Machine Learning-Powered Combinatorial Clock Auction

Ermis Nikiforos Soumalias[1,3†], **Jakob Weissteiner**[1,3†], **Jakob Heiss**[2,3], **Sven Seuken**[1,3]

[1]University of Zurich
[2]ETH Zurich
[3]ETH AI Center
ermis@ifi.uzh.ch, weissteiner@ifi.uzh.ch, jakob.heiss@math.ethz.ch, seuken@ifi.uzh.ch

Abstract

We study the design of *iterative combinatorial auctions (ICAs)*. The main challenge in this domain is that the bundle space grows exponentially in the number of items. To address this, several papers have recently proposed machine learning (ML)-based preference elicitation algorithms that aim to elicit only the most important information from bidders. However, from a practical point of view, the main shortcoming of this prior work is that those designs elicit bidders' preferences via *value queries* (i.e., "What is your value for the bundle $\{A, B\}$?"). In most real-world ICA domains, value queries are considered impractical, since they impose an unrealistically high cognitive burden on bidders, which is why they are not used in practice. In this paper, we address this shortcoming by designing an *ML-powered combinatorial clock auction* that elicits information from the bidders only via *demand queries* (i.e., "At prices p, what is your most preferred bundle of items?"). We make two key technical contributions: First, we present a novel method for training an ML model on demand queries. Second, based on those trained ML models, we introduce an efficient method for determining the demand query with the highest clearing potential, for which we also provide a theoretical foundation. We experimentally evaluate our ML-based demand query mechanism in several spectrum auction domains and compare it against the most established real-world ICA: the *combinatorial clock auction (CCA)*. Our mechanism significantly outperforms the CCA in terms of efficiency in all domains, it achieves higher efficiency in a significantly reduced number of rounds, and, using linear prices, it exhibits vastly higher clearing potential. Thus, with this paper we bridge the gap between research and practice and propose the first practical ML-powered ICA.

1 Introduction

Combinatorial auctions (CAs) are used to allocate multiple items among several bidders who may view those items as complements or substitutes. In a CA, bidders are allowed to submit bids over *bundles* of items. CAs have enjoyed widespread adoption in practice, with their applications ranging from allocating spectrum licences (Cramton 2013) to TV ad slots (Goetzendorff et al. 2015) and airport landing/take-off slots (Rassenti, Smith, and Bulfin 1982).

One of the key challenges in CAs is that the bundle space grows exponentially in the number of items, making it infeasible for bidders to report their full value function in all but the smallest domains. Moreover, Nisan and Segal (2006) showed that for general value functions, CAs require an exponential number of bids in order to achieve full efficiency in the worst case. Thus, practical CA designs cannot provide efficiency guarantees in real world settings with more than a modest number of items. Instead, the focus has shifted towards *iterative combinatorial auctions (ICAs)*, where bidders interact with the auctioneer over a series of rounds, providing a limited amount of information, and the aim of the auctioneer is to find a highly efficient allocation.

The most established mechanism following this interaction paradigm is the *combinatorial clock auction (CCA)* (Ausubel, Cramton, and Milgrom 2006). The CCA has been used extensively for spectrum allocation, generating over $\$20$ Billion in revenue between 2012 and 2014 alone (Ausubel and Baranov 2017). Speed of convergence is a critical consideration for any ICA since each round can entail costly computations and business modelling for the bidders (Kwasnica et al. 2005; Milgrom and Segal 2017; Bichler, Hao, and Adomavicius 2017). Large spectrum auctions following the CCA format can take more than 100 bidding rounds. In order to decrease the number of rounds, many CAs in practice use aggressive price update rules (e.g., increasing prices by up to 10% each round), which can harm efficiency (Ausubel and Baranov 2017). Thus, it remains a challenging problem to design a practical ICA that elicits information via demand queries, is efficient, and converges in a small number of rounds. Specifically, given the value of resources allocated in such real-world ICAs, increasing their efficiency by even one percentage point already translates into monetary gains of hundreds of millions of dollars.

1.1 ML-Powered Preference Elicitation

To address this challenge, researchers have proposed various ways of using machine learning (ML) to improve the efficiency of CAs. The seminal works by Blum et al. (2004) and Lahaie and Parkes (2004) were the first to frame preference elicitation in CAs as a learning problem. In the same strand of research, Brero, Lubin, and Seuken (2018,

*The full paper including appendix is available on arXiv via: https://arxiv.org/abs/2308.10226.

