A Grey-Box Approach to Automated Mechanism Design☆,☆☆

J. Niu[a,b,*], K. Cai[a], S. Parsons[a,c], M. Fasli[d], X. Yao[e]

[a]*Department of Computer Science, Graduate Center, CUNY, 365 Fifth Avenue, New York, NY 10016, USA*
[b]*CAISS, City College, CUNY, 137th Street and Convent Avenue, New York, NY 10031, USA*
[c]*Department of Computer and Information Science, Brooklyn College, CUNY, 2900 Bedford Avenue, Brooklyn, NY 11210, USA*
[d]*School of Computer Science and Electronic Engineering, University of Essex, Wivenhoe Park, Colchester CO4 3SQ, UK*
[e]*School of Computer Science, University of Birmingham, Edgbaston, Birmingham B15 2TT, UK*

Abstract

This paper presents an approach to automated mechanism design in the domain of double auctions. We describe a novel parameterized space of double auctions, and then introduce an evolutionary search method that searches this space of parameters. The approach evaluates auction mechanisms using the framework of the TAC Market Design Game and relates the performance of the markets in that game to their constituent parts using reinforcement learning. Experiments show that the strongest mechanisms we found using this approach not only win the Market Design Game against known, strong opponents, but also exhibit desirable economic properties when they run in isolation.

Keywords: Double auction, Mechanism design, Trading agent competition

---

1. Introduction

Auctions play an important role in electronic commerce, and have been used to solve problems in distributed computing. A major problem to solve in these fields is: *Given a certain set of restrictions and desired outcomes, how can we design a good, if not optimal, auction mechanism; and when the restrictions and goals alter, how can the current mechanism be improved to handle the new scenario?*

☆This work was started as part of the first author's doctoral research at the CUNY Graduate Center, and continued as part of the first author's post-doctoral research at the University of Birmingham and the University of Essex.

☆☆This is a revised and substantially extended version of a paper presented at the Twelfth International Workshop on Agent-Mediated Electronic Commerce [1], an extended abstract of which was also presented at the Ninth International Joint Conference on Autonomous Agent and Multi-Agent Systems [2].

*Corresponding author. Tel: +1-212-650-5378.
Email addresses: jniu@gc.cuny.edu (J. Niu), kcai@gc.cuny.edu (K. Cai), parsons@sci.brooklyn.cuny.edu (S. Parsons), mfasli@essex.ac.uk (M. Fasli), x.yao@cs.bham.ac.uk (X. Yao)
URL: http://www.cs.gc.cuny.edu/~jniu/ (J. Niu), http://www.sci.brooklyn.cuny.edu/~parsons/ (S. Parsons), http://cswww.essex.ac.uk/staff/mfasli/ (M. Fasli), http://www.cs.bham.ac.uk/~xin/ (X. Yao)

The traditional answer to this question has been in the domain of auction theory [3]. A mechanism is designed by hand, analyzed theoretically, and then revised as necessary. The problems with the approach are exactly those that dog any manual process — it is slow, error-prone, and restricted to just a handful of individuals with the necessary skills and knowledge. In addition, there are classes of commonly used mechanisms, such as the double auctions that we discuss here, which are too complex to be analyzed theoretically, at least for interesting cases [4].

Automated mechanism design (AMD) aims to overcome the problems of the manual process by designing auction mechanisms automatically. AMD considers design to be a search through some space of possible mechanisms. For example, Cliff [5] and Phelps *et al.* [6, 7] explored the use of evolutionary algorithms to optimize different aspects of the continuous double auction. Around the same time, Conitzer and Sandholm [8] were examining the complexity of building a mechanism that fitted a particular specification.

These different approaches were all problematic. The algorithms that Conitzer and Sandholm considered dealt with exhaustive search, and naturally the complexity was exponential. In contrast, the approaches that Cliff and Phelps *et al.* pursued were computationally more appealing, but gave no guarantee of success and were only searching tiny sections of the search space for the mechanisms they considered. As a result, one might consider the work of Cliff and Phelps *et al.*, and indeed the work we describe here, to be what Conitzer and Sandholm [9] call "incremental" mechanism design, where one starts with an existing mechanism and incrementally alters parts of it, aiming to iterate towards an optimal mechanism. Similar work, though work that uses a different approach to searching the space of possible mechanisms has been carried out by Vorobeychik *et al.* [10] and has been applied to several different mechanism design problems [11].

The problem with taking the automated approach to mechanism design further is how to make it scale — though framing it as an incremental process is a good way to look at it, it does not provide much practical guidance about how to proceed. Our aim in this paper is to provide more in the way of practical guidance, showing how it is possible to build on a previous analysis of the most relevant components of a complex mechanism in order to set up an automated mechanism design problem, and then describing one approach to solving this problem.

2. Grey-box AMD

We propose a *grey-box* AMD approach, which emerged from our previous work on the analyses of the CAT games.

2.1. *From analyses of* CAT *games towards a grey-box approach*

The CAT game, a.k.a. the Trading Agent Competition Market Design game, which has run annually since 2007, asks entrants to design a market for a set of automated traders to trade between each other. The game is broken up into a sequence of *days*, and each day every trader picks a market to trade in, using a market selection strategy that models the situation as an *n*-armed bandit problem [12, Chapter 2]. Each day lasts a certain number of *rounds*, and each round every trader has a chance to place a shout or modify its placed, unmatched shout in the market it chose until the trader completes trading on that day. Traders use standard algorithms for making shouts in a double auction, including ZI-C [13], ZIP [14], RE [15], and GD [16]. Markets are allowed to charge traders in a variety of ways and are scored on the basis of the number of traders they attract (market share), the profits that they make from traders (profit share), and the number of successful transactions they broker relative to the total number of shouts placed in them (transaction success rate). Full details of the game can be found in [17].

We picked the CAT game as the basis of our work for four main reasons. First, the double auctions that are the focus of the design are a widely used mechanism. Second, the competition is run using an open source software package called JCAT which was developed at CUNY and is a good basis for implementing our ideas. Third, after four years of competition, a number of auction mechanisms have been made available by their authors, giving us a library of mechanisms to test against. Fourth, there have been a number of publications that analyze different aspects of previous entrants, giving us a good basis from which to start searching for new mechanisms.

With colleagues we have carried out two previous studies of CAT games [18, 19], which mirror the white-box and black-box analyses from software engineering. [18] provides a white-box analysis, looking inside each market mechanism in order to identify which components it contains, and relating the performance of each mechanism to the operation of its components. [19] provides a black-box analysis, which ignores the detail of the internal components of each market mechanism, but provides a much more extensive analysis of how the markets perform. These analyses make a good combination for examining the strengths and weaknesses of auction mechanisms. The white-box approach is capable of relating the internal design of a mechanism to its performance and revealing which part of the design may cause vulnerabilities, but it requires knowledge of the internal structure of the mechanism and involves manual examination. The black-box approach does not rely upon the accessibility of the internal design of a mechanism. It can be applied to virtually any strategic game, and is capable of evaluating a design in many more situations. However, the black-box approach tells us little about what may have caused a strategy to perform poorly and provides little in the way of hints as to how to improve the strategy. It is desirable to combine these two approaches in order to benefit from the advantages of both. Following the GA-based approach to trading strategy acquisition and auction mechanism design in [5, 7, 20], we propose what we call a *grey-box* approach to automated mechanism design that solves the problem of automatically creating a complex mechanism by searching a structured space of auction components. In other words, we concentrate on the components of the mechanisms as in the white-box approach, but take a black-box view of the components, evaluating their effectivenesses by looking at their performance against that of their peers.

More specifically, we view a market mechanism as a combination of auction rules, each as an atomic building block. We consider the problem: *how can we find a combination of rules that is better than any known combination according to a certain criterion, based on a pool of existing building blocks?* The black-box analysis in [19] maintains a population of strategies and evolves them generation by generation based on their fitnesses. Here we intend to follow a similar approach, maintaining a population of components or building blocks for strategies, associating each block with a *quality score*, which reflects the fitnesses of auction mechanisms using this block, exploring the part of the space of auction mechanisms that involves building blocks of higher quality, and keeping the best mechanisms we find.

Having sketched our approach at a high level, we now look in detail at how it can be applied in the context of the CAT game.

2.2. A search space of double auctions

The first issues we need to address are *what composite structure is used to represent auction mechanisms?* and *where can we obtain a pool of building blocks?*

Viewing an auction as a structured mechanism is not a new idea. Wurman *et al.* [21] introduced a conceptual, parameterized view of auction mechanisms. We extended this framework for auction mechanisms competing in CAT games [18] and provided a classification of entries in the first CAT competition that was based on it. The extended framework includes multiple intertwined components, or *policies*, each regulating one aspect of a market. We adopt this framework, include more candidates for each type of policy and take into consideration parameters that are used by these policies. These policies, each a building block, form a solid foundation for the grey-box approach.

Figure 1 illustrates the building blocks as a tree structure which we describe after we review the blocks themselves.

[Figure 1 about here.]

Below we describe the different types of policies just briefly due to space limitations.

Matching policies, denoted as M in Figure 1, define how a market matches shouts made by traders, including *equilibrium matching* (ME), *max-volume matching* (MV), and *theta matching* (MT). ME clears the market at the equilibrium price, matching asks (offers to sell) lower than the price with bids (offers to buy) higher than the price. MV maximizes transaction volume by considering also less-competitive shouts that would not be matched in ME. MT uses a parameter, $\theta \in [-1, 1]$, to realize a transaction volume that is proportional to 0 and those realized in ME and MV.

