Automated Design of Affine Maximizer Mechanisms in Dynamic Settings

Michael Curry[*1, 2, 4], **Vinzenz Thoma**[*3, 4], **Darshan Chakrabarti**[5], **Stephen McAleer**[6], **Christian Kroer**[5], **Tuomas Sandholm**[6, 7], **Niao He**[3], **Sven Seuken**[2, 4]

[1]Harvard University
[2]University of Zurich
[3]ETH Zurich
[4]ETH AI Center
[5]Columbia University
[6]Carnegie Mellon University, Computer Science Department
[7]Optimized Markets, Strategy Robot, Strategic Machine
curry@ifi.uzh.ch, vinzenz.thoma@ai.ethz.ch

Abstract

Dynamic mechanism design is a challenging extension to ordinary mechanism design in which the mechanism designer must make a sequence of decisions over time in the face of possibly untruthful reports of participating agents. Optimizing dynamic mechanisms for welfare is relatively well understood. However, there has been less work on optimizing for other goals (e.g. revenue), and without restrictive assumptions on valuations, it is remarkably challenging to characterize good mechanisms. Instead, we turn to *automated mechanism design* to find mechanisms with good performance in specific problem instances. We extend the class of affine maximizer mechanisms to MDPs where agents may untruthfully report their rewards. This extension results in a challenging bilevel optimization problem in which the upper problem involves choosing optimal mechanism parameters, and the lower problem involves solving the resulting MDP. Our approach can find truthful dynamic mechanisms that achieve strong performance on goals other than welfare, and can be applied to essentially any problem setting—without restrictions on valuations—for which RL can learn optimal policies.

1 Introduction

Dynamic mechanism design studies sequential decision-making problems, where decisions are based on the self-reported preferences of agents. A typical model is that the environment consists of a *Markov decision process (MDP)*, and the mechanism controls the process given reported utilities by the agents. This has important applications, such as ad auctions or more generally online pricing (e.g. Bergemann and Välimäki 2019) but also problems of decentralised decision making in RL (e.g. Chang et al. 2020).

Much work in dynamic mechanism design has focused on maximizing welfare (Athey and Segal 2013; Nisan et al. 2007a; Lyu et al. 2022b) subject to *strategyproofness* (there should be no incentive for untruthful reports by agents). Some other work considers different goals, notably revenue (Bergemann and Välimäki 2010; Kakade, Lobel, and Nazerzadeh 2013; Hajiaghayi, Kleinberg, and Parkes 2004; Hajiaghayi, Kleinberg, and Sandholm 2007) but needs to make restrictive assumptions about the space of agent types. Work on dynamic mechanism design, for general goals and for broad spaces of agent types, is much more limited.

Dynamic mechanism design includes as a special case static mechanism design, and here the situation is similar. To maximize welfare while ensuring strategyproofness, one can use the celebrated and well-understood Vickrey-Clarke-Groves (VCG) mechanism (Vickrey 1961; Clarke 1971; Groves 1973). For optimizing revenue, Myerson (1981) completely settles the question under the restrictive assumption that agents' types are single-dimensional (essentially, that there is only one type of item up for sale); beyond this there has been little progress except for very specific problem instances (Yao 2017). But for static settings, *automated mechanism design (AMD)* (Conitzer and Sandholm 2002; Sandholm 2003; Curry, Sandholm, and Dickerson 2023) has been used: this is a data-driven search through some class of mechanisms in order to find one that performs well while satisfying the constraints of strategyproofness and individual rationality. Automated mechanism design has in some cases found the highest-performing mechanisms known so far, and can recover optimal mechanisms in special cases where they are known (Duetting et al. 2019; Ivanov et al. 2022; Shen, Tang, and Zuo 2019).

Given the successes of automated mechanism design for static problems, it is surprising that its use for dynamic problems is relatively underexplored.

Our Contributions

Our present paper develops automated dynamic mechanism design techniques which can be applied to a very broad range of problems. In particular, we consider mechanism design on general MDPs. Our model works with many loss functions (including revenue but also other domain-specific loss functions), not just the easier goal of welfare, and we do not assume one-dimensional agent types. Our assumptions are sufficiently general to capture essentially all of static

[*]These authors contributed equally.

multi-parameter mechanism design as a special case[1].

Optimal mechanism design over all possible mechanisms entails the very difficult problem of computing equilibria in imperfect information games, to understand whether or not any agent has any incentive to deviate from truthful reporting. Inspired by prior work for static mechanism design, we sidestep this issue by focusing on the class of affine maximizers (AMAs), which we define on MDPs.[2] These mechanisms are always strategyproof.

We justify this restriction in two ways. First, as mentioned, our problem assumptions capture multi-parameter mechanism design—but finding optimal mechanisms in this setting has proven extremely difficult, so it makes sense to search within a more tractable class of mechanisms. Second, our framing of the problem is broad enough that it captures problem instances where Roberts' theorem (Roberts 1979) applies, which states that under general agent type spaces, *only* affine maximizers can be strategyproof.

We frame the search for a high-performing **dynamic affine maximizer mechanism** as a **bilevel optimization problem**, where the outer problem consists of choosing weights and (possibly state-dependent) boosts—the AMA parameters—to minimise a given loss function. The inner problem consists of learning to control an MDP to maximise affinely-transformed social welfare (the definition of an affine maximizer), given the weights, boosts, and agents' type reports (see equation 1 below). The derivatives of this inner problem do not exist at many points: however, we show that for the important case of revenue, the *expected* loss over the distribution (with a continuous density) of agent valuations is differentiable.

We solve the inner problem via (possibly regularized) linear programming. We also propose a variety of ways to solve the outer problem: by grid search, by differentiating through the regularized LP, or by using zeroth-order methods to approximate the LP gradient. These latter two approaches explicitly or implicitly smooth the objective and avoid the problem of nonexistent derivatives. In experiments on several dynamic mechanism design settings, such as sequential auctions, task scheduling and navigating a gridworld, our approaches result in truthful mechanisms that outperform the VCG baseline.

2 Related Work

Maximizing Welfare in Dynamic Mechanisms

Athey and Segal (2013) consider a dynamic mechanism design setting where agents update their beliefs over time, and where the goal is an efficient and budget-balanced outcome. Parkes (Nisan et al. 2007b) describes a dynamic mechanism design setting where the focus is on agents who may arrive and depart at different periods. Both of these approaches simply assume an optimal policy is available. More recently, Lyu et al. (2022b) presents a model for learning this policy in an offline RL setting; another work focuses on the online case (Lyu et al. 2022a). Chang et al. (2020) consider a dynamic VCG mechanism for decentralised RL where agents bid on the MDP transitions. Bergemann and Välimäki (2010) presents a VCG-like "dynamic pivot mechanism".There are different modeling choices and goals in each of these approaches, but the common theme is that the allocation problem involves making decisions on some MDP after observing agent reports. All of these papers consider a dynamic analogue to the VCG mechanism, i.e. the mechanism designer acts according to the welfare-optimal policy and charges agents their externalities to ensure incentive compatibility. This is in contrast to our work, where we are concerned with goals beyond welfare.

