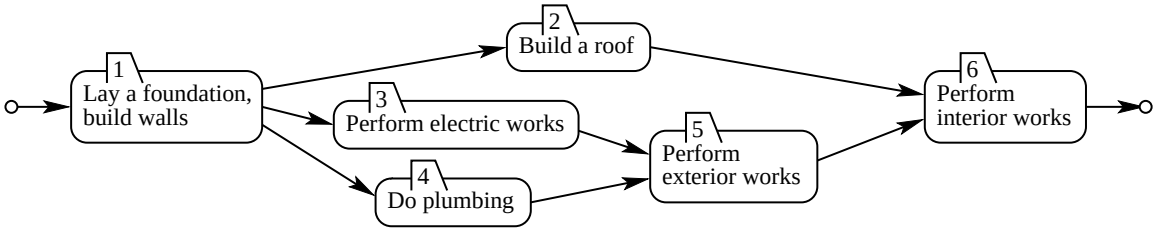

Figure 1.1: Sample agent's plan: a simplified process of house building.

We distinguish between two agent roles, the *customer* and the *supplier*[1]. A cus-

[1]The suggested distinction does not prevent an agent from accepting both roles, e.g., to compose

tomer needs resources outside its direct control in order to carry out its plans. It proceeds by soliciting the help of other self-interested agents, suppliers, by issuing a *request for quotes* (RFQ). In response to a customer's RFQ, a supplier might bid to provide one or more of the requested resources or services for a specified lump sum price, over specified time intervals. Upon receiving bids from suppliers, the customer solves a *winner-determination problem* [Sandholm *et al.*, 2002] and awards the best combination of bids.

To formalize the financial side of the process we assume that the customer makes one payment to a supplier after awarding its bid, and another payment upon successful completion of all tasks covered in this bid. If all tasks in the customer's plan are completed successfully, the customer gets a final reward. However, if one or more tasks fail, the customer does not get any payment, and is still liable for those tasks that are in progress[2].

The objective of the agents is to maximize their (or rather their owners') profits while predicting and managing the financial risk exposure. This task is complicated by a staggering number of factors that influence the performance of an agent. In the rest of the introduction and, in essence, in the bulk of this work, we concentrate on the decision problem of the customer agent. We cover a few of the most important factors, and suggest a way of testing a viability of the proposed solutions in a large-scale evolutionary simulation.

1.5 Customer Agent's Problem

In MAGNET a customer agent resolves its resource limitations by running a first-price sealed-bid reverse auction. The timeline of one negotiation and execution session is shown in Figure 1.2. In our research we focus on the procurement part of the process for its theoretic value.

The procurement phase consists of three stages. In the first stage, the customer agent analyses a plan that it needs to fulfill, and the current state of the market in which it operates; upon completing the analysis it forms and sends an RFQ to potential suppliers. In the second stage, supplier agents decide whether to respond to the customer agent's RFQ. In the third stage, the customer agent solves the winner-

several resource commitments and resell the resulting package in a secondary market.

[2]We will discuss a possibility of plan repair in Section 2.7.2.

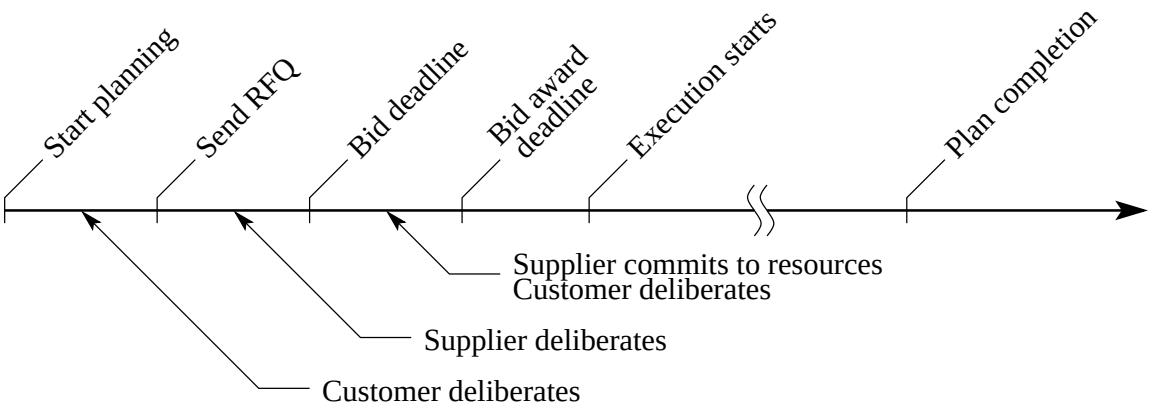

Figure 1.2: The timeline of a single negotiation and execution session. The first three intervals constitute the procurement phase.

determination problem, while those supplier agents who submitted bids stand ready to deliver if their bids are awarded.

The final stage of the procurement is its most time-stressed part. During this stage all bidding suppliers should maintain some level of resource commitment. One might reasonably expect that the faster the customer agent announces its decisions, the lower prices it will, ultimately, face. Under a close examination the speed of a customer agent's decision process reflects on many more parameters of the system than just the prices.

Figure 1.3 brings together factors that are most relevant to our discussion, as well as cause-effect relationships between them. In this figure we distinguish between three types of concepts: those that were studied by the members of our research group previously (or are trivially derived using other concepts), those that we discuss in this work, and those that we plan to cover in our future research. We label every relationship with $\uparrow\uparrow$ symbol when two concepts change, in general, in the same direction, and with $\uparrow\downarrow$ if the opposite is true.

Note that the figure shows only the most direct relationships between the concepts. For example, although the speed of the winner-determination (WD in the figure) clearly impacts the customer's bottom line, we prefer to break this relationship in a chain of direct causes and effects. In this particular case one of possible chains is the following: if the speed of a winner-determination routine increases, the supplier's cost of committing to bids falls, and so does the price of the winning bundle, hence

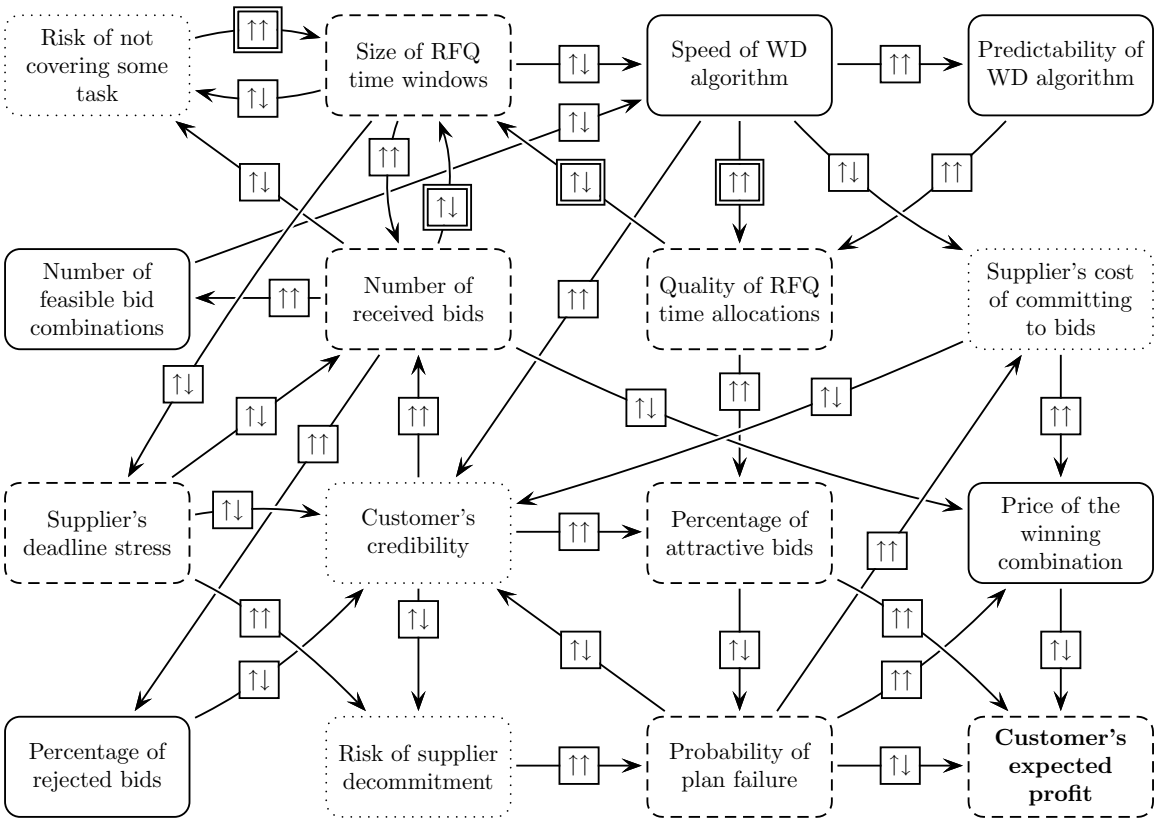

Figure 1.3: Major factors contributing to a customer agent's decision process, and cause-effect relationships between them. Solid boxes frame concepts that were studied previously; dashed boxes show concepts that we consider in this work; dotted boxes denote concepts that we plan to cover in future. Symbol ↑↑ means that two related concepts commonly change in the same direction; symbol ↑↓ shows that they move in opposite directions.

a customer's expected profit grows.

Of the concepts in Figure 1.3, the number of feasible combinations, the speed of multiple winner-determination algorithms, and the predictability of algorithms' run-time were thoroughly studied by other members of the MAGNET group (refer to [Collins, 2002] for the comprehensive roundup of the findings). In brief, four conceptually different winner-determination algorithms were designed, and examined for their performance and the predictability of the results. The algorithms are based on simulated annealing, integer programming, A* and IDA* methodologies. As of this moment we are not aware of any alternative algorithms capable of operating on bids for tasks with temporal and precedence constraints. All four algorithms, however, rely on

cost-minimization as on the sole criterion of customer's preferences.

The concepts that we address in this work stem from the need to augment the existing customer-side algorithms with the notions of risk and expected profit. We also devote special attention to the question of how our theoretical reasonings can be assessed in the face of the scarcity of the real-world data on combinatorial auctions.

1.6 Overview of the Thesis

The main body of this thesis consists of two parts: in Chapter 2 we propose to use the Expected Utility Theory as a basis for risk estimation in the MAGNET framework, in Chapter 3 we suggest a way of testing and improving agents' strategies in a large-scale evolutionary simulation.

Chapter 4 lays out questions that were not addressed in sufficient detail, and discusses possible research directions. Finally, Chapter 5 covers related research, and Chapter 6 concludes with the summary of contributions.

Chapter 2

Risk Assessment in Customer Agent's Problem

2.1 Introduction

In this chapter we describe a way of constructing optimal RFQs to reduce the likelihood of receiving unattractive bids, while maximizing the percentage of bids that are likely to be awarded. In the outlook, such approach allows a customer agent to try minimizing the number of solicited bids, thus decreasing the percentage of rejections, improving the agent's credibility in the suppliers' eyes, and, in the long run, enhancing profit expectations (refer to Figure 1.3).

We assume that a customer agent's plan is given in the form of a *task network* — a collection of tasks and transitive precedence relations between them. A customer agent's RFQ specifies preferred time windows for tasks in the task network (see Figure 2.1).

An RFQ is not specifically constrained by the precedence relations[1]. In fact, the agent might benefit from specifying overlapping time windows in situations where offers for one or more tasks are rarely available in the market, like, for example, it is the case with roofing services after a major hailstorm. Conversely, specifying larger time windows for tasks that are expected to attract many bids is not beneficial — only one bid will be accepted by a winner-determination routine, the rest will just

[1] There are some commonsense rules that still apply, e.g., specifying in RFQ that a task can start before its predecessors start does not bring any useful bids. Indeed, each extra solicited bid will lead to a violation of feasibility conditions when bundled with bids covering preceding tasks.

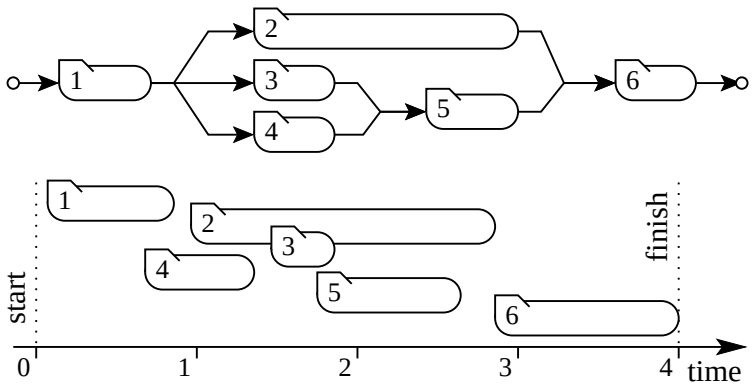

Figure 2.1: Formalized version of the house construction plan from Figure 1.1 (top) with a sample RFQ (bottom). Time windows for tasks in RFQ are not restricted by precedence relations.

needlessly slow down the routine, and discontent suppliers who spend resources to prepare their bids. In essence, giving overly large windows in RFQ is a modern way of crying wolf — in the outcome it's the customer's reputation that suffers.

Finding the right combination of time windows thus is a paramount goal in preparing an RFQ. The current mechanism of RFQ generation in MAGNET uses a variation of the Critical Path Method (CPM) [Hillier and Lieberman, 1990]. Our CPM-based approach proved to be useful in creating test problems for various winner-determination algorithms [Collins, 2002], yet it has its problems, and is hardly applicable to real-life situations.

The algorithm we are looking for should improve upon the CPM approach with cost-minimization criterion in, at least, the following areas:

1. It should derive required inputs from market data that are relatively easy to collect, better yet, are publicly available.

2. It must account for risks, such as: risks due to supplier's decommitment, risks due to supplier's failure to deliver, and risks due to cascading failures to start tasks whose predecessor task was not finished on time.

3. Finally, it should produce RFQs that balance the sizes of time windows to maximize the percentage of *desirable* bids. Here by "desirable" we imply bids that can feasibly be bundled together in a low-cost combination with a high probability of successful completion.

Figure 2.2 shows a speculative example of how an algorithm can adjust time windows to ensure a higher fraction of desirable bid combinations. Each of four parts of this figure corresponds to some choice of RFQ windows for two related tasks, and shows all possible combinations of expected bids in a grid. Gray cells of the grid show desirable combinations, a white circle indicates the best combination of expected bids. The sequence of figures demonstrates the process of shifting and resizing the time windows, and cutting away bids with a low "desirability potential," to increase the probability of a desirable outcome. The ratio of desirable combinations to the total number of possibilities is displayed in each case.

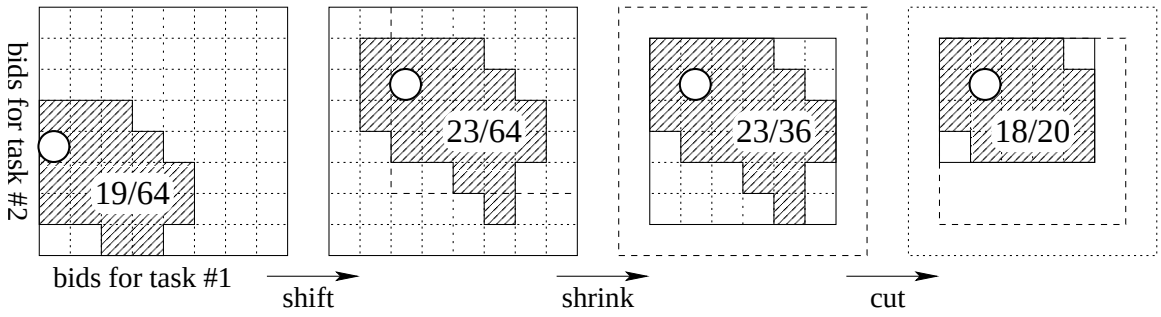

Figure 2.2: Speculative illustration of RFQ adjustment to solicit higher percentage of desirable bid combinations. Gray cells show desirable combinations, a white circle corresponds to the best combination.

We do not expect an algorithm to actually go through a step-by-step process of adjusting time windows, mainly because there are no actual bids to adjust to. Our approach is to define a reasonable concept of bid's desirability, and use it to directly select time windows in an RFQ. We refer to the RFQs selected based on their desirability as to *rational* RFQs.

In the rest of this chapter we proceed with deriving a measure of RFQ's worthiness based on the expected utility formulation; then we discuss maximization issues that arise from using the derived criterion; in the conclusion we discuss possible extensions to our formulation.

2.2 Expected Utility Formulation

2.2.1 Time, Payments and Probabilities

A task network is a connected directed acyclic graph, where individual tasks, plan start and finish points are vertices, and precedence relations are edges (Figure 2.1). It is convenient to disregard the start and finish, as well as all edges connected to them. A task network then can be expressed as a tuple $\langle N, \prec \rangle$ of N tasks and a strict partial ordering on them, such that for any tasks i and j in N, $i \prec j$ implies that i immediately precedes j. We use N to stand for a set of tasks and for their number.

We denote a plan *start time* as t^{s} and a *finish time* as t^{f}. The placement of task i in the schedule is characterized by a *task start time* t_i^{s} and a *task finish time* t_i^{f}, subject to the precedence constraints:

$$t^{\mathrm{s}} \leq t_j^{\mathrm{f}} \leq t_i^{\mathrm{s}}, \qquad \forall j \in P_1(i) \tag{2.1}$$

$$t_i^{\mathrm{f}} \leq t_j^{\mathrm{s}} \leq t^{\mathrm{f}}, \qquad \forall j \in S_1(i) \tag{2.2}$$

Here $P_1(i)$ is the a set of *immediate predecessors* of i, $P_1(i) = \{j \in N \mid j \prec i\}$. $S_1(i)$ is defined similarly to be the set of *immediate successors* of task i.

The probability of task i completion by time t, conditional on its successful completion given infinite time, is distributed according to the cumulative distribution function $\Phi_i = \Phi_i\left(t_i^{\mathrm{s}}; t\right)$, $\lim_{t\to\infty} \Phi_i\left(t_i^{\mathrm{s}}; t\right) = 1$. We define Φ_i to be explicitly dependent on the start time t_i^{s}. To see the rationale, consider the probability of successful delivery for letters that were mailed to the same address on different days of a week. One may rightfully expect that a letter sent on Monday has higher probability to be delivered in two days than the one that was sent on Friday.

To account for tasks that fail no matter how much extra time is given to the supplier, we associate the unconditional *probability of success* p_i with the percentage of tasks that are successfully completed, given infinite time. We then define unconditional probability of task i completion by time t as the product of p_i and Φ_i (Figure 2.3). In our experiments, we assumed a Weibull probability distribution for Φ_i, however the form of the distribution is not tied in the theory. In fact, we expect that the success probabilities would be derived from the available market information.

We assume the total cost of task i to the customer consists of two parts. The first

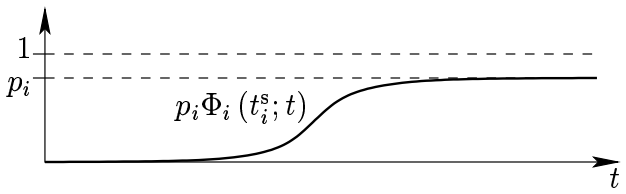

Figure 2.3: Unconditional distribution for successful completion probability.

part is a deposit paid when the bid is accepted to secure supplier's commitment. The second is a *cost* c_i due at task finish time t_i^{f} if the task is successfully completed. Both amounts are derived from the market statistics and do not change between different schedules. In the following derivations we omit deposits from the consideration, since they are paid unconditionally at a fixed time and thus do not contribute to our contingency analysis. We further consider c_i to be a single value — market average[2].

For each cost c_i there is an associated *rate of return* q_i that is used to calculate the *discounted present value* PV at time t as

$$\mathrm{PV}(c_i;t) := c_i(1+q_i)^{-t} \tag{2.3}$$

There is a single *final reward* V, paid as soon as all tasks in N are successfully completed, i.e. at time $\tilde{t}^{\mathrm{f}} = \max_i t_i^{\mathrm{f}}$. We associate the rate of return q with the final payment.

In general, the probabilities of success and the present values influence scheduling in opposite directions. The more time is given to suppliers, the higher are the customer's odds of receiving the final reward. Contrary, the difference between the present value of the reward and the present values of all payments to suppliers is likely to decrease with the length of the schedule. In the next section we suggest a way to quantify the relation between payments and probabilities.

2.2.2 Expected Utility and Certainty Equivalent

We assume that the customer agent's preferences over payments can be represented by the Bernoulli *utility function*[3] u. We further assume that the Arrow-Pratt *coefficient*

[2]We discuss a possible relaxation of this restriction later in Section 2.7.3 to include supplier price distributions.

[3]This and related definitions are due [Mas-Colell *et al.*, 1995, pp. 168–194].

of absolute risk aversion $r(x) := -u''(x)/u'(x)$ of the utility function is defined and constant for any value of the argument x. We choose u satisfying the assumptions above as follows:

$$u(x) = \begin{cases} -e^{-rx} & \text{for} \quad r \neq 0 \\ x & \text{for} \quad r = 0 \end{cases} \tag{2.4}$$

It is imperative to note that we will not compare utility values directly; the counter-intuitive (i.e. decreasing in monetary terms) form of the utility for $r < 0$ is a tradeoff for simple notation.

We assume that a future state of the world is described by the set S of all possible events. Each event $s \in S$ happens with a non-zero probability and results in some monetary payoff. We define a *lottery* to be a set of payoff-probability pairs L,

$$L = \{(x_s, p_s)\} \qquad \text{s.t.} \qquad p_s > 0 \quad \text{and} \quad \sum p_s = 1 \tag{2.5}$$

The expectations of the utility values over a lottery L are captured by the von Neumann-Morgenstern *expected utility function*:

$$\mathrm{E}u[L] := \sum_{(x_s, p_s) \in L} p_s u(x_s) \tag{2.6}$$

The *certainty equivalent* CE of a lottery L is defined as the single payoff whose utility matches the expected utility of the entire lottery, i.e. $u\left(\mathrm{CE}[L]\right) := \mathrm{E}u[L]$. Under our assumptions,

$$\mathrm{CE}[L] = \begin{cases} -\frac{1}{r} \log \sum_{(x_s, p_s) \in L} p_s e^{-r x_s} & \text{for} \quad r \neq 0 \\ \sum_{(x_s, p_s) \in L} p_s x_s & \text{for} \quad r = 0 \end{cases} \tag{2.7}$$

Certainty equivalent values represent payments in certain and current money. Naturally, one will not accept a lottery with a negative certainty equivalent, and higher values of the certainty equivalent will correspond to more attractive lotteries. For this reason, we choose to compare different schedules based on their certainty equivalents rather than on hardly intuitive expected utility values.