†These authors contributed equally.

2021), Weissteiner and Seuken (2020) and Weissteiner et al. (2022b) proposed ML-powered ICAs. At the heart of those approaches lies an ML-powered preference elicitation algorithm that uses an ML model to approximate each bidder's value function and to generate the next value query, which in turn refines that bidder's model. Weissteiner et al. (2022a) designed a special network architecture for this framework while Weissteiner et al. (2023) incorporated a notion of uncertainty (Heiss et al. 2022) into the framework, further increasing its efficiency. Despite their great efficiency gains compared to traditional CA designs, those approaches suffer from one common limitation: they fundamentally rely on value queries of the form "What is your value for bundle $\{A, B\}$". Prior research in auction design has identified demand queries (DQs) as the best way to run an auction (Cramton 2013). Their advantages compared to value queries include elimination of tacit collusion and bid signaling, as well as simplified bidder decision-making that keeps the bidders focused on what is most relevant: the relationship between prices and aggregate demand. Additionally, value queries are cognitively more complex, and thus typically impractical for real-world ICAs. For these reasons, DQs are the most prominent interaction paradigm for auctions in practice.

Despite the prominence of DQs in real-world applications, the only prior work on ML-based DQs that we are aware of is that of Brero and Lahaie (2018) and Brero, Lahaie, and Seuken (2019), who proposed integrating ML in a price-based ICA to generate the next price vector in order to achieve faster convergence. Similar to our design, in these works the auctioneer maintains a model of each agent's value function, which are updated as the agents bid in the auction and reveal more information about their values. Then, those models are used in each round to compute new prices and drive the bidding process. Unlike our approach, the design of this prior work focuses solely on clearing potential, as the authors do not report efficiency results. Additionally, their design suffers from some significant limitations: (i) it does not exploit any notion of similarity between bundles that contain overlapping items, (ii) it only incorporates a fraction of the information revealed by the agents' bidding. Specifically, it only makes use of the fact that for the bundle an agent bids on, her value for that bundle must be larger than its price, and (iii) their approach is computationally intractable already in medium-sized auction domains, as their price update rule requires a large number l of posterior samples for *expectation maximization* and then solving a linear program whose number of constraints for each bidder is proportional to l times the number of bids by that agent. These limitations are significant, as they can lead to large efficiency decreases in complex combinatorial domains. Moreover, their design cannot be easily modified to alleviate these limitations. In contrast, our approach effectively addresses all of these limitations.

1.2 Our Contributions

In this paper, we address the main shortcomings of prior work by designing an ML-powered combinatorial clock auction. Our auction elicits information from bidders via *demand queries (DQs)* instead of value queries, while simultaneously, unlike prior work on ML-based DQs, being computationally feasible for large domains and incorporating the *complete information* the demand query observations provide into the training of our ML models. Concretely, we use *Monotone-Value Neural Networks (MVNNs)* (Weissteiner et al. 2022a) as ML models, which are tailored to model monotone and non-linear combinatorial value functions in CAs.

The main two technical challenges are (i) training those MVNNs only on *demand query* observations and (ii) efficiently determining the next demand query that is most likely to clear the market based on the trained MVNNs. In detail, we make the following contributions:

1. We first propose an adjusted MVNN architecture, which we call *multiset MVNNs (mMVNNs)*. mMVNNs can be used more generally in *multiset domains* (i.e., if multiple indistinguishable copies of the same good exist) (Section 3.1) and we prove the universality property of mMVNNs in such multiset domains (Theorem 1).
2. We introduce a novel method for training *any* MIP-formalizable and gradient descent (GD)-compatible ML model (e.g., mMVNNs) on *demand query* observations (Section 3.2).[1] Unlike prior work, our training method provably makes use of the *complete* information provided by the demand query observations.
3. We introduce an efficient method for determining the price vector that is most likely to clear the market based on the trained ML models (Section 4). For this, we derive a simple and intuitive price update rule that results from performing GD on an objective function which is minimized exactly at clearing prices (Theorem 3).
4. Based on Items 2 and 3, we propose a practical ML-powered clock auction (Section 5).
5. We experimentally show that compared to the CCA, our ML-powered clock auction can achieve substantially higher efficiency on the order of 9% points. Furthermore, using linear prices, our ML-powered clock auction exhibits significantly higher clearing potential compared to the CCA (Section 6).