Dynamic Mechanism Design for Goals Other Than Welfare

There is some existing work in this direction as well, with a particular focus on revenue. Pavan, Segal, and Toikka (2014) and Kakade, Lobel, and Nazerzadeh (2013) both consider cases where the private information about the value of the item is 1-dimensional. This allows for a Myerson-style analysis of the actual profit-maximizing mechanism. Other work considers optimal selling of items to agents who arrive and depart over time from the perspective of optimal stopping, but again considers single-parameter item valuations (Hajiaghayi, Kleinberg, and Parkes 2004; Hajiaghayi, Kleinberg, and Sandholm 2007; Kleinberg 2005). Bergemann and Välimäki (2019) surveys other results that make similar assumptions for tractability. Pai and Vohra (2008) considers a similar setting and finds the optimal Bayes-Nash incentive compatible mechanism. Still other work considers settings where the mechanism designer may update the mechanism online over time (e.g. by changing reserve prices) (Shen et al. 2017) and where bidders may even strategically attempt to manipulate the learning process (Amin, Rostamizadeh, and Syed 2014). Overall, these approaches are restrictive in terms of their assumed value model and frequently focus on analytic results, while our computational approach allows more general values and loss functions.

Preference Elicitation from Multiple Agents and Multistage Mechanisms

Another line of work considers iterative preference elicitation (Conen and Sandholm 2001; Sandholm and Boutilier 2006): based on past agent reports, the mechanism can query what preference information it needs most. Existing approaches make use of an analogy to query learning (Zinkevich, Blum, and Sandholm 2003; Blum et al. 2004; Lahaie and Parkes 2004), or leverage machine learning (Soumalias et al. 2023; Weissteiner et al. 2023; Brero, Lubin, and Seuken 2019). In this context, Sandholm, Conitzer, and Boutilier (2007) use automated mechanism design to first find good mechanisms (for general type spaces) and then convert them into multistage mechanisms. While these ap-

[1] If we restrict the MDP to have a single state, then we recover ordinary mechanism design, with each possible action corresponding to an outcome.

[2] The acronym AMA refers to "affine maximizer auctions". We consider affine maximizer *mechanisms*, but we stick to the AMA acronym because it is widely used and to avoid confusion with AMM used to refer to "automated market makers".

proaches are typically focused on making a single final decision, but eliciting agent's preferences over multiple stages, our automated dynamic mechanism design approach is concerned with making a sequence of decisions over time, while eliciting preferences once.

Static Automated Mechanism Design

Due to the difficulty of analytically finding optimal mechanisms, a number of works have instead attempted to treat static mechanism design as a computational optimization problem, starting with Conitzer and Sandholm (2002) and Sandholm (2003), and learn good mechanisms from samples, starting with Likhodedov and Sandholm (2004). One line of work makes use of static affine maximizers and achieves good performance in multi-item multi-bidder auctions (Sandholm and Likhodedov 2015; Curry, Sandholm, and Dickerson 2023; Duan et al. 2023). There is also a line of learning theory research on choosing the AMA parameters given samples from the valuation distribution (Balcan, Sandholm, and Vitercik 2016, 2018; Balcan et al. 2021). Another direction is to start with a potentially non-strategyproof mechanism and iteratively modify it to improve strategyproofness. This is known as *incremental mechanism design* (Conitzer and Sandholm 2007). One line of work in this direction makes use of rich function approximators to learn mechanisms. Duetting et al. (2019) presents one influential direction, which uses neural networks to optimize revenue and a penalized loss to approximately enforce strategyproofness, with many followups (Curry et al. 2021, 2020; Ivanov et al. 2022; Rahme, Jelassi, and Weinberg 2021; Rahme et al. 2021). Shen, Tang, and Zuo (2019) presents an alternative approach for single-bidder settings which can cope with broader classes of utility functions. As mentioned above, the success of these techniques in static settings is our motivation to develop such approaches for the dynamic setting.

3 Preliminaries

Below, we describe our mechanism design problem. The nature of the problem and our mechanism design desiderata motivate our choice to restrict attention to affine maximizer mechanisms. We then describe in more detail how these mechanisms work in a dynamic setting.

Formal Model of Problem Setting

Environment and policy/allocation rule Consider some MDP $\mathcal{M} = (S, A, P)$ (with reward not yet specified), where S is a set of states, A is a set of actions, and P is a transition function. There are n agents, each with their own reward function $r_i : S \times A \to \mathbb{R}$ drawn from a distribution with density f_i. We emphasize that these agents are not themselves taking actions in the MDP—this is done by the mechanism. Their only choice will be which rewards to report to the mechanism. We assume the mechanism designer wants to minimize some loss function (which will often be the negative of some objective to be maximized) $\mathcal{L}$ in expectation over the f_i, which in general depends on the agents' rewards and the chosen policy π on the MDP. The latter corresponds to the allocation rule in traditional mechanism design.

Mechanism Design Desiderata

Incentive compatibility and payments In order to achieve its goal, the mechanism will need access to the true $r_i(s, a)$. However, in general, we should expect the agents to misreport their reward function if they think it will benefit them. Thus, we will allow for some side payments to be made based on the MDP's solution and the agents' reports, in order to ensure that there is no incentive to misreport, that is, making the mechanism *incentive compatible (IC)* or *strategyproof*. (Such payments only exist for some choices of π.) We assume agent utility is quasilinear, that is, positive payments just correspond to negative reward.

Individual rationality We also want to guarantee individual rationality (IR), meaning that agents should not be charged so much that they receive negative utility and would be better off not participating in the mechanism.

Remark. *The mechanism designer's goal may also relate to the payments. For example, a canonical mechanism design goal is to maximize revenue—in our setting, choose a policy π^* such that payments can be made as high as possible while still ensuring IC and IR.*

Background on Affine Maximizers

As motivated above our goal is to choose some π such that IC/IR side payments can be constructed, while also performing as well as possible on the mechanism designer's higher-level objective.

Unlike prior work (e.g., Kakade, Lobel, and Nazerzadeh 2013; Bergemann and Välimäki 2010), we do not make any assumptions about the structure of the reward functions. Our setting is therefore general enough to incorporate many hard problems such as optimal multi-item mechanism design, and is thus at least as hard as those. Therefore hoping to get the truly best-performing π, even only in an infinite-sample/asymptotic sense, is too much. Thus it is appropriate to restrict attention to a more tractable class of mechanisms.

Also, our problem setting is general enough to include situations where a result known as Roberts' theorem applies (Roberts 1979). It states that under certain conditions (arbitrary rewards, at least 3 outcomes), the only allocation rules that can possibly have IC payments take the form of affine maximizers.