Figure 2.4 illustrates the concept of certainty equivalent by showing how it depends

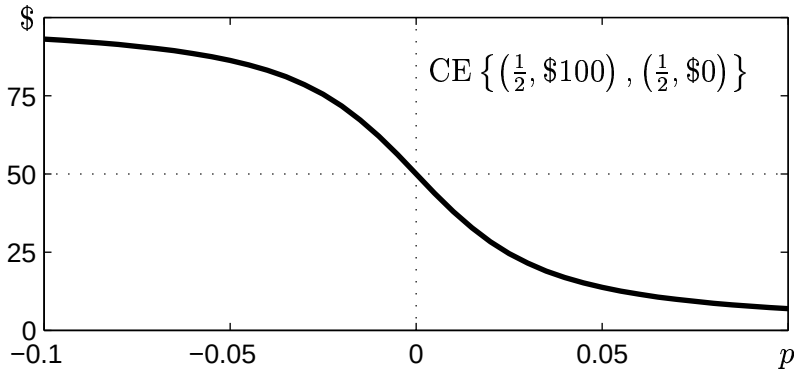

Figure 2.4: Certainty equivalent of a simple lottery as a function of a risk aversity.

on the risk aversity r of agents. In this figure we consider a lottery that yields either \$100 or nothing with equal probabilities. Agents with positive r values are risk averse, and as such are willing to accept smaller certain payments in exchange for the right to play the lottery. Agents with zero risk aversity are risk neutral, they equate the lottery to its weighted mean of \$50. Those with negative values of r are risk loving.

2.2.3 Payoff-Probability Pairs

To find the certainty equivalent of a lottery we need to compute all payoff-probability pairs constituting this lottery. Each event corresponds to a different set of failed and completed tasks. We assume that once some task is failed, the agent cancels all tasks that were not yet started. Each task in progress is allowed to proceed until its scheduled finish time, and is paid for on success.

The payoff c_i for task i is scheduled at time t_i^{f}, therefore its present value $\tilde{c}_i$[4] is

$$\tilde{c}_i := c_i \left(1 + q_i\right)^{-t_i^{\mathrm{f}}} \tag{2.8}$$

The conditional probability that task i succeeds when given a chance to start is determined by its scheduled finish time:

$$\tilde{p}_i := p_i \Phi_i(t_i^{\mathrm{s}}; t_i^{\mathrm{f}}) \tag{2.9}$$

The probability of successful completion of every task i precursor and of task i itself

[4]Hereafter we "wiggle" variables that depend on the current task schedule.

are considered independent events. The unconditional probability that task i will be completed successfully is

$$\tilde{p}_i^{\mathrm{c}} := \tilde{p}_i \times \prod_{j \in \tilde{P}(i)} \tilde{p}_j, \tag{2.10}$$

where $\tilde{P}(i)$ is a set of the *precursors* of task i — all tasks that finish before task i starts in the schedule,

$$\tilde{P}\,(i) := \{\, j \in N \;\big|\; t_j^{\mathrm{f}} \leq t_i^{\mathrm{s}} \,\} \tag{2.11}$$

The probability of receiving the final reward V is equal to the probability that all tasks in N are completed:

$$\tilde{p} = \prod_{i \in N} \tilde{p}_i \tag{2.12}$$

2.2.4 Example and Discussion

We illustrate the definitions above on the example of the task network in Figure 2.1 by considering two sample schedules shown in Figure 2.5. In this figure the x-axis is time, the y-axes show both the task numbers and cumulative distributions of the unconditional probability of completion (compare to Figure 2.3). Circle markers show start times t_i^{s}. Crosses indicate both finish times t_i^{f} 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. The thick part of each CDF shows the time allocated to each task, i.e. an RFQ window.

The customer agent needs a way of collecting the market information necessary to build and use the probability model. The probability of success is relatively easy to observe in the market. This is the reason for introducing the cumulative probability of success Φ_i and probability of success p_i, instead of, e.g., the average project life span, or a probability of failure, or a hazard rate, etc. Indeed, it is rational for a supplier to report a successful completion immediately in order to maximize the present value of a payment. It is equally rational *not* to report a failure until the very last moment, due to the possibility of earning the payment by rescheduling, outsourcing, or fixing the problem in some way.

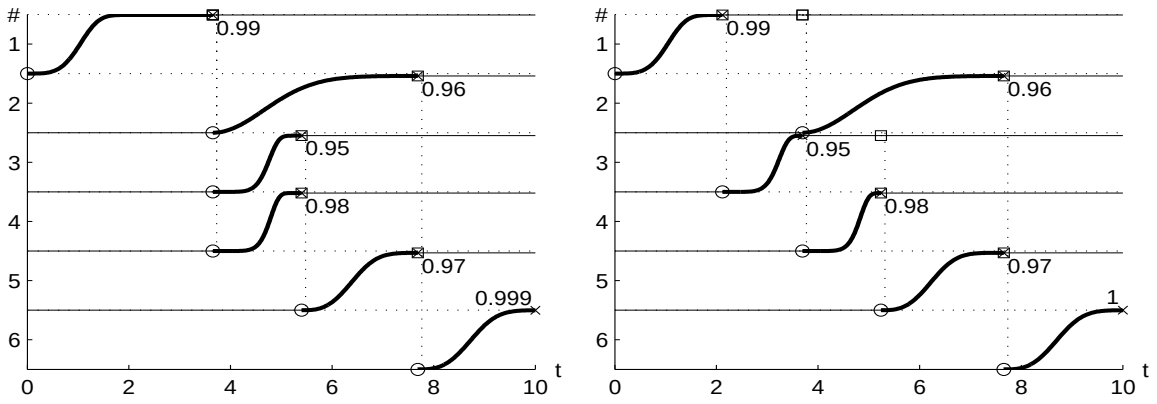

Figure 2.5: CE maximizing time allocations for the plan in Figure 2.1 for $r = -0.01$ (left) and $r = 0.02$ (right).

For example, it is possible for a curious observer to approximate p_i's and Φ_i's for various dishes in a restaurant — the time between ordering and receiving a dish is easily perceived. The observer does not need to worry about obtaining detailed (and private) information such as: exact cooking times, a workflow, and a number of attempts it took a novice cook to finally get a tricky dish right.

2.2.5 Event Tree

Figure 2.6 displays two lotteries corresponding to schedules in Figure 2.5 as event trees. The left tree is more convoluted, since its corresponding schedule has tasks 3 and 4 executed in parallel, whereas in the right schedule these tasks are sequential, and also task 3 precedes task 2.

Examining the first task in the right tree, we note that with probability $1 - \tilde{p}_1$ task 1 fails, the customer agent does not pay or receive anything and stops the execution (path $\bar{1}$ in the tree). With probability $\tilde{p}_1^{\mathrm{c}} = \tilde{p}_1$ the agent proceeds with task 3 (path 1 in the tree). Before task 3 is completed, the agent starts task 4. If task 3 fails, the agent is liable for paying c_1 and c_4 (paths $1 \to \bar{3} \to 4$) or c_1, if task 4 also fails (path $1 \to \bar{3} \to \bar{4}$). In turn, if task 3 succeeds, the agent starts task 2 before task 4 is completed. If both tasks 4 and 2 fail, the resulting path in the tree is $1 \to 3 \to \bar{4} \to \bar{2}$ and the corresponding payoff-probability pair is framed in the figure.

Writing the explicit description of the certainty equivalent as a function of lotteries is overly complicated and relies on the order of task completions. Instead we propose

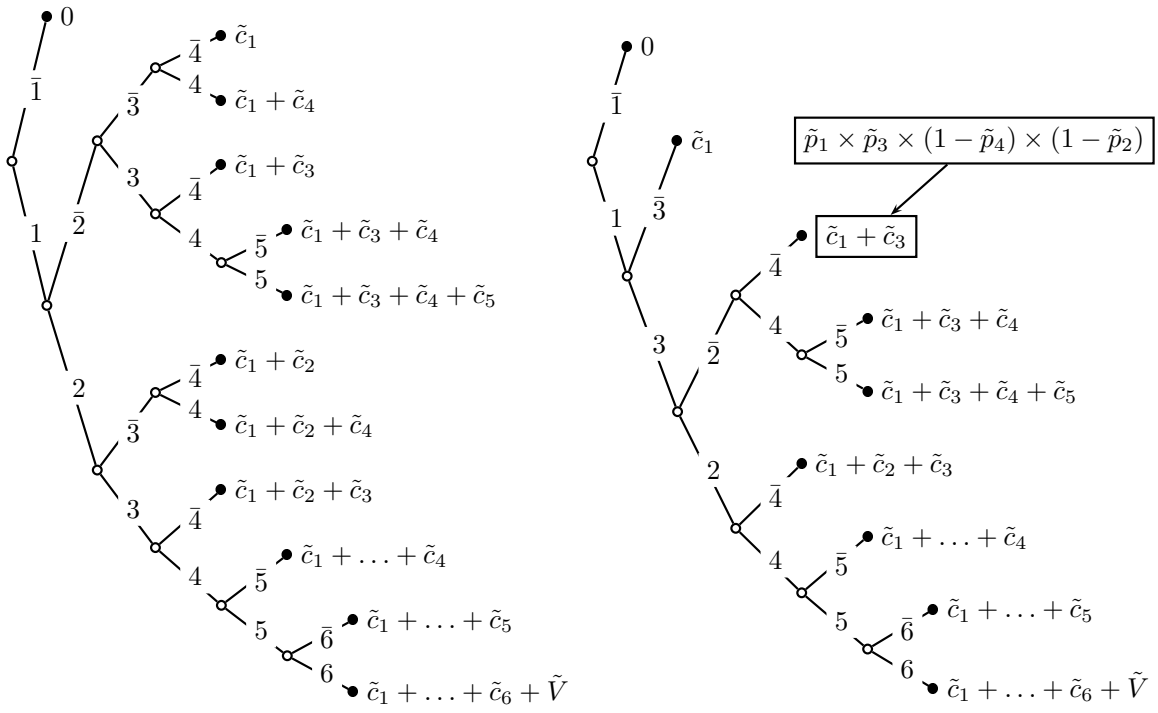

Figure 2.6: Payoff-probability trees corresponding to two schedules in Figure 2.5. Branch x corresponds to successful completion of task x, branch $\bar{x}$ — to failure.

a simple recursive algorithm that calculates these lotteries (see Algorithm 1). We then maximize the certainty equivalent over the space of all feasible schedules and the corresponding lotteries.

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. Each call of the algorithm corresponds to one node in an event tree. If this node is a leaf the algorithm returns a certain event, otherwise it branches a tree by The operation of the algorithm is best illustrated by examining event trees in Figure 2.6.

2.2.6 Maximization Issues

The complexity of **calcGamble** algorithm is $O\left(N2^{K-1}\right)$, 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 is due to the assumption that there are no more than $K-1$ tasks running in parallel to the one that failed, hence no other tasks will start

Algorithm: $G \leftarrow$ **calcGamble**(T, D)
  Requires: T "*tasks to process*", D "*processed tasks*"
  Returns: G "*subtree gamble*"

$M \leftarrow \{m \in T | \tilde{P}(m) \subset D\}$
if $M \neq \varnothing$ "*it's a branch*"
  $n \leftarrow$ ***first***$\{M\}$ "*according to some ordering*"
  $T \leftarrow T \setminus \{n\}$
  $G \leftarrow \varnothing$
  $E \leftarrow$ **calcGamble**(T, D) "*follow* $\ldots \rightarrow \bar{n}$ *path*"
  forall $(x, p) \in E$
    $G \leftarrow G \cup \{(x, p \times (1 - \tilde{p}_n)\})$
  endfor
  $I \leftarrow$ **calcGamble**$(T, D \cup \{n\})$ "*follow* $\ldots \rightarrow n$ *path*"
  forall $(x, p) \in I$
    $G \leftarrow G \cup \{(x + \tilde{c}_n, p \times \tilde{p}_n)\}$
  endfor
  return G "*subtree is processed*"
else "*it's a leaf*"
  if $N = D$ "*all tasks are done*"
    return $\{(V, 1)\}$
  else "*some task failed*"
    return $\{(0, 1)\}$
  endif
endif

Algorithm 1: Event tree based algorithm for computing payoff-probability pairs.

after the failure.

Reducing the complexity of **calcGamble** is critical, since it will be executed in the inner loop of any CE maximization procedure, unless we somehow depart from the event tree representation. We assert that for commercial projects the ratio K/N is likely to be low, since not many of them exhibit a high degree of parallelism. Our preliminary experiments allow us to conclude that the K/N ratio is lower for risk-averse agents (presumably, businessmen) than for risk-lovers (gamblers alike). These two considerations may reduce the need for a faster algorithm, though additional work to improve the algorithm is planned.

Another major issue related to the CE maximization is the presence of multiple local maxima of CE even in cases where task networks are fairly simple. The large number of local maxima is caused mainly by two factors: major variations due to different

ordering of tasks that significantly impact CE values[5] and small variations due to different scheduling of tasks that are not stressed by time. Any two tasks that are not ordered by the precedence constraints can be scheduled in three ways: parallel and two sequential. Scheduling tasks in parallel increases the probability of successful completion, while sequential scheduling minimizes overall payments whenever one of the 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 a lower probability of success than task 2; everything else is the same. The resulting graph of CE is shown in Figure 2.7. There are 3 local maxima with positive CE values in this figure: one on the left side corresponds to task 2 being scheduled first in sequential order; another on the right side corresponds to task 1 being first; yet another one in the furthermost corner of the graph represents both tasks being scheduled to start at time 0 and executed in parallel. The number of local maxima grows considerably (in the worst case scenario — exponentially) with the number of the tasks that are not constrained by precedence relations.

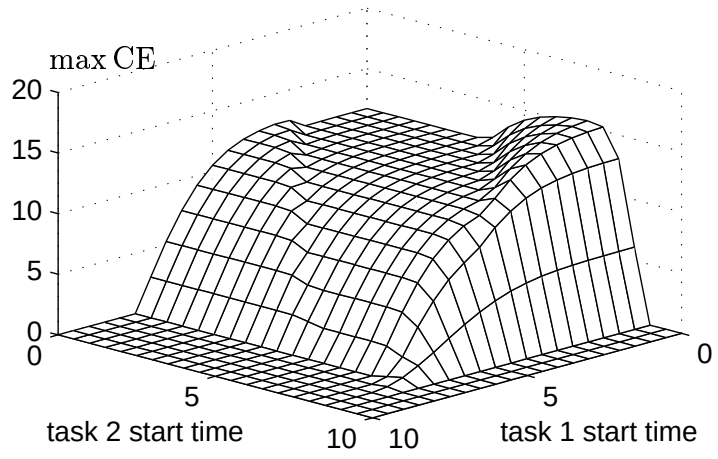

Figure 2.7: Local maxima for two parallel tasks.

We approach the problem of CE maximization in the presence of multiple local maxima as follows: in the next section we suggest a way to enumerate all possible task orderings, so that it is possible to near-globally optimize CE by examining each and every of those. The suggested maximization method is extremely computationally

[5]One may think of these tasks as if they are on the critical path, but this analogy is misleading since in our formulation the impact of a task on CE value depends on a schedule.

expensive. However it can be usefully employed when studying the properties of the problem domain, which is what we use it for in Section 2.4. In Section 2.5 we observe that it is possible to improve the performance of a global optimization by considering maximum intervals of continuity of CE function. Finally, in Section 2.7.1 we outline a stochastic maximization algorithm based on the ideas from the Genetic Algorithms and Simulated Annealing maximization approaches.

2.3 Counting of Task Orderings

In a given task network, there may be many different ways to order the individual tasks that are consistent with the precedence relationships and yet result in different payoff-probability trees. We divide task networks into two categories: reducible, which can be reduced to a single equivalent task by recursively merging sequential and parallel tasks, and irreducible. Examples of reducible and irreducible task networks are shown in Figure 2.8.

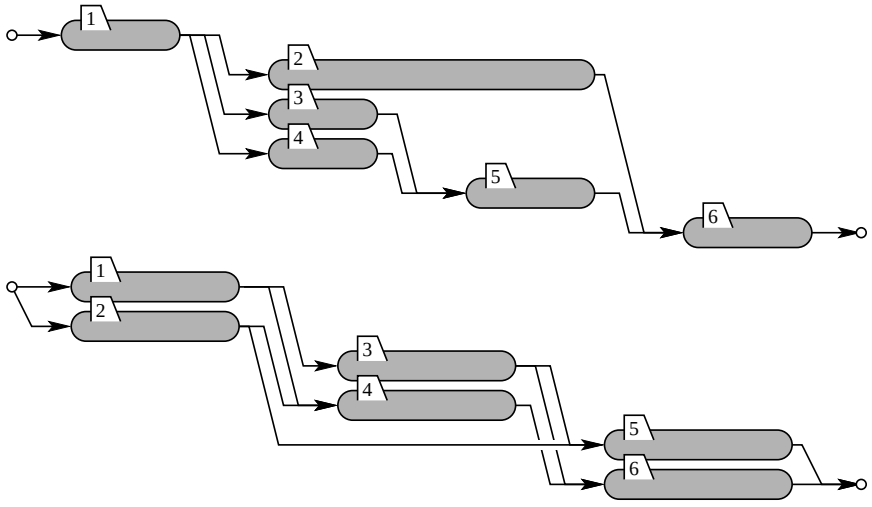

Figure 2.8: Examples of reducible (top) and irreducible (bottom) task networks. The reducible task network has the same structure as in Figures 1.1 and 2.1.

To remove ambiguity in distinguishing between task orderings, we also assume that no two task start and/or finish times can be exactly equal. Under this assumption schedules where some start and finish times coincide will belong to two or more task orderings.

We start by considering simple cases of reducible networks and build up an apparatus

to enumerate task orderings for arbitrary task networks, whether they are reducible or not.

2.3.1 Counting in Reducible Networks

Consider first a task network that consists of k parallel sequences of N_i, $i = 1, \ldots, k$ tasks each. We associate a set of $2N_i$ balls with each of k sequences, one ball to denote the start time of some task and another to denote its finish time. Since inside each sequence the order of task start and finish times is uniquely defined by precedence relations, we label all $2N_i$ balls in each sequence by the number i (see Figure 2.9(a)).

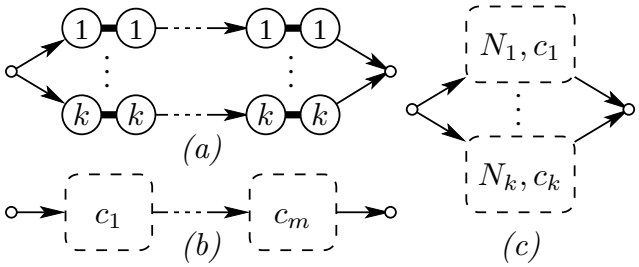

Figure 2.9: Counting scenarios.

The number of orderings $C^{(a)}$ of task start and finish times corresponds to the number of orderings of k sets of balls, i.e.

$$C^{(a)} = \frac{\left(2\sum_{i\in N} k_i\right)!}{\prod_{i\in N}(2k_i)!} \tag{2.13}$$

The number $C^{(b)}$ of orderings for m sequential task networks of an arbitrary (not necessarily reducible) configuration, where each network has c_j, $j = 1, \ldots, m$ possible orderings by itself (Figure 2.9(b)) is clearly

$$C^{(b)} = \prod_{j=1}^{m} c_j \tag{2.14}$$

Finally, we count the number $C^{(c)}$ of orderings in the case of k parallel task networks of arbitrary configuration, where each network i, $i = 1, \ldots, k$ consists of N_i tasks with c_i possible combinations on them (Figure 2.9(c)). To do this we first enumerate the possible orderings assuming that the order of task start and finish times is fixed

Algorithm: $C^{\forall} \leftarrow$ **calcOrderings**(T, D)
  Requires: T "*started tasks*", D "*finished tasks*"
  Returns: $C^{\forall}$ "*number of orderings*"
$X \leftarrow \{i \in N | P_1(i) \subset D, i \notin T \cup D\}$ "*tasks to start*"
if $X \cup T = \varnothing$ "*all tasks are done*"
  $C^{\forall} \leftarrow 1$
else "*can start or finish some tasks*"
  $C^{\forall} \leftarrow 0$
  foreach $i \in X$ "*start task* i"
    $C^{\forall} \leftarrow C^{\forall} +$ **calcOrderings**$(T \cup \{i\}, D)$
  endfor
  foreach $i \in T$ "*finish task* i"
    $C^{\forall} \leftarrow C^{\forall} +$ **calcOrderings**$(T \setminus \{i\}, D \cup \{i\})$
  endfor
endif
return $C^{\forall}$

Algorithm 2: Counting of task orderings in irreducible task networks.

inside each network i, and then account for the number of ways to fix the order independently in each task network i, i.e. c_i,

$$C^{(c)} = \frac{\left(2\sum_{i=1}^{k} n_i\right)!}{\prod_{i=1}^{k}(2n_i)!} \prod_{i=1}^{k} c_i \tag{2.15}$$

2.3.2 Counting in Irreducible Networks

For the cases where either it is impossible to reduce a task network or, in general, when one wants a universal way of enumerating orderings of task start and finish times in arbitrary networks, we propose Algorithm 2. It starts by assuming that no task was started or finished yet (i.e. $T = D = \varnothing$), and proceeds recursively by examining cases when one of tasks that can be started next is started or one of the tasks that have been started already is completed.

2.3.3 Examples

Let's consider a few examples of using the counting methods, starting with the reducible task network in Figure 2.8. First we find the number of orderings for two

parallel tasks 3 and 4 to be $C^{(a)}\{3,4\}$, i.e. six[6]. Adding task 5 to subnetwork $\{3,4\}$ does not change the number of orderings: $C^{(b)}(\{3,4\},5) = 6$. Next, we consider a subnetwork $\{2,(\{3,4\},5)\}$ as consisting of two parallel networks, one with one task and one ordering, and the other with 3 tasks and 6 orderings: $C^{(c)}\{2,(\{3,4\},5)\} = 168$. By a trivial application of rule $C^{(b)}$ we get a total number of orderings equal to 168 (out of $7,484,400$ orderings possible in a network of 6 tasks without precedence constraints).

The number of orderings in the irreducible task network in Figure 2.8 has to be calculated using the `calcOrderings` algorithm. It is equal to 517, and is quite different from the one calculated before, although the number of tasks, 6, and the number of precedence constraints, 7, are equal for both cases.

Clearly, the number of orderings can grow exponentially with the number of tasks. However, precedence constraints may reduce this growth; in the case where all the tasks are in a single sequence, there is only one possible ordering. This observation emphasizes the need for understanding the problem domain, in order to construct efficient maximization algorithms that will not require the exploration of all possible orderings to find desirable maxima[7].

2.4 Maximization Methods

Our initial attempt to explore the local maxima space was to use unconstrained maximization[8] with all precedence and order constraints internalized within the calculation of the certainty equivalent CE. This approach made it possible to find the global maximum of CE after a large number of restarts from random points in a $2N$–dimensional space of task start and finish times [Babanov *et al.*, 2002a]. However, it also produced many "blind maxima" — points which wrongfully appeared to be maxima to the maximization algorithm because of constraint handling inside the calculations of CE. This approach was abandoned until we find a better way of internalizing constraints.