Quote policies, denoted as Q in Figure 1, determine the quotes issued by markets, including *two-sided quoting* (QT), *one-sided quoting* (QO), and *spread-based quoting* (QS). Typical quotes are ask and bid quotes, which respectively specify the upper bound for asks and the lower bound for bids that may be placed in a quote-driven market. QT defines the quotes based on information from both the seller side and the buyer side, while QO does so considering only information from a single side. QS extends QT to maintain a higher ask quote and a lower bid quote for use with MV.

Shout accepting policies, denoted as A in Figure 1, judge whether a shout made by a trader should be permitted in the market, including *always accepting* (AA), *never accepting* (AN), *quote-beating accepting* (AQ), *self-beating accepting* (AS), *equilibrium-beating accepting* (AE), *average-beating accepting* (AD), *history-based accepting* (AH), *transaction-based accepting* (AT), and *shout type-based accepting* (AY). AE uses a parameter, w, to specify the size of a sliding window in terms of the number of transactions, and a second parameter, δ, to relax the restriction on shouts [22]. AD is basically a variant of AE and uses the standard deviation of transaction prices in the sliding window rather than w to relax the restriction on shouts. AH is derived from the GD trading strategy and accepts only shouts that will be matched with probability no lower than a specified threshold, $\tau \in [0, 1]$. AY stochastically allows shouts based merely on their types, i.e., asks or bids, and uses a parameter, $q \in [0, 1]$, to control the chances that shouts of either type are allowed to place.

Clearing conditions, denoted as C in Figure 1, define when to clear the market and execute transactions between matched asks and bids, including *continuous clearing* (CC), *round clearing* (CR), and *probabilistic clearing* (CP). CP uses a parameter, $p \in [0, 1]$, to define a continuum of clearing rules with CR and CC being the two ends.

Pricing policies, denoted as P in Figure 1, set transaction prices for matched ask-bid pairs, including *discriminatory k-pricing* (PD), *uniform k-pricing* (PU), *n-pricing* (PN), and *side-biased pricing* (PB). Both PD and PU use a prefixed parameter, $k \in [0,1]$, to control the bias in favor of buyers or sellers, and PB adjusts an internal k aiming to obtain a balanced demand and supply. PN was introduced in [22] and sets the transaction price as the average of the latest n pairs of matched asks and bids.

Charging policies, denoted as G in Figure 1, determine the charges imposed by a market, including *fixed charging* (GF), *bait-and-switch charging* (GB), and *charge-cutting charging* (GC), *learn-or-lure-fast charging* (GL) [23]. GF imposes fixed charges while the rest three policies adapt charges over time in different ways. GL relies upon two parameters, τ and r, to achieve dynamic adjustments. All these charging policies require an initial set of fees on different activities, including fee for registration, fee for information, fee per shout, fee per transaction, and fee on profit, denoted as f_r, f_i, f_s, f_t, and f_p respectively in Figure 1.

These policies were either inferred from the literature [5, 24, 25] (ME, QT, QO, AQ, AY, CC, CR, PD, and PU), contributed by entrants to the CAT competitions (AD is based on personal conversations with the PSUCAT team in CAT 2007), or taken from our previous work [18, 19, 22, 23, 26] (all the rest of the policies listed above). The implementations of ME, QT, AQ, CC, CR, PD, and PU were based on JASA, an open-source single-market simulator that was built by Phelps [27] and contributed by some of us, and all the rest were our original work. An in-depth knowledge of these policies is not required in understanding the grey-box approach, but a full description of these policies can be found in [28].

2.3. *The* GREY-BOX-AMD *algorithm*

The tree model of double auctions in Figure 1 illustrates how building blocks are selected and assembled level by level. There are *and* nodes, *or* nodes, and *leaf* nodes in the tree. An *and* node, rounded and filled, combines a set of building blocks, each represented by one of its child nodes, to form a compound building block. The root node, for example, is an *and* node to assemble policies, one of each type described in the previous section, to obtain a complete auction mechanism. An *or* node, rectangular and filled, represents the decision making of selecting a building block from the candidates represented by the child nodes of the *or* node based on their quality scores. This selection occurs not only for those major aspects of an auction mechanism, i.e. M, Q, A, P, C, and G (at G's child node 'policy' in fact), but also for minor components, for example, a learning component for an adaptive policy (in a similar way to that in which Phelps *et al.* learnt a trading strategy [20]), and for determining optimal values of parameters in a policy, like θ in MT and k in PD. A *leaf* node represents an atomic block that can either be for selection at its parent *or* node or be further assembled into a bigger block by its parent *and* node. A special type of *leaf* node in Figure 1 is that with a label in the format of $[x, y]$. Such a *leaf* node is a convenient representation of a set of *leaf* nodes that have a common parent — the parent of this special *leaf* node — and take values evenly distributed between x and y for the parameter labeled at the parent node.

or nodes contribute to the variety of auction mechanisms in the search space and are where exploitation and exploration occur. We model each *or* node as an n-armed bandit learner that chooses among candidate blocks, and uses the simple softmax method [12, Section 2.3] to solve this learning problem.

Given a set of building blocks, $\mathbb{B}$, and a set of fixed markets, $\mathbb{FM}$, as targets to beat, we define the skeleton of the grey-box algorithm in Algorithm 1. The GREY-BOX-AMD algorithm runs a certain number of steps. At each step, a single CAT game is created and a set of markets are prepared for the game. This set of markets includes all markets in $\mathbb{FM}$, a certain number of markets sampled from the search space, denoted as $\mathbb{SM}$, and a certain number of markets, denoted as $\mathbb{EM}$, chosen from a Hall of Fame, $\mathbb{HOF}$. All these markets are put into the game, which is run to evaluate the performance of these markets. The $\mathbb{HOF}$ has a fixed capacity, and maintains markets that performed well in games at previous steps in terms of their average scores across games they participated in. The $\mathbb{HOF}$ is empty initially, updated after each game, and returned in the end as the result of the grey-box process.

Each market in $\mathbb{SM}$ is constructed based on the tree model in Figure 1. After an 'empty' market mechanism, M, is created, building blocks can be incorporated into M. There are a certain number of different policy types, and from each group of policies of the same type, denoted as $\mathbb{B}_t$ where t specifies the type, a building block is chosen for M. For simplicity, this algorithm illustrates only what happens to the *or* nodes at the high level, including `M`, `Q`, `A`, `C`, and `P`. Markets in $\mathbb{EM}$ are chosen from the $\mathbb{HOF}$ in a similar way.

After a CAT game, G, completes at each step, the game score of each participating market $M \in \mathbb{SM} \cup \mathbb{EM}$, SCORE($G$, M), is recorded and the game-independent score of M, `Score(`M`)`, is updated. If M is not currently in the $\mathbb{HOF}$ and `Score(`M`)` is higher than the lowest score of markets in the $\mathbb{HOF}$, it replaces that corresponding market.

SCORE(G, M) is also used to update the quality score of each building block used by M. Both `Update-Market-Score()` and `Update-Block-Score()` in Algorithm 1 calculate respectively game-independent scores of markets and quality scores of building blocks by averaging feedback `Score(`G, M`)` over time. Because choosing building blocks occurs only at *or* nodes in the tree, only child nodes of an *or* node have quality scores and receive feedback after a CAT game. Initially, quality scores of building blocks are all 0, so that the probabilities of choosing them are even. As the exploration proceeds, fitter blocks score higher and are chosen more

Algorithm 1: The GREY-BOX-AMD algorithm.

Input: $\mathbb{B}$, $\mathbb{FM}$

Output: $\mathbb{HOF}$

```
HOF ← ∅
for s ← 1 to NUM_OF_STEPS do
    G ← Create-Game()
    SM ← ∅
    for m ← 1 to NUM_OF_SAMPLES do
        M ← Create-Market()
        for t ← 1 to NUM_OF_POLICYTYPES do
            B ← Select(B_t, 1)
            Add-Block(M, B)
        SM ← SM ∪ {M}
    EM ← Select(HOF, NUM_OF_HOF_SAMPLES)
    Run-Game(G, FM ∪ EM ∪ SM)
    foreach M ∈ EM ∪ SM do
        Update-Market-Score(M, Score(G, M))
        if M ∉ HOF then
            HOF ← HOF ∪ {M}
        if CAPACITY_OF_HOF < |HOF| then
            HOF ← HOF − {Worst-Market(HOF)}
        foreach B used by M do
            Update-Block-Score(B, Score(G, M))
```

often to construct better mechanisms.

3. Experiment Set I: Learning against classic double auction mechanisms

We carried out two sets of experiments to acquire auction mechanisms using the grey-box approach. The first set of experiments searches the space of auction mechanisms presented above and learn mechanisms for CAT games against classic double auction mechanisms.

3.1. *Experimental setup*

We extended JCAT with the parameterized framework of double auctions and all the individual policies described in Section 2.2. To reduce the computational cost, we eliminated the exploration of charging policies by focusing on mechanisms that impose a fixed charge of 10% on trader profit, which we denote as $\mathsf{GF}_{0.1}$. Analysis of CAT games [19] and what entries have typically charged in actual CAT competitions, especially in the latest two events, suggest that such a charging policy is a reasonable choice to avoid losing either intra-marginal or extra-marginal traders. Even with this cut-off, the search space still contains more than 1,200,000 different kinds of auction mechanisms, due to the variety of policies for aspects other than charging and the choices of values for parameters.