Below, we give the standard definition of affine maximizer mechanisms in terms of the allocation and payment rules, modified for our dynamic problem setting.

Definition 3.1 (Affine maximizers)**.** *Given so-called weights $w \in \mathbb{R}^n_+$ and boosts $b \in \mathbb{R}^{|S|\times|A|}$, a dynamic affine maximiser mechanism (AMA) takes reported reward functions $\boldsymbol{r} \in \mathbb{R}^{|S|\times|A|\times n}$ and returns a policy $\pi^*(w, b, \boldsymbol{r})$ on the MDP that maximizes the affine social welfare $\mathsf{asw}^{(\pi)}(w, b, \boldsymbol{r})$ where*

$$\mathsf{asw}^{(\pi)}(w, b, \boldsymbol{r}) = \mathbb{E}_\pi \left[\sum_{t=0}^{T} \left(\sum_{i=1}^{n} w_i r_i(s_t, a_t) \right) + b(s_t, a_t) \right]$$

Defining $\mathsf{asw}(w, b, \boldsymbol{r}) = \mathsf{asw}^{(\pi^*(w,b,\boldsymbol{r}))}(w, b, \boldsymbol{r})$ as the maximum affine social welfare for reports $\boldsymbol{r}$, the resulting payment is then

$$\boldsymbol{p}_i(w, b, \boldsymbol{r}) = \frac{1}{w_i}\left(\mathsf{asw}^{(-i)}(w, b, \boldsymbol{r}) - \left(\mathbb{E}_{\pi^*(w,b,\boldsymbol{r})}\left[\sum_{t=0}^{T}\left(\sum_{j\neq i} w_j r_j(s_t, a_t)\right) + b(s_t, a_t)\right]\right)\right)$$

where $\mathsf{asw}^{(-i)}(w, b, \boldsymbol{r})$ *defined as* $\max_\pi \mathbb{E}_\pi\left[\sum_{t=0}^{T}\left(\sum_{j\neq i} w_j r_j(s_t, a_t)\right) + b(s_t, a_t)\right]$ *is the maximum affine social welfare, when disregarding* i*.*

No matter the choice of weights and boosts, every resulting AMA is strategyproof ex-post—after learning the rewards of the other agents, reporting truthfully is a dominant strategy—and IR in expectation over the MDP trajectories. The proof can be found in Appendix A.

4 Dynamic Mechanism Design as Bilevel Optimisation

The problem of searching for a performant mechanism within the class of AMAs can be naturally formulated as stochastic bilevel optimization as follows[3]:

$$\min_{w,b} \mathbb{E}_{\boldsymbol{r}\sim f}[\mathcal{L}(\pi^*, w, b)] \text{ s.t. } \pi^* \in \arg\max_\pi \mathbb{E}_{s_t,a_t\sim\pi}\left[\sum_{t=0}^{T}\left(\sum_{i=1}^{n} w_i r_i(s_t, a_t)\right) + b(s_t, a_t)\right] \forall \boldsymbol{r} \quad (1)$$

Given the bilevel structure, we can think of the problem as a game between a leader and a follower.

- The leader knows only the joint distribution $f = \prod_i f_i$ from which the set of reward functions are drawn, and chooses weights w_i and boosts $b(s, a)$.
- For any draw of r_i and the weights and boosts, the follower acts optimally ("best responds") in the MDP according to the AMA objective.

To clarify (and to contrast with many models of mechanism design where the agents making reports are treated as followers): the leader and follower are only "notional". In reality, there is only one mechanism designer. We nevertheless speak in terms of a "leader" and "follower" because in order for strategyproofness to be attained, some component of the system must successfully maximize affine social welfare, which is a goal distinct from the true goal of the mechanism designer.

In general, the problem above—bilevel optimization, with stochasticity in the leader's objective—is quite difficult. Indeed, the case of revenue-maximisation in one-round auctions with multiple goods, which is a very special case of our much more general problem, remains essentially unsolved beyond a few very simple special cases (Yao 2017). Therefore finding a globally optimal solution to the above problem is too much to hope for, but we show that derivatives exist for important expected loss functions, which enables us to use gradient-based optimization techniques to find local optima.

We then consider three complementary methods for optimizing the AMA parameters: random grid search, zeroth-order methods to approximate the derivatives, and differentiation through a smoothed LP.

[3]We constrain the policy to be in the *set* of best responses, because there is a null set of possible $\boldsymbol{r}$ where π^* is not unique. However, because it is a null set, the choice of π^* does not influence the expected value in the outer problem so that it is well-defined.

Existence of Derivatives

So far, we have not made any assumptions about the loss function. A very natural desideratum would be that $\mathcal{L}(\pi^*(w, b, \boldsymbol{r}), w, b)$ is differentiable, so that we can perform stochastic gradient descent. However, in general this is not true. This is due to the fact that $\pi^*(w, b, \boldsymbol{r})$ is in general not even continuous in w, b—since the optimal policy on an MDP is always going to be deterministic—and therefore neither is $\mathcal{L}$. However, if $\mathcal{L}$ has a certain shape, which is the case for the loss functions we consider in this work, we can prove that the relaxed condition that $E_{\boldsymbol{r}}[\mathcal{L}(\pi^*(w, b, \boldsymbol{r}), w, b)]$ is differentiable.

Theorem 4.1. *Let* $\mathcal{L}$ *be a loss function for the problem in Equation* (1) *and assume it can be decomposed as follows*

$$\mathcal{L}(\pi^*, w, b) = \mathsf{asw}(w, b, \boldsymbol{r}) + \sum_{k=1}^{K} a_k g_k(\pi^*(w, b, \boldsymbol{r}), \boldsymbol{r})$$

where it holds for all k *that* $g_k = \mathcal{O}(\|\boldsymbol{r}\|_\infty)$*. Assume further that the support of* $\boldsymbol{r}$ *is compact or* f *decays sufficicently quickly such that* $\mathbb{E}[\|\boldsymbol{r}\|_\infty]$ *exists. Then* $E_{\boldsymbol{r}}[\mathcal{L}(\pi^*(w, b, \boldsymbol{r}), w, b)]$ *is differentiable almost everywhere.*

Proof Idea. The full proof is in Appendix B. It consists of two parts. First we show that asw is Lipschitz continuous in w, b. This essentially follows because it is the maximum over a set of functions which are linear in w, b. For g_k we argue that for any w, b, the set of rewards for which two alternative policies give the same affine social welfare, lie on a hyperplane. Changing the weights and boosts may change the optimal policy, and therefore may result in differing social welfare for some subset of the rewards. We would like to bound the probablility mass of the set of rewards where this can happen. Indeed, the rewards where the optimal policy may change, all lie in between the aforementioned hyperplanes. The distance between the hyperplanes depends on the change in w and b, so the mass can be bounded—assuming the probability density decays sufficiently fast—which results in a local Lipschitz constant. Differentiability follows from Rademacher's theorem and holds equivalently for the sum of these terms because of the linearity of the expectation and derivative. □