[6]It was asserted earlier that there are *three* ways to position two unrelated tasks: two sequential and one parallel. This is not a contradiction — there are four distinct ways of arranging start and finish times for two parallel tasks.

[7]Our favorite motivational example is that a simple addition of a sequence of two tasks parallel to the irreducible task network in Figure 2.8 increases the number of orderings from mere 517 to $940,940$ — nearly $2,000$-fold.

[8]In particular, the `Matlab` implementation of Nelder-Mead direct search method in `fminsearch`.

In this section we consider a variety of methods that we designed to tackle the problem of maximizing the CE function. The methods are illustrated using the experimental data from exploring our sample house building plan. The general results were tested and remained applicable to several other networks including the irreducible task network in Figure 2.8.

2.4.1 Constrained Methods

We designed and tested three maximization approaches that utilize the ability to enumerate all possible orderings of task start and finish times. We call these methods *"loose," "strict"* and *"nice"* for reasons that will become clear later. Each method is initialized by random schedules that satisfy the corresponding set of constraints.

"Loose" and "Strict" Maximizations

The "loose" method has its set of constraints defined by the precedence relations and time constraints, as specified in Section 2.2.1. The number of such constraints for a generic N-task network is, clearly, $2N$.

The "strict" maximization method further restricts the "loose" method by adding task ordering constraints to the set, thus ensuring that every maximum found will have the same ordering of task times as in the initial schedule. The number of additional constraints introduced by this method depends on the selected ordering. For example, for our sample house building problem it varies from 2 to 7.

Unfortunately, both of these methods have a major drawback preventing us from using them to study the domain reliably. The issue is that they produce a large number of pseudo-maxima in cases where the maximization algorithm[9] is stuck on a plateau.

The results produced by the "loose" method for our sample problem and a risk neutral agent are shown in Figure 2.10, where each dot corresponds to some local or pseudo-maximum. In this figure the y-axis shows CE values and x-axis represents a quadratic measure of schedule $\{t_i^{\mathrm{s}}, t_i^{\mathrm{f}}\}_{i \in N}$ parallelism, which attempts to capture all 12 dimensions of task start and finish times in one measure.

[9] We use `fmincon` from `Matlab` that implements a sequential quadratic programming method with a numerical updating of the Hessian on every step.

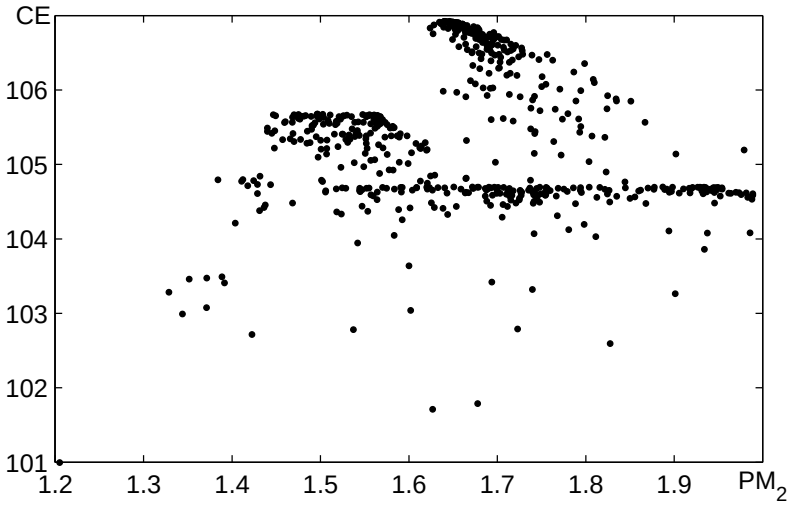

Figure 2.10: Local and pseudo-maxima found by the "loose" method for the house building network and a risk neutral agent.

We define the measure of parallelism of degree x as

$$\mathrm{PM}_x(\cdot) = \left\{ \frac{1}{t^{\mathrm{f}} - t^{\mathrm{s}}} \int_{t^{\mathrm{s}}}^{t^{\mathrm{f}}} \left[\sum_{i \in N} 1_{t \in [t_i^{\mathrm{s}}, t_i^{\mathrm{f}}]} \right]^x dt \right\}^{\frac{1}{x}} \tag{2.16}$$

The use of a quadratic ($x = 2$) measure is motivated by our expectation that schedules with higher number of parallel tasks will be less attractive to risk averse agents due to a high risk of a failure of one or more parallel tasks.

"Nice" Maximization

The "nice" method was created in the attempt to fix the pseudo-maxima problem by projecting a set of precedence and order constraints in a space where they do not interact directly. To achieve this we fix the order of task start and finish times, and build a correspondence between points $\bar{x}$ of a $(2N + 1)$–dimensional unit cube and the ordered vector of $2N$ task times. This many-to-one correspondence reflects proportions in which task start and finish times divide the $[t^{\mathrm{s}}, t^{\mathrm{f}}]$ interval. Figure 2.11 illustrates the suggested variable transformation for the network consisting of two sequential tasks.

This approach works remarkably well and produces nice, hence its name, local max-

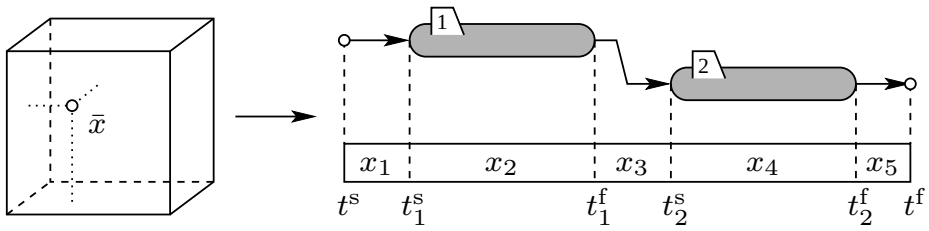

Figure 2.11: Transformation of a point in a 5-dimensional unit cube to a schedule for a 2-task network.

ima without artifacts like "blind maxima" or pseudo-maxima described above. The transition from zero of the unit cube is not defined. However, in our experience this never caused a problem, presumably because the CE function is always flat in the direction of zero.

The results produced by the "nice" maximization for the house construction problem and a risk neutral agent are shown in Figure 2.12 in the same scale as the results for the "loose" method in Figure 2.10. The comparison of Figures 2.10 and 2.12 reveals that the bulk of points in the former are, indeed, pseudo-maxima corresponding to unsuccessful attempts to reach clusters of maxima distinctly displayed in the latter.

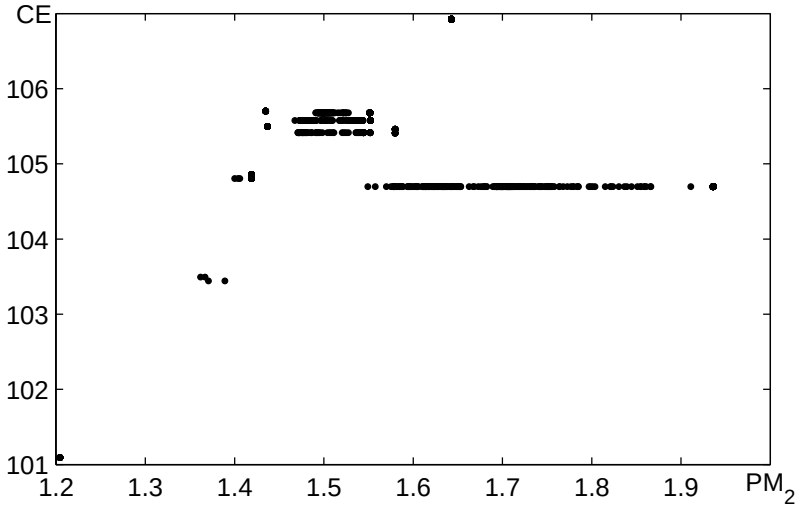

Figure 2.12: Local maxima found by the "nice" method for the house building example and a risk neutral agent. Compare to Figure 2.10.

2.4.2 Domain-specific Methods

One important observation is derived from our experiments that renders all three methods above nearly useless for the case of risk averse agents. All three methods are "lazy," meaning that, being initialized with a random schedule that results in a negative CE value, they converge to a degenerate local maximum with one or more task durations equal to zero. This behavior may be interpreted as a way of signaling rejection of the plan, and, although being potentially useful, it dominates maximization results.

We approached the problem of poor maximization performance by thoroughly studying the properties of the CE function in the task network environment. These efforts led to the creation of two approaches that we refer to as *"boost"* and *"copycat"* methods. Both approaches are, in essence, variations of homotopy, or continuation methods [Allgower and Georg, 1990].

"Boosted" Maximization

This method arises from the intuition that lowering the final payoff might change a schedule somewhat, but will not lead to abandoning it completely, unless its CE value becomes negative. Guessing this, we "boost" the search space by increasing the final payoff to get a maximizing schedule. We then repeat the maximization procedure starting with the obtained maximizing schedule, using the actual payoff to get an adjusted schedule.

Table 2.1 shows the results of "boosting" each of the three constrained maximization methods described before by doubling the reward. We use the sample house construction problem and a risk neutral agent. Each number in the table shows the number of local CE maxima with positive CE values found after 100 restarts for each of 168 possible orderings (refer to Section 2.3.3).

"Copycat" Maximization

The "copycat" method arises from the intuition that, although the exact location and relative attractiveness of different maxima may change as the agent's risk aversity changes, these changes won't likely be large. To verify this guess we used maxima for risk loving agents as initial points for maximizing the CE functions of agents with

Method	# plain	# "boosted"
"loose"	404	660
"strict"	366	687
"nice"	410	1562

Table 2.1: Number of local maxima for plain and "boosted" maximization methods, for the house building example and a risk neutral agent.

higher risk aversity.

Table 2.2 shows an example of applying the "copycat" improvement to our "nice" method in the sample problem. The set of maxima for $r = -0.03$ was used to improve the "nice" method's performance for higher risk aversity cases. The results show that not only the number of maxima found rises dramatically, but the maximum CE value among all local maxima increases as well.

	"nice"		"copycat nice"	
r	# max	max CE	# max	max CE
0.06	0	0.00	149	3.66
0.05	0	0.00	6194	9.70
0.04	0	0.00	6195	19.32
0.03	0	0.00	6198	34.35
0.02	1	54.50	6198	57.46
0.01	36	84.77	6198	85.12
0.00	410	106.92	6198	106.92
−0.01	1602	118.13	6198	118.13
−0.02	3736	123.16	6198	123.16
−0.03	6198	125.64	—	—.—

Table 2.2: Number of local maxima for "nice" and "copycat nice" maximization methods, for the house building example.

2.4.3 Properties of the Domain

We have employed the "nice" method with "boost" and "copycat" extensions to thoroughly study the properties of task scheduling using CE function maximization. In this section we illustrate our findings using the sample house construction network in Figure 2.1 and the assumption of risk neutrality. The reason we choose the risk neutral agent's case is due to our empirical results showing that CE maximizing

schedules will not change much with r, although their value relative to each other may and will change.

Figure 2.13 exposes the structure of schedules behind some of the CE maxima shown in Figure 2.12. By examining this figure we conclude that CE maximization discovered several distinct strategies of scheduling tasks 2 to 5. The most successful strategy in this setup, A, schedules as many tasks in parallel as possible to get a final payment as soon as possible, hence increasing its present value. Indeed, for $r = 0$ a random schedule has almost 80% probability of converging to schedules A, C or I, in which tasks 3 and 4 are parallel. K, though, has too many tasks scheduled sequentially, thus running against the time limit.

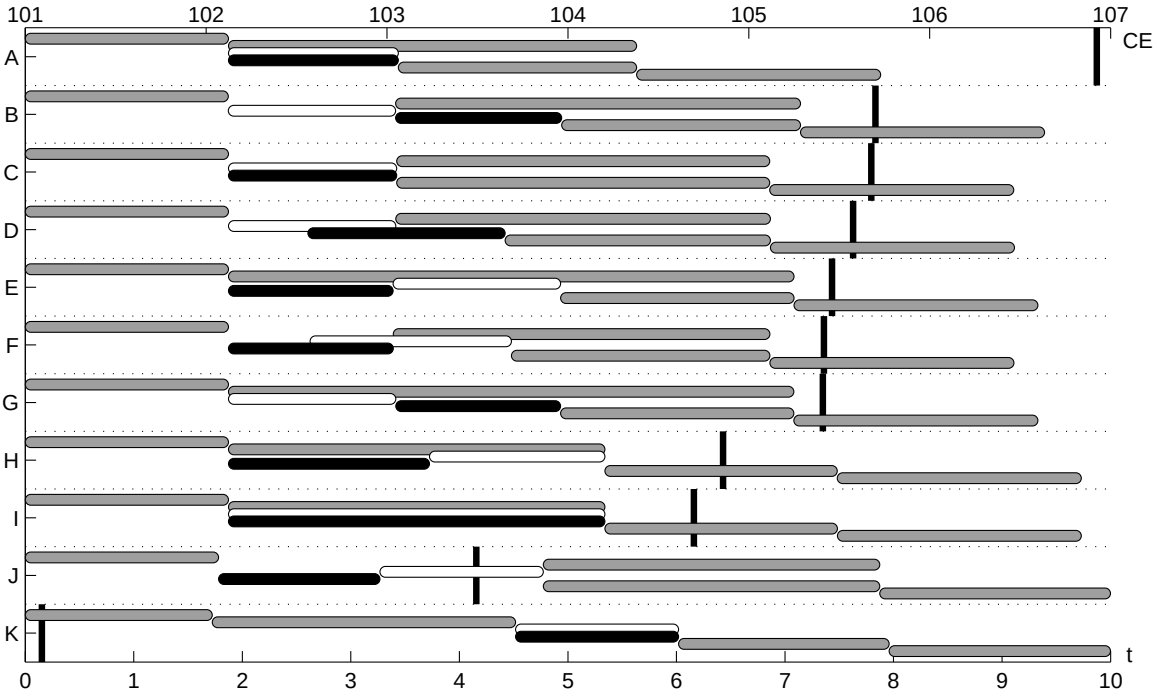

Figure 2.13: Structure and CE values for different clusters of CE maximizing schedules corresponding to maxima in Figure 2.12. The bottom x–axis shows time. Schedules are represented by rounded horizontal bars in the same manner as in the previous figures, and separated by dotted lines. Each schedule is assigned a letter on the y–axis. In each schedule, task 3 is highlighted in white color and task 4 in black. Top x–axis represents the certainty equivalent, and vertical black bars show the CE values attained in the corresponding schedules by a risk neutral agent.

One alternative is to schedule tasks 3 and 4 sequentially, giving task 2 more time to run by scheduling it parallel to three (as in E and G), two (B, H) or, at least, one

and a half (D, F) of other tasks. Attempts to give task 2 less time by scheduling it parallel to task 4 reveal that scheduling tasks 3 and 4 in parallel is important to ensure necessary schedule flexibility (compare J to C).

However, a schedule having highest CE value is not necessary the most preferable one. In fact, by examining Figure 2.14 we see that schedule A has hardly any flexibility, which is indicated by its very small support on the parallelism measure axis. In contrast, strategy I allows for a great flexibility in choosing time windows for tasks 3 and 4, hence its large cluster of maxima points in CE–PM_2 space.

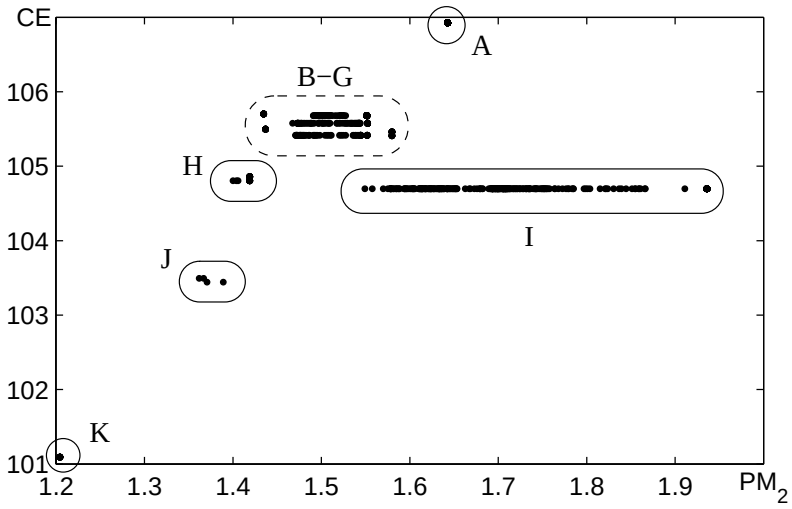

Figure 2.14: Local maxima in Figure 2.12 annotated with schedule labels from Figure 2.13.

The particular choice of one schedule over another must depend on the availability of bids for each task. For example, if a market has an abundance of task 3 and 4 suppliers, a risk neutral agent may safely choose schedule A as a base for an RFQ that will solicit desirable bid combinations. In the case when one or both of these tasks are in rare supply, the agent might prefer schedule I and trade its expectations of incoming bid quality, as expressed by the certainty equivalent, for larger RFQ time windows and, thus, a higher expected number of bids.

We want to stress that an agent's preferences over CE maximizing schedules vary greatly with risk aversity. For example, in Figure 2.15, schedule A turns to be less attractive in the eye of a more risk averse agent than schedules H, I and even K. One cause of such change is that risk averse agents generally prefer a "fail fast" approach that distributes more time to the tasks that are late in the schedule.

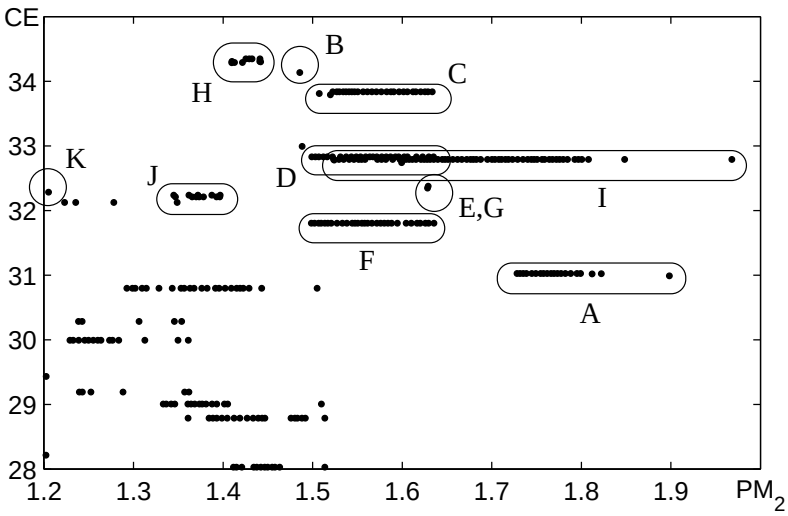

Figure 2.15: Change of local maxima structure in Figure 2.14 for $r = 0.03$.

2.4.4 Heuristics

Our findings show that CE maximizing schedules have different properties with regard to quality and quantity of bids that are potentially solicited by the corresponding RFQs. We also demonstrated that naive CE maximization approaches do not perform well in interesting cases, but their performance can be significantly improved by using domain properties. We also observed that the maximization methods used to explore the problem domain are hardly applicable in practice, as they rely on the task ordering enumeration and therefore are exponentially hard in the worst case.

As a result, we plan on designing heuristics that will let agents effectively explore maximizing schedules of different nature and choose desired quality-quantity combinations. One possible heuristic might be based on the observation that there exist fairly obvious transitions between many maximizing schedules (see Figure 2.16), such as rescheduling two tasks from sequential ordering to parallel or the other way around. When designed, such heuristic will reduce the search problem from considering all task orderings and random schedules to finding a few maxima and exploring the near-optimal schedules suggested by the heuristic.

Another potential approach would be to discover the exact mechanism behind the changes in preferences over maximizing schedules for different degrees of risk aversity. This heuristic would allow us to concentrate search efforts on specific classes of schedules that are expected to be of interest for agents with known risk aversity.

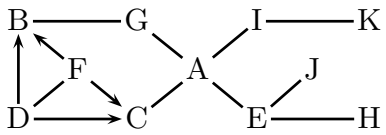

Figure 2.16: Some simple transitions between schedules in Figure 2.13.

We employ the idea of domain-specific heuristics to articulate a possible design of a stochastic maximization algorithm is Section 2.7.1. In the next section we suggest an alternative way of bringing near-optimal maximization up to speed by reducing the number of distinct task orderings.

2.5 Counting of Event Trees

Consider two two alternative schedules for the house building problem shown in Figure 2.17. These schedules look very different, yet they correspond to the same event tree — the right tree in Figure 2.6.

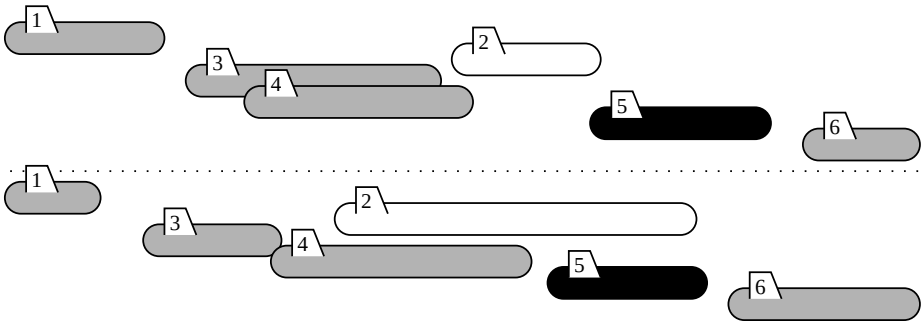

Figure 2.17: Two schedules for the task network in Figure 2.1.

It is easy to see that the certainty equivalent is a continuous function of task start and finish times as far as it is restricted to one particular event tree. We argue that it is also true that in almost all generic problems there is a discontinuity of CE value between two arbitrary close schedules with different corresponding event trees.

Indeed, event trees change when the start time of some task is scheduled in a different order with the finish time of another task. For example, consider an exchange of the start point of task 5 with the finish point of task 2 in the first schedule in Figure 2.17 — a seemingly small change. In the outcome, however, the certainty equivalent of the schedule is expected to increase, since an agent is not any longer liable for paying

for the success of task 5, if it is known that task 2 failed.

To put it in numbers, in the set of payoff-probability pairs two events with probabilities $\tilde{p}_1\tilde{p}_3(1-\tilde{p}_2)\tilde{p}_4(1-\tilde{p}_5)$ and $\tilde{p}_1\tilde{p}_3(1-\tilde{p}_2)\tilde{p}_4\tilde{p}_5$, and respective payoffs $\tilde{c}_1+\tilde{c}_3+\tilde{c}_4$ and $\tilde{c}_1+\tilde{c}_3+\tilde{c}_4+\tilde{c}_5$ are replaced by a single event with a probability $\tilde{p}_1\tilde{p}_3(1-\tilde{p}_2)\tilde{p}_4$ and a payoff $\tilde{c}_1+\tilde{c}_3+\tilde{c}_4$. This change leads to a positive gain as long as $\tilde{c}_5$ and $\tilde{p}_5$ are both positive.

