A Grey-Box Approach to Automated Mechanism Design

Jinzhong Niu[1*], Kai Cai[2], and Simon Parsons[2,3]

[1] School of Computer Science, University of Birmingham
Edgbaston, Birmingham B15 2TT, UK
jniu@cs.bham.ac.uk
[2] Department of Computer Science, Graduate Center, City University of New York
365, 5th Avenue, New York, NY 10016, USA
[3] Department of Comp. and Info. Science, Brooklyn College, City University of New York
2900 Bedford Avenue, Brooklyn, NY 11210, USA
parsons@sci.brooklyn.cuny.edu

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.

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; or when the restrictions and goals alter, how can the current mechanism be improved to handle the new scenario?*

The traditional answer to this question has been in the domain of auction theory [9]. 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 [21].

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 [2] and Phelps *et al.* [16, 17] explored the use of evolutionary algorithms to optimize different aspects of the continuous double auction. Around the same time, Conitzer and Sandholm [4] were examining the complexity of building a mechanism that fitted a particular specification.

*Part of this work was done while the author was a student at the City University of New York.

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 [5] 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 [20] and has been applied to several different mechanism design problems [18].

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 for the last three years, asks entrants to design a market for a set of automated traders which are based on standard algorithms for buying and selling in a double auction, including ZI-C [8], ZIP [3], RE [6], and GD [7]. 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 [19, Section 2]. Markets are allowed to charge traders in a variety of ways and are scored based on 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 [1].

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 is a good basis for implementing our ideas. Third, after three years of competition, a number of specialists 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 [11, 13], which mirror the white-box and black-box analyses from software engineering. [13] 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. [11] 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 specialists. The white-box approach is capable of relating the internal design of a strategy to its performance and revealing which part of the design may cause vulnerabilities, but it requires internal structure and involves manual examination. The black-box approach does not rely upon the accessibility of the internal design of a strategy. 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 [2, 15, 17], 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 [11] 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.* [22] introduced a conceptual, parameterized view of auction mechanisms. Niu *et al.* [13] extended this framework for auction mechanisms competing in CAT games 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 are either inferred from the literature [10], taken from our previous work [11, 13, 14], or contributed by entrants to the CAT competitions. The set of 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. Below we describe the different types of policies just briefly due to space limitations. An in-depth understanding of these policies is not required in understand the grey-box approach, but a full description of these policies can be found in the extended version of this paper [12].

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 [14]. 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