In Section 5 we will study two loss functions in particular: revenue and makespan. The revenue of a dynamic AMA

mechanism is given by

$$\begin{aligned}\mathsf{rev}(w,b,\boldsymbol{r}) &= \sum_{i=1}^{n} \boldsymbol{p}_i(w,b,\boldsymbol{r}) \\ &= -\sum_{i=1}^{n}\left(\frac{1}{w_i}\mathsf{asw}(w,b,\boldsymbol{r})\right) + \mathsf{sw}^{(\pi^*(w,b,\boldsymbol{r}))}(\boldsymbol{r}) \\ &\quad + \sum_{i=1}^{n}\frac{1}{w_i}\mathsf{asw}^{(-i)}(w,b,\boldsymbol{r})\end{aligned}$$

Here sw, the non-transformed social welfare, is just asw when weights are all 1 and boosts are all 0, i.e. $\mathsf{sw}^{(\pi)} = \mathbb{E}_\pi\left[\sum_{t=0}^{T}\sum_{i=1}^{n} r_i(s_t,a_t)\right]$.

It is easy to verify that revenue fulfills all assumptions of Theorem 4.1.

Lemma 4.2. *The expected revenue is differentiable almost everywhere.*

Proof. Revenue is a essentially a sum of the three terms $\mathsf{asw}, \mathsf{sw}$ and $\mathsf{asw}^{(-i)}$. The first asw is directly as stated in the assumptions of Theorem 4.1. For $\mathsf{asw}^{(-i)}$ the proof extends canonically as it equivalently is the maximal affine social welfare, albeit for a smaller set of agents. Last but not least $\mathsf{sw}^{(\pi^*(w,b,\boldsymbol{r}))}(\boldsymbol{r})$ is a function of only $(\pi^*(w,b,\boldsymbol{r}))$ and $\boldsymbol{r}$, for which it holds that

$$\begin{aligned}\mathsf{sw}^{(\pi^*(w,b,\boldsymbol{r}))}(\boldsymbol{r}) &= \mathbb{E}_\pi\left[\sum_{t=0}^{T}\left(\sum_{i=1}^{n} w_i r_i(s_t,a_t)\right) + b(s_t,a_t)\right] \\ &\leq Tn\|\boldsymbol{r}\|_\infty\end{aligned}$$

thereby concluding the proof. □

Once we have introduced the task scheduling problem we will analogously show that makespan also satisfies the assumptions of Theorem 4.1.

Linear Programming Formulation

We now describe the linear-programming formulation for an MDP, which can be either infinite horizon or episodic with states partially ordered by time (that is, the time step is encoded in the state) (Altman 1999). In particular, we suppose the follower solves the following LP to find the optimal state-action occupancy measure $\nu_{\pi^*(w,b,\boldsymbol{r})}$,[4] which determines the revenue (or whichever loss function is chosen):

$$\begin{aligned}&\max \sum_{s\in S, a\in A}\left(\sum_i w_i r_i(s,a) + b(s,a)\right)\nu(s,a) \text{ s.t.} \\ &\sum_{a\in A}\nu(s,a) = \sum_{s',a'} P(s|s',a')\nu(s',a') + \mu_0(s) \quad \forall s\in S \\ &\nu(s,a)\geq 0 \quad \forall s\in S, a\in A\end{aligned} \tag{2}$$

where μ_0 denotes the initial state distribution.

[4]This corresponds to the optimal policy by $\pi^*(a|s) = \frac{\nu_{\pi^*(w,b,\boldsymbol{r})}(s,a)}{\sum_a \nu_{\pi^*(w,b,\boldsymbol{r})}(s,a)}$.

Note that if $\mathcal{L}$ were itself differentiable, we could take the gradient inside the expectation and apply the implicit function theorem to get $\nabla_{w,b}\mathbb{E}_{\boldsymbol{r}}\left[\mathcal{L}\left(\pi^*,w,b\right)\right] = \mathbb{E}_{\boldsymbol{r}}\left[\nabla_2\mathcal{L}\left(\pi^*,w,b\right) + \nabla_{w,b}\pi^*(w,b,\boldsymbol{r})\nabla_1\mathcal{L}\left(\pi^*,w,b\right)\right]$. Then we could estimate the gradient of the expected value from a sum of gradients for different sampled $\boldsymbol{r}$. Indeed as we show in Appendix C, for revenue we can even compute the partial derivatives of $\mathcal{L}$ with respect to w, b analytically. However, because $\mathcal{L}$ is not differentiable (since π^* is not), the Dominated Convergence Theorem does not hold and we cannot in fact exchange gradient and expected value. Instead we propose two alternatives to compute the gradient of the expected value. First using zeroth-order estimates and second introducing regularization, which makes the optimal policy in the inner problem unique $\mathcal{L}$ differentiable and thus actually allows us to estimate the gradient from samples. The pseudocode of our approach is given by Algorithm 1.

Zeroth-order methods For bilevel problems, Sow, Ji, and Liang (2022) present a zeroth-order approach, which we adapt to our own setting. The key observation of their approach is that the derivative of the leader's objective is the sum of two partial derivatives. One of these, the derivative of the leader's objective with respect to their own solution, is usually easy to evaluate. The other requires differentiating through the follower's best-response map and inverting a Jacobian, which is challenging—and this second portion can be separately estimated using zeroth-order perturbations.

Zeroth-order methods, due to the use of random perturbations, also implicitly smooth the function (Duchi, Bartlett, and Wainwright 2012), so that when using them there is no further need for regularization to ensure that derivatives can be estimated. [5]

Regularized linear program As an alternative method, we propose regularizing our problem to get a smooth surrogate of the derivative. We add a small entropy regularizer to the follower's objective. [6] The result is now a convex program (see 13 in the appendix): When the objective is strongly convex, the solution map $\nu_r^* : (w,b) \mapsto \nu_{\pi^*(w,b,\boldsymbol{r})}$ is smooth. Therefore, derivatives of revenue can now be estimated from the gradients at sampled type profiles, calculated using reverse-mode automatic differentiation (Agrawal et al. 2020, 2019).

Adding a small amount of regularisation does not change the follower's problem significantly. As has been shown by Weed (2018), the distance between the solutions to the regularized and unregularized inner problem decays exponentially fast in the regularisation constant α, for sufficiently small α.

Under certain assumptions we claim this convergence translates to the outer problem, so that choosing a small α ensures the objective is not disturbed too much.

[5]They have been proposed in certain related (static) settings (Bichler et al. 2021; Martin and Sandholm 2023) to cope with similar problems related to nonexistence of derivatives.