Assuming now that an event tree represent the maximum interval of CE continuity, we proceed by finding a method for enumerating all possible event trees, and by comparing the numbers of event trees with the number of task orderings.

2.5.1 Counting Method

As of this moment, our efforts to find an analytical way of enumerating event trees, similar to a method of enumerating task orderings in reducible task networks (refer to Section 2.3.1) were not successful. Instead we propose an algorithmic approach to the problem (see Algorithm 3).

The algorithm starts with a call to **startTasks** with the assumption that no task is started or finished yet. It proceeds by alternating recursive calls to **startTasks** and **finishTasks**, so that during every call at least one task is started or finished.

Figure 2.18 illustrates the execution path of this algorithm that leads to a schedule that is equivalent to the right tree in Figure 2.6 and both schedules in Figure 2.17. Note that calls to **startTasks** (shown by vertical lines with letter *'s'*) and **finishTasks** (letter *'f'*) alternate and always start or finish a non-empty set of tasks.

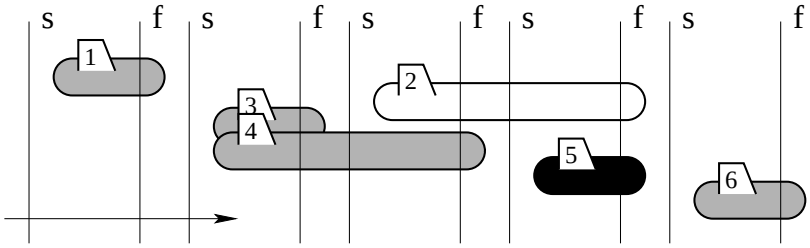

Figure 2.18: Execution path of the event tree enumeration algorithm corresponding to the right tree in Figure 2.6 and two schedules in Figure 2.17. Vertical lines show the recursive steps of the algorithm from left to right.

Algorithm: $C \leftarrow$ **startTasks**(T, D)
  Requires: T "*started tasks*", D "*finished tasks*"
  Returns: C "*number of orderings*"

$C \leftarrow 0$
$X \leftarrow \{i \in N | P_1(i) \subset D, i \notin T \cup D\}$ "*tasks to start*"
foreach $Y \in 2^X \setminus \{\varnothing\}$
  $C \leftarrow C +$ **finishTasks**$(T \cup Y, D)$
endfor
return C

Algorithm: $C \leftarrow$ **finishTasks**(T, D)
  Requires: T "*started tasks*", D "*finished tasks*"
  Returns: C "*number of orderings*"

if $X \cup T = \varnothing$ "*no tasks left to start*"
  $C \leftarrow 1$
else "*can start some tasks*"
  $C \leftarrow 0$
  foreach $Y \in 2^T \setminus \{\varnothing\}$
    $C \leftarrow C +$ **startTasks**$(T \setminus Y, D \cup Y)$
  endfor
endif
return C

Algorithm 3: Algorithm for enumerating distinct event trees.

2.5.2 Examples

Table 2.3 shows the results of executing Algorithm 3 on a few sample problems compared against numbers from Section 2.3.3. The number of event trees is never greater and, in most cases, much lower than the number of task orderings. Also, as it was argued before, event trees correspond to the maximum regions of continuity of the certainty equivalent.

To summarize our findings, it is beneficial to use event tree enumeration instead of task ordering enumeration as a basis for near-global optimization algorithms, such as those described in Section 2.4. It is also worth investigating if CE can be analytically maximized on its intervals of continuity, in order to turn near-global optimization in a formally global one.

Task network	# orderings	# event trees
Two sequential tasks	1	1
Two parallel tasks	6	3
House construction task network	168	32
Irreducible task network in Figure 2.8	517	70
—"— parallel to two sequential tasks	940,940	24,913
Six parallel tasks	7,484,400	81,663

Table 2.3: Numbers of task orderings and event trees for sample problems.

2.6 RFQ Generation

In the previous sections we discussed methods of finding CE maximizing schedules of task execution. For a given value of risk aversity and known market statistics, such schedules ensure the highest expected quality of the bids that satisfy them. By *quality*, we assume some function of the expectations over the cost, the probability of successful completion, and the profitability of the incoming bids when combined with other bids.

A maximizing schedule, however, cannot serve as a rational RFQ, since it is unlikely that bids will be available to cover precisely the same intervals as mandated by this schedule. In order to construct a viable RFQ using a maximizing schedule as a basis, the customer agent might choose to lower its expectations of the bid quality to some level by widening the RFQ time windows around the time windows in the maximizing schedule, thus increasing[10] the expected number of incoming bids. In this section we discuss criteria that allow for rationalizing the selection among all such RFQs.

We approach the individually rational (i.e. agent-dependent) RFQ generation as follows:

1. Measure the sensitivity of the expected bid quality to deviations from the CE maximizing schedule.

2. Derive the relationship between the quality of incoming bids and the size of RFQ time windows.

[10]At least to some extent, — there is a fair chance that the number of the incoming bids will cease to increase whenever RFQ time windows become too large to inspire confidence on the part of suppliers.

3. Choose a rational quality-quantity combination.

In addition, we search for a solution concept that generates viable RFQs, *and* is comprehensible to a human user of the system.

2.6.1 Sensitivity to Schedule Changes

We propose measuring the sensitivity of CE by investigating how CE values change with variations of a single task i start time t_i^{s} in the CE maximizing schedule. For the sake of brevity, the resulting dependency of CE values is denoted by $\mathrm{CE}\left(t_i^{\mathrm{s}}\right)$. Figure 2.19 shows $\mathrm{CE}\left(t_i^{\mathrm{s}}\right), i = 1 \ldots 6$ for our sample house building problem, for $r = -0.01$ and $r = 0.02$ respectively.

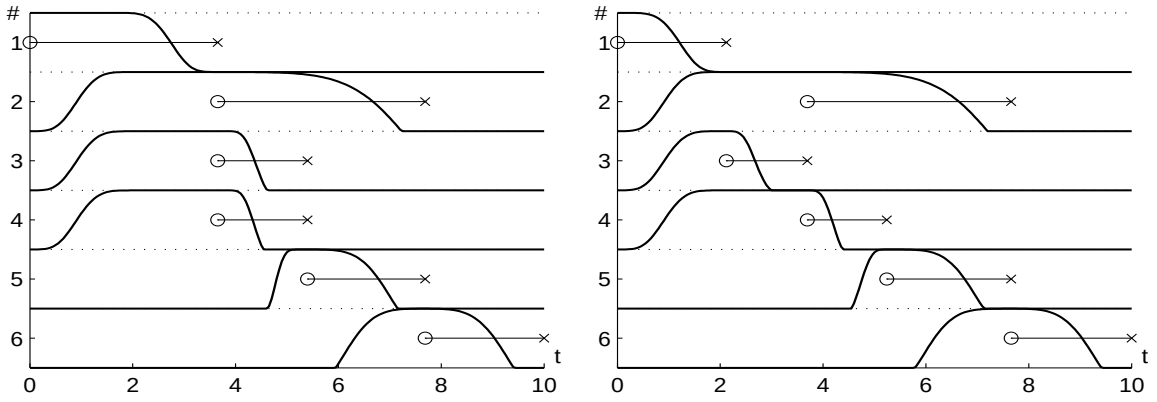

Figure 2.19: $\mathrm{CE}\left(t_i^{\mathrm{s}}\right)$ graphs for the corresponding maximizing schedules in Figure 2.5. y-axis of each horizontal stripe i represents the percentage of the maximum CE value as t_i^{s} varies; x-axis represents t_i^{s}; the horizontal lines with circle and cross ends show the corresponding maximizing schedules.

Tasks 1, 3 and 5 in the right graph are relatively sensitive to the start times of the bids that can be bundled with other bids without considerably impairing the resulting bundle's value. However, the fact that task 2 in the right graph is more flexible does not guarantee that it will attract a higher number of bids, since the latter depends both on the size of the corresponding time window and on the market properties of the task: resource availability, number of prospective bidders, seasonal changes, etc.

We assert that for the purpose of creating a rational RFQ, it is admissible to choose time windows based on the sensitivity of CE to deviations of a single time constraint from the reasonable schedule. The rationale is that the relations between tasks are

already encapsulated in the calculations of CE, so the change of one constraint will approximate the rescheduling of several related tasks in the neighborhood of the maximizing schedule.

2.6.2 Quality vs. Quantity

Observe that the time window for the task i, $\{t_i^s | \mathrm{CE}\,(t_i^s) \geq x\}$, grows as the lowest expected CE value, x, decreases. The relation between these two variables for tasks 3 and 4 of the test problem is shown in Figure 2.20. The corresponding relation between the lowest expected CE value and the expected number of bids as a function of the window size is shown in Figure 2.21. In the right graph we assumed, for the sake of example, that the supply of task 3 in the market is higher than of task 4, hence the difference in relative positions of task 3 and task 4 graphs in two figures.

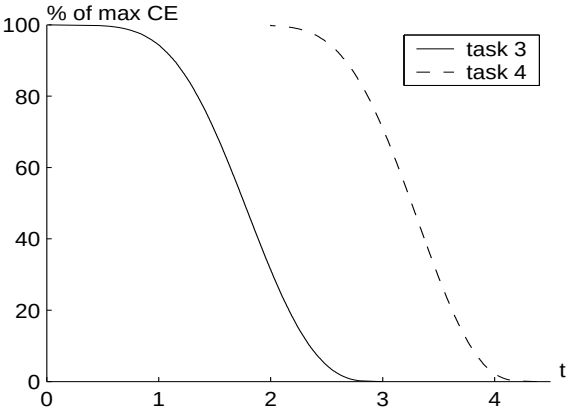

Figure 2.20: Relationship between the RFQ window size (shown in units of time on x-axis) and the lowest admissible percentage of the maximum CE value.

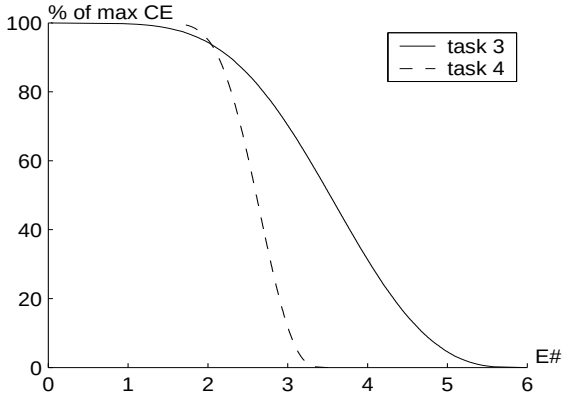

Figure 2.21: Relationship between the expected number of bids (shown on x-axis) and the lowest admissible percentage of the maximum CE value.

The graph in Figure 2.21 is an example of the relation between the quality and the quantity of bids we are searching for. Indeed, the only independent variable in this graph is t_i^s. The quantity of bids depends on the size and positions of RFQ time windows that, in turn, depend on the decision about the lowest acceptable CE value. The quality of bids is a function of the RFQ choice and the properties of the plan. Finally, it is expected that the customer agent will prefer a point on the graph to any point below and to the left of it, hence the best choice should lie on the graph.

2.6.3 Rational Choice of RFQ

We illustrate the decision process of the customer agent in Figure 2.22, where the customer agent's preferences over quality-quantity combinations are represented by a family of indifference curves, and the graph of underlying quality-quantity relationship is as derived in the previous section. Each indifference curve shows quality-quantity pairs that are equivalent from the agent's point of view. In particular, points A, B and C are considered to be equally attractive. The intuition is that although in point A agent receives much smaller number of bids compared with C and is exposed to a higher risk of not covering some tasks, this is offset by the positive effect of a lower percentage of bid rejections on the agent's reputation. Also the winner determination problem is exponentially easier to solve [Collins, 2002] for the lower expected number of bids in point A.

For all points below the maximum expected quality line (horizontal dashed line in the graph) agent's preferences increase in the direction of point M. Thus a curve through point D is preferred over one through point C and even more so over one through point E. The rational choice belongs to the intersection of the quality-quantity graph and the highest indifference curve (point B in the graph).

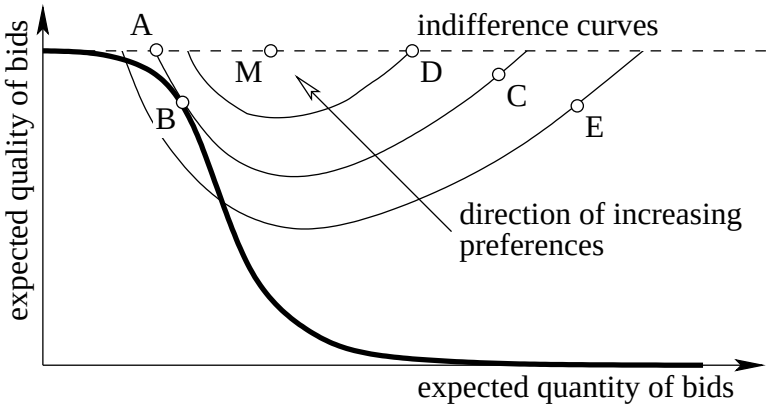

Figure 2.22: Quality-quantity graph with three indifference curves.

After the rational choice of the quality-quantity combinations for all tasks in the plan is revealed, we proceed with constructing the RFQ time windows. The *early start time* t_i^{es} and the *late start time* t_i^{ls} are determined by the value of the reciprocal of the CE (t_i^{s}) at the minimum admissible CE choice for the task i. The *late finish time* t_i^{lf} is chosen to be at the reasonable time window length distance from t_i^{ls}. Figure 2.23 shows two sample RFQs for the house building example.

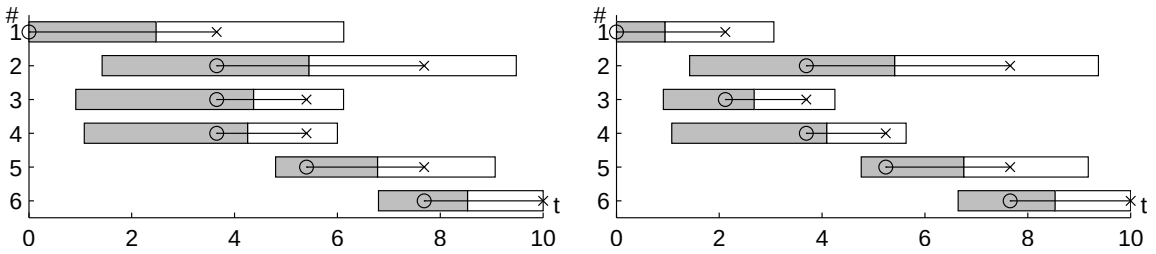

Figure 2.23: Rational RFQs for the corresponding maximizing schedules in Figure 2.5, and the following vector of the maximum CE percentages: $(80\%, 95\%, 50\%, 70\%, 50\%, 90\%)$. Gray bars show start time intervals, $[t_i^{\mathrm{es}}, t_i^{\mathrm{ls}}]$, the ends of white bars correspond to late finish times, t_i^{lf}, and the horizontal lines with circle and cross ends show the corresponding maximizing schedules.

Our choice of the RFQ may not be optimal in the quantitative sense, however it is individually rational for the customer agent, it is also fast to compute, and arguably easy to grasp for a human user of the system. It should be emphasized here that the choice of the RFQ is based on uncertain market information, hence any quantitatively "optimal" solution is itself a compromise.

2.7 Extensions

In this section we discuss possible extensions to the theory presented in the chapter. The exact choice of which of these we will investigate next will depend on the needs of the MAGNET project, as well as on the input from the academic community.

2.7.1 Stochastic Maximization Methods

To summarize all previously outlined properties of the domain that influence the design of effective stochastic search algorithms:

- local maxima are due to different scheduling order of tasks off the critical path (Section 2.2.6);
- RFQs based on a global maximization might be overly restrictive (Section 2.4.3);
- groups of local maxima might have similar CE values (Section 2.4.3);
- it is possible to "jump" from one local maxima to near another (Section 2.4.4).

The properties of the domain frame the properties of the search algorithm that we design to fit this domain. Namely, the search algorithm must be able to test different orderings of tasks, it should know how to explore groups of similar local maxima and, whenever possible, it should provide alternative schedules with CE values close to the global maximum.

We propose a search algorithm based on the ideas of Simulated Annealing [Reeves, 1993] and Genetic Algorithms [Forrest, 1993]. The algorithm will combine the stochastic temperature-driven nature of Simulated Annealing with the simultaneous search space exploration of Genetic Algorithms.

The proposed search algorithm explores several alternative schedules in parallel. The initial set of alternatives can be generated in many ways: random generation, hill-climbing from a random schedule, CPM, etc. The execution of the algorithm proceeds in steps by randomly applying one of the six transformation rules to each alternative schedule. The probabilities of choosing some rule can be adjusted to tune the algorithm's behavior in a wide range.

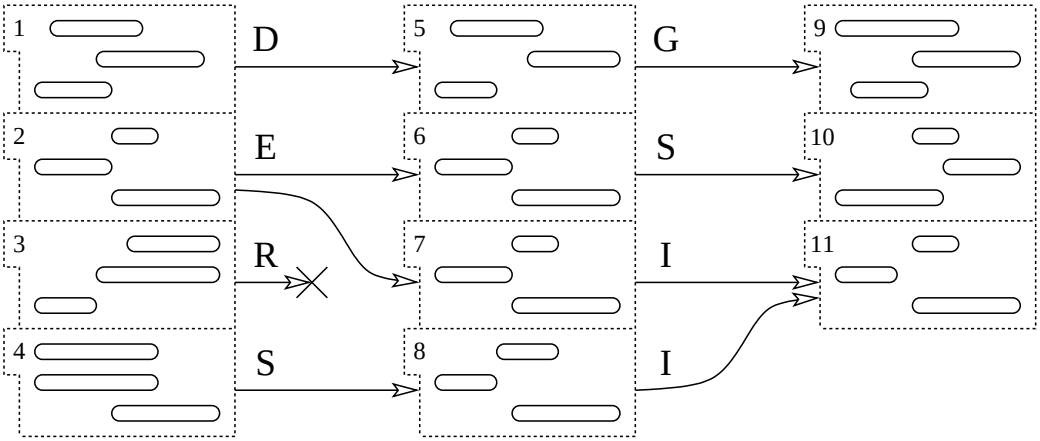

Figure 2.24: Two steps of the stochastic search algorithm execution.

Figure 2.24 illustrates the algorithm for the case of three pairwise independent tasks. In this figure columns represent three sequential steps of the algorithm, each column lists several alternative schedules. Arrows and letters next to them denote various transformations from the following list:

Distortion alters start and finish times of one or several tasks as well as adjusts time windows of all related tasks to maintain precedence constraints ($1 \rightarrow 5$

in Figure 2.24). Distortion mimics the basic step of the SA algorithm in the continuous $2N$–dimensional space of task start and finish times.

Gradient following is one or more steps of a generic numerical maximization method ($5 \rightarrow 9$). This step is very computationally intensive, as it requires many calls to **calcGamble** algorithm in the process of calculating numerical derivatives. By varying the relative probabilities of the distortion and gradient following we may choose different stochastic properties.

Shuffling changes the relative scheduling of two or more tasks, wherever it is permitted by the precedence constraints. Shuffling can switch the ordering of tasks ($6 \rightarrow 10$), change from sequential ordering to parallel, or reschedule parallel tasks to be executed sequentially ($4 \rightarrow 8$). The major role of shuffling is to explore distinct local maxima by "jumping" between different event tree forms.

Explosion adds a copy of the subject schedule to the list of alternatives ($2 \rightarrow 6, 7$). Explosion complements shuffling by allowing for simultaneous exploration of groups of similar schedules. We may choose to decrease the rate of explosions with the annealing temperature to focus on improving the current set of solution after the search space was explored to some extent.

Implosion merges two similar[11] schedules in one. Implosion helps reducing computational expenses from crowding several alternative schedules around one maximum ($7, 8 \rightarrow 11$). The rate of implosions will change in the opposite direction to the rate of explosions.

Removal eliminates alternatives that do not score well relative to others ($3 \rightarrow \varnothing$). This transformation takes care of the schedules that are stuck in local maxima with low CE values. The rate of removals grows as the annealing temperature decreases.

Each of the first five transformations is tested against the SA temperature rule whenever it leads to a decrease in the CE value. In case it is discarded, other transformations are chosen at random and applied until one of them increases CE or passes the temperature rule.

[11]Similarity is a function the distance between two schedules as between two points in the $2N$-dimensional time space.

The probabilities of transformations as well as details of the proposed search algorithm's properties are subject to further research. It is reasonable to believe though that a comprehensive study of the RFQ generation mechanism is only possible in a dynamic market environment. In Chapter 3 we describe the approach to large-scale testing of the MAGNET system that we presently research, and that might provide us with the necessary data for testing stochastic maximization approaches.

2.7.2 Partial Plans and Plan Repair

It is possible to extend the assumptions that our theory inherits from the MAGNET framework by making task networks and the negotiation process more flexible. We envision the following interesting and relatively feasible extensions:

- **Partial plans.** A customer agent might submit a sequence of dependent RFQs. For example, the first RFQ might solicit bids for some scarce resource giving suppliers a great flexibility; the second RFQ might use the information obtained from the first one to pinpoint additional requests for the rest of the required resources.

- **Plan repair.** An agent might try to repair its plan after one or more suppliers defaulted on the delivery. This will require factoring extra time slack in an RFQ to account for possible failures. A new planning process will require modifications to the current CE maximization formulation.

- **Soft precedence constraints.** One limitation of MAGNET is the assumption of the strict precedence of tasks. In reality it is often the case that some task is started after a certain percentage of a preceding task is completed. One can try to work around such limitations by breaking the preceding task in two parts. This, however, will strain suppliers who might want a different breakup; it will also require a relatively rich bid description language.

2.7.3 Supplier Price Distributions

Another interesting extension is to relax the assumption that task costs are represented by a market average. We performed a few preliminary tests of a formulation that allows for specifying arbitrary price distributions instead of single values. The

resulting form of the certainty equivalent is a distribution that is a function of the given price distributions.

Further development of this idea requires less computationally expensive maximization algorithms, and a higher involvement of the analytical methods in the maximization routine.

2.7.4 Human-Computer Interaction

The last proposed extension is, perhaps, the most vital for the future development of the project. While describing the theory and numerical experiments in this chapter, we deliberately avoided questions of obtaining data on agents' preferences, risk aversity, etc. These questions will be hard to evade if we plan to make the theory applicable to real situations.

To offset this problem we plan to initially develop the theory in a mixed-initiative environment, i.e. to deploy our intelligent software agents in parallel and under control of a human decision maker. The goal of the agents will be to perceive the decisions made by a human and adjust their vision of human's preferences. A midway step between the current state and the deployment of a mixed-initiative system will be the large-scale experimentation framework described in greater detail in the next chapter.