The experiments that we ran to search the space each last 200 steps. At each step, we sample two auction mechanisms from the space, and run a CAT game to evaluate them against four fixed, well known, mechanisms plus two mechanisms that performed well at previous steps and are from the Hall of Fame. The scores of the sampled and Hall of Fame mechanisms are used as feedback for every building block that an individual mechanism uses and is associated with a quality score.

To sample auction mechanisms, the softmax exploration method used by *or* nodes starts with a relatively high temperature ($\tau = 10$) so as to explore randomly, then gradually cools down, τ scaling down by 0.96 (α) each step, and eventually maintains a temperature ($\tau = 0.5$) that guarantees a non-negligible probability of choosing even the worst action any time.[1] After all, our goal in the grey-box approach is not to converge quickly to a small set of mechanisms, but to explore the space as broadly as possible and avoid being trapped in local optima.

The fixed set of four markets in every CAT game includes two CH markets — CH_l and CH_h — and two CDA markets — CDA_l and CDA_h — with one of each charging 10% on trader profit, like $\mathsf{GF}_{0.1}$ does, and the other charging 100% on trader profit (denoted as $\mathsf{GF}_{1.0}$). The CH and CDA mechanisms are two common double auctions and have been used in the real world for many years, in financial marketplaces in particular due to their high allocative efficiency. Earlier experiments we ran, involving CH and CDA markets against entries into CAT competitions, indicate that it is not trivial to win over these two standard double auctions. Markets with different charge levels are included to avoid any sampled mechanisms taking

[1] In calculating the probabilities of choosing actions, the softmax method adjusts the estimated returns of actions in the way in which the maximal return is 1.0 and other returns are set proportionally. Thus the value of the temperature parameter can be set without considering the absolute returns of actions.

advantage otherwise. Based on the parameterized framework in Section 2.2, the CH and CDA markets can be represented as follows:

$$\mathrm{CH}_l = \mathsf{ME} + \mathsf{QT} + \mathsf{AQ} + \mathsf{CR} + \mathsf{PU}_{k=0.5} + \mathsf{GF}_{0.1}$$
$$\mathrm{CH}_h = \mathsf{ME} + \mathsf{QT} + \mathsf{AQ} + \mathsf{CR} + \mathsf{PU}_{k=0.5} + \mathsf{GF}_{1.0}$$
$$\mathrm{CDA}_l = \mathsf{ME} + \mathsf{QT} + \mathsf{AQ} + \mathsf{CC} + \mathsf{PD}_{k=0.5} + \mathsf{GF}_{0.1}$$
$$\mathrm{CDA}_h = \mathsf{ME} + \mathsf{QT} + \mathsf{AQ} + \mathsf{CC} + \mathsf{PD}_{k=0.5} + \mathsf{GF}_{1.0}$$

The Hall of Fame that we maintain during the search contains ten 'active' members and a list of 'inactive' members. After each CAT game, the two sampled mechanisms are compared with those active Hall of Famers. If the score of a sampled mechanism is higher than the lowest average score of the active Hall of Famers, the sampled mechanism is inducted into the Hall of Fame and replaces the corresponding Hall of Famer, which becomes inactive and ineligible for CAT games at later steps. An inactive Hall of Famer may be reactivated if an identical mechanism happens to be sampled from the space again and scores high enough to promote its average score to surpass the lowest score of active Hall of Famers. In addition, the softmax method used to choose two Hall of Famers out of the ten active ones involves a constant $\tau = 0.3$. Since the scores of the Hall of Famers gradually converge in the experiments and the difference between the best and the worst Hall of Famers is less than 25% (see Figure 2b below), this value of τ guarantees that the bias towards the best Hall of Famers is modest and all Hall of Famers have a fairly large chance of being chosen.

Each CAT game is populated by 120 trading agents, using ZI-C, ZIP, RE, and GD strategies, a quarter of the traders using each strategy. Half the traders are buyers, half are sellers. The supply and demand schedules are both drawn from a uniform distribution between 50 and 150. Each CAT game lasts 500 days with ten rounds for each day. This setup is similar to that of actual CAT competitions except for a smaller trader population that helps to reduce computational costs. A 200-step grey-box experiment takes around sixteen hours on a WINDOWS PC that runs at 2.8GHz and has a 3GB memory. To obtain reliable results, we ran the grey-box experiments for 40 iterations and the results that are reported in the next section are averaged over these iterations.[2]

Table 1 summarizes the values of parameters and inputs of Algorithm 1 in our experiments.

[Table 1 about here.]

[2] As we ran these experiments on a busy Linux cluster at the CUNY Graduate Center and our jobs had to run side by side with other jobs, some of which lasted days to complete, we were not able to run the grey-box experiment as many as hundreds of times.

3.2. Experimental results

We collected data and checked whether the grey-box approach is successful in searching for good auction mechanisms in four different ways.

First, we measured the performance of the generated mechanisms indirectly, through their effect on other mechanisms. Since the four standard markets participate in all the CAT games, their performance over time reflects the strength of their opponents — they will do worse as their opponents get better — which in turn reflects whether the search generates increasingly better mechanisms. Figure 2a shows that the scores of the four markets (more specifically the average daily scores of the markets in a game) decrease over 200 games, especially over the first 100 games, suggesting that the mechanisms we are creating get better as the learning process progresses.

Second, we measured the performance of the set of mechanisms we created more directly. The mechanisms that are active in the Hall of Fame at a given point represent the best mechanisms that we know about at that point and their performance tells us more directly how the best mechanisms evolve over time. Figure 2b shows the scores of the ten active Hall of Famers at each step over 200-step runs.[3] As in Figure 2a, the first 100 steps sees a clear, increasing trend. Even the scores of the worst of the ten at the end are above 0.35, higher than the highest of the four fixed markets from Figure 2a. Indeed, Table 2 lists respectively the average scores of the best fixed market, and the best and worst Hall of Famers at the end of the grey-box experiments as well as the standard deviations. At the 95% confidence level, the score of the worst Hall of Famers is significantly higher than that of the best fixed market, CDA_l.

[Table 2 about here.]

Thus we know that our approach will create mechanisms that outperform standard mechanisms, though we should not read too much into this since we trained our new mechanisms directly against them.

It should be noted that in Figure 2b and in Figure 2d the scores of the top Hall of Famers descend slightly or reach a plateau after around 100 steps. This happens for two reasons. First, these Hall of Famers face stronger and stronger opponents as the grey-box experiments go on and better mechanisms are sampled and put into the games at latter steps — the same reason caused the descending scores of the fixed markets.

[3] Note that the active Hall of Famers may be different mechanisms at different steps in the process, so, for example, the curve for the best Hall of Famer in the figure may reflect the scores of many different mechanisms, the highest we know of up to the the point when we collected the data.

Second, as the grey-box experiments go on, no new mechanisms can be found and inducted into the Hall of Fame that are able to produce significantly better performance than those existing Hall of Famers. The time when the plateau begins and the level where the plateau resides are both quantitative indicators of the effectiveness of the search process, and provide guidance on, for example, how long a grey-box experiment should run to obtain stable results.

A better test of the new mechanisms than running them against the fixed mechanisms is to run them against those mechanisms that we know to be strong in the context of CAT games, asking what would have happened if our Hall of Fame members had been entered into prior CAT competitions and had run against the carefully hand-coded entries in those competitions. We chose three Hall of Famers from the ten active Hall of Famers obtained in one of the 40 runs to test in this way. These Hall of Famers are internally labeled as SM7.1, SM88.0, and SM127.1 and can be represented in the parameterized framework in Section 2.2 as follows:

$$\texttt{SM7.1} = \mathsf{ME} + \mathsf{QO} + \mathsf{AH}_{\tau=0.4} + \mathsf{CP}_{p=0.3} + \mathsf{PN}_{n=11} + \mathsf{GF}_{0.1}$$

$$\texttt{SM88.0} = \mathsf{ME} + \mathsf{QT} + \mathsf{AA} + \mathsf{CP}_{p=0.4} + \mathsf{PU}_{k=0.7} + \mathsf{GF}_{0.1}$$

$$\texttt{SM127.1} = \mathsf{ME} + \mathsf{QS} + \mathsf{AS} + \mathsf{CP}_{p=0.4} + \mathsf{PU}_{k=0.7} + \mathsf{GF}_{0.1}$$

These three mechanisms were not the top three Hall of Famers produced by that run of grey-box experiment, but were mechanisms that performed consistently well based on our manual examination of the experimental log file.[4] The policies used by these three mechanisms may indicate that they are better choices than their peers, but this should not be over-interpreted. We plotted the probabilities of choosing individual policies at *or* nodes over time through the grey-box experiments and collected statistics on how frequently individual policies appear in the 400 Hall of Famers from the 40 runs. We do not elaborate on these here due to space limitations, but we did observe that certain policies obtained high quality scores over the search process and had more appearances than their peers in the Hall of Famers, e.g., AS (in more than 50% of the Hall of Famers) and CP (with $p = 0.3$ or 0.4 in about 50% of the Hall of Famers). More than two thirds of the Hall of Famers used some version of PN and almost 15% of them used a PU, so the fact that $\mathsf{PU}_{k=0.7}$ appeared in both SM88.0 and SM127.1 and this policy sets the transaction prices in favor of the seller side (in contrast to the common practise with $k = 0.5$) should not be interpreted as a prevailing phenomenon across all the grey-box experiments. A mistake we made in configuring the first set of grey-box experiments was that matching policies were not properly sampled and as a result all market mechanisms used the default ME policy, which explains why the three Hall of Famers here all used ME. We ran these three mechanisms against the best

[4] A more systematic way to choose among the Hall of Famers will be discussed in Section 5 as a piece of future work.

recreation of past CAT competitions that we could achieve given the contents of the TAC agent repository,[5] where competitors are asked to upload their entries after the competition. The CAT games were set up in a similar way to the competitions, populated by 500 traders that are evenly split between buyers and sellers and between the four trading strategies — ZI-C, ZIP, RE, and GD — and the private values of sellers or buyers were drawn from a uniform distribution between 50 and 150. For the recreated competitions, we ran three games for 2007 and 2008 (like in the actual competitions) and ten games for 2009.[6,7]

[Figure 2 about here.]