[6]A technique equivalent to such regularization has also been used to deal with a similar issue in first-order computation of equilibria of non-truthful static auctions (Kohring, Pieroth, and Bichler 2023).

Algorithm 1: Gradient-based Dynamic Mechanism Design

Input: MDP $\mathcal{M}$, number of agents n, loss $\mathcal{L}$, number of initialisations m

Initialize: $w \in \mathbb{R}^n$, $b \in \mathbb{R}^{|S|\times|A|}$
$(w_i, b_i)_{1\le i\le m} \leftarrow$ **grid search**$(\mathcal{M}, \mathcal{L})$
{Multiple starting points to avoid local minima}
for `num_iterations` **do**
  for $i = 1$ **to** m **do**
    $(w_i, b_i) \leftarrow (w_i, b_i) + \gamma$ **gradientstep**$(\mathcal{M}, w_i, b_i)$
    {Zeroth or first order gradient estimate}
  end for
end for
return $\arg\min_{i\in\{1,\dots,m\}} \mathbb{E}[\mathcal{L}(w_i, b_i)]$

Theorem 4.3 (Pointwise convergence of regularised loss)**.** *Assume that $\mathcal{L}(w, b, \boldsymbol{r})$ can be represented as an inner product between a vector $\boldsymbol{q}$ in $\mathbb{R}^{|S|\times|R|}$ and the state-action occupancy measure ν, such that $\|q\|_\infty \le \mathcal{O}(\|r\|_\infty)$. Let $L_\alpha(w, b, \boldsymbol{r})$ denote the loss achieved with the optimal policy $\pi^\alpha(w, b, \boldsymbol{r})$ for the regularised problem (see equation 13 in Appendix E). Assume further that $\mathbb{E}[\|\boldsymbol{r}\|_\infty]$ exists, then*

$$\lim_{\alpha\to 0} \mathbb{E}\left[\mathcal{L}_\alpha(w, b, \boldsymbol{r})\right] = \mathbb{E}\left[\mathcal{L}(w, b, \boldsymbol{r})\right]$$

The proof of the Theorem can be found in Appendix D. Given that both affine social welfare and social welfare can be represented as an inner product of ν, Theorem 4.3 implies the convergence of regularised revenue. Similarly, we will later show the same holds for makespan.

Corollary 4.4. *Let $\mathsf{rev}_\alpha(w, b)$ denote the revenue achieved with the optimal policy $\pi^\alpha(w, b, \boldsymbol{r})$. Assume that $\mathbb{E}[\|\boldsymbol{r}\|_\infty]$ exists, then*

$$\lim_{\alpha\to 0} \mathsf{rev}_\alpha(w, b) = \mathsf{rev}(w, b)$$

5 Experiments

Methods

We optimize AMAs in dynamic mechanism design settings on tabular MDPs. We compare three different mechanism design settings with different reward distributions. Our optimization methods consist of a naïve grid search baseline, and two gradient-based methods (using either zero-order or regularized gradient estimates).

Implementation Details

Grid search As a methodological benchmark, we implement a naïve grid search algorithm. We sample 10000 different weights and boosts from a Sobol sequence (Sobol' 1967). To assess the expected performance of each draw, we solve the associated dual linear program (LP), as detailed in Equation (2), for 2000 randomly sampled reward profiles.

Zeroth-order methods We estimate derivatives using 20 perturbations, sampled from a Gaussian distribution with standard deviation 0.05 per estimate on 20 sampled type profiles. We use a learning rate of 0.1.

n	m	VCG	Grid	0-order	reg. LP	Best imp.
2	2	0.0000	0.4902	**0.4939**	0.4893	N/A
3	2	0.4999	0.6079	**0.6777**	0.6755	35.56%
4	2	0.8000	0.7977	**0.8783**	0.8328	9.79%
5	2	0.9996	1.0050	**1.0078**	0.9967	0.83%
2	3	0.0000	**0.4715**	0.3825	0.4168	N/A
3	3	0.0000	0.6107	0.6446	**0.7240**	N/A
4	3	0.6007	0.7323	0.8875	**0.9104**	51.54%
5	3	0.9988	0.7142	1.0631	**1.0743**	7.56%

Table 1: Results for optimizing auction revenue in a sequential sales setting (n agents, m sales) with symmetric uniformly-distributed types. Standard errors were < 0.007. Runtime was < 31 hours for grid search, < 0.2 hours for zeroth-order and < 1.1 hours for first order.

Regularized LP. We compute derivatives with respect to social welfare using the regularized LP, with smoothing parameter 10^{-2} except where mentioned. We solve the regularized program using MOSEK (ApS 2023) and use the DiffOpt package within JuMP (Lubin et al. 2023) to differentiate. For the affine social welfare terms in the expected revenue we use the partial derivatives given in Appendix C. For each stochastic gradient step, we sample 20 type profiles and optimize with learning rate 10^{-2}.

In all cases, when evaluating the objective, we sample 10000 type profiles and do not use regularization. Thus at evaluation time, the LP solution is exactly correct, ensuring strategyproofness. Computational details and hyperparameters are described in appendix F.

Environments and Results

Sequential Sales We begin with a simple setting in which identical items are sold sequentially to unit-demand bidders. The states consist of a record of who has received the item; the allowed actions are to sell the item to some bidder, or to no one. The welfare-maximizing mechanism thus involves giving the items to the highest bidder, but by altering the boosts, revenue can be increased by sometimes withholding the item. We consider a distribution of type profiles drawn uniformly from $[0, 1]$, with results in Table 1.

We observe that optimizing the boosts can consistently improve performance compared to VCG, especially when there are no tight supply constraints. Intuitively, if there is a large supply of goods, VCG revenue should be low, as agents do not cause much externality on other agents. By setting boosts to effectively withhold goods (like a reserve price), revenue can be increased.

We also consider a distribution where agent i's type is uniformly distributed on $[0, i]$. In this setting, we also allow the bidder weights to vary, with results in Table 2. Again, we observe improved performance by optimizing the AMA parameters.