2.8 Summary

In this chapter we presented a way of generating individually rational requests for quotes on the part of a customer agent. The approach that we take uses the expected utility formulation. We have shown a collection of maximization methods that can be used to support the creation of RFQs, and outlined future development of the theory.

The idea to use expected utility to factor risks in the customer's decision criteria was originally mentioned in [Collins *et al.*, 2001], and was subsequently formalized in [Babanov *et al.*, 2002a]. The first detailed study of the maximization methods was done in [Babanov *et al.*, 2003c]. The approach to building rational RFQs on the basis of CE maximizing schedules will appear in [Babanov *et al.*, 2003b].

Chapter 3

Evolutionary Framework for Large-scale Experimentation

3.1 Introduction

The goal of our research is to usefully employ intelligent agents to support human decision making in real markets. A theoretical analysis of algorithms and sample strategies could help us understand what benefit could be derived from their use. However, to thoroughly examine and tune algorithms one needs extensive test suites, preferably based on realistic data, and supported by a clear understanding of market mechanisms. Such data are not readily available in the case of MAGNET, since we are not aware of any real-world application using reverse combinatorial auctions with a time component.

To support theoretical analysis, it is common to assume that a new strategy does not have a large impact on the market. Under such assumption the analysis can be conducted as if no other agent changes its strategy. Another approach is to assume that everyone in the market adopts the same strategy, and use the representative agent approach to solve for an equilibrium.

The problem arises when we try to analyze a market in which many different strategies are present, and none of them clearly dominates the others. The relative strength of each strategy would then depend greatly on its ability to learn and adapt, on interaction with other strategies, as well as on the general market situation. Thus

each strategy should preferably be assessed in a dynamic market environment.

A good example to illustrate the issue is the case reported in [Roth, 2002] about his study of the labor market clearinghouse for American physicians. As reported in the study, none of the theorems from the theoretical model could be applied directly to the real market data, since they used much simpler models. Nevertheless, the theory proved useful to design more complex experiments and to validate the experimental results. The computational methods allowed to compare the experimental results with the theory. Showing that departures from the theory were small was an important result of the entire study.

We propose to use an evolutionary approach as a computational framework for assessing multi-agent systems. The major advantage of an evolutionary framework is that it is a natural choice for studying complex societies of entities, where the structure of the society changes during the simulation, and the entities change as well in an effort to adapt to the changing environment. Evolutionary game theory [Weibull, 1995, Vega-Redondo, 1996] provides tools for analysis in dynamic environments. Applications studied by other researchers range from the emergence of cooperation in an otherwise selfish society [Axelrod, 1984, Axelrod, 1997] with possible formation of spatial patterns of strategic interaction [Lindgren, 1997], to ostracism and neighborhood effects [Gaylord and D'Andrea, 1998], and design of auction mechanisms [Phelps *et al.*, 2002].

A drawback of most evolutionary systems is that they need a homogeneous representation of the strategies used by the agents. In fact, it is often prohibitively hard to come up with a convenient representation of strategies for well studied problems [Forrest, 1993]. It is even harder to make several strategies evolve simultaneously — figuratively speaking, cats and dogs just don't crossbreed correctly. On the other hand, in the real world the same cats and dogs coexist, however nicely or not, by excelling in different "market" niches.

The homogeneity requirement is dictated by the agent reproduction rules, and causes most evolutionary environments to be in-house projects that do not benefit from the cooperation of several research teams. We propose an extension that allows one to use strategies that are represented heterogeneously, and we illustrate our ideas with a case study of a society of service providers and customers.

Our proposed extension adds a layer of evolutionary learning atop disjointly evolving types of agents that use different strategies. Every type of agent maintains its separate source of "genetic" information. This information could be a gene pool, if the type is based on the genetic algorithms paradigm; it could be some statistical data, as it is the case in the test model we introduce later in this paper; it could even be a neural network, which is continuously trained on the performance of its "children."

The purpose of the additional evolutionary layer is to learn the probabilities with which representatives of each type should enter the market, by observing the distribution of surviving agents by type. Each time a new agent is introduced to the market, the top layer decides on its type and uses the corresponding reproduction rule. Agents who fail to satisfy a predefined performance criterion are removed at regular time intervals and, eventually, replaced by more fit entities.

We illustrate the idea of two-layered evolutionary framework in greater detail in Section 3.3 where we introduce our case study. In the next section we discuss a way of converting the existing mixed-initiative setup of the MAGNET system to evolutionary settings. The approach that we demonstrate is general enough to be used in our test model, which is based on a rather different market structure.

3.2 Framework Design

Figure 3.1 shows a rough outline of the MAGNET system. At this level the structure is very generic, many other systems of trading intelligent agents have the same general structure. We intentionally choose this level of details to make sure that our speculations on building a large-scale experimentation framework on top of MAGNET can also be applied to outside projects.

In Figure 3.1 a collection of customer and supplier agents is governed by humans. A market agent is responsible for coordination of tasks, i.e. distributing customers' RFQs among an appropriate selection of suppliers, collecting and timing bids, monitoring interactions during task execution phase, etc. A data warehouse agent collects information on the transactions, and makes it available in a form of statistical data to agents and their owners.

In order for MAGNET to operate in the evolutionary framework, we need to add components that will manage the evolutionary aspects of the system and exclude

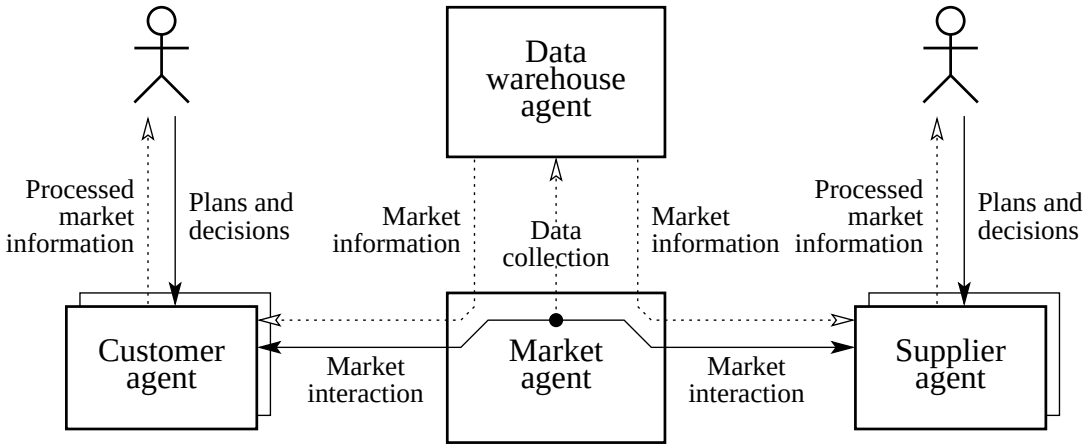

Figure 3.1: Mixed-initiative architecture of the MAGNET system.

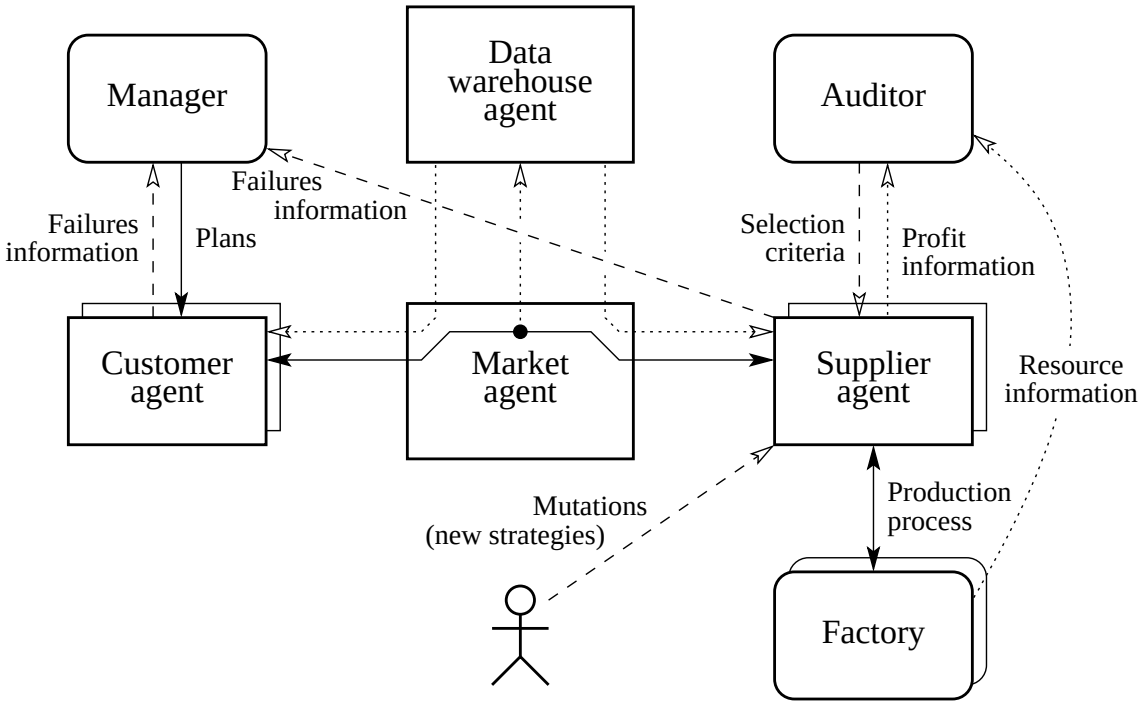

Figure 3.2: Evolutionary wrapper for the MAGNET architecture. Newly introduced components are indicated by rounded boxes.

human decision makers from the loop. Figure 3.2 shows the resulting architecture, and the following list summarizes the required changes:

The **Manager** generates and distributes tasks to customer agents. It observes the rate of customers' and suppliers' failures to complete their evaluations of RFQs and bids during specified deliberation periods (refer to Figure 1.2). The manager adjusts the frequency of issuing RFQs to keep the rate of failures reasonably low, yet not zero. Having a rate of failures greater than zero puts some pressure on agents that use computationally overly intensive strategies. The frequency of generating the RFQs determines the size of the market population.

The **Auditor** evaluates the performance of supplier agents' strategies based on suppliers' average profit over a specified period of simulation time. Agents that make negative profit are removed from the market. Whenever the average profit in the market exceeds some specified value, the auditor introduces a new supplier agent with a strategy that is chosen from the pool of all available strategies. The auditor maintains a pool of "retired" strategies, and eventually tries to put them back in the market.

The **Customer agent** makes all market decisions without help from its human supervisor and reports the rate of failures to the manager.

The **Supplier agent** also operates without human supervision and reports on its computational failures to the manager. The supplier agent coordinates its resource commitments with its own factory.

One instance of the **Factory** is assigned to each supplier agent to keep track of resource availability and existing commitments. The size and types of products produced in a factory are determined by the auditor upon creation of the corresponding supplier agent.

Human participants submit new strategies to the pool of possible **mutations**. "Mutant" strategies are introduced to the market in the same way as "retired" ones.

The **Data warehouse agent** collects data the same way as in the mixed-initiative configuration. In addition, it replies to data queries from the manager, the auditor and, possibly, human observers whenever it is required.

The choice of this particular architecture is determined by the need to stabilize the size of the market, and by our specific interest in testing customer-side algorithms. To satisfy these requirements we fix the population of customer agents and let the supplier agents evolve to meet the demand. The demand, in turn, is limited from above by the computational capacity of the system (possibly, distributed) that runs the agents' software. In case the load is overly high, the rate of agents' failures to complete calculations will signal the manager to decrease the rate of issuing RFQs, thus effectively shrinking the market.

Whether we decide to study the other side of the market, the architecture might be changed to make the auditor manage the evolving population of customer agents, while the manager governs the resource availability to suppliers. The exact choice and composition of the evolutionary components is not formalized at present, however we plan to improve the methodology as we study other prospective domains.

Two candidate domains are considered in the next sections. In Section 3.3 we demonstrate the results from a simple supply-demand model that is build in accordance with the guidelines given above. In Section 3.4 we touch upon a possibility of using the suggested approach in a new version of a well-established Trading Agent Competition [TAC, 2001, TAC, 2002, Stone, 2002].

3.3 Concept Check: Citysim

In this section we use a test model called *Citysim* to verify the suggested approach to evolutionary simulation of multi-agent systems. We demonstrate that in this model it is possible for heterogenous strategies to coexist in the market and to evolve to occupy specific niches. We examine the emergence of said behavior over a wide range of initial conditions and support the experimental results with theoretic analysis of the model.

3.3.1 The Model

Citysim is a model of a city inhabited by two types of economic agents: providers of a particular service, for example, hairdressers, and their customers. According to the design guidelines in Section 3.2 we assume that the properties of the customer side of the society do not change in the course of a simulation. At the same time the society

of suppliers is expected to evolve to meet the demands of the customers.

The assumptions of our test model are similar to those from the urban economics model of a monocentric city [O'Sullivan, 1999], and from the model of market areas with free entry [Mills and Lav, 1964]. We intentionally built our model upon classical urban economics models to ensure its practical meaning. It is important to note though that the observations of the model's behavior are not essential, they are offered mainly to demonstrate that the results obtained by the means of the proposed approach are, in fact, meaningful.

One issue with using evolutionary models is that the number of control variables might easily become uncontrollable. Typically, a researcher can guide an evolutionary model by changing: (a) the parameters of the environment, (b) the rules of interaction between agents, (c) the reproduction rules, (d) the initial parameters and distributions, and/or (e) the internal settings of the agents and agents' strategies.

The number of parameters is high and it is a daunting task to study the space of parameters thoroughly (and convincingly). The fact that the evolution of a system might take different paths depending on random events, such as those used in the reproduction scheme or in pairing agents during market activities, adds to the complexity. Worse yet, it is not always clear what parameters control the major trends in the system and what merely cause small perturbations.

In this section we tackle the problem of managing control variables by backing up the evolutionary system by the corresponding analytical model. We proceed by building a model of a monocentric city that serves as a market for suppliers of some service. Suppliers are given free entry to the market, and are allowed to position themselves when they enter the market and change prices when necessary. Suppliers may enter the market with different *sizes*, meaning that they can serve a different number of customers simultaneously. They use scheduling to provide service to the customers at the earliest possible time. The net price to customers accounts for the supplier's price, transportation cost, and service delay.

Next we apply a market area analysis to find what comparative advantage each size supplier has, and under which circumstances. We then conduct evolutionary experiments to verify how well the experimental evidence supports the theory and to extend the theory to more complex cases. We further explain how the individual properties

of each of two test strategies contributed to the observed discrepancy in the data.

3.3.2 General Terms

We consider a society of economic agents who live and interact in a circular city of radius R. Anonymous customers come to the market for a single transaction at random intervals governed by a stationary Poisson process with a fixed frequency λ^{c}:

$$t^{\mathrm{c}}_{i+1} = t^{\mathrm{c}}_i - \frac{1}{\lambda^{\mathrm{c}}} \log U[0,1] \tag{3.1}$$

where $U[x,y]$ is a random variable distributed uniformly on the interval $[x,y]$. The location of a new customer in polar coordinates is determined by the following rules:

$$r \sim U[0,R] \qquad \text{and} \qquad \alpha \sim U[0,2\pi] \tag{3.2}$$

The rules imply that the density of customers is inversely proportional to the distance from the center of the city[1].

Upon entry, a customer looks for suppliers and chooses the one that provides the service at the minimum cost. We define the net cost of service c as a function of the supplier's price p, distance to the customer d, and delay due to servicing previously scheduled customers Δt:

$$c = p + d \times c^{\mathrm{mile}} + \Delta t \times c^{\mathrm{hour}} \tag{3.3}$$

where c^{mile} and c^{hour} are cost per mile of travel and cost per hour delay respectively.

Suppliers, in turn, enter the market at intervals governed by a random process similar to the one used for the customers. A supplier is characterized by its pricing strategy and the number of customers it can serve simultaneously, also referred as the *supplier's size*. Serving one customer takes size s supplier one continuous hour and costs $c^{\mathrm{work}}(s)$, staying idle for an interval of time of any length costs $c^{\mathrm{idle}}(s)$ per hour. Both costs

[1]Studies in urban economics suggest and support by empirical evidence the use of a reversed exponential relation between population density and distance from the city center (see, for example, [Anas *et al.*, 1998, O'Sullivan, 1999]). We adopt a hyperbolic distance-density relation for convenience of the analysis.

decrease with size to simulate economies of scale:

$$c^{\text{work}}(s) = c^{\text{work}}(1) \times (1-g)^{s-1} \quad \text{for} \quad s \geq 2 \tag{3.4}$$

$$c^{\text{idle}}(s) = c^{\text{idle}}(1) \times (1-g)^{s-1} \quad \text{for} \quad s \geq 2 \tag{3.5}$$

where $c^{\text{work}}(1)$ and $c^{\text{idle}}(1)$ are constants and g determines a gain due to the supplier's size.

A supplier maintains a schedule of all future deliveries and their prices; it is paid after delivering the service. Each supplier is audited at regular periods and removed from the market if its profit becomes negative.

3.3.3 Analytical Model

We impose the following set of assumptions on the equilibrium state of the market:

Assumption 1 *Suppliers operate at zero profit without idle periods, and do not discriminate between customers.* This assumption implies that a size s supplier sets its price at a constant level of the work cost $c^{\text{work}}(s)$.

Assumption 2 *The market area of a single supplier is small relative to the size of the whole city,* hence the density of customer arrivals might be considered constant over each supplier's market area.

Assumption 3 *Market areas are circular and do not interfere.*

Assumption 4 *Inside each market area customers arrive at regular intervals.*

Under assumption 1 a supplier of size s should attract a minimum of s customers in its market area per hour. We refer to the smallest circular area that can support a particular supplier as its *support*. The relation between the radius of the support ρ and the frequency of customers' arrival per square mile per hour ν is expressed as

$$s = \pi \rho^2 \nu \tag{3.6}$$

We can use the equations (3.2) to find the frequency ν as a function of the distance from the center r. Consider a part of a city ring with middle radius r and width Δ. For small Δ-s a part of such ring with angular size Δ/r is a nearly square region that accounts for a fraction $\frac{\Delta}{R} \times \frac{\Delta}{2\pi r}$ of all customer arrivals. The frequency ν then can be expressed as the limiting ratio of the fraction of customers appearing in the region to the area of this region by a square:

$$\nu(r) = \lim_{\Delta \to 0} \frac{\frac{\Delta}{R} \times \frac{\Delta}{2\pi r} \lambda^{\mathrm{c}}}{\Delta^2} = \frac{\lambda^{\mathrm{c}}}{2\pi r R} \tag{3.7}$$

From equations 3.6 and 3.7 we derive the radius of the support as

$$\rho(s, r) = \sqrt{\frac{2rRs}{\lambda^{\mathrm{c}}}} \tag{3.8}$$

The next step is to note that under assumptions 1 and 4 in equilibrium the average delay that customers experience inside the support of size s supplier is equal to $1/2s$. Hence the net price that customers face on the boundary of the support is

$$p(s, r) = c^{\mathrm{work}}(1-g)^{s-1} + c^{\mathrm{mile}}\rho(s, r) + c^{\mathrm{hour}}\frac{1}{2s} \tag{3.9}$$

Now we can equate prices at the boundaries of the supports of supplier sizes s_1 and s_2 to find the distance from the center r at which they don't have any advantage over each other. Assuming $s_1 > s_2$ and solving for r we get

$$\bar{r}(s_1, s_2) = \frac{\lambda^{\mathrm{c}}}{2R}\left[\frac{c^{\mathrm{work}}\left[(1-g)^{s_1-1} - (1-g)^{s_2-1}\right] + c^{\mathrm{hour}}\left[\frac{1}{2s_1} - \frac{1}{2s_2}\right]}{c^{\mathrm{mile}}\left[\sqrt{s_2} - \sqrt{s_1}\right]}\right]^2 \tag{3.10}$$

Although equation 3.10 allows us to find the boundaries between each pair of different supplier sizes, we also need a criterion that will show the comparative advantage of one strategy over another as a function of the distance from the center. To derive such criterion we equate prices at the boundaries with size s_2's radius "inflated" by a coefficient γ to show the extra (over strictly necessary) area where s_2 is preferred by customers over s_1. Solving for γ we get a coefficient that is equal to 1 at $\bar{r}(s_1, s_2)$,

and is increasing with size s_2 gaining a comparative advantage over s_1:

$$\gamma(s_1, s_2, r) = \frac{c^{\text{work}}\left[(1-g)^{s_1-1} - (1-g)^{s_2-1}\right] + c^{\text{hour}}\left[\frac{1}{2s_1} - \frac{1}{2s_2}\right]}{c^{\text{mile}}\sqrt{\frac{2rRs_2}{\lambda^c}}} + \sqrt{\frac{s_1}{s_2}} \qquad (3.11)$$

3.3.4 Analysis of the Analytical Model

Note that there are four degrees of freedom in both equations 3.10 and 3.11, namely: (λ^c/R), $(c^{\text{work}}/c^{\text{mile}})$, $(c^{\text{hour}}/c^{\text{mile}})$ and g. The first one determines the size of the simulated city and it is only of interest from the scalability point of view. The other three variables influence the shapes and relative positioning of the city zones where having a particular size of service is relatively advantageous.

Figure 3.3 illustrates how the size gain parameter g and the cost ratios influence the preferences of suppliers. The figure shows three graphs in which size 1 to 5 has the highest advantage over size 3 as a function of the distance from the center and some parameter. In each graph we restrict (λ^c/R) and two other degrees of freedom such that the horizontal line drawn at the middle of each graph is the common intersection of all three.

By studying the graphs in Figure 3.3 we may choose a set of parameters that is of particular interest for us. For example, the common intersection line seems to have all five sizes occupying noticeable niches in the market. Drawing such conclusion, however, requires that the theory and the experimental results have a reasonable degree of compliance. Therefore the next section is devoted to showing that this, in fact, is the case in the suggested model.

3.3.5 Evolutionary Environment

Our test model is a continuous time discrete-event simulation of a society of economic agents: suppliers of a service and their customers. The customer side of the society is assumed to be in equilibrium and does not change its properties in the course of a simulation. The society of suppliers is the one that evolves over time to meet the demands of the customers.