[Table 3 about here.]

Tables 3a, 3b and 7a list the average cumulative scores of all the markets across the games along with the standard deviations of those scores against entries into CAT 2007, 2008, and 2009 respectively.[8] The three new mechanisms we obtained from the grey-box experiments beat the actual entries into CAT 2007 and CAT 2008 by a comfortable margin in both cases. The fact that we can take mechanisms that we generate in one series of games (against the fixed opponents and other new mechanisms) and have them perform well against a separate set of mechanisms suggests that the grey-box approach learns robust mechanisms. The three new mechanisms failed to win the competition against entries into CAT 2009, but were able to perform better than some of them. The second set of grey-box experiments that is to be described in the next section aims to search for mechanisms that perform well against entries into CAT 2009.

In passing, we note that the rankings of the entries from the repository do not reflect those in the actual CAT competitions. This is to be expected since the entries now face new opponents and different markets will, in general, respond differently to this. Excluding the markets that attempt to impose invalid fees and are marked with '*', we can see that the overall performance of entries from the two recent, actual CAT competitions is significantly better than that of those from the competitions in the previous year respectively when they face the three new, strong, opponents, reflecting the improvement in the entries over time.

[5] `http://www.sics.se/tac/showagents.php`.

[6] It is desirable to run more games for each recreated competition. However some of the entries use a graphical interface, e.g., `MyFuzzy` for CAT 2008 and `IAMwildCAT` and `UMTac` for CAT 2009, which makes it difficult to run games involving these entries repeatedly in an automated manner on our cluster.

[7] When we ran these experiments, CAT 2010 had been held but no entries had been made available in the TAC agent repository so we were unable to recreate the latest competition.

[8] The data for CAT 2009 is placed in a separate table so as to be compared with the data from the second set of experiments that is to be described later.

Mertacor, which did not win the actual 2009 CAT competition, surprisingly beat all other mechanisms by a huge margin. It is unclear whether this is due to a different, improved version of Mertacor uploaded to the TAC agent repository, or some other reason.

Finally, we tested the performance of SM7.1, SM88.0, and SM127.1 when they are run in isolation, applying the same kind of test that auction mechanisms are traditionally subject to. We tested the mechanisms both for allocative efficiency and, following our work in [22], for the extent to which they trade close to theoretical equilibrium as measured by the coefficient of convergence, α, even when populated by minimally rational traders. In [22] we proposed a class of double auctions, called NCDAEE, which can be represented as:

$$\text{NCDAEE} = \mathsf{ME} + \mathsf{AE}_{w,\delta} + \mathsf{CC} + \mathsf{PN}_n$$

The advantage of NCDAEE is that it can give significantly lower α — faster convergence of transaction prices — and higher allocative efficiency (E_a) than a CDA when populated respectively by homogeneous ZI-C traders and can perform comparably to a CDA when populated by homogeneous GD traders.

We replicated these experiments using JCAT and ran additional ones for the three new mechanisms with similar configurations. The results of these experiments are shown in Table 4.[9] The best result in each column is shaded. We can see that both SM7.1 with ZI-C traders and SM88.0 with GD traders give higher E_a than the best of the existing markets respectively, and both of these increases are statistically significant at the 95% level. Both cases also lead to low α, not the lowest in the column but close to the lowest, and the differences between them and the lowest are not statistically significant at the 95% level. Thus the grey-box approach can generate mechanisms that perform as well in the single market case as the best mechanisms from the literature.

[Table 4 about here.]

4. Experiment Set II: Learning against entries from CAT 2009

As the mechanisms we found in the first experiment fail to win over entries in CAT 2009, we carried out a second set of experiments to show how the grey-box approach scales by searching in an extended space that includes policies used in the auction mechanisms of strong entries from CAT 2009. Although

[9] The results we get there are slightly different from those we reported in [22] (in which we used a different platform), but the pattern of these results still holds. In addition, we ran an NCDAEE variant ($\delta = 30$) that was not tested in [22], observing that those with $\delta \leq 20$ do not perform well when populated by GD traders.

there is no formal guarantee, we do expect, in running the second set of grey-box experiments, either to find mechanisms that are able to beat all CAT 2009 entries in a reproduced competition or to confirm that certain entries from CAT 2009 are indeed strong and are identified among the best mechanisms found in the search.

4.1. Experimental setup

When a grey-box search fails to produce mechanisms that meet our goal, just as the mechanisms we found in the first set of experiments are unable to win in the reproduced CAT 2009 competition, there are at least two improvements we can make: first to introduce new auction policies into the search space, and second, to use stronger mechanisms in the fixed set of markets. We consider both types of improvement in the second set of grey-box experiments.

Although the search space in the first set of experiments already includes a variety of policies and some of them are further parameterized, all these policies are simple and fixed, and do not adapt over time within a duration of a single CAT game. The entries in the actual CAT competitions, on the other hand, often adapt the values of parameters in their policies, or switch to different policies over time in response to the adaptation of their opponents [18]. Intuitively, to combat against these complex mechanisms, the policies in our space should incorporate comparable complexity. As our focus in the work of grey-box search is how to automatically search for effective combinations of building blocks, we do not endeavor to design new, complex building blocks manually, which is contrary to our intention of having an approach of automated design. What we can do however is to directly incorporate policies used by these CAT 2009 entries into our search space.

We intended to incorporate at least policies used by those entries that ranked higher than the mechanisms we found in the first set of experiments as shown in Table 7a, including `Mertacor`, `cestlavie`, `IAMwildCAT`, `jackaroo`, `UMTac`. Both `IAMwildCAT` and `UMTac` however include a graphical component, which will make it impossible to run grey-box experiments on our cluster iteratively. So we eventually considered policies used by the other three entries.

`Mertacor` relies upon collecting information about shouts and transactions in the markets regulated by its opponents. This is different from all the mechanisms we considered so far, in which only information from the market itself is collected and used in its decision making. We introduce a new type of auction policy into the parameterized framework presented in Section 2.2 that regulates this aspect. We call the new type of policy a *subscribing policy*, denoted as `S`, and the default choice, *self subscribing* or `SS`.

Policies used by the three CAT 2009 entries are either among those we introduced previously or their own brew. We name policies in the latter case in such a scheme as, for example, Gaj for the charging policy of jackaroo and Sam for the subscribing policy of Mertacor.

We also introduce a new matching policy, *adaptive matching* or MA, which is a variant of MT. MA sets its parameter θ at 0 to clear the market at the equilibrium point in the first few rounds of a day and increases the value of θ modestly in later rounds of the day so as to increase the transaction success rate.

We add all these new policies into the search space and depict this extension of the tree model in Figure 3. The three CAT 2009 entries can thus be represented respectively as follows:

$$\texttt{Mertacor} = \mathsf{ME} + \mathsf{Q^*} + \mathsf{Aam} + \mathsf{Cam} + \mathsf{Pam} + \mathsf{Gam} + \mathsf{Sam}$$

$$\texttt{cestlavie} = \mathsf{ME} + \mathsf{Q^*} + \mathsf{AE}_{w=10,\delta=25} + \mathsf{CP}_{p=0.7} + \mathsf{Pac} + \mathsf{Gac} + \mathsf{SS}$$

$$\texttt{jackaroo} = \mathsf{ME} + \mathsf{QT} + \mathsf{Aaj} + \mathsf{CR} + \mathsf{Paj} + \mathsf{Gaj} + \mathsf{SS}$$

where Q* represents an arbitrary quote policy as neither Mertacor nor cestlavie use the market quotes.

[Figure 3 about here.]

In addition to extending the search space, we replace three members in the fixed set of markets in the first set of experiments with Mertacor, cestlavie, and jackaroo, and keep the best one, CDA_l, only. Stronger fixed markets may help to speed up the search in the extended space and to some extent avoid the search being trapped in local optima.

The second set of experiments are set up in a similar way to the first set of experiments except that each run of these experiments lasts 600 steps as the search space is bigger and the fixed markets are more difficult to beat. Table 5 lists the part of configuration that differs from that in the first set of experiments.

[Table 5 about here.]

4.2. Experimental results

As previously, we generate the plots of the scores of the fixed markets and the top ten Hall of Famers in the second set of experiments averaged across 30 iterations, which are shown in Figure 4. The best member of the fixed set of markets in the first set of experiments, CDA_l, achieves the lowest score as we expected among the new fixed set of markets, 0.1795 at the last step, which is much lower than its score in the first set of experiments, 0.3101. Also unsurprisingly, as shown in both Figure 4 and Table 6, Mertacor obtains the highest score among the fixed set of markets, 0.4628 at the last step. This score is slightly lower than the score of the top Hall of Famers, 0.4708, however the difference is not significant at the 95% confidence level.