In both settings, the gradient-based approaches generally outperform the random grid search both in terms of results and runtime, in particular for the larger settings, which makes sense considering the high number of dimensions.

n	m	VCG	Grid	0-order	reg. LP	Best imp.
2	2	0.0000	0.3327	0.3302	**0.3665**	N/A
3	2	0.2348	0.3217	**0.4123**	0.4116	75.55%
4	2	0.3089	0.3328	0.4021	**0.4508**	45.95%
5	2	0.3350	0.3355	0.4348	**0.4645**	38.65%
2	3	0.0000	**0.3208**	0.2544	0.3202	N/A
3	3	0.0000	0.3304	0.3556	**0.4354**	N/A
4	3	0.2276	0.3431	0.4263	**0.4751**	108.77%
5	3	0.3193	0.2713	0.3770	**0.4976**	55.85%

Table 2: Results for optimizing auction revenue in a sequential sales setting (n agents, m sales) with asymmetric uniformly-distributed types. Standard errors were < 0.004. Runtime was < 32 hours for grid search, < 0.2 hours for zeroth-order and < 1.1 hours for first order.

n	m	VCG	Grid	0-order	reg. LP	Best imp.
2	4	1.0336	1.0326	**0.9132**	0.9288	-11.65%
2	5	1.0116	1.0142	**0.8820**	0.9286	-12.81%
3	4	0.6111	0.6196	**0.5967**	0.6184	-2.36%
3	5	0.5662	0.5632	**0.5429**	0.5538	-4.12%

Table 3: Results for minimizing makespan in the dynamic truthful task scheduling setting (n agents, m sales) with symmetric uniformly-distributed types. Standard errors were < 0.02. Runtime was < 16 hours for grid search, < 0.1 hours for zeroth-order and < 4.1 hours for first order. AMA outperforms VCG because the makespan is smaller.

n	m	VCG	Grid	0-order	reg. LP	Best imp.
2	4	1.8312	1.8260	**1.5978**	1.6121	-12.75%
2	5	1.9651	1.9837	**1.6347**	1.6886	-16.81%
3	4	1.4299	1.4426	**1.3242**	1.3243	-7.39%
3	5	1.4644	1.4729	**1.3256**	1.3629	-9.48%

Table 4: Results for minimizing makespan in the dynamic truthful task scheduling setting (n agents, m sales) with asymmetric uniformly-distributed types. Standard errors were < 0.03. Runtime was < 17 hours for grid search, < 0.1 hours for zeroth-order and < 4 hours for first order. AMA outperforms VCG because the makespan is smaller.

n	m	VCG	Grid	0-order	reg. LP	Best imp.
2	3	0.7547	1.0607	1.3486	**1.5464**	104.90%
3	3	1.2812	1.4348	1.5575	**1.9251**	50.25%
4	3	1.8683	1.8729	1.8853	**2.3009**	23.15%
5	3	2.3563	2.3689	1.9951	**2.5989**	10.30%
2	4	1.0402	1.0446	1.4935	**1.6134**	55.11%
3	4	1.4898	1.4904	1.6821	**2.0171**	35.40%
4	4	1.8610	1.8757	1.9427	**2.3413**	25.81%
5	4	2.2472	2.2133	2.0256	**2.5911**	15.30%

Table 5: Results for minimizing makespan in the gridworld environment. Standard errors were < 0.05. Runtime was < 30 hours for grid search, < 0.1 hours for zeroth-order and < 0.1 hours for first order.

Dynamic truthful task scheduling The next setting we consider is a dynamic version of the classic truthful task scheduling problem (Nisan and Ronen 2001). In the static problem, workers report the time it takes them to complete certain tasks. A mechanism then has to assign the tasks and give payments[7] to incentivize truthful reports with the goal of minimizing the makespan of all jobs. We formulate a dynamic version of the truthful task scheduling problem. There are n workers and T tasks. Each round, one of the tasks arrives and has to be assigned. Each worker has a cost vector $\theta_i = (t_{i,1}, \dots, t_{i,T})$ distributed according to some prior f_i, which consists of the times they take to finish each task. Each round τ the mechanism designer takes an allocative action $\boldsymbol{x}_\tau \in \{0,1\}^n$, s.t. $\sum_{i=1}^n \boldsymbol{x}_{\tau,i} = 1$. At time τ, this causes agent i to receive reward $r_{i,\tau} = -\boldsymbol{x}_{\tau,i} t_{i,\tau}$.

The objective of the leader is to minimise total makespan. For this define $\tilde{t}_i$ as the time i has already worked on its jobs, when the last task has been assigned in round T. The leader's loss is then given by $\max_{i\in\{1,\dots,n\}} \left(\left(\sum_{\tau=1}^T \boldsymbol{x}_{\tau,i} t_{i,\tau}\right) - \tilde{t}_i\right)$.

In order to justify using our approach for minimizing makespan, we need to show that Theorems 4.1 and 4.3 apply. To see that makespan fulfills the assumption of Theorem 4.1, notice that it is a function of only $\boldsymbol{t}$ (which is $-\boldsymbol{r}$) and $\pi^*(w, b, \boldsymbol{r})$ and that the total makespan is always bound by $T\|\boldsymbol{r}\|_\infty$. Therefore we can conclude that it is differentiable almost everywhere. Further notice that the makespan of a single agent (i.e. without the $\max$ operator in front) fulfills the assumptions of Theorem 4.3. Let the maximum difference across all agent-specific makespans between the unregularised and regularised optimal policy be ϵ, then the difference in the total makespan can be at most 2ϵ. But $\epsilon \to 0$ according to Theorem 4.3 which shows the makespan under the regularized optimal policies will converge to the correct makespan as $\alpha \to 0$.

As a benchmark, we consider a dynamic VCG mechanism that chooses the solution, which minimizes the total work done by the agents—an objective which is not the same as minimizing makespan. (It can been shown, however, that VCG is an n-approximation of the optimal static mechanism (Nisan and Ronen 2001).)

For a participant-symmetric valuation distribution (uniform on $[0, 3]$), results are in Table 3. Across all environments, we see an improvement in makespan for the optimized AMAs over the VCG mechanism. We also consider an asymmetric distribution with disutilities distributed on $[0, 3i]$ for bidder i, with results in Table 4. Across all environments, we see a significant improvement in makespan for the best AMAs. In both settings, gradient-based approaches outperform the naïve grid search both in terms of results and runtime, as was observed in the sequential auction environments.

[7] This in contrast to the auction environment where the mechanism could charge payments, because here agents suffer costs for which they need to be compensated.

Navigating a grid with multiple tasks One of the most canonical environments in RL is the *gridworld*, where an agent deterministically navigates a two-dimensional grid with rewards for reaching certain states (Sutton 2018). We consider the following variant: the mechanism moves ("up","down","left","right") in a grid with n agents observing the trajectories. It starts in state s_0. Each agent i draws a goal state $s_i \in S \setminus \{s_0\}$ and a reward $r_i \sim \mathcal{U}(0,1)$, which they receive when the mechanism reaches s_i. Given (w, b), the mechanism finds a policy to maximize affine social welfare $\pi^* \in \arg\max_\pi \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t \left(\sum_{i=1}^{n} w_i r_i(s_t, a_t) + b(s_t, a_t)\right)\right]$ where γ is the discount factor to account for the infinite time horizon. One can interpret this environment as an auctioneer navigating an environment with different replenishing goods which agents wish to collect. The agents now bid to influence trajectories, and the auctioneer tries to maximize revenue. The results are in Table 5 and show that we can increase revenue by at least 10% in every setting considered. Moreover, the gradient-based approaches are once more much faster and achieve better results than grid search. We further observed that the optimized boosts correspond to a preference of the mechanism for staying close to s_0. This can be interpreted roughly as a reserve price.