The restriction on the customer side is imposed to fix the scale of the simulation as well as to avoid imposing extra assumptions on the behavior of customers. In fact,

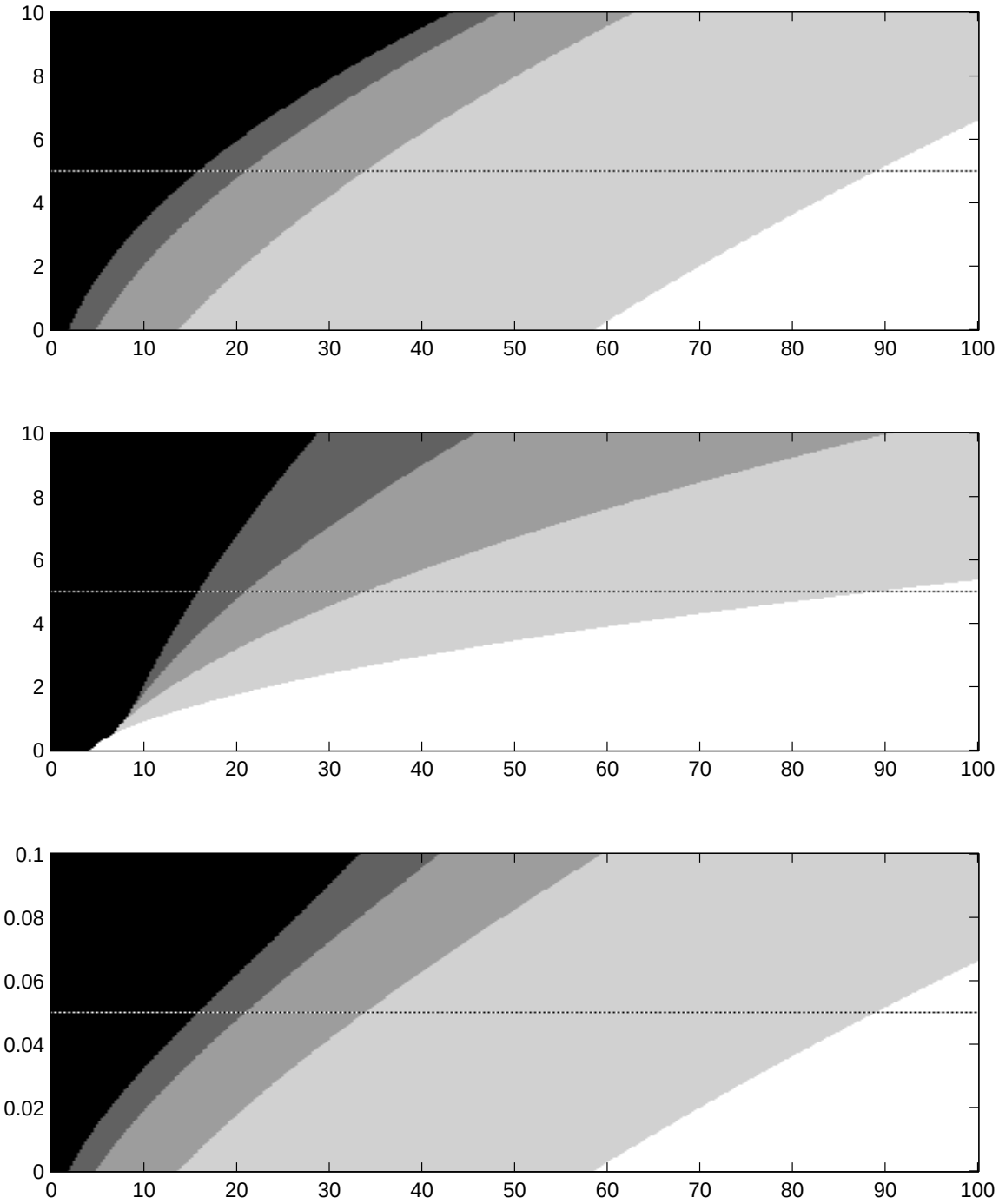

Figure 3.3: Graphs displaying which of sizes from 1 (white) to 5 (black) has the highest comparative advantage over size 3 (medium gray). The x-axis is the distance from the center; the y-axis is $(c^{\text{work}}/c^{\text{mile}})$ for the top figure, $(c^{\text{hour}}/c^{\text{mile}})$ for the middle, and g for the bottom one. The horizontal line in the middle of each graph is the common intersection of all.

it is customary for the urban economics models to assume a fixed distribution of the population density and a particular form of the customers' utility function. In our model though we have the freedom of changing the parameters of the customer side during the simulation.

3.3.6 Supplier Strategies and Generators

We define a pair of supplier's size and pricing strategy to be a *supplier type*. A type is represented in the market by the corresponding *supplier generator*. Each generator maintains a pool of information concerning the history and the current state of its type suppliers. It uses the collected information to create new suppliers of its type and provide them with an initial location, price and, perhaps, other parameters specific to the type.

The probability that a supplier of a particular type will enter the market next is proportional to the number of suppliers of its type that are surviving in the market. There is also a small probability, referred to as *noise*, that a new supplier is assigned a type at random. The noise allows types that were once retired from the market completely to enter it again, perhaps, at a more favorable time. It also suggests a way for completely new types to enter the market, as we demonstrate later in Section 3.3.8.

In the experiments referred to in this chapter we have used two strategies that exhibit sufficiently different behaviors. In both strategies a supplier accepts whatever location and price was suggested by its generator and never alters them. Such restriction simplifies the analysis of the results, since only generators are capable of learning and adapting to the market situation.

The first strategy, code named *market sampler*, involves sampling the city in several locations to maximize the potential revenue flow given the state of the market.The price and the number of samples it takes to place a supplier are assumed to be distributed normally. The corresponding generator periodically collects information on the price and the number of samples from the current suppliers that passed at least one audit. The information is used to estimate the normal distributions mentioned above and merge them with the historical ones.

The second strategy, *price seeker*, assumes that the "right" price and density of the suppliers depend on the distance from the center of the city. The price seeker

generator attempts to estimate the most beneficial distributions of price and density by observing the number and prices of the surviving suppliers. Consequently, the location of a new supplier is chosen to reflect the beliefs of its generator about the density, and the price is selected depending on the distance from the center.

To summarize the important properties of the strategies, the market sampler strategy is capable of pinpointing the best location to deploy a new supplier, yet it assumes the same price distribution for every location. The price seeker strategy can avoid unattractive locations and choose better prices on a global scale, while lacking the precision in positioning its suppliers relative to their rivals.

3.3.7 "Reality Check" Experiments

Our first series of experiments with the Citysim model are aimed at verifying if the derivations from the previous section are close to the results from actual simulations.

The experiments were conducted for the following values of the previously defined parameters: $R = 100$ miles, $\lambda^{\mathrm{c}} = 1200$ or 1600 customers per hour, $c^{\mathrm{mile}} = 2$ \$/mile, $c^{\mathrm{hour}} = 10$ \$/hour. Costs c^{idle} and c^{work} for size 1 suppliers were set to 5 and 10 \$/hour respectively and reduced by $g = 5\%$ with each unit size gain.

In the following we will refer to four experiments: two with λ^{c} equal to 1200 and two with λ^{c} equal to 1600. For each frequency, one experiment involved price seekers of sizes 1 to 5, and the other — market samplers of the same five sizes. The setup with customer frequency λ^{c} equal to 1600 corresponds to the common intersection line in Figure 3.3.

All experiments were sampled after 200 artificial years[2], which corresponds to approximately 2.4 billions of customer entries for $\lambda^{\mathrm{c}} = 1200$ and 3.2 billions for 1600.

Figures 3.4 and 3.5 show the results of the four reference experiments together with theoretic predictions for $\gamma(3, s, \cdot)$ for $s = 1, \ldots, 5$. We use the percentage of supplied capacity as a measure of supplier size's dominance at particular distance from the center.

Observe that the trends in the experimental results follow the theory rather closely.

[2]Technical note: we were using Java 1.4 (Sun JDK) for the simulation software. One 200 "year" experiment takes between 3 and 4 days of Pentium III 650 MHz processor computational time.

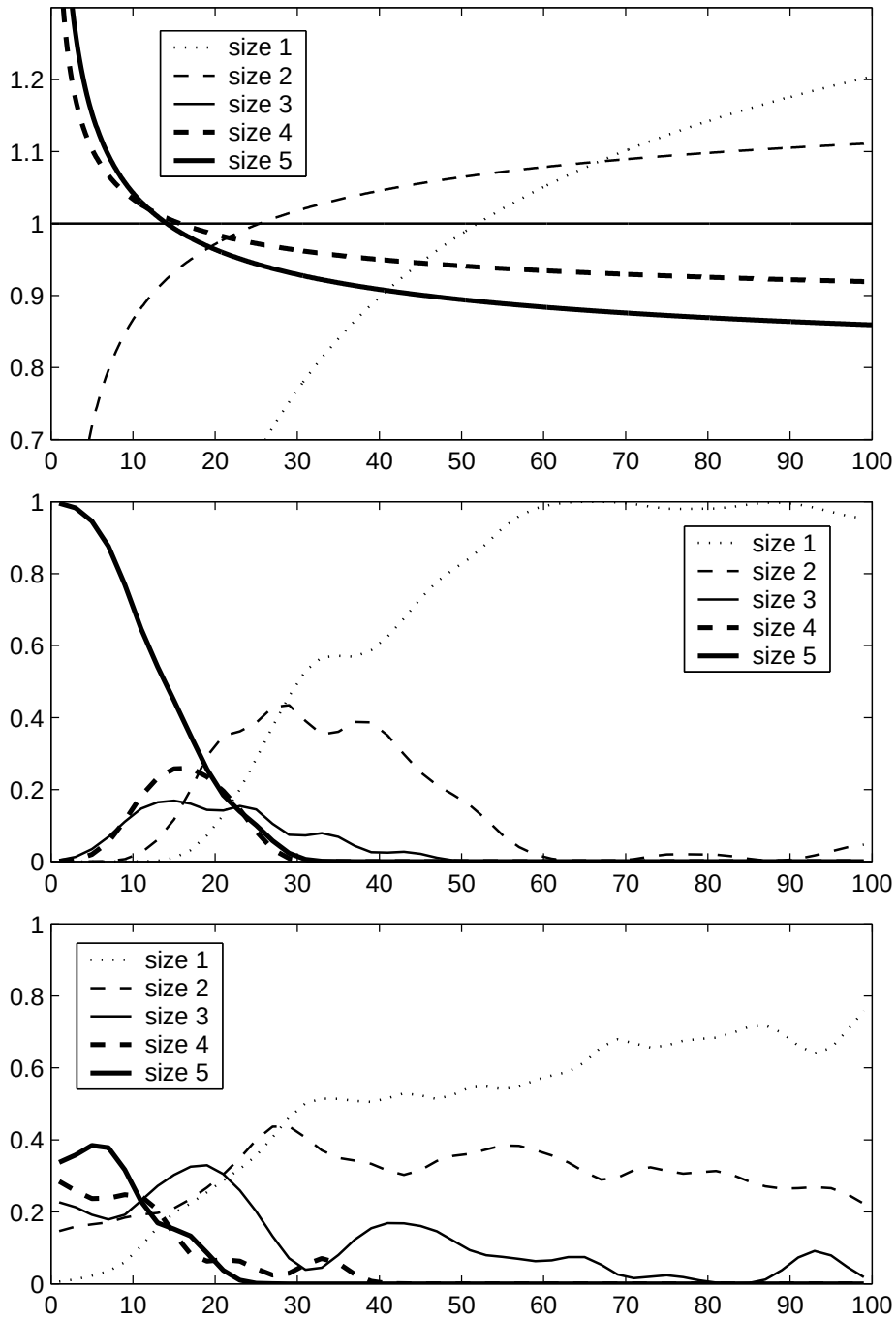

Figure 3.4: Graphs of $\gamma(3, s, \cdot)$ for $s = 1, \ldots, 5$ (top graph), and of approximate percentages of capacity provided by suppliers of each size as a function of the distance from the center. The middle graph is for price seekers, the bottom is for market samplers. Customer frequency λ^c is 1200, simulation time is 200 artificial years.

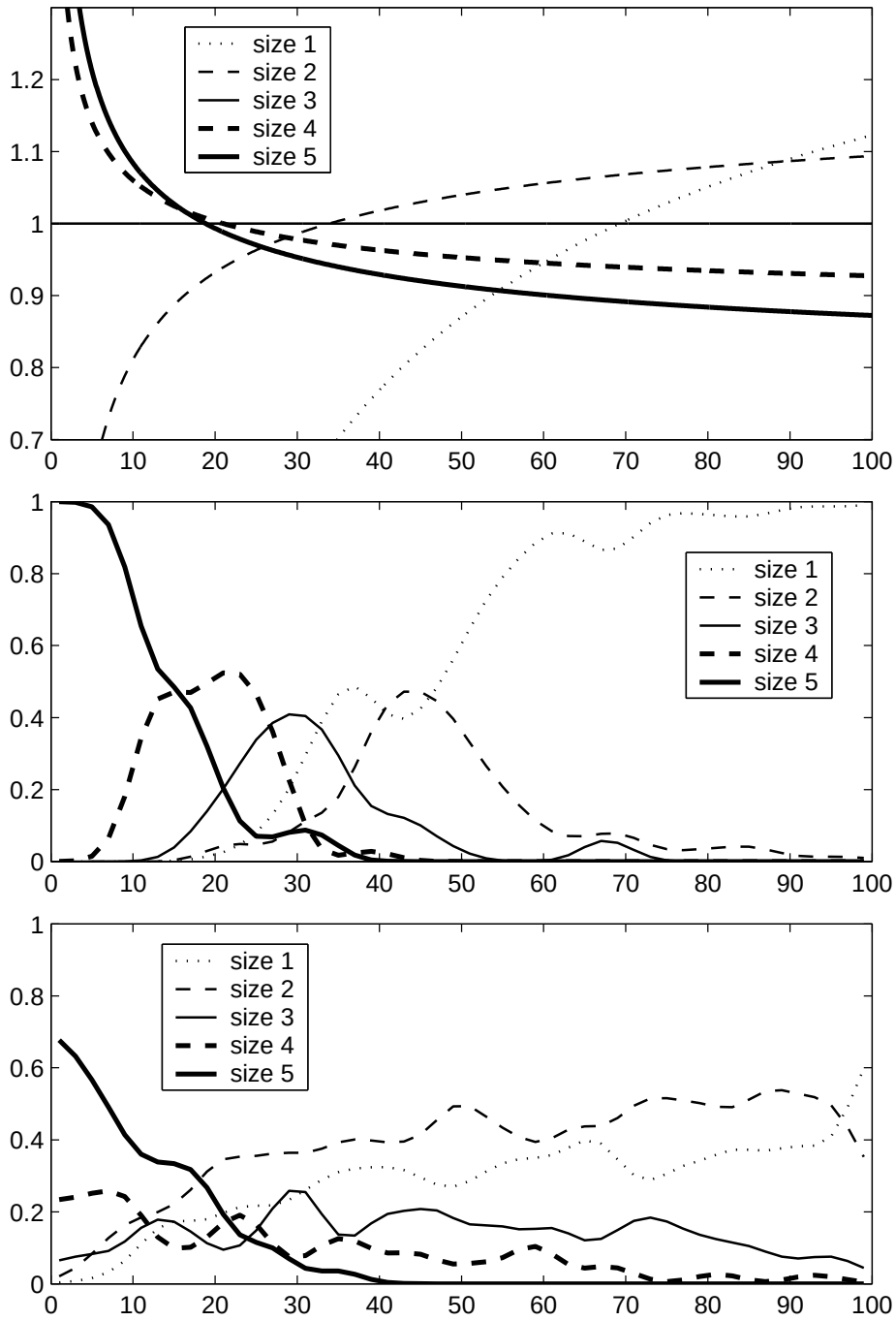

Figure 3.5: Graphs of $\gamma(3, s, \cdot)$ for $s = 1, \ldots, 5$ (top graph), and of approximate percentages of capacity provided by suppliers of each size as a function of the distance from the center. The middle graph is for price seekers, the bottom is for market samplers. Customer frequency λ^c is 1600, simulation time is 200 artificial years.

First, all five sizes acquire considerable shares of the market, with larger suppliers tending to stay in the center of the city. Second, the interval around which size 3 has the highest representation in the market in the experimental results is not much different from the region where $\gamma(3, s, \cdot) \leq 1$ for all $s \neq 3$.

The emerging trends become more clear as the simulations continue. Figure 3.6 shows the states of the simulations for $\lambda^c = 1600$ after 600 artificial years.

It is important to note that although the theory predicts the general state of the market quite satisfactorily, there exist noticeable discrepancies between the results obtained using different supplier strategies. Most spectacular, the separation of zones where one size is preferred over others is much more distinct in the price seeker experiments than in the market sampler ones. This, however, is a sign that the evolutionary experimenting provides a richer framework than the original theoretic model did.

The key to the differences in the results of simulating our two test strategies is in their deployment and pricing mechanisms. In particular, market samplers are designed such that upon deployment they explore the market in many spots and choose the one that promises them a higher revenue flow. This leads to a market structure where smaller suppliers can survive in between market areas of the bigger rivals, much alike a basket of ping-pong balls mixed with a basket of tennis balls fits in less than a two basket volume (see Figure 3.7).

Price seekers, in turn, lack the ability to pick the exact location of their deployment, hence their survival depends greatly on the ability to secure the supporting area by offering the right price. This explains why they follow the presented theory closer — the theory suggests at what distance from the center one should deploy to be able to sell at the minimum reasonable price. Indeed, observing the example of the price seekers' city in Figure 3.8 we may note that it is more common for suppliers to deploy near one another, thus stressing the ability to choose the right price.

One more interesting trend is that from the way the supports, depicted in the bottom part of Figure 3.7, overlap, it is clear that in the simulation the suppliers are able of surviving in higher density then explained by the theory. One reason for this is that in the simulation suppliers are not restricted by assumption 1 of the theory, hence they charge higher prices and derive non-zero profits. Such behavior is necessary for

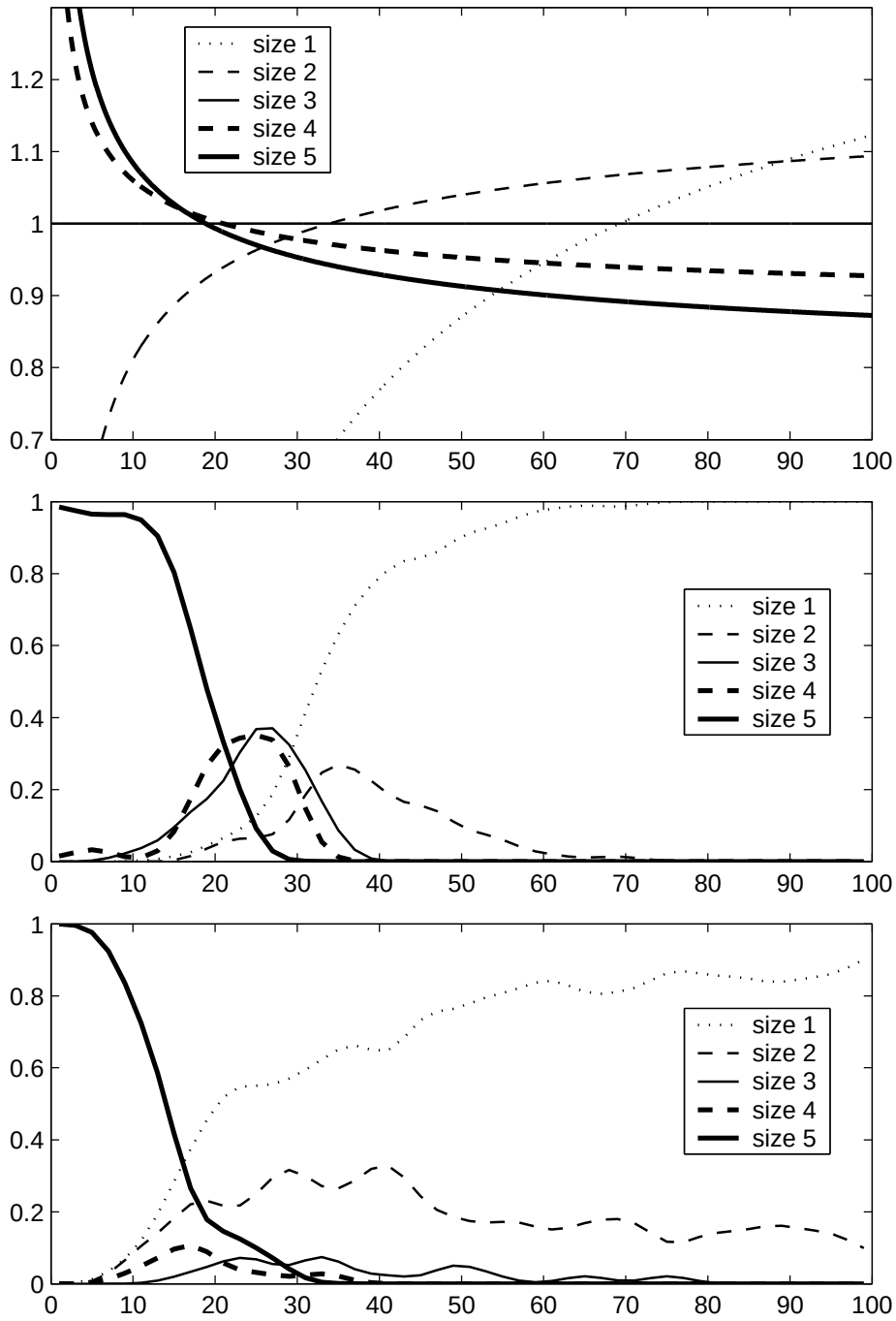

Figure 3.6: Graphs of $\gamma(3, s, \cdot)$ for $s = 1, \ldots, 5$ (top graph), and of approximate percentages of capacity provided by suppliers of each size as a function of the distance from the center. The middle graph is for price seekers, the bottom is for market samplers. Customer frequency λ^c is 1600, simulation time is 600 artificial years.