GitHub Our source code is publicly available on GitHub at https://github.com/marketdesignresearch/ML-CCA.

1.3 Further Related Work

In the field of *automated mechanism design*, Dütting et al. (2015, 2019), Golowich, Narasimhan, and Parkes (2018) and Narasimhan, Agarwal, and Parkes (2016) used ML to learn new mechanisms from data, while Cole and Roughgarden (2014); Morgenstern and Roughgarden (2015) and Balcan, Sandholm, and Vitercik (2023) bounded the sample complexity of learning approximately optimal mechanisms. In contrast to this prior work, our design incorporates an ML algorithm into the mechanism itself, i.e., the ML algorithm is part of the mechanism. Lahaie and Lubin (2019) suggest an adaptive price update rule that increases price expressivity as the rounds progress in order to improve efficiency and

[1] Namely, this includes neural networks with any piecewise linear activation function.

speed of convergence. Unlike that work, we aim to improve preference elicitation while still using linear prices. Preference elicitation is a key market design challenge outside of CAs too. Soumalias et al. (2023) introduce an ML-powered mechanism for course allocation that improves preference elicitation by asking students comparison queries.

1.4 Practical Considerations and Incentives

Our ML-powered clock phase can be viewed as an alternative to the clock phase of the CCA. In a real-world application, many other considerations (beyond the price update rule) are also important. For example, the careful design of *activity rules* is vital to induce truthful bidding in the clock phase of the CCA (Ausubel and Baranov 2017). The payment rule used in the supplementary round is also important, and it has been argued that the use of the VCG-nearest payment rule, while not strategy-proof, induces good incentives in practice (Cramton 2013). Similar to the clock phase of the CCA, our ML-powered clock phase is not strategyproof. If our design were to be fielded in a real-world environment, we envision that one would combine it with carefully designed activity and payment rules in order to induce good incentives. Thus, we consider the incentive problem orthogonal to the price update problem and in the rest of the paper, we follow prior work (Brero, Lahaie, and Seuken 2019; Parkes and Ungar 2000) and assume that bidders follow myopic best-response (truthful) bidding throughout all auction mechanisms tested.

2 Preliminaries

2.1 Formal Model for ICAs

We consider *multiset* CA domains with a set $N = \{1, \dots, n\}$ of bidders and a set $M = \{1, \dots, m\}$ of distinct items with corresponding *capacities*, i.e., number of available copies, $c = (c_1, \dots, c_m) \in \mathbb{N}^m$. We denote by $x \in \mathcal{X} = \{0, \dots, c_1\} \times \dots \times \{0, \dots, c_m\}$ a bundle of items represented as a positive integer vector, where $x_j = k$ iff item $j \in M$ is contained k-times in x. The bidders' true preferences over bundles are represented by their (private) value functions $v_i : \mathcal{X} \to \mathbb{R}_{\geq 0},\ i \in N$, i.e., $v_i(x)$ represents bidder i's true value for bundle $x \in \mathcal{X}$. We collect the value functions v_i in the vector $v = (v_i)_{i \in N}$. By $a = (a_1, \dots, a_n) \in \mathcal{X}^n$ we denote an allocation of bundles to bidders, where a_i is the bundle bidder i obtains. We denote the set of *feasible* allocations by $\mathcal{F} = \left\{a \in \mathcal{X}^n : \sum_{i \in N} a_{ij} \leq c_j,\ \forall j \in M\right\}$. We assume that bidders have quasilinear utility functions u_i of the form $u_i(a_i) = v_i(a_i) - \pi_i$ where v_i can be highly nonlinear and $\pi_i \in \mathbb{R}_{\geq 0}$ denotes the bidder's payment. This implies that the (true) *social welfare* $V(a)$ of an allocation a is equal to the sum of all bidders' values $\sum_{i \in N} v_i(a_i)$. We let $a^* \in \operatorname{argmax}_{a \in \mathcal{F}} V(a)$ denote a social-welfare maximizing, i.e., *efficient*, allocation. The *efficiency* of any allocation $a \in \mathcal{F}$ is determined as $V(a)/V(a^*)$.