[Figure 4 about here.]

[Table 6 about here.]

Further examination of the Hall of Famers from the 30 runs of the grey-box experiments shows that the mechanism of Mertacor was picked as the top Hall of Famer in steadily more runs over time and was picked in almost half of the runs by the end of the experiment (Figure 5).

[Figure 5 about here.]

The mechanisms that are identified as the top Hall of Famer at the last step in the other runs, though not identical to the mechanism of Mertacor, adopt many of the individual policies of Mertacor:

HM0 = ME + Q* + Aam + Cam + Pam + *Gac* + Sam

HM1 = *MA* + Q* + Aam + Cam + Pam + Gam + Sam

HM2 = ME + Q* + *AS* + Cam + Pam + Gam + Sam

HM3 = $MT_{\theta=0.2}$ + Q* + Aam + Cam + Pam + Gam + Sam

where *italic* indicates the policies that differentiate the mechanisms from that of Mertacor. HM1 and HM2 appeared in five and nine runs respectively while HM0 and HM3 appeared in a single run each.

In the same way as we examine the performance of the mechanisms we found in the first set of experiments, we ran a reproduced CAT 2009 competition between the CAT 2009 entries and the Hall of Famers listed above. Table 7b shows the cumulative scores of these mechanisms averaged over ten games. Mertacor still claims the victory, but it scores much less this time than previously if we compare Table 7b with Table 7a. This is to a great extent due to the strong competition from HM0 – HM3, which are virtually variants of Mertacor itself.[10] These Mertacor variants take the second place through the fifth, pushing down those entries that performed well previously, such as cestlavie, jackaroo, and IAMwildCAT. These observations, together with the high scores of Mertacor as shown in Figure 4a, suggest that Mertacor may be the best mechanism that can be found in our extended space of auction mechanisms for CAT games. It also suggests that the competitiveness of Mertacor in CAT games is attributed to its mechanism as a whole and does not hinge upon one or two individual policies alone, as replacing one policy in the mechanism tends to lower the performance of the overall mechanism. It is noteworthy that the score of Mertacor is very close to that

[10]This is also to some extent due to a larger set of players in the games.

of the runner-up, HM2. This indicates that Aam, the only policy that distinguishes Mertacor from HM2, brings little improvement to the mechanism of Mertacor compared to the known policies like AS in HM2.[11]

[Table 7 about here.]

Overall, identifying Mertacor as potentially the best mechanism in the search space suggests that our grey-box approach is effective in exploring the search space and scales well when new building blocks are introduced into the search space.

5. Summary and future work

This paper describes a practical approach to the automated design of complex mechanisms. The approach that we propose breaks a mechanism down into a set of components each of which can be implemented in a number of different ways, some of which are also parameterized. Given a method to evaluate candidate mechanisms, the approach then uses machine learning to explore the space of possible mechanisms, each composed from a specific choice of components and parameters. The key difference between our approach and previous approaches to this task is that the score from the evaluation is not only used to grade the candidate mechanisms, but also the components and parameters, and new mechanisms are generated in a way that is biased towards components and parameters with high scores.

The specific case-study that we used to develop our approach is the design of new double auction mechanisms. Evaluating the candidate mechanisms using the infrastructure of the TAC Market Design competition, we showed that we could either learn mechanisms that can outperform the standard mechanisms that were used to evaluate the learned mechanisms and the best entries in past Market Design competitions or confirm the high competitiveness of a known mechanism in the search space. Even when no better mechanisms can be found than the best known mechanism, the evolved mechanisms could be the starting point for designing better and more complex mechanisms that are beyond the current search space. We also showed that the best mechanisms we learned could outperform mechanisms from the literature even when the evaluation did not take place in the context of the Market Design game. These results make us confident that we can generate robust double auction mechanisms and, as a consequence, that the grey-box approach is an effective approach to automated mechanism design.

[11] Aam is actually a hybrid of AS and AE. It behaves in the same way as AS most of the time and switches to AE only for new shouts that are placed during a certain period of time in a game.

This grey-box search also has potential in identifying weaknesses of a particular mechanism. Mechanisms like the CDA and CH markets, for example, were used in some of our grey-box experiments to evaluate and acquire effective auction mechanisms, which in turn can be viewed as high quality 'attackers' that help to thoroughly examine aspects of those fixed mechanisms. For instance, if we find in a given CAT game that a fixed mechanism receives a score that is much lower than it usually does in other games, we may zoom into the dynamics of the game in a way that is similar to the white-box analysis to see whether a new mechanism takes advantage of flaws in the fixed one. The grey-box method comes in handy in serving this purpose in that it automatically produces a variety of new mechanisms. In this scenario, it does not matter much whether or not these new mechanisms are strong competitors in CAT games, but only matters if they are 'trouble makers'.

There are limitations in our approach and experiments, which motivate several pieces of future work. First, we update the quality scores of building blocks in a mechanism *equally* — every building block receives exactly the same feedback regardless of their contributions — and *independently* — there is no record whether positive or negative feedback comes along with the existence of another policy in the mechanism. This may lead to ineffective feedback and inefficient exploration. One improvement is that heuristic rules may be applied to generate different feedback for updating quality scores of different building blocks in a mechanism. For instance, a mechanism that obtains a bigger profit share than its opponents in a CAT game and charges only on shouts may either have charged higher fees or have had more shouts placed in the market. As a result, stronger feedback should be given to its charging policy and shout accepting policy than that to other parts of the mechanism. Another improvement is that combinations of building blocks may be viewed as composite building blocks and added into the tree model in Figure 1, which helps in recognizing symbiotic building blocks. Auction policies listed in Section 2.2 and those introduced in Section 4.1, more often than not need cooperation of certain other policies, and their contributions to the performance of a market mechanism may hinge on the existence of its buddies. Strong mechanisms are certainly potential places where such symbiotic relations take place. We may add possible combinations of building blocks from these mechanisms into the tree as new branches, and later on identify those mistaken combinations and cut them off using reinforcements from other mechanisms. Here we do not mean to explore all possible combinations. After all, that will lead to an exponentially large search space and does not differ, in essence, from an exhaustive search. What we intend to do is to leverage symbiosis between building blocks, to some extent, so as to produce more accurate causal feedback and explore the space more effectively.

Second, different runs of the grey-box experiment will very likely produce different sets of Hall of Famers and after dozens of runs the number of Hall of Famers will be huge. The three market mechanisms from the first set of grey-box experiments were chosen rather arbitrarily from the 400 Hall of Famers we obtained from 40 runs, and the top Hall of Famers from the second set of experiments won out after a series of games for which the set of players are composed rather randomly. A question that arises is how to choose the best of the best in the end as the output of the grey-box experiments. One way to do so is to use evolutionary game theory [4, 29] and follow an iterative process that is similar to the one in [30] to obtain those Hall of Famers that are more robust than others. The small set of Hall of Famers that are obtained this way may be further used as the fixed markets in another iteration of grey-box experiments so that better mechanisms used as targets may lead to new better mechanisms over iterations.

Third, the fact that new mechanisms that we obtained through the grey-box experiments failed to win games against entries from CAT 2009, `Mertacor` in particular, suggests that novel, better building blocks should be introduced into the pool of building blocks so that better mechanisms can be constructed. Designing brand new building blocks requires domain knowledge and does not contribute much to the state of the art in a broad range of research fields, however more intelligence and complexity can be incorporated by supporting building blocks of some type that *mixes* the existing ones of the same type. There are at least two kinds of mixing: *concurrent* and *sequential*. A concurrent mixed block selects one of multiple pure blocks stochastically with a distribution of probabilities to fulfill its task, as in the concept of mixed strategies in the context of game theory, while a sequential mixed block keeps using one particular pure block over a period of time and switches to another for the next period. Indeed, these two kinds of mixing methods can be integrated in the framework of Markov decision processes in reinforcement learning. These RL-based mixed building blocks are able to significantly contribute to the variety of auction mechanisms, no matter whether these blocks are fixed or allowed to adapt after being incorporated into an auction mechanism. One piece of work that is related to this is [31], where a genetic algorithm is used to acquire diverse, simple strategies for the Iterated Prisoner's Dilemma Tournament and these strategies are then combined to form a meta strategy which chooses the best response among these strategies based on its recent interaction against its opponent. This work provides insights upon how simple solutions can be utilized to build composite, adaptive solutions, although the much more complex interactions in CAT games present challenges.

Fourth, sometimes a solution model or part of it can be over-parameterized, which means that too many parameters are involved and need to be optimized, making it difficult to find a good solution within a reasonable time. This is exactly what happened when we ran grey-box experiments based on the entire tree model, i.e., without cutting off the part of the space that involves charging policies. What we observed in those experiments was that the performance of sampled mechanisms increased very slowly and the convergence did not occur even after the experiment ran for days, which we did not report here. An intuitive way to avoid this, as we did in the first set of grey-box experiments, is to limit the exploration in this part of the search space and instead adopt a known good combination of parameter values, and perhaps to come back to explore only this part of the space when a good understanding of the rest of the space is obtained. We believe that this problem can be dealt with in a more systematic way. For example, the search method can be designed to act automatically in a way similar to what we did manually, although how to partition the entire search space and which part of the space to explore at a certain time needs to be decided intelligently and dynamically.