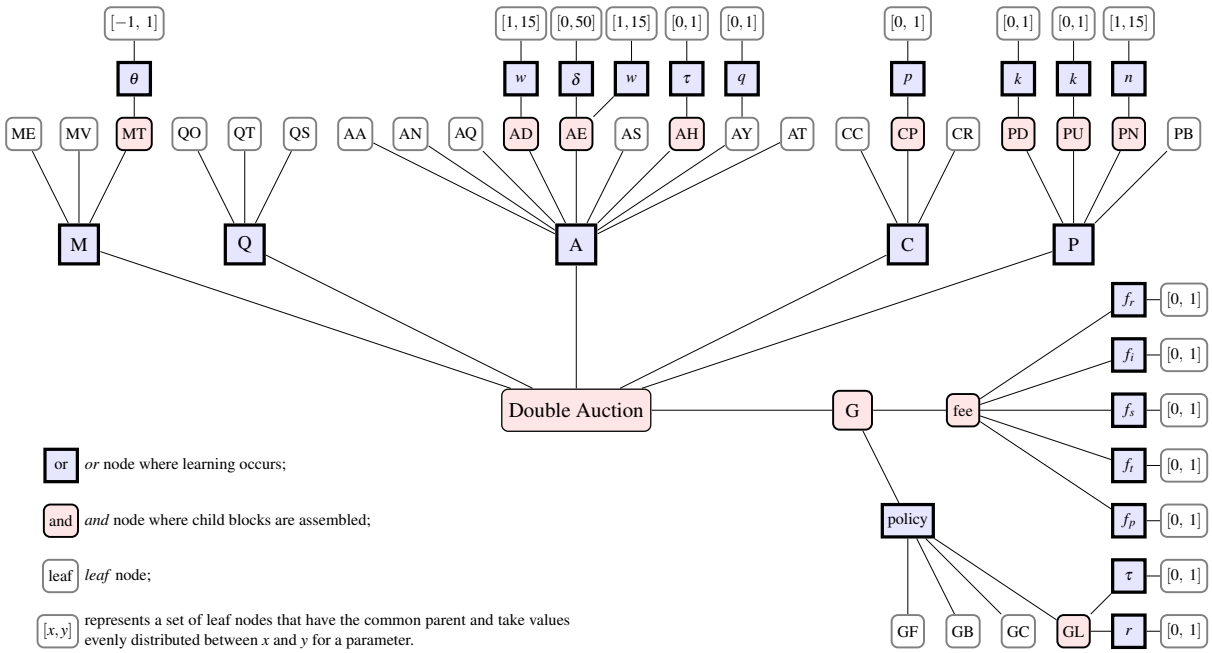


Fig. 1: The search space of double auctions modeled as a tree, discussed in details in Section 2.

obtain a balanced demand and supply. PN was introduced in [14] 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). 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 on registration, fee on information, fee on shout, fee on transaction, and fee on profit, denoted as f_r, f_i, f_s, f_t, and f_p respectively in Figure 1.

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 of 'policy' in fact), but also for minor components, for example, a learning component for an adaptive policy (following Phelps *et al.*'s work on acquiring a trading strategy [15]), 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 *or* parent node or be further assembled into a bigger block by its *and* parent 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 use the simple softmax method [19, 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 (NUM_OF_STEPS in Line 2). At each step, a single CAT game is created (CREATE-GAME() in Line 3) and a set of markets are prepared for the game. This set of markets includes all markets in $\mathbb{FM}$, a certain number (NUM_OF_SAMPLES in Line 5) of markets sampled from the search space, denoted as $\mathbb{SM}$, and a certain number (NUM_OF_HOF_SAMPLES in Line 11) 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 (RUN-GAME(G, $\mathbb{FM} \cup \mathbb{EM} \cup \mathbb{SM}$) in Line 12). $\mathbb{HOF}$ has a fixed capacity, CAPACITY_OF_HOF, and maintains markets that performed well in games at previous steps in terms of their average scores across games

```
GREY-BOX-AMD(𝔹, 𝔽𝕄)
 1  ℍ𝕆𝔽 ← {}
 2  for s ← 1 to NUM_OF_STEPS
 3  do G ← CREATE-GAME()
 4     𝕊𝕄 ← {}
 5     for m ← 1 to NUM_OF_SAMPLES
 6     do M ← CREATE-MARKET()
 7        for t ← 1 to NUM_OF_POLICYTYPES
 8        do B ← SELECT(𝔹_t, 1)
 9           ADD-BLOCK(M, B)
10        𝕊𝕄 ← 𝕊𝕄 ∪ {M}
11     𝔼𝕄 ← SELECT(ℍ𝕆𝔽, NUM_OF_HOF_SAMPLES)
12     RUN-GAME(G, 𝔽𝕄 ∪ 𝔼𝕄 ∪ 𝕊𝕄)
13     for each M in 𝔼𝕄 ∪ 𝕊𝕄
14     do UPDATE-MARKET-SCORE(M, SCORE(G, M))
15        if M not in ℍ𝕆𝔽
16          then ℍ𝕆𝔽 ← ℍ𝕆𝔽 ∪ {M}
17        if CAPACITY_OF_HOF < |ℍ𝕆𝔽|
18          then ℍ𝕆𝔽 ← ℍ𝕆𝔽 − {WORST-MARKET(ℍ𝕆𝔽)}
19        for each B used by M
20        do UPDATE-BLOCK-SCORE(B, SCORE(G, M))
21  return ℍ𝕆𝔽
```