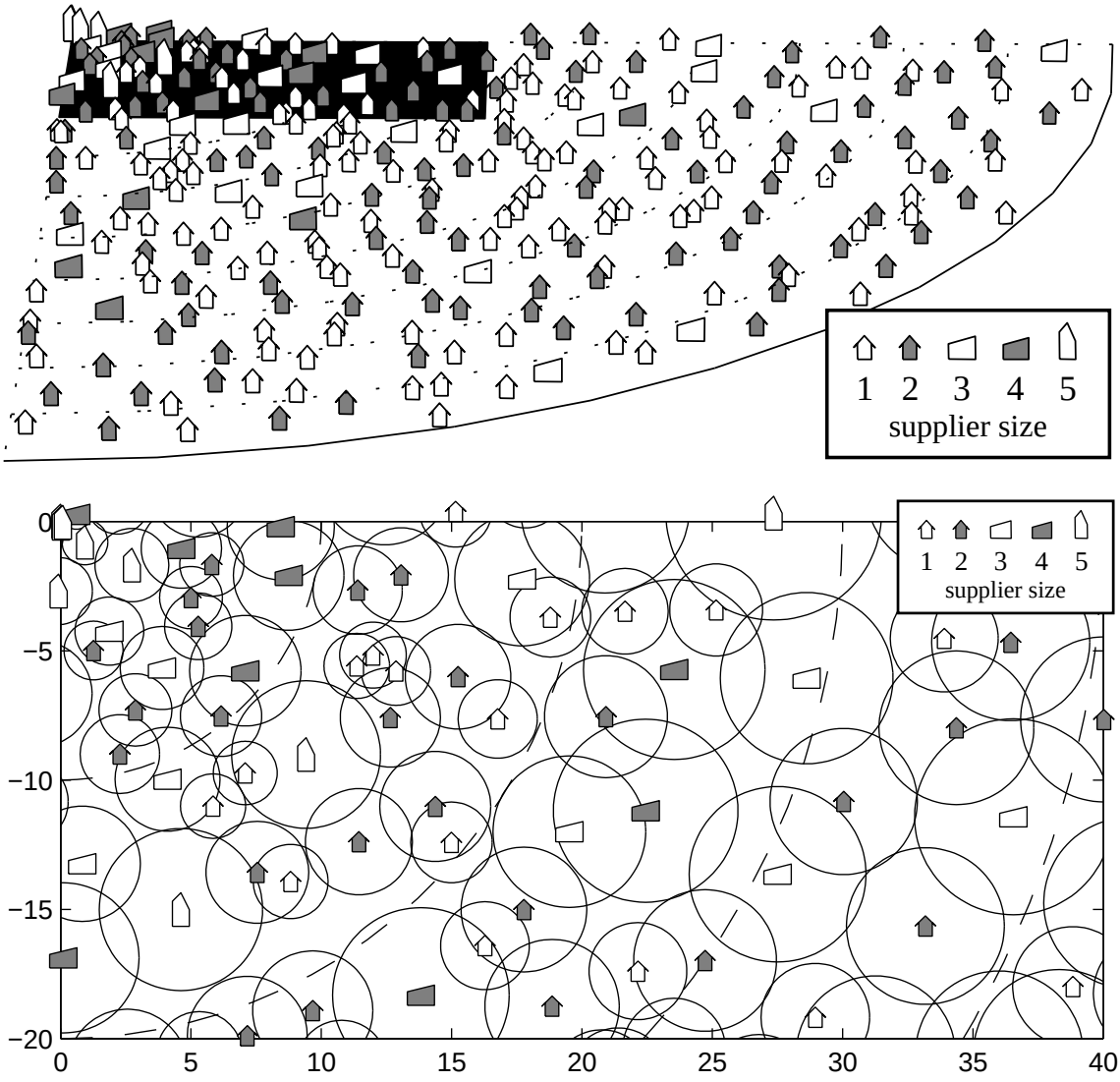

Figure 3.7: One quarter of the city populated by 5 sizes of market samplers after 200 artificial years (top graph). The bottom graph shows a black area of the top graph with theoretic prediction of supplier supports (refer to formula (3.10)). Customer frequency λ^c is 1600.

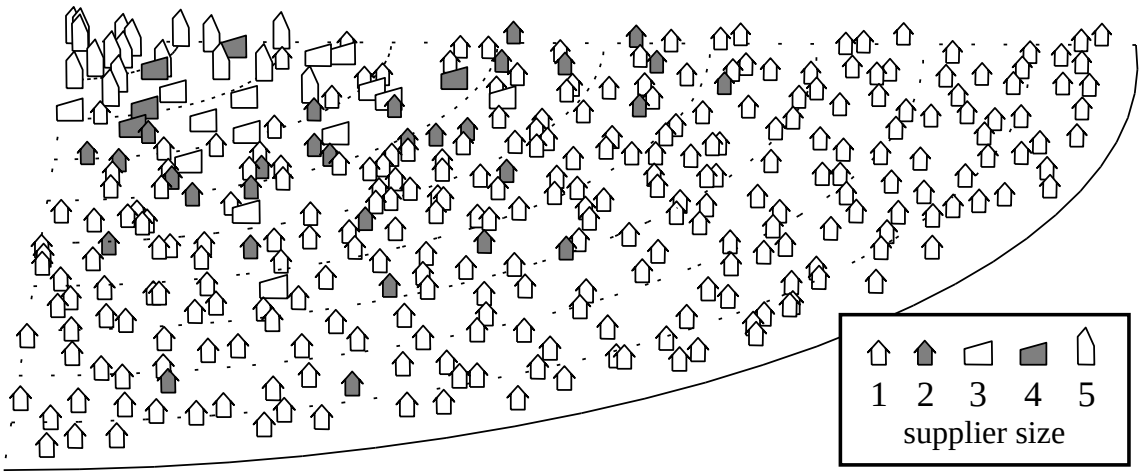

Figure 3.8: One quarter of the city populated by 5 sizes of price seekers after 200 artificial years. Customer frequency λ^c is 1600.

surviving in the evolving market, since the arrival of customers is random, and the positions and prices of rivals are not guaranteed.

Our final observation is the distribution of average service delays faced by the customers upon entry. Figures 3.9 and 3.10 show the distributions of delays for customer entry frequency 1600, two supplier types, and two time points: 200 and 600 years. We can derive several conclusions from the data:

1. average delays are longer near city boundaries than in the center;
2. in some areas delays are close to zero, confirming our guess that suppliers are packed closer than predicted, since at least some of them have idle periods;
3. in general, average delays are not much different from what is assumed in theoretical derivations, i.e. $1/2s$ for a supplier of size s;
4. in market seekers' case, time delays become considerably more evenly distributed, as a result of more even distribution of suppliers.

3.3.8 Supplier Strategy Entry Experiments

We devote our next set of experiments to verifying the expectation that new and retired strategies can enter the market by the means of randomly selecting one of them

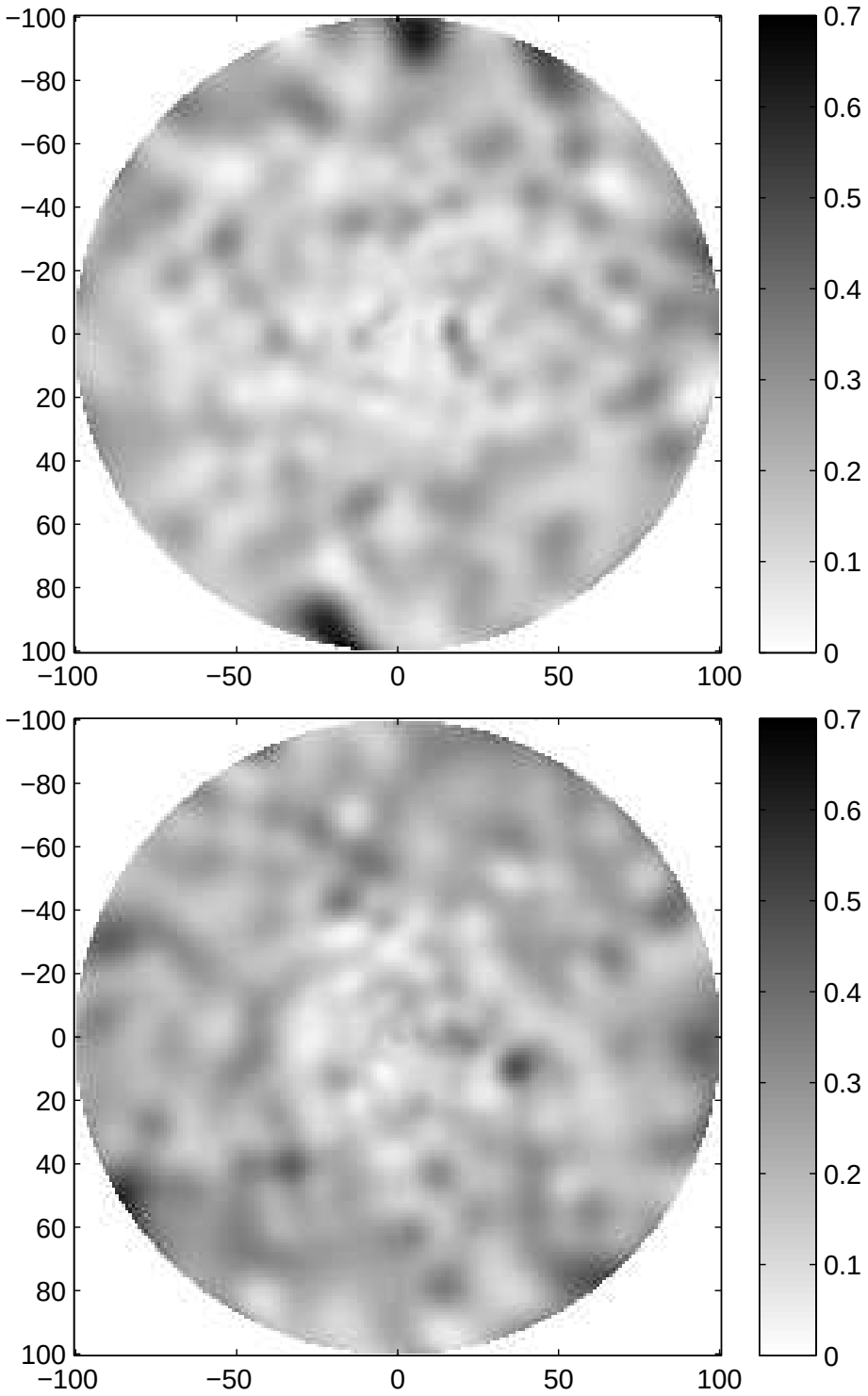

Figure 3.9: Distributions of delays for 5 price seeker strategies, for 200 years (top) and 600 years (top) ($\lambda^c = 1600$).

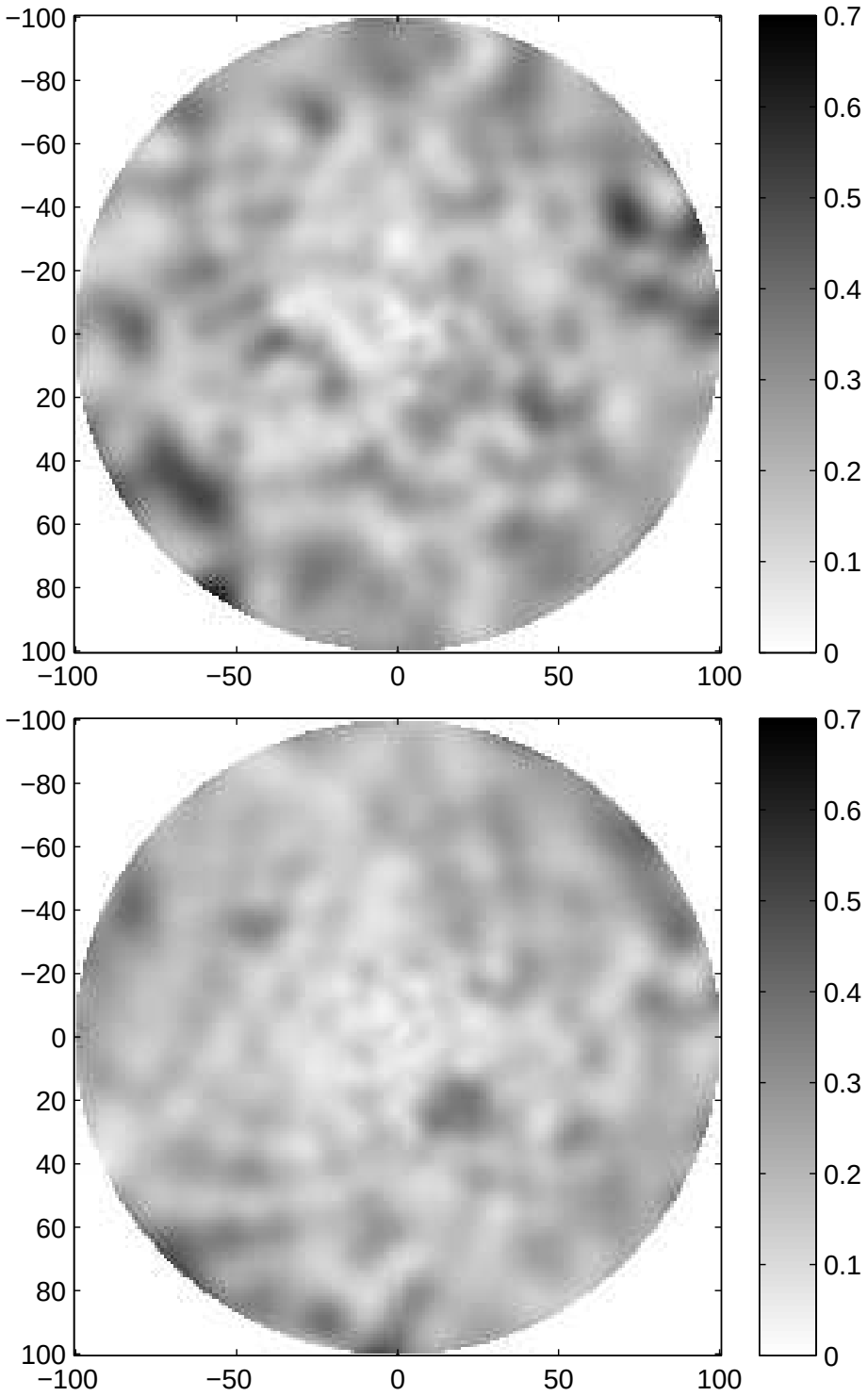

Figure 3.10: Distributions of delays for 5 market sampler strategies, for 200 years (top) and 600 years (top) ($\lambda^c = 1600$).

with a small probability, or noise (refer to Section 3.3.5). In a properly functioning model it should be possible for new types of suppliers to acquire a niche in an existing market. This does not imply that every strategy must win a noticeable representation in the market, only that a reasonably competitive strategy should eventually get its share when the market state is favorable.

In order to verify the viability of our ideas we have conducted a large number of experiments using different market parameters, such as frequencies and costs defined previously. The following analysis applies to an experiment[3], which starts with a city populated by price seeker suppliers of sizes 1, 2 and 3. Later, market sampler suppliers of the same three sizes are introduced to the market via a 10% noise factor (i.e. initially each of the new types gets less than 2% chance every time a supplier is created).

The resulting distribution of entry probabilities for each of the six types is shown in Figure 3.11. Each artificial year, or *milestone* (m/s) in the figure corresponds to $10,000$ simulated hours and roughly 2.5 million of transactions[4]. The market samplers enter the market at milestone 100 when the situation is relatively stable and try to find their niche. Eventually, the size 3 market sampler type proves itself to be competitive and captures a sizable share. In the following, we consider two simulation milestones: one at 110, soon after the market samplers' entry, and the other at 290, at the point of the major success of size 3 market sampler types. These two milestones are depicted in Figure 3.11 by vertical lines.

Figure 3.12 suggests a schematic view of the city at milestone 110. Each supplier is depicted by a house of a particular shape, depending on the supplier's type; price seeker types are colored in white and market samplers are gray. The concentric circles divide the city in ten equally wide zones with each zone getting roughly the same number of customer entries per unit of time.

There are two important observations to be made from Figure 3.12. Firstly, price seekers dominate the market and, indeed, their size 1 suppliers tend to the rim, while size 3 operate mostly in the middle of the city and size 2 form a ring in between

[3]For the previously introduced parameters we used the following values: $R = 50$ miles, $\lambda^c = 250$ customers per hour, $c^{\text{mile}} = 0.5$ \$/mile and $c^{\text{hour}} = 1$ \$/hour. Costs c^{idle} and c^{work} for size 1 suppliers were set to 5 and 10 \$/hour respectively and reduced by 3% with each size unit gain.

[4]The probabilities are kept constant for the first 10 milestones to let the generators adjust their (or rather our) pretty volatile initial guesses about the market situation.

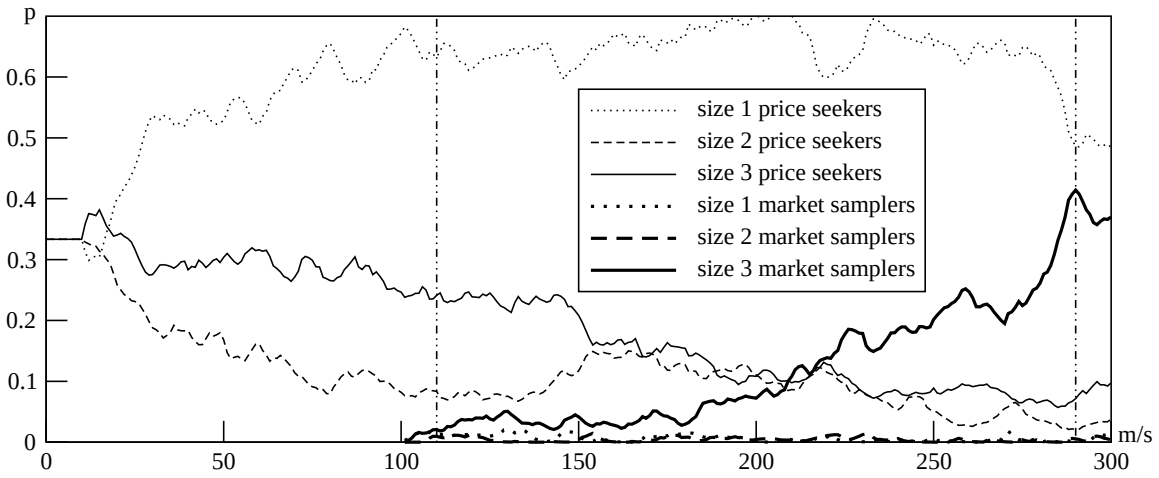

Figure 3.11: Probabilities of a new supplier entry for different supplier types as a function of milestone numbers. Market sampler suppliers are introduced at milestone 100. Vertical lines denote milestones 110 and 290.

the other two types. Secondly, the distribution of suppliers is quite uneven with dense clusters and wide open areas situated at the same distance from the center. The summary of the observations confirms that the price seeker generators have little regard to the exact placement of suppliers, while converging to the right distributions as a whole.

Figure 3.13 presents a view of the city at milestone 290 when the market is dominated by size 3 market samplers and size 1 price seekers. The structure of the market is evidently different from the one we witnessed 10 milestones after the market samplers were introduced — the size 3 market samplers are distributed regularly across the city with the size 1 price seekers surviving in between and in several remaining clusters.

To complete the comparison of the market situations we introduce Figure 3.14 with information on the prices and the costs that customers face at the two considered milestones. The left graph shows average prices and standard deviations for the ten concentric city zones. The right graph shows the *decaying averages*[5] for the customer

[5]The concept of the decaying average is introduced to avoid storage requirements of calculating moving averages in a large scale discrete event simulation. We define the decaying average da_i with a half-life T for the time interval $[t_i, t_{i+1})$ iteratively as

$$w_i = 1 + w_{i-1} 2^{-\frac{t_i - t_{i-1}}{T}} \quad \text{and} \quad da_i = \frac{v_i + (w_i - 1) da_{i-1}}{w_i}$$

where t_i and v_i are respectively the value and the time of the event i, and w_i is the weight of all events from 1 to i at time t_i.

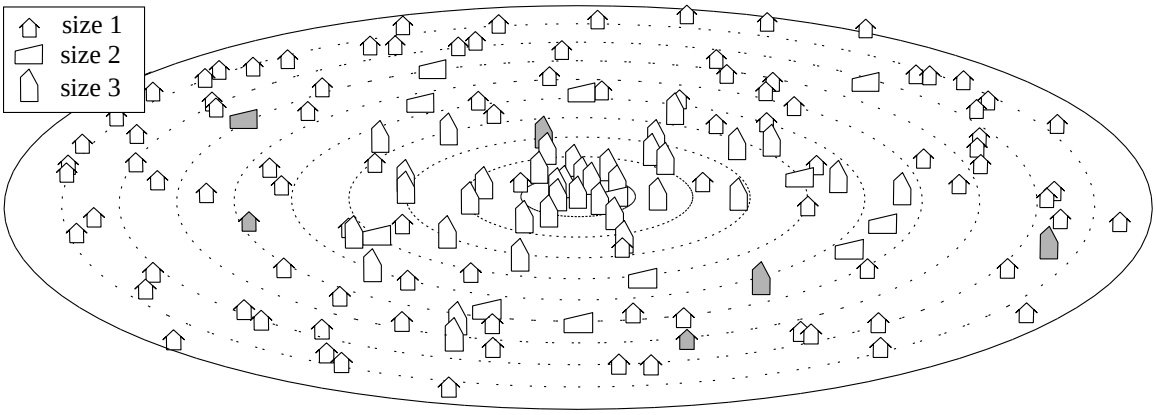

Figure 3.12: City snapshot at milestone 110. Price seeker suppliers are white, market samplers are gray.

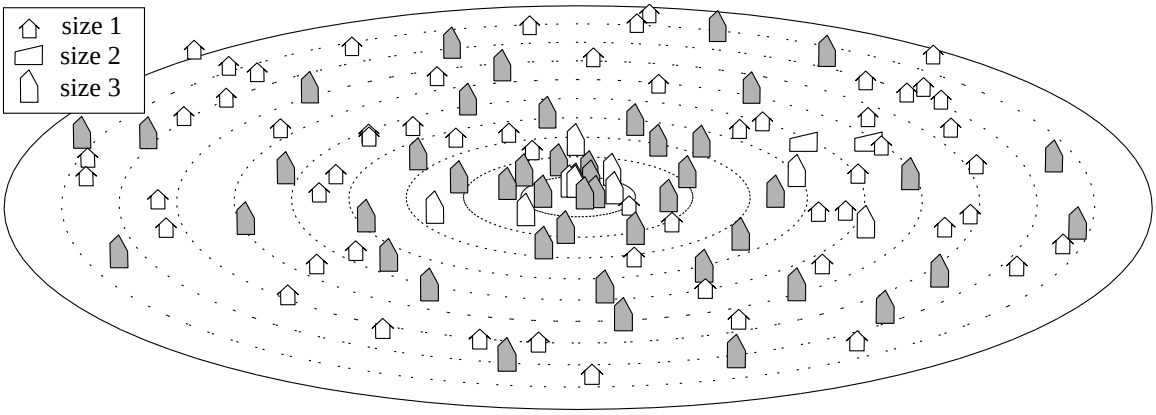

Figure 3.13: City snapshot at milestone 290. Price seeker suppliers are white, market samplers are gray.

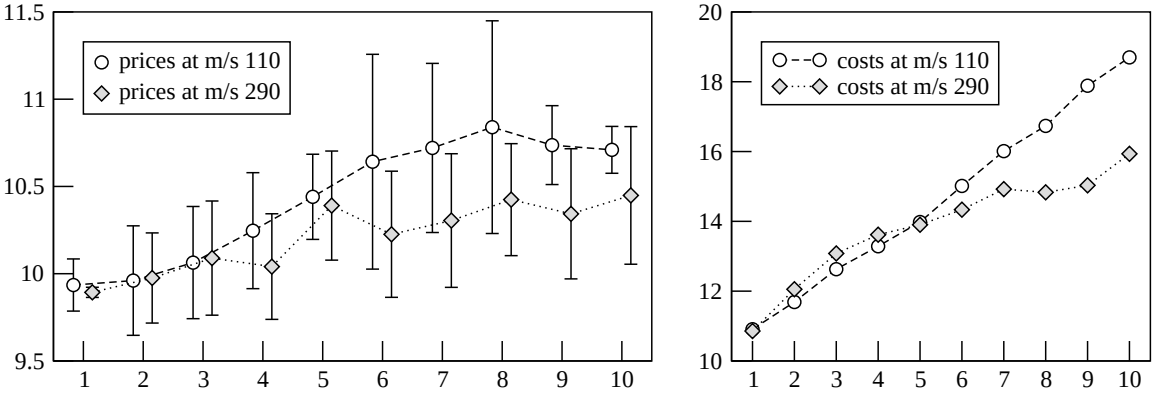

Figure 3.14: Average supplier prices with standard deviations (left) and 25 hour half-life decaying averages of customer costs (right) for 10 concentric city zones at milestones 110 and 290.

costs in the same zones. Both graphs suggest that the market populated with market sampler types exhibits more even price and, subsequently, customer cost distributions.