Fifth and finally, we can allow the strategies of traders to evolve in parallel to the market mechanisms. We have used a fixed set of trading strategies in both the CAT games during the grey-box experiments and those CAT games against entries from prior CAT competitions. The results may vary when different configurations are used for those games. Indeed, as reported in [28, 32], entries from CAT competitions are to some extent sensitive to what their opponents are and how the population of traders are composed. In the real world, traders tend to adapt their strategies based on their experience so as either to take advantage of weaknesses of market mechanisms or to behave more robustly. To this end, we can model the search space of trading strategies as a tree similar to the grey-box approach to auction mechanism design. The existing trading strategies in the literature, their implementations in JCAT, and prior work on trading strategy acquisition [30, 33] together make this task easier. Then two search processes, one in the space of auction mechanisms and the other in the space of trading strategies, can run alternately and iteratively. That is, for example, at one step, we fix the space of trading strategies, generate a population of trading agents according to the landscape that is defined by the quality scores of building blocks of trading strategies, and allow exploration in the space of auction mechanisms through CAT games using those trading agents until a good set of Hall of Famers are obtained or the search reaches a plateau based on certain criteria, and at the next step, we fix the space of auction mechanisms, run parallel markets using the Hall of Famers obtained from the previous step, and allow exploration in the space of trading strategies. These alternate and iterative steps may run either for a number of iterations or until the search process in one of the two search spaces stops producing significantly different landscapes at two adjacent exploration steps. Given the large number of possible strategies and auction mechanisms, solution concepts like Nash equilibrium as used in [30] may not be readily applicable in this scenario. However this alternate, iterative approach is promising to help obtain insights into the complex interaction between markets and trading agents, especially how one side responds to the changes on the other side, a topic on which so far as we are aware, little work has been done.

Acknowledgments

The authors acknowledge financial support from the US NSF under grant IIS-0329037, *Tools and Techniques for Automated Mechanism Design*, and from the UK EPSRC under grants GR/T10657/01 and GR/T10671/01, *Market Based Control of Complex Computational Systems*, and resources from the computational facility at the CUNY Graduate Center. Our grateful thanks go to the entrants to the Market Design game for releasing binaries of their market agents and to the anonymous reviewers for their valuable comments.

References

[1] J. Niu, K. Cai, S. Parsons, A grey-box approach to automated mechanism design, in: Proceedings of the Twelfth Workshop on Agent-Mediated Electronic Commerce (AMEC XII), Toronto, Canada, 2010.

[2] J. Niu, K. Cai, S. Parsons, A grey-box approach to automated mechanism design, in: Proceedings of the Ninth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010), Toronto, Canada, 2010, extended Abstract.

[3] V. Krishna, Auction Theory, Academic Press, San Diego, CA, 2002.

[4] W. Walsh, R. Das, G. Tesauro, J. O. Kephart, Analyzing complex strategic interactions in multi-agent systems, in: P. Gmytrasiewicz, S. Parsons (Eds.), Proceedings of 2002 Workshop on Game-Theoretic and Decision-Theoretic Agents (GTDT-02), AAAI, Edmonton, Alberta Canada, 2002.

[5] D. Cliff, Evolution of market mechanism through a continuous space of auction-types, Tech. Rep. HPL-2001-326, Hewlett-Packard Research Laboratories, Bristol, England (December 2001).

[6] S. Phelps, P. McBurney, S. Parsons, E. Sklar, Co-evolutionary auction mechanism design: a preliminary report, in: Proceedings of the Workshop on Agent Mediated Electronic Commerce IV (AMEC IV), 2002.

[7] S. Phelps, S. Parsons, E. Sklar, P. McBurney, Using genetic programming to optimise pricing rules for a double auction market, in: Proceedings of the Workshop on Agents for Electronic Commerce, Pittsburgh, PA, 2003.

[8] V. Conitzer, T. Sandholm, Automated mechanism design: Complexity results stemming from the single-agent setting, in: Proceedings of the Fifth International Conference on Electronic Commerce (ICEC'03), ACM Press, Pittsburgh, PA, USA, 2003, pp. 17–24. `doi:http://doi.acm.org/10.1145/948005.948008`.

[9] V. Conitzer, T. Sandholm, Incremental mechanism design, in: Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07), Hyderabad, India, 2007, pp. 1251–1256.

[10] Y. Vorobeychik, D. M. Reeves, M. P. Wellman, Constrained automated mechanism for infinite games of incomplete information, in: Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence, 2007, pp. 400–407.

[11] J. L. Schvartzman, M. P. Wellman, Exploring large strategy spaces in empirical game modeling, in: Proceedings of the 11th Workshop on Agent-Mediated Electronic Commerce, 2009.

[12] R. S. Sutton, A. G. Barto, Reinforcement Learning: An Introduction, MIT Press, Cambridge, MA, 1998.

[13] D. K. Gode, S. Sunder, Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality, Journal of Political Economy 101 (1) (1993) 119–137.

[14] D. Cliff, J. Bruten, Minimal-intelligence agents for bargaining behaviours in market-based environments, Tech. Rep. HPL-97-91, Hewlett-Packard Research Laboratories, Bristol, England (August 1997).

[15] I. Erev, A. E. Roth, Predicting how people play games: Reinforcement learning in experimental games with unique, mixed strategy equilibria, The American Economic Review 88 (4) (1998) 848–881.

[16] S. Gjerstad, J. Dickhaut, Price formation in double auctions, Games and Economic Behavior 22 (1998) 1–29.

[17] K. Cai, E. Gerding, P. McBurney, J. Niu, S. Parsons, S. Phelps, Overview of CAT: A market design competition, Tech. Rep. ULCS-09-005, Department of Computer Science, University of Liverpool, Liverpool, UK, version 2.0. (2009).

[18] J. Niu, K. Cai, S. Parsons, E. Gerding, P. McBurney, Characterizing effective auction mechanisms: Insights from the 2007 TAC Mechanism Design Competition, in: Padgham, Parkes, Müller, Parsons (Eds.), Proceedings of the Seventh International Joint Conference on Autonomous Agents and Multi-agent Systems (AAMAS 2008), Estoril, Portugal, 2008, pp. 1079–1086.

[19] J. Niu, K. Cai, P. McBurney, S. Parsons, An analysis of entries in the First TAC Market Design Competition, in: Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT), Sydney, Australia, 2008.

[20] S. Phelps, M. Marcinkiewicz, S. Parsons, P. McBurney, A novel method for automatic strategy acquisition in n-player non-zero-sum games, in: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'06), ACM Press, New York, NY, USA, 2006, pp. 705–712. `doi:http://doi.acm.org/10.1145/1160633.1160760`.

[21] P. R. Wurman, M. P. Wellman, W. E. Walsh, A parametrization of the auction design space, Games and Economic Behavior 35 (2001) 304–338.

[22] J. Niu, K. Cai, S. Parsons, E. Sklar, Reducing price fluctuation in continuous double auctions through pricing policy and shout improvement rule, in: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), Hakodate, Japan, 2006, pp. 1143–1150.

[23] J. Niu, K. Cai, S. Parsons, E. Sklar, Some preliminary results on competition between markets for automated traders, in: Proceedings of AAAI-07 Workshop on Trading Agent Design and Analysis (TADA-07), Vancouver, Canada, 2007.

[24] K. A. McCabe, S. J. Rassenti, V. L. Smith, Designing a uniform price double auction, in: D. Friedman, J. Rust (Eds.), The Double Auction Market: Institutions, Theories and Evidence, Santa Fe Institute Studies in the Sciences of Complexity, Westview Press, Perseus Books Group, Cambridge, MA, 1993, Ch. 11, pp. 307–332.

[25] P. R. Wurman, W. E. Walsh, M. P. Wellman, Flexible double auctions for electronic commerce: Theory and implementation, Decision Support Systems 24 (1) (1998) 17–27.

[26] J. Niu, S. Parsons, Maximizing matching in double-sided auctions, Working paper, Department of Computer Science, Graduate School and University Center, City University of New York, New York, NY (2011).

[27] S. Phelps, JASA — Java Auction Simulation API, `http://jasa.sourceforget.net/` (2005).

[28] J. Niu, K. Cai, S. Parsons, P. McBurney, E. Gerding, What the 2007 TAC Market Design Game tells us about effective auction mechanisms, Journal of Autonomous Agents and Multiagent Systems 21 (2) (2010) 172–203. `doi:10.1007/s10458-009-9110-0`.

[29] S. Phelps, S. Parsons, P. McBurney, An evolutionary game-theoretic comparison of two double-auction market designs, in: Proceedings of the Workshop on Agent Mediated Electronic Commerce VI (AMEC VI), 2004.

[30] J. L. Schvartzman, M. P. Wellman, Stronger CDA strategies through empirical game-theoretic analysis and reinforcement learning, in: Proceedings of the Eighth International Joint Conference on Autonomous Agents and Multi-Agent Systems, ACM Press, Budapest, Hungary, 2009.

[31] P. J. Darwen, X. Yao, Speciation as automatic categorical modularization, IEEE Transactions on Evolutionary Computation 1 (2) (1997) 101–108.

[32] E. Robinson, P. McBurney, X. Yao, How specialised are specialists? Generalisation properties of entries from the 2008 TAC Market Design Competition, in: Proceedings of the International Trading Agent Design and Analysis Workshop (TADA 2009), Pasadena, CA, USA, 2009.