Overall, we conclude that searching in the class of dynamic AMAs produces performant mechanisms in a variety of settings, which consistently outperform dynamic VCG. On a methodological level, a naïve grid search can perform well in settings with small dimensionality, but cannot scale up and takes a magnitude more runtime compared to our gradient-based approaches, which perform well across all settings. The best choice between zeroth-order and regularized LP depends on the mechanism design setting.

6 Conclusion

In this paper, we have proposed an approach for automated dynamic mechanism design. In contrast to earlier work, this formulation allows for a wide array of possible objectives (not just maximising social welfare) and works without strong restrictions on the type space. In principle, it captures essentially all problems of static mechanism design as a special case. By focusing on the class of AMAs, we can frame the problem as stochastic bilevel optimization, where the mechanism designer acting in the outer problem chooses parameters to maximize their objective in expectation over possible rewards and the inner problem consists of optimally solving the MDP.

For the most prominent objective in mechanism design—expected revenue—we have further proven differentiability, which allows for gradient-based optimisation approaches to converge to locally optimal mechanisms. Because we restrict to the class of AMAs, all these mechanisms are guaranteed to be exactly IC and IR. To solve the bilevel problem, we have presented randomized grid search, as well as a zeroth and first order gradient-based algorithm to find well-performing mechanisms, which can beat the benchmark dynamic VCG mechanism across a broad range of environments we consider. The gradient-based methods we propose also consistently outperform naïve grid search, which suffers from the curse of dimensionality.

The method we have presented is appropriate for any problem that can be formulated as controlling an MDP in the face of possibly untruthful preferences. This covers a wide range of interesting scenarios. In particular, the use of affine maximizers and the bilevel problem formulation are applicable to a broader range of settings, including those beyond the reach of tabular methods. Future work could apply deep RL for both the leader and follower, enabling scaling to significantly larger and more complicated problems, or apply our techniques to novel mechanism design settings.

Acknowledgments

This paper is part of a project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (Grant agreement No. 805542). N.H. is supported by ETH research grant and Swiss National Science Foundation (SNSF) Project Funding No. 200021-207343. V.T. is partly supported by the ETH AI Center. D.C. was supported by the National Science Foundation Graduate Research Fellowship Program under award number DGE-2036197. C.K. was supported by the Office of Naval Research awards N00014-22-1-2530 and N00014-23-1-2374, and the National Science Foundation awards IIS-2147361 and IIS-2238960. T.S. is supported by the Vannevar Bush Faculty Fellowship ONR N00014-23-1-2876, National Science Foundation grants RI-2312342 and RI-1901403, ARO award W911NF2210266, and NIH award A240108S001.

References

Agrawal, A.; Amos, B.; Barratt, S.; Boyd, S.; Diamond, S.; and Kolter, J. Z. 2019. Differentiable Convex Optimization Layers. In *Neural Information Processing Systems (NeurIPS)*.

Agrawal, A.; Barratt, S.; Boyd, S.; Busseti, E.; and Moursi, W. M. 2020. Differentiating Through a Cone Program. arxiv:1904.09043.

Altman, E. 1999. *Constrained Markov decision processes*, volume 7. CRC press.

Amin, K.; Rostamizadeh, A.; and Syed, U. 2014. Repeated Contextual Auctions with Strategic Buyers. In *Neural Information Processing Systems (NeurIPS)*.

ApS, M. 2023. *The MOSEK optimization toolbox, version 10.0*.

Athey, S.; and Segal, I. 2013. An efficient dynamic mechanism. *Econometrica*, 81(6): 2463–2485.

Balcan, M.-F.; DeBlasio, D.; Dick, T.; Kingsford, C.; Sandholm, T.; and Vitercik, E. 2021. How much data is sufficient to learn high-performing algorithms? Generalization guarantees for data-driven algorithm design. In *ACM Symposium on Theory of Computing (STOC)*.

Balcan, M.-F.; Sandholm, T.; and Vitercik, E. 2018. A general theory of sample complexity for multi-item profit maximization. In *Economics and Computation (EC)*.

Balcan, M.-F. F.; Sandholm, T.; and Vitercik, E. 2016. Sample complexity of automated mechanism design. In *Neural Information Processing Systems (NeurIPS)*.

Bergemann, D.; and Välimäki, J. 2010. The dynamic pivot mechanism. *Econometrica*, 78(2): 771–789.

Bergemann, D.; and Välimäki, J. 2019. Dynamic Mechanism Design: An Introduction. *Journal of Economic Literature*, 57(2): 235–274.

Bichler, M.; Fichtl, M.; Heidekrüger, S.; Kohring, N.; and Sutterer, P. 2021. Learning equilibria in symmetric auction games using artificial neural networks. *Nature Machine Intelligence*, 3(8): 687–695.

Blum, A.; Jackson, J. C.; Sandholm, T.; and Zinkevich, M. 2004. Preference Elicitation and Query Learning. *Journal of Machine Learning Research*.

Brero, G.; Lubin, B.; and Seuken, S. 2019. Machine Learning-powered Iterative Combinatorial Auctions. *arXiv preprint arXiv:1911.08042*.

Chang, M.; Kaushik, S.; Weinberg, S. M.; Griffiths, T.; and Levine, S. 2020. Decentralized Reinforcement Learning: Global Decision-Making via Local Economic Transactions. In III, H. D.; and Singh, A., eds., *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, 1437–1447. PMLR.

Clarke, E. H. 1971. Multipart pricing of public goods. *Public Choice*, 17–33.

Conen, W.; and Sandholm, T. 2001. Preference Elicitation in Combinatorial Auctions: Extended Abstract. In *Economics and Computation (EC)*, 256–259. More detailed description of algorithmic aspects in IJCAI-01 Workshop on Economic Agents, Models, and Mechanisms, pp. 71–80.

Conitzer, V.; and Sandholm, T. 2002. Complexity of Mechanism Design. In *Uncertainty in Artificial Intelligence (UAI)*.

Conitzer, V.; and Sandholm, T. 2007. Incremental Mechanism Design. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Curry, M.; Chiang, P.-y.; Goldstein, T.; and Dickerson, J. P. 2020. Certifying Strategyproof Auction Networks. In *Neural Information Processing Systems (NeurIPS)*.

Curry, M.; Sandholm, T.; and Dickerson, J. 2023. Differentiable Economics for Randomized Affine Maximizer Auctions. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Curry, M. J.; Lyi, U.; Goldstein, T.; and Dickerson, J. 2021. Learning Revenue-Maximizing Auctions With Differentiable Matching. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Duan, Z.; Sun, H.; Chen, Y.; and Deng, X. 2023. A Scalable Neural Network for DSIC Affine Maximizer Auction Design. .