It should be emphasized here that the most important conclusion of our experiments is that a society of agents that use strategies based on different approaches, information pools, and reproduction methods does evolve as one society and produces meaningful results.

3.4 Concept Check: Trading Agents Competition

Another prospective application of the idea of building a large-scale evolutionary experimentation environment on top of existing systems could be a year 2003 version of the Trading Agent Competition (TAC) environment [TAC, 2003].

In TAC-03 there are three types of agents: customers that order some custom-assembled computers, suppliers of parts, and assemblers. The first two types of agents are fixed and simulated by the market, the challenge is to design competitive strategies for assemblers. To introduce the evolutionary paradigm into the TAC-03 environment we will need to modify the general structure in Figure 3.2 to support three types of agents.

TAC-03 is only a sample application that can benefit from evolutionary testing of heterogeneous strategies. In general, we might divide the technology space in three parts by degree of compatibility of each known or future technology to our evolutionary approach. The first subspace includes technologies that are based on one or another evolutionary methodology, such as Genetic Algorithms or Celluar Automata. Examples of such technologies include ZIP [Cliff and Bruten, 1997], our pilot discrete-event Citysim model or, perhaps, its possible asynchronous implementation.

The second collection of technologies can be restructured to be compatible with the evolutionary paradigm. MAGNET [Collins *et al.*, 2002], as it was demonstrated in the previous section, is one perspective member of this subspace, and the new 2003 revision of TAC is another.

In the third part of the space we count those technologies that are not easily convertable to the evolutionary setup, such as the older versions of TAC. Also in this subspace are technologies whose compatibility was not yet examined, e.g., TAEMS

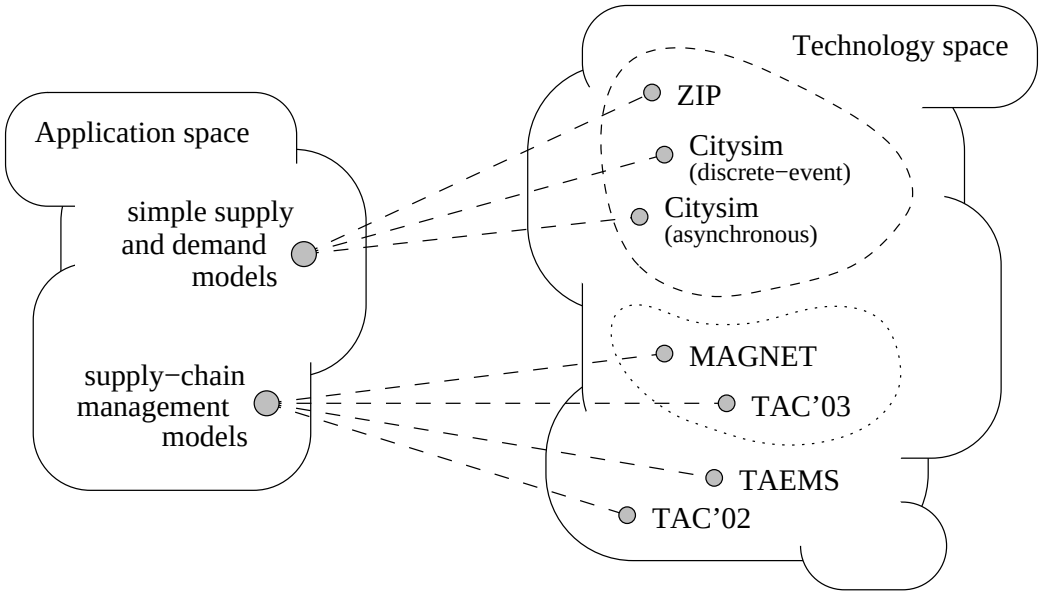

Figure 3.15: The division of the technology space by the degree of compatibility with the evolutionary approach.

framework [Decker, 1996, Lesser *et al.*, 2000].

3.5 Summary

In this chapter we introduced the idea of building a large-scale evolutionary framework for experimentation with heterogenous strategies on top of existing multi-agent systems. We plan to eventually apply this idea to the MAGNET system, however, the complexity of such task suggests that we test the idea in simpler environments. In this chapter we described building a sample system for a simple supply-demand model of the urban economics descent.

The original idea of using the evolutionary framework in application to MAGNET appeared in [Babanov *et al.*, 2002b], and was further refined in [Babanov *et al.*, 2003a, Ketter *et al.*, 2003]. In the outlook, we consider the development of this topic to be in its early stage, and plan to continue pursuing it further on.

Chapter 4

Future Plans

The top priority issue in our plans is to complete the comprehensive study of the expected utility function properties, and to derive sound maximization heuristics as described in Section 2.4.4. We will also examine the possibility of applying stochastic search methods, such as the one based on the combination of the simulated annealing and genetic algorithms approaches that we describe in Section 2.7.1.

The rest of our research agenda can be divided in three relatively independent parts:

1. Develop extensions of the RFQ creation mechanism to include plan repair in case of supplier's failure to deliver, generation of partial and dynamic RFQs, and estimation of the characteristics of a human decision maker, such as her willingness to take risks (Section 2.7.4).

2. Integrate the suggested RFQ construction theory with other MAGNET subprojects, in particular, with the evolutionary framework for large-scale experimentation from Chapter 3. This integration will be used to examine and further refine the theory.

3. Investigate other prospective applications of the developed theory, for example, the Trading Agent Competition as described in Section 3.4. Another family of prospective applications includes real-time systems that traditionally use centralized and collaborative planning mechanisms: resource distribution in multiprocessor systems, bandwidth sharing, planning of transactions in distributed database systems.

The exact sequence in which we will pursue the issues in our research agenda depends on the progress of the related parts of the MAGNET project. We will also adhere to the interest of the academic community to one or another part of our project, and seek opportunities of cooperation with other research groups.

Chapter 5

Related Work

Auctions are becoming a widely accepted mechanism not just for agent-mediated electronic commerce [Guttman *et al.*, 1998, Sandholm, 2002], but also for allocation of tasks to cooperative agents [Hunsberger and Grosz, 2000, Gerkey and Matarić, 2002]. Despite the abundance of work in auctions [McAfee and McMillan, 1987], limited attention has been devoted to auctions over tasks with complex time constraints and interdependencies.

In [Parkes and Ungar, 2001], a method is proposed to auction a shared track line for train scheduling. The problem is formulated with mixed integer programming, with many domain-specific optimizations. Bids are expressed by specifying a price to enter a line and a time window. The bidding language, which is similar to what we use in MAGNET, avoids use of discrete time slots.

Time slots are used in [Wellman *et al.*, 2001], where a protocol for decentralized scheduling is proposed. The study, however, is limited to scheduling a single resource, while MAGNET agents deal with multiple resources. Walsh *et al* [Walsh *et al.*, 2000] propose a protocol for combinatorial auctions for supply chain formation, using a game-theoretical perspective. They allow complex task networks, but do not include time constraints.

Agents in MASCOT [Sadeh *et al.*, 1999] coordinate scheduling with the user. Their major objective is to show policies that optimize schedules locally. Our objective is to optimize the expected customer's utility before bids are submitted and schedules

are finalized.

The results reported in [Watson *et al.*, 2002] on the problem difficulty in job-shop scheduling show many similarities to the problems we encountered in maximizing the certainty equivalent. Our problem is not job-shop scheduling; we are not scheduling resources the agent has, but we are producing a schedule of tasks that other agents will do. Our objective is to schedule tasks for the RFQ in a way that optimizes the expected utility of the agent. Knowledge of the market, in terms of expected number of bidders and probability of success, play a role in our formulation, as well as knowledge of the risk aversity of the agent.

Expected Utility Theory [Pratt, 1964] is an established and well-studied field of Economics, that has attracted many supportive as well as critical studies, both theoretical [Machina, 1987, Machina, 1989] and empirical [Smith and Desvousges, 1987, Jullien and Salanié, 2000]. We believe that expected utility will play an increasing role in automated auctions, since it provides a practical way of describing risk estimations and temporal preferences.

Much research has been done in the last few years in studying how agents should price their services or products, and in understanding how interactions among agents affect pricing [Kephart *et al.*, 2000, Kephart and Greenwald, 2001]. Because of the complexity of providing an analytical treatment most of the studies are limited to a small number of agents, and to relatively simple strategies, or are based on simulation. An example is the simulation of dynamic pricing strategies in finite time horizon markets [DiMicco *et al.*, 2001], where results are evaluated in terms of overall profit. There are so many variables in the simulation that it is hard to assess the generality of the results obtained.

Another important result is that even simple strategies can be quite effective. For instance, Cliff's [Cliff and Bruten, 1997] Zero-Intelligence Plus trader agents have minimal intelligence, yet they have been successfully used in continuous double auctions, where they performed very well even when compared to human traders [Das *et al.*, 2001]. More complex strategies can be developed using machine learning techniques, such as, Decision Trees, Q-learning, Neural Networks, and other methods [Russell and Norvig, 2003].

The use of evolutionary methods is proposed in [Park *et al.*, 1999], who simulates

the evolution of the agent population as they adapt their strategy by observing what happens in the environment. Cliff [Cliff, 2001] uses genetic algorithms to learn the parameters that control how his trader agents evolve their pricing strategies. Along similar lines, an evolutionary system based on Genetic Programming is presented in [Phelps *et al.*, 2002]. The major difference with the work presented here, is that we aim at providing a methodology for extending theoretical modeling to deal effectively with complex multi-agent systems with a large number of agents.

Our Citysim has some similarities to the model for electronic service markets presented in [Markopoulos and Ungar, 2001], in the sense that we also take into account customer waiting time and not only the price of the service. In addition to modeling the waiting times, our model includes a geographical component. Transportation costs affect customer choices and consequently affect the decision of where each supplier locates its business.

Chapter 6

Conclusion

We investigated two related topics in applications of groups of intelligent agents to auction-based electronic markets. Our first development concerned the construction of individually rational requests for quotes by a customer agent, which solicit a preferred number of "desirable" bids. The developed mechanism can be useful in environments where agents operate with complex tasks under temporal and precedence constraints.

Our second effort was directed towards designing an experimentation framework for examining agents' strategies, and obtaining data for improving the RFQ generation algorithms. The suggested evolutionary design can be applied to our domain of interest, as well as to other multi-agent electronic markets.

Bibliography

[Allgower and Georg, 1990] E.L. Allgower and K. Georg. *Numerical Continuation Methods: an Introduction*. Springer-Verlag, New York, 1990.

[Anas *et al.*, 1998] Alex Anas, Richard Arnott, and Kenneth A. Small. Urban spatial structure. *Journal of Economic Literature*, 36(3):1426–1464, September 1998.

[Axelrod, 1984] R. M. Axelrod. *The evolution of cooperation*. Basic Books, 1984.

[Axelrod, 1997] Robert Axelrod. *The complexity of cooperation*. Princeton University Press, 1997.

[Babanov *et al.*, 2002a] Alexander Babanov, John Collins, and Maria Gini. Risk and expectations in a-priori time allocation in multi-agent contracting. In *Proc. of the First Int'l Conf. on Autonomous Agents and Multi-Agent Systems*, volume 1, pages 53–60, Bologna, Italy, July 2002.

[Babanov *et al.*, 2002b] Alexander Babanov, Wolfgang Ketter, and Maria Gini. An evolutionary framework for large-scale experimentation in multi-agent systems. In *Toward an Application Science: MAS Problem Spaces and Their Implications to Achieving Globally Coherent Behavior*, Bologna, Italy, July 2002.

[Babanov *et al.*, 2003a] A. Babanov, W. Ketter, and M. Gini. An evolutionary framework for large-scale experimentation in multi-agent systems. In T. Wagner, editor, *MAS Problem Spaces and Their Implications to Achieving Globally Coherent Behavior*. Kluwer, 2003.

[Babanov *et al.*, 2003b] Alexander Babanov, John Collins, and Maria Gini. Asking the right question: Risk and expectation in multi-agent contracting. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 17(4), September 2003.

[Babanov *et al.*, 2003c] Alexander Babanov, John Collins, and Maria Gini. Scheduling tasks with precedence constraints to solicit desirable bid combinations. In *Proc. of the Second Int'l Conf. on Autonomous Agents and Multi-Agent Systems*, Melbourne, Australia, July 2003.

[Cliff and Bruten, 1997] D. Cliff and J. Bruten. Minimalintelligence agents for bargaining behaviors in marketbased environments. Technical Report HPL-97-91, Hewlett Packard Labs, 1997.

[Cliff, 2001] Dave Cliff. Evolutionary optimization of parameter sets for adaptive software-agent traders in continuous double auction markets. Technical Report HPL-2001-99, Hewlett Packard Labs, 2001.

[Collins *et al.*, 2001] John Collins, Corey Bilot, Maria Gini, and Bamshad Mobasher. Decision processes in agent-based automated contracting. *IEEE Internet Computing*, pages 61–72, March 2001.

[Collins *et al.*, 2002] John Collins, Wolfgang Ketter, and Maria Gini. A multi-agent negotiation testbed for contracting tasks with temporal and precedence constraints. *Int'l Journal of Electronic Commerce*, 7(1):35–57, 2002.

[Collins, 2002] John Collins. *Solving Combinatorial Auctions with Temporal Constraints in Economic Agents.* PhD thesis, University of Minnesota, June 2002.

[Das *et al.*, 2001] Rajarshi Das, James E. Hanson, Jeffrey O. Kephart, and Gerald Tesauro. Agent-human interactions in the continuous double auction. In *Proc. of the 17th Joint Conf. on Artificial Intelligence*, Seattle, WA, USA, August 2001.

[de Vries and Vohra, 2001] Sven de Vries and Rakesh Vohra. Combinatorial auctions: a survey. Technical report, Technische Universität München, 2001.

[Decker, 1996] Keith Decker. Taems: A framework for environment centered analysis and design of coordination mechanisms. In *Foundations of Distributed Artificial Intelligence*, pages 429–448. John Wiley & Sons, Inc., January 1996.

[DiMicco *et al.*, 2001] Joan Morris DiMicco, Amy Greenwald, and Pattie Maes. Dynamic pricing strategies under a finite time horizon. In *Proc. of ACM Conf on Electronic Commerce (EC'01)*, October 2001.

[Forrest, 1993] Stephanie Forrest. Genetic algorithms: Principles of natural selection applied to computation. *Science*, 261:872–878, 1993.

[Gaylord and D'Andrea, 1998] Richard J. Gaylord and Louis J. D'Andrea. *Simulating Society*. Springler-Verlag, 1998.

[Gerkey and Matarić, 2002] Brian P. Gerkey and Maja J Matarić. Sold!: Auction methods for multi-robot coordination. *IEEE Trans. Robotics and Automation*, 18(5):758–786, October 2002.

[Guttman *et al.*, 1998] Robert H. Guttman, Alexandros G. Moukas, and Pattie Maes. Agent-mediated electronic commerce: a survey. *Knowledge Engineering Review*, 13(2):143–152, June 1998.

[Hillier and Lieberman, 1990] Frederick S. Hillier and Gerald J. Lieberman. *Introduction to Operations Research*. McGraw-Hill, 1990.

[Hunsberger and Grosz, 2000] Luke Hunsberger and Barbara J. Grosz. A combinatorial auction for collaborative planning. In *Proc. of 4th Int'l Conf on Multi-Agent Systems*, pages 151–158, Boston, MA, 2000. IEEE Computer Society Press.

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

[Kephart and Greenwald, 2001] Jeffrey O. Kephart and Amy R. Greenwald. Shopbot economics. In S. Parsons, P. Gmytrasiewicz, and M.J. Wooldridge, editors, *Game Theory and Decision Theory in Agent-based Systems*. Kluwer Academic Publishers, Netherlands, 2001.

[Kephart *et al.*, 2000] Jeffrey O. Kephart, James E. Hanson, and Amy R. Greenwald. Dynamic pricing by software agents. *Computer Networks*, 32(6):731–752, 2000.

[Ketter *et al.*, 2003] Wolfgang Ketter, Alexander Babanov, and Maria Gini. An evolutionary framework for studying behaviors of economic agents. In *Proc. of the Second Int'l Conf. on Autonomous Agents and Multi-Agent Systems*, Melbourne, Australia, July 2003.

[Krishna, 2002] Vijay Krishna. *Auction Theory*. Academic Press, London, UK, 2002.

[Lesser *et al.*, 2000] Victor Lesser, Bryan Horling, Frank Klassner, Anita Raja, Thomas Wagner, and Shelley XQ. Zhang. BIG: A resource-bounded information gathering and decision support agent. *Artificial Intelligence*, 118(1–2):197–244, May 2000.

[Lindgren, 1997] Kristian Lindgren. Evolutionary dynamics in game-theoretic models. In *The Economy as an Evolving Complex System II*, pages 337–367, 1997.

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

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

[Maes *et al.*, 1999] Pattie Maes, Robert H. Guttman, and Alexandros G. Moukas. Agents that buy and sell: Transforming commerce as we know it. *Comm. of the ACM*, 42(3), March 1999.

[Markopoulos and Ungar, 2001] P. M. Markopoulos and L. H. Ungar. Shopbots and pricebots in electronic service markets. In *Game theory and decision theory in agent-based systems*. Kluwer Academic Publishers, 2001.

[Mas-Colell *et al.*, 1995] Andreu Mas-Colell, Michael D. Whinston, and Jerry R. Green. *Microeconomic Theory*. Oxford University Press, 1995.

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

[Mills and Lav, 1964] Edwin S. Mills and Michael R. Lav. A model of market areas with free entry. *Journal of Political Economy*, 72(3):278–288, 1964.

[O'Sullivan, 1999] Arthur O'Sullivan. *Urban Economics*. McGraw-Hill Higher Education, 1999.

[Park *et al.*, 1999] Sunju Park, Edmund H. Durfee, and William P. Birmingham. An adaptive agent bidding strategy based on stochastic modeling. In *Proc. of the Third Int'l Conf. on Autonomous Agents*, 1999.

[Parkes and Ungar, 2001] David C. Parkes and Lyle H. Ungar. An auction-based method for decentralized train scheduling. In *Proc. of the Fifth Int'l Conf. on Autonomous Agents*, pages 43–50, Montreal, Quebec, May 2001. ACM Press.

[Phelps *et al.*, 2002] Steve Phelps, Peter McBurney, Simon Parsons, and Elizabeth Sklar. Co-evolutionary mechanism design: a preliminary report. In J. Padget, Onn Shehory, David Parkes, Norman Sadeh, and William Walsh, editors, *Agent Mediated Electronic Commerce IV*, volume LNAI2531, pages 123–142. Springer-Verlag, 2002.

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

[Reeves, 1993] Colin R. Reeves. *Modern Heuristic Techniques for Combinatorial Problems*. John Wiley & Sons, New York, NY, 1993.

[Roth, 2002] Alvin E. Roth. The economist as engineer: Game theory, experimentation, and computation as tools for design economics. *Econometrica*, 70(4):1341–1378, July 2002.

[Russell and Norvig, 2003] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach, 2nd Ed.* Prentice-Hall, Upper Saddle River, NJ, 2003.

[Sadeh *et al.*, 1999] Norman M. Sadeh, David W. Hildum, Dag Kjenstad, and Allen Tseng. MASCOT: an agent-based architecture for coordinated mixed-initiative supply chain planning and scheduling. In *Workshop on Agent-Based Decision Support in Managing the Internet-Enabled Supply-Chain, at Agents '99*, pages 133–138, 1999.

[Sandholm *et al.*, 2002] T. Sandholm, S. Suri, A. Gilpin, and D. Levine. Winner determination in combinatorial auction generalizations. In *Proc. of the First Int'l Conf. on Autonomous Agents and Multi-Agent Systems*, volume 1, pages 69–76, Bologna, Italy, July 2002.

[Sandholm, 2002] Tuomas Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence*, 135:1–54, 2002.

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

[Stone *et al.*, 2002] Peter Stone, Robert E. Schnapire, Micheal L. Littman Janos A. Csirik, and David McAllester. ATTac-2001: A learning, autonomous bidding agent. Submitted to the Eigtheenth National Conference on Artificial Intelligence (AAAI-2002), January 2002.

[Stone, 2002] Peter Stone. Multiagent competitions and research: Lessons from RoboCup and TAC. In *The 6th RoboCup International Symposium*, 2002.

[TAC, 2001] Trading agent competition 2001. http://auction2.eecs.umich.edu/, 2001.

[TAC, 2002] Trading agent competition 2002. http://www.sics.se/tac/, 2002.

[TAC, 2003] Trading agent competition 2003. http://www.sics.se/tac/TAC03_spec.PDF, 2003.

[Veeramani and Joshi, 1998] D. Veeramani and P. Joshi. Emerging agent-based business model for Internet-enabled supply webs. In *Proc. 2nd Intn'l Conf. on Engineering Design and Automation*, August 1998.

[Vega-Redondo, 1996] Fernando Vega-Redondo. *Evolution, Games and Economic Behavior*. Oxford University Press, New York, 1996.

[Walsh *et al.*, 2000] William E. Walsh, Michael Wellman, and Fredrik Ygge. Combinatorial auctions for supply chain formation. In *Proc. of ACM Conf on Electronic Commerce (EC'00)*, October 2000.

[Watson *et al.*, 2002] Jean Paul Watson, J. Christopher Beck, Adele Howe, and L. Darrell Whitley. Problem difficulty for tabu search in job-shop scheduling. *Artificial Intelligence*, 2002.

[Weibull, 1995] Jörgen W. Weibull. *Evolutionary Game Theory*. The MIT Press, 1995.

[Wellman *et al.*, 2001] Michael P. Wellman, William E. Walsh, Peter R. Wurman, and Jeffrey K. MacKie-Mason. Auction protocols for decentralized scheduling. *Games and Economic Behavior*, 35:271–303, 2001.

[Wellman *et al.*, 2003] MP Wellman, JK MacKie-Mason, DM Reeves, and S Swaminathan. Exploring bidding strategies for market-based scheduling. In *Proc. of Fourth ACM Conf on Electronic Commerce*, 2003.

[Zlot *et al.*, 2002] R. Zlot, A. Stentz, M. B. Dias, and S. Thayer. Multi-robot exploration controlled by a market economy. In *Proc. Int'l Conf. on Robotics and Automation*, 2002.