Algorithm 1: The GREY-BOX-AMD algorithm.

they participated. $\mathbb{HOF}$ 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 (CREATE-MARKET() in Line 6), building blocks can be incorporated into *M* (ADD-BLOCK(*M*,*B*) in Line 9, where $B \in \mathbb{B}$). NUM_OF_POLICYTYPES in Line 7 defines the number of different policy types, and from each group of policies of same type, denoted as $\mathbb{B}_t$ where *t* specifies the type, a building block is chosen for *M* (SELECT($\mathbb{B}_t$, 1) in Line 8). 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 $\mathbb{HOF}$ in a similar way (SELECT($\mathbb{HOF}$, NUM_OF_HOF_SAMPLES) in Line 11).

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 (UPDATE-MARKET-SCORE(*M*, SCORE(*G*, *M*)) in Line 14). If *M* is not currently in $\mathbb{HOF}$ and SCORE(*M*) is higher than the lowest score of markets in $\mathbb{HOF}$, it replaces that corresponding market (WORST-MARKET($\mathbb{HOF}$) in Line 18).

SCORE(*G*, *M*) is also used to update the quality score of each building block used by *M* (UPDATE-BLOCK-SCORE(*B*, SCORE(*G*, *M*)) in Line 20). Both UPDATE-MARKET-SCORE and UPDATE-BLOCK-SCORE 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 often to construct better mechanisms.

3 Experiments

This section describes the experiments that are carried out to acquire auction mechanisms using the grey-box approach.

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 charge of 10% on trader profit, which we denote as $\mathsf{GF}_{0.1}$. Analysis of CAT games [11] 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 on 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 from the Hall of Fame. 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. 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 advantage otherwise. Based on the parameterized framework in Section 2.2, the CH and CDA markets can be represented as follows:

$$\begin{aligned} \text{CH}_l \,/\, \text{CH}_h &= \mathsf{ME} + \mathsf{QT} + \mathsf{AQ} + \mathsf{CR} + \mathsf{PU}_{k=0.5} + \mathsf{GF}_{0.1} \,/\, \mathsf{GF}_{1.0} \\ \text{CDA}_l \,/\, \text{CDA}_h &= \mathsf{ME} + \mathsf{QT} + \mathsf{AQ} + \mathsf{CC} + \mathsf{PD}_{k=0.5} + \mathsf{GF}_{0.1} \,/\, \mathsf{GF}_{1.0} \end{aligned}$$

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 (lines 15–18 in Algorithm 1). 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 fairly big chances to be 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.

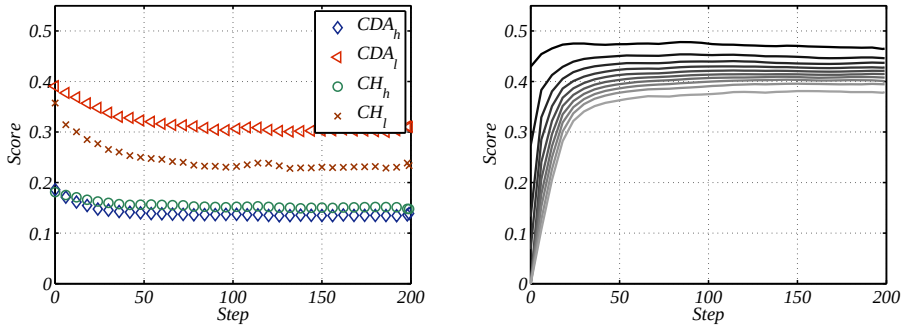
3.2 Experimental results

We carried out four experiments to check whether the grey-box approach is successful in searching for good auction mechanisms.