Duchi, J. C.; Bartlett, P. L.; and Wainwright, M. J. 2012. Randomized smoothing for stochastic optimization. *SIAM Journal on Optimization*, 22(2): 674–701.

Duetting, P.; Feng, Z.; Narasimhan, H.; Parkes, D. C.; and Ravindranath, S. S. 2019. Optimal Auctions through Deep Learning. In *International Conference on Machine Learning (ICML)*.

Groves, T. 1973. Incentives in teams. *Econometrica: Journal of the Econometric Society*, 617–631.

Hajiaghayi, M.; Kleinberg, R.; and Parkes, D. C. 2004. Adaptive limited-supply online auctions. In *Electronic Commerce (EC)*.

Hajiaghayi, M.; Kleinberg, R.; and Sandholm, T. 2007. Automated online mechanism design and prophet inequalities. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Ivanov, D.; Safiulin, I.; Balabaeva, K.; and Filippov, I. 2022. Optimal-er Auctions through Attention. *arXiv preprint arXiv:2202.13110*.

Kakade, S. M.; Lobel, I.; and Nazerzadeh, H. 2013. Optimal dynamic mechanism design and the virtual-pivot mechanism. *Operations Research*, 61(4): 837–854.

Kleinberg, R. 2005. A Multiple-Choice Secretary Algorithm with Applications to Online Auctions. In *Symposium on Discrete Algorithms (SODA)*.

Kohring, N.; Pieroth, F. R.; and Bichler, M. 2023. Enabling First-Order Gradient-Based Learning for Equilibrium Computation in Markets. *arXiv preprint arXiv:2303.09500*.

Lahaie, S. M.; and Parkes, D. C. 2004. Applying Learning Algorithms to Preference Elicitation. In *Proceedings of the 5th ACM Conference on Electronic Commerce*, EC '04. New York, NY, USA: Association for Computing Machinery.

Likhodedov, A.; and Sandholm, T. 2004. Methods for boosting revenue in combinatorial auctions. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Lubin, M.; Dowson, O.; Garcia, J. D.; Huchette, J.; Legat, B.; and Vielma, J. P. 2023. JuMP 1.0: Recent improvements to a modeling language for mathematical optimization. *Mathematical Programming Computation*. In press.

Lyu, B.; Meng, Q.; Qiu, S.; Wang, Z.; Yang, Z.; and Jordan, M. I. 2022a. Learning Dynamic Mechanisms in Unknown Environments: A Reinforcement Learning Approach.

Lyu, B.; Wang, Z.; Kolar, M.; and Yang, Z. 2022b. Pessimism meets VCG: Learning Dynamic Mechanism Design via Offline Reinforcement Learning. *arXiv preprint arXiv:2205.02450*.

Martin, C.; and Sandholm, T. 2023. Finding mixed-strategy equilibria of continuous-action games without gradients using randomized policy networks. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Myerson, R. B. 1981. Optimal auction design. *Mathematics of Operations Research*, 6(1): 58–73.

Nisan, N.; and Ronen, A. 2001. Algorithmic Mechanism Design. *Games and Economic Behavior*, 35(1-2): 166–196.

Nisan, N.; Roughgarden, T.; Tardos, E.; and Vazirani, V. V. 2007a. *Algorithmic Game Theory*. Cambridge University Press.

Nisan, N.; Roughgarden, T.; Tardos, E.; and Vazirani, V. V. 2007b. *Algorithmic Game Theory*, chapter 16. Cambridge University Press.

Pai, M.; and Vohra, R. V. 2008. Optimal Dynamic Auctions. Working Paper 1461, Discussion Paper.

Pavan, A.; Segal, I.; and Toikka, J. 2014. Dynamic mechanism design: A Myersonian approach. *Econometrica*, 82(2): 601–653.

Rahme, J.; Jelassi, S.; Bruna, J.; and Weinberg, S. M. 2021. A Permutation-Equivariant Neural Network Architecture For Auction Design. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Rahme, J.; Jelassi, S.; and Weinberg, S. M. 2021. Auction learning as a two-player game. In *International Conference on Learning Representations (ICLR)*.

Roberts, K. 1979. The characterization of implementable choice rules. *Aggregation and Revelation of Preferences*, 12(2): 321–348.

Sandholm, T. 2003. Automated mechanism design: A new application area for search algorithms. In *International Conference on Principles and Practice of Constraint Programming*.

Sandholm, T.; and Boutilier, C. 2006. Preference Elicitation in Combinatorial Auctions. In Cramton, P.; Shoham, Y.; and Steinberg, R., eds., *Combinatorial Auctions*, 233–263. MIT Press. Chapter 10.

Sandholm, T.; and Likhodedov, A. 2015. Automated design of revenue-maximizing combinatorial auctions. *Operations Research*, 63(5): 1000–1025.

Sandholm, T. W.; Conitzer, V.; and Boutilier, C. 2007. Automated Design of Multistage Mechanisms. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Shen, W.; Peng, B.; Liu, H.; Zhang, M.; Qian, R.; Hong, Y.; Guo, Z.; Ding, Z.; Lu, P.; and Tang, P. 2017. Reinforcement Mechanism Design, with Applications to Dynamic Pricing in Sponsored Search Auctions. *CoRR*, abs/1711.10279.

Shen, W.; Tang, P.; and Zuo, S. 2019. Automated mechanism design via neural networks. In *International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*.

Sobol', I. 1967. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4): 86–112.

Soumalias, E.; Zamanlooy, B.; Weissteiner, J.; and Seuken, S. 2023. Machine Learning-powered Course Allocation.

Sow, D.; Ji, K.; and Liang, Y. 2022. On the Convergence Theory for Hessian-Free Bilevel Algorithms. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Neural Information Processing Systems (NeurIPS)*.

Sutton, R. S. 2018. *Reinforcement learning : : an introduction*. Adaptive computation and machine learning series. Cambridge, Massachusetts: The MIT Press, second edition. edition. ISBN 0-262-35270-2.

Vickrey, W. 1961. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of Finance*, 16(1): 8–37.

Weed, J. 2018. An explicit analysis of the entropic penalty in linear programming. In Bubeck, S.; Perchet, V.; and Rigollet, P., eds., *Proceedings of the 31st Conference On Learning Theory*, volume 75 of *Proceedings of Machine Learning Research*, 1841–1855. PMLR.

Weissteiner, J.; Heiss, J.; Siems, J.; and Seuken, S. 2023. Bayesian Optimization-based Combinatorial Assignment. *arXiv preprint arXiv:2208.14698*.

Yao, A. C.-C. 2017. Dominant-strategy versus bayesian multi-item auctions: Maximum revenue determination and comparison. In *Economics and Computation (EC)*.

Zinkevich, M.; Blum, A.; and Sandholm, T. 2003. On Polynomial-Time Preference Elicitation with Value Queries. In *Economics and Computation (EC)*, 176–185.