[33] S. Phelps, M. Marcinkiewicz, S. Parsons, P. McBurney, Using population-based search and evolutionary game theory to acquire better-response strategies for the double-auction market, in: Proceedings of IJCAI-05 Workshop on Trading Agent Design and Analysis (TADA-05), 2005.

List of Figures

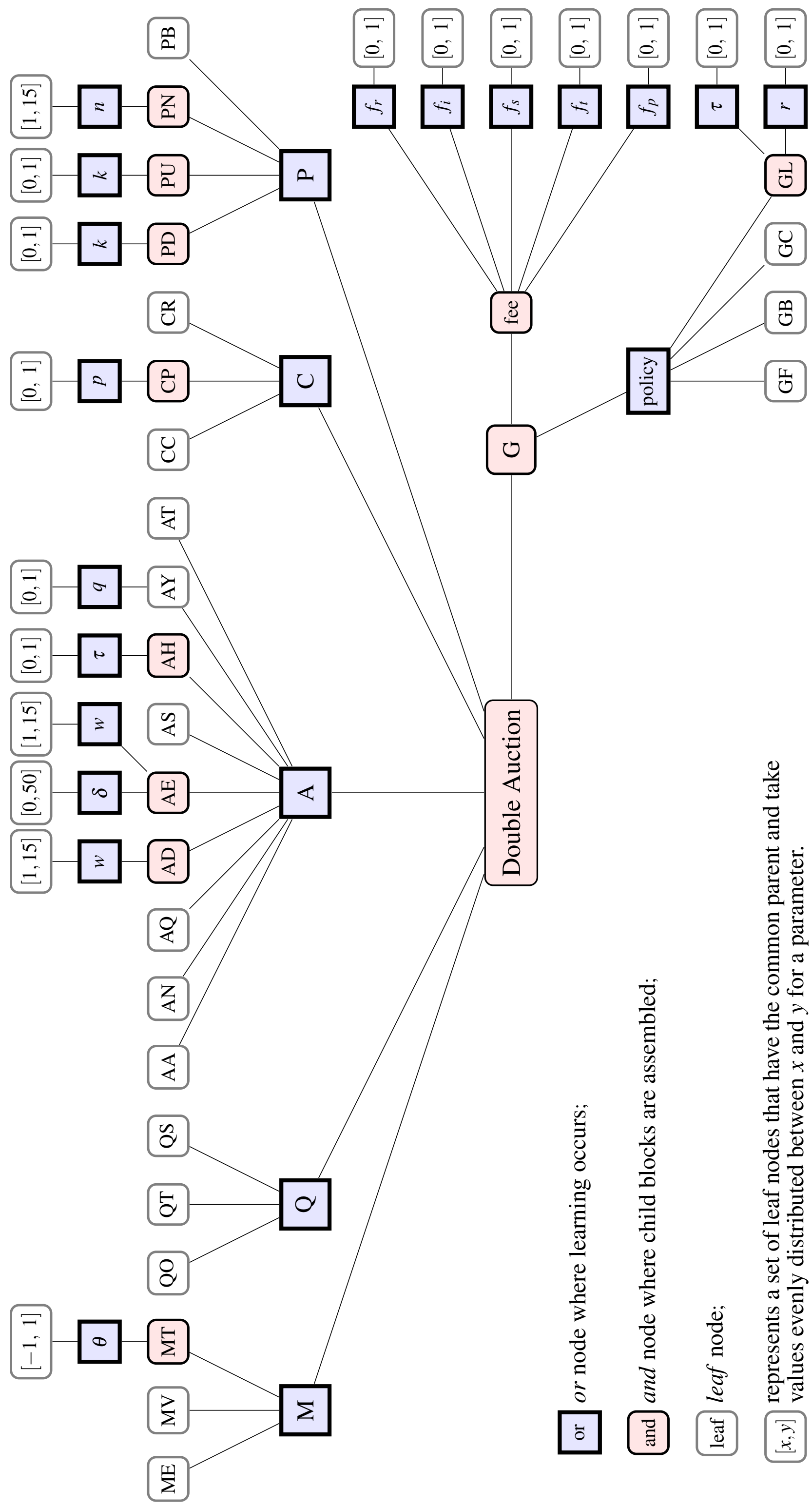
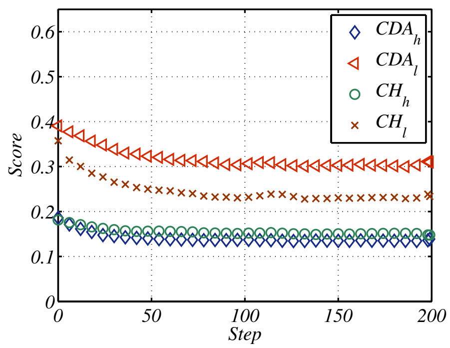
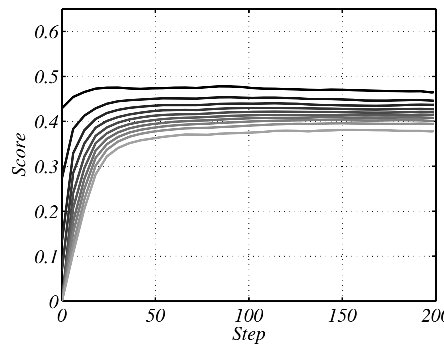


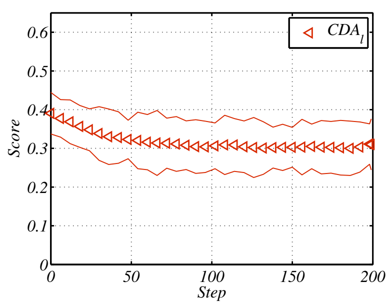
Figure 1: The search space of double auctions modeled as a tree, discussed in detail in Section 2.



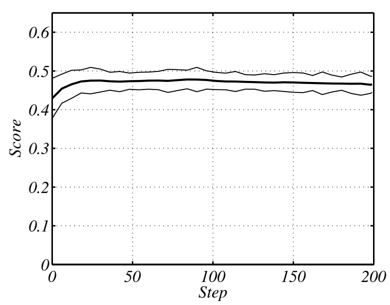
(a) Fixed markets.



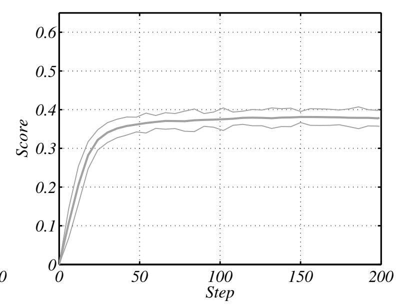
(b) Hall of Famers with the best at the top and the worst at the bottom.



(c) The best fixed market.



(d) The best Hall of Famer(s).



(e) The worst Hall of Famer(s).

Figure 2: Scores of market mechanisms in the first set of grey-box experiments across 200 steps, averaged over 40 runs.

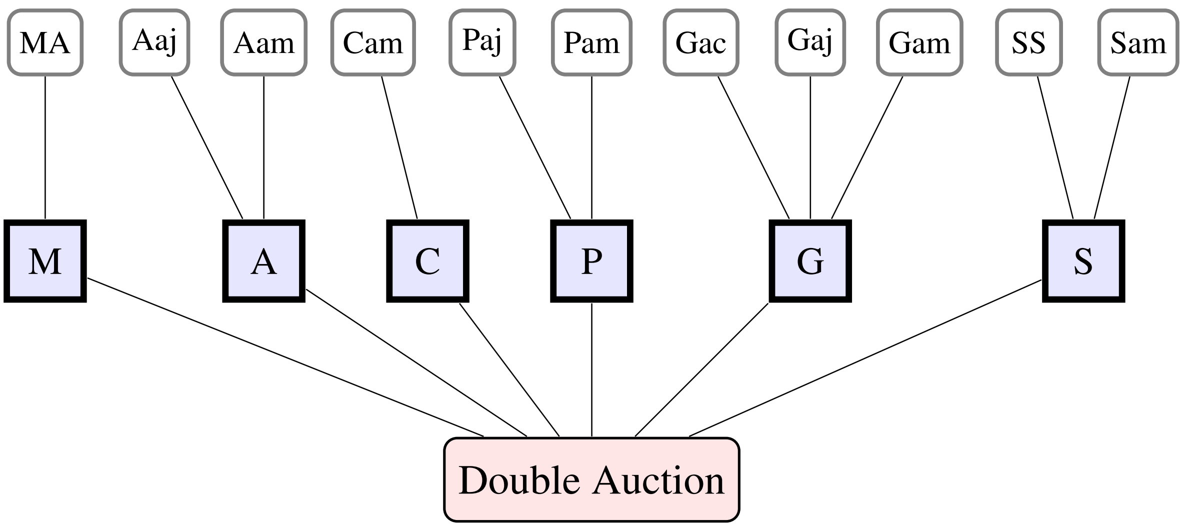
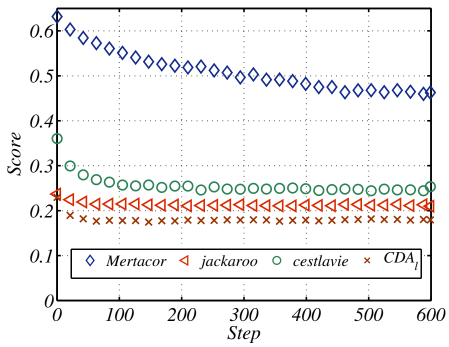
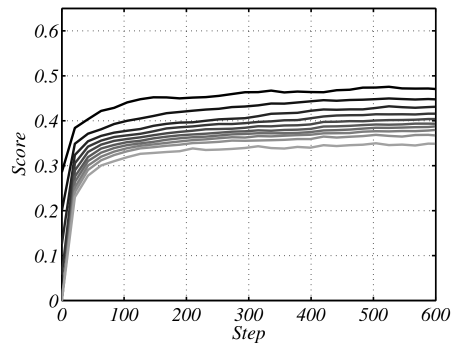


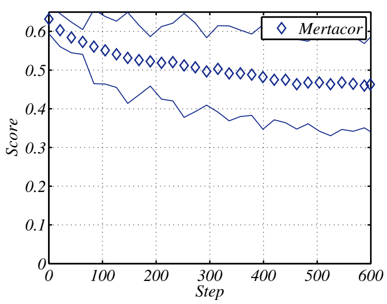
Figure 3: The extension of the search space of double auctions in the second grey-box experiment.