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.[4] 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

[4]Note that the active Hall of Famers will be different mechanisms at different steps in the process, so what we see in the figure is the performance of the best mechanisms we know of up to the point we collected the data.



(a) The four fixed auction mechanisms.

(b) The top ten active Hall of Famers.

Fig. 2: Scores of market mechanisms across 200 steps (games), averaged over 40 runs.

are above 0.35, higher than the highest score of the four fixed markets from Figure 2a, and the difference is statistically significant at the 95% confidence level. 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.

Third, a better test of the new 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, which 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:

$$\texttt{SM7.1} = \mathsf{MV} + \mathsf{QO} + \mathsf{AH}_{\tau=0.4} + \mathsf{CP}_{p=0.3} + \mathsf{PN}_{n=11} + \mathsf{GF}_{0.1}$$
$$\texttt{SM88.0} = \mathsf{MT}_{\theta=0.4} + \mathsf{QT} + \mathsf{AA} + \mathsf{CP}_{p=0.4} + \mathsf{PU}_{k=0.7} + \mathsf{GF}_{0.1}$$
$$\texttt{SM127.1} = \mathsf{MV} + \mathsf{QS} + \mathsf{AS} + \mathsf{CP}_{p=0.4} + \mathsf{PU}_{k=0.7} + \mathsf{GF}_{0.1}$$

We ran these three mechanisms against the best 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. There were enough entries in the repository at the time we ran the experiments to create reasonable facsimiles of the 2007 and 2008 competitions, but there were not enough entries from the 2009 competition for us to recreate that year's 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 each recreated competition, we ran three games.

Table 1 lists the average cumulative scores of all the markets across their three games along with the standard deviations of those scores. The three new mechanisms we obtained from the grey-box approach beat the actual entries for CAT 2007 and CAT

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

Table 1: 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.

(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
CrocodileAgent	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
CrocodileAgent*	71.4820	5.8687
PSUCAT*	68.3143	6.7389

* IAMwildCAT from CAT 2007, and CrocodileAgent 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 competition — these markets still perform poorly, in contrast to their rankings in the actual competitions.

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.

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 much stronger 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 into the 2008 CAT competition is better than that of those into the 2007 CAT competition when they face the three new, strong, opponents, reflecting the improvement in the entries over time.

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 [14], 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 [14] we investigated a class of double auctions, called NCDAEE, which can be represented as:

Table 2: Properties of the best mechanisms from the grey-box experiments and the auction mechanisms explored in [14]. 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
NCDAEE$_{\delta=0}$	98.336	3.262	4.219	3.141	9.756	28.873	14.098	1.800
NCDAEE$_{\delta=10}$	98.912	2.605	5.552	2.770	23.344	41.727	7.834	5.648
NCDAEE$_{\delta=20}$	98.304	2.562	7.460	3.136	89.128	30.867	4.826	3.487
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

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

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 2.[6] 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.

4 Conclusions 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

[6]Our results are slightly different from those in [14], but the pattern of these results still holds. In addition, we ran an NCDAEE variant ($\delta = 30$) that was not tested in [14], observing that those with $\delta \leq 20$ do not perform well when populated by GD traders.

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 learn mechanisms that can outperform the standard mechanisms against which learning took place and the best entries in past Market Design competitions. 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.

Now that we can learn mechanisms effectively, we plan to adapt the approach to also learn trading strategies, allowing us to co-evolve mechanisms and the traders that operate within them.

Acknowledgments

This work is partially supported by NSF under grant IIS-0329037, *Tools and Techniques for Automated Mechanism Design*, and EPSRC under grants GR/T10657/01 and GR/T10671/01, *Market Based Control of Complex Computational Systems*. We thank the entrants to the Market Design game for releasing binaries of their market agents and the reviewers for their valuable comments.

References

1. K. Cai, E. Gerding, P. McBurney, J. Niu, S. Parsons, and S. Phelps. Overview of CAT: A market design competition. Technical Report ULCS-09-005, Department of Computer Science, University of Liverpool, Liverpool, UK, 2009. Version 2.0.
2. D. Cliff. Evolution of market mechanism through a continuous space of auction-types. Technical Report HPL-2001-326, Hewlett-Packard Research Laboratories, Bristol, England, Dec. 2001.
3. D. Cliff and J. Bruten. Minimal-intelligence agents for bargaining behaviours in market-based environments. Technical Report HPL-97-91, Hewlett-Packard Research Laboratories, Bristol, England, Aug. 1997.
4. V. Conitzer and 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)*, pages 17–24, Pittsburgh, PA, USA, September 2003. ACM Press.
5. V. Conitzer and T. Sandholm. Incremental mechanism design. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07)*, pages 1251–1256, Hyderabad, India, January 2007.

6. I. Erev and A. E. Roth. Predicting how people play games: Reinforcement learning in experimental games with unique, mixed strategy equilibria. *The American Economic Review*, 88(4):848–881, September 1998.
7. S. Gjerstad and J. Dickhaut. Price formation in double auctions. *Games and Economic Behavior*, 22:1–29, 1998.
8. D. K. Gode and S. Sunder. Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality. *Journal of Political Economy*, 101(1):119–137, 1993.
9. V. Krishna. *Auction Theory*. Academic Press, San Diego, CA, 2002.
10. K. A. McCabe, S. J. Rassenti, and V. L. Smith. Designing a uniform price double auction. In D. Friedman and J. Rust, editors, *The Double Auction Market: Institutions, Theories and Evidence*, Santa Fe Institute Studies in the Sciences of Complexity, chapter 11, pages 307–332. Westview Press, Perseus Books Group, Cambridge, MA, 1993.
11. J. Niu, K. Cai, P. McBurney, and 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, December 2008.
12. J. Niu, K. Cai, and S. Parsons. A grey-box approach to automated mechanism design. *Computing Research Repository (CoRR)*, abs/1002.0378, 2010.
13. J. Niu, K. Cai, S. Parsons, E. Gerding, and P. McBurney. Characterizing effective auction mechanisms: Insights from the 2007 TAC Mechanism Design Competition. In Padgham, Parkes, Müller, and Parsons, editors, *Proceedings of the Seventh International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, pages 1079–1086, Estoril, Portugal, May 2008.
14. J. Niu, K. Cai, S. Parsons, and 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)*, pages 1143–1150, Hakodate, Japan, May 2006.
15. S. Phelps, M. Marcinkiewicz, S. Parsons, and 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)*, pages 705–712, New York, NY, USA, 2006. ACM Press.
16. S. Phelps, P. McBurney, S. Parsons, and E. Sklar. Co-evolutionary auction mechanism design: a preliminary report. In *Proceedings of the Workshop on Agent Mediated Electronic Commerce IV (AMEC IV)*, 2002.
17. S. Phelps, S. Parsons, E. Sklar, and 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.
18. J. L. Schvartzman and M. P. Wellman. Exploring large strategy spaces in empirical game modeling. In *Proceedings of the 11th Workshop on Agent-Mediated Electronic Commerce*, 2009.
19. R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.
20. Y. Vorobeychik, D. M. Reeves, and M. P. Wellman. Constrained automated mechanism for infinite games of incomplete information. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence*, pages 400–407, 2007.
21. W. Walsh, R. Das, G. Tesauro, and J. O. Kephart. Analyzing complex strategic interactions in multi-agent systems. In P. Gmytrasiewicz and S. Parsons, editors, *Proceedings of 2002 Workshop on Game-Theoretic and Decision-Theoretic Agents (GTDT-02)*, Edmonton, Alberta Canada, July 2002. AAAI.
22. P. R. Wurman, M. P. Wellman, and W. E. Walsh. A parametrization of the auction design space. *Games and Economic Behavior*, 35:304–338, 2001.