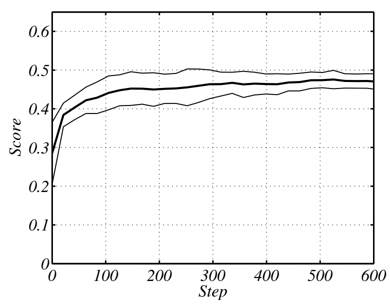
(a) Fixed markets.



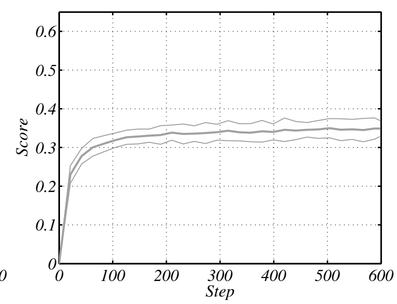
(b) Hall of Famers with the best at the top and the worst at the bottom.



(c) The best fixed market.



(d) The best Hall of Famer(s).



(e) The worst Hall of Famer(s).

Figure 4: Scores of market mechanisms in the second set of grey-box experiments across 600 steps, averaged over 30 iterations.

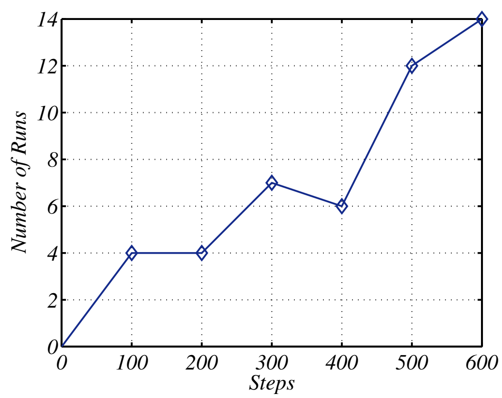


Figure 5: The number of runs of grey-box search out of a total of 30 runs in the second set of experiments that pick `Mertacor` as the best mechanism.

List of Tables

Table 1: The values of parameters and inputs of the GREY-BOX-AMD algorithm in the first set of experiments.

Parameter/Input	Value
NUM_OF_STEPS	200
NUM_OF_SAMPLES	2
NUM_OF_HOF_SAMPLES	4
CAPACITY_OF_HOF	10
NUM_OF_POLICYTYPES	5
initial τ_0 *	10
minimal τ_0 *	0.5
α_0 *	0.96
τ_1 †	0.3
α_1 †	1
$\mathbb{FM}$	$\{\mathrm{CH}_l, \mathrm{CH}_h, \mathrm{CDA}_l, \mathrm{CDA}_h\}$

* τ_0 and α_0 are parameters in the softmax solver used by the SELECT($\mathbb{B}_t$, 1) function.

† τ_1 and α_1 are parameters in the softmax solver used by the SELECT($\mathbb{HOF}$, NUM_OF_HOF_SAMPLES) function.

Table 2: The average daily scores of the best fixed market and the best and worst Hall of Famers in the CAT games at the end of the first set of grey-box experiments.

Market	**Mean**	**SD**
Best fixed market (CDA$_l$)	0.3101	0.0659
Best Hall of Famers	0.4652	0.0210
Worst Hall of Famers	0.3790	0.0219

Table 3: The scores of markets in CAT games including the best mechanisms from the grey-box approach and entries in prior CAT competitions, averaged over three CAT games respectively for 2007 and 2008.

(a) Against CAT 2007 entries.

Market	Score	SD
SM7.1	199.4500	5.9715
SM88.0	191.1083	10.3186
SM127.1	180.1277	9.0289
MANX	154.6953	1.3252
Croc'Agent	142.0523	9.0867
TacTex	138.4527	5.8224
PSUCAT	133.1347	5.6565
PersianCat	124.3767	11.2409
jackaroo	108.8017	8.6851
IAMwildCAT*	106.8897	4.4006
Mertacor	89.1707	4.9269

(b) Against CAT 2008 entries.

Market	Score	SD
SM7.1	196.7240	9.2843
SM88.0	186.9247	4.2184
SM127.1	183.5887	9.7835
jackaroo	177.5913	2.5722
Mertacor	161.5440	5.8741
MANX	147.3050	15.7718
IAMwildCAT	142.9167	8.9581
PersianCat	139.1553	17.9783
DOG	130.2197	18.9782
MyFuzzy	125.9630	1.9221
Croc'Agent*	71.4820	5.8687
PSUCAT*	68.3143	6.7389

* IAMwildCAT from CAT 2007, and CrocodileAgent (abbreviated as Croc'Agent in the table) and PSUCAT from CAT 2008 worked abnormally during the games and tried to impose invalid fees, probably due to competition from the three new, strong opponents. Although we modified JCAT to avoid kicking out these markets on those trading days when they impose invalid fees — which JCAT does in an actual CAT tournament — these markets still perform poorly, in contrast to their rankings in the tournaments.

Table 4: Economic properties of the best mechanisms from the first set of grey-box experiments and the auction mechanisms explored in [22]. All NCDAEE mechanisms are configured to have $w = 4$ in their AE policies and $n = 4$ in their PN policies. The best result in each column is shaded. Data in the first four rows are averaged over 1,000 runs and those in the last four are averaged over 100 runs.

Market	ZI-C				GD			
	E_a		α		E_a		α	
	Mean	**SD**	**Mean**	**SD**	**Mean**	**SD**	**Mean**	**SD**
CDA	97.464	3.510	13.376	4.351	99.740	1.553	4.360	3.589
$\text{NCDAEE}_{\delta=0}$	98.336	3.262	4.219	3.141	9.756	28.873	14.098	1.800
$\text{NCDAEE}_{\delta=10}$	98.912	2.605	5.552	2.770	23.344	41.727	7.834	5.648
$\text{NCDAEE}_{\delta=20}$	98.304	2.562	7.460	3.136	89.128	30.867	4.826	3.487
$\text{NCDAEE}_{\delta=30}$	97.708	3.136	8.660	3.740	99.736	1.723	4.498	3.502
SM7.1	99.280	1.537	4.325	2.509	58.480	47.983	4.655	4.383
SM88.0	98.320	2.477	11.007	4.251	99.920	0.560	4.387	2.913
SM127.1	97.960	3.225	11.152	4.584	99.520	1.727	4.751	3.153

Table 5: The values of parameters and inputs of the GREY-BOX-AMD algorithm in the second set of experiments that differ from those in the first set of experiments.

Parameter/Input	**Value**
NUM_OF_STEPS	600
$\mathbb{FM}$	$\{$Mertacor, cestlavie, jackaroo, CDA$_l\}$

Table 6: The average daily scores of the best fixed market and the best and worst Hall of Famers in the CAT games at the end of the second set of grey-box experiments.

Market	**Mean**	**SD**
Best fixed market (`Mertacor`)	0.4628	0.1216
Best Hall of Famers	0.4708	0.0197
Worst Hall of Famers	0.3488	0.0200

Table 7: The scores of markets in CAT games including the best mechanisms from the grey-box experiments and entries from CAT 2009, averaged over ten CAT games in both cases.

(a) With mechanisms from the first set of experiments.

Market	Score	SD
Mertacor	241.5715	10.5360
cestlavie	178.8957	3.3455
IAMwildCAT	171.4209	8.3065
jackaroo	161.3124	13.0854
UMTac†	158.6552	7.7849
SM88.0	157.4959	7.9758
SM127.1	150.6758	12.5501
SM7.1	149.7483	15.1307
CUNY.CS	137.5801	5.6975
PSUCAT	134.5170	11.1125
TWBB‡	113.2514	19.8423

(b) With mechanisms from the second set of experiments.

Market	Score	SD
Mertacor	176.5365	24.1721
HM2	176.4945	20.6140
HM3	156.1061	21.1483
HM1	152.3192	18.0645
HM0	152.1263	27.6663
cestlavie	126.8365	14.6078
IAMwildCAT	114.6787	18.2257
jackaroo	114.5572	8.4117
CUNY.CS	93.2921	6.5482
UMTac†	91.5155	17.1831
PSUCAT	90.6562	22.9281
TWBB‡	68.0193	17.6970

† UMTac from CAT 2009 uses fuzzy logic in its mechanism and has a graphical interface to accept certain parameters. As we do not know what parameters should be used, we ran UMTac simply without setting those parameters. It may perform better if the parameters are properly set.

‡ TWBB from CAT 2009 requires a MySQL database in its market making. We were not able to run this on the cluster where we ran the experiments, so the scores of TWBB may not reflect its full capabilities.