An ICA *mechanism* defines how the bidders interact with the auctioneer and how the allocation and payments are determined. In this paper, we consider ICAs that iteratively ask bidders *linear demand queries*. In such a query, the auctioneer presents a vector of item prices $p \in \mathbb{R}^m_{\geq 0}$ and each bidder i responds with her utility-maximizing bundle, i.e.,

$$x_i^*(p) \in \operatorname*{argmax}_{x \in \mathcal{X}} \{v_i(x) - \langle p, x \rangle\}\ i \in N, \tag{1}$$

where $\langle \cdot, \cdot \rangle$ denotes the Euclidean scalar product in $\mathbb{R}^m$.

Even though our approach could conceptually incorporate any kind of (non-linear) price function $p : \mathcal{X} \to \mathbb{R}_{\geq 0}$, our concrete implementation will only use linear prices (i.e., prices over items). Linear prices are most established in practice since they are intuitive and simple for the bidders to understand (e.g., (Ausubel, Cramton, and Milgrom 2006)).

For bidder $i \in N$, we denote a set of $K \in \mathbb{N}$ such elicited utility-maximizing bundles and price pairs as $R_i = \{(x_i^*(p^r), p^r)\}_{r=1}^K$. Let $R = (R_1, \dots, R_n)$ be the tuple of elicited demand query data from all bidders. The ICA's (inferred) optimal feasible allocation $a^*(R) \in \mathcal{F}$ and payments $\pi_i := \pi_i(R) \in \mathbb{R}^n_+$ are computed based on the elicited reports R *only*. Concretely, $a^*_R \in \mathcal{F}$ is defined as

$$a^*(R) \in \operatorname*{argmax}_{\substack{\mathcal{F} \ni (x_i^*(p^{r_i}))_{i=1}^n \\ : r_i \in \{1, \dots, K\}\ \forall i \in N}} \sum_{i \in N} \langle p^{r_i}, x_i^*(p^{r_i}) \rangle . \tag{2}$$

In words, a bidder's response to a demand query provides a lower bound on that bidder's value for the bundle she requested. That lower bound is equal to the bundle's price in the round the bundle was requested. The ICA's optimal (inferred) feasible allocation $a^*(R) \in \mathcal{F}$ is the one that maximizes the corresponding lower bound on social welfare, based on all elicited demand query data R from the bidders. As payment rule $\pi_i(R)$ one could use any reasonable choice (e.g., VCG payments, see Appendix A). As the auctioneer can only ask a limited number of demand queries $|R_i| \leq Q^{\max}$ (e.g., $Q^{\max} = 100$), an ICA needs a practically feasible and smart preference elicitation algorithm.

2.2 The Combinatorial Clock Auction (CCA)

We consider the CCA (Ausubel, Cramton, and Milgrom 2006) as the main benchmark auction. The CCA consists of two phases. The initial *clock phase* proceeds in rounds. In each round, the auctioneer presents anonymous item prices $p \in \mathbb{R}^m_{\geq 0}$, and each bidder is asked to respond to a *demand query*, declaring her utility-maximizing bundle at p. The clock phase of the CCA is parametrized by the *reserve prices* employed in its first round, and the way prices are updated. An item j is *over-demanded* at prices p, if, for those prices, its total demand based on the bidders' responses to the demand query exceeds its capacity, i.e., $\sum_{i \in N} (x_i^*(p))_j > c_j$. The most common price update rule is to increase the price of all over-demanded items by a fixed percentage, which we set to 5% for our experiments, as in many real-world applications (e.g., (Industry Canada 2013)).

The second phase of the CCA is *the supplementary round*. In this phase, each bidder can submit a finite number of additional bids for bundles of items, which are called *push bids*. Then, the final allocation is determined based on the combined set of all inferred bids of the clock phase, plus all submitted push bids of the supplementary round. This design aims to combine good price discovery in the clock phase with good expressiveness in the supplementary round. In

Algorithm 1: TRAINONDQS

Input : Demand query data $R_i = \{(x_i^*(p^r), p^r)\}_{r=1}^{K}$, Epochs $T \in \mathbb{N}$, Learning Rate $\gamma > 0$.

1 $\theta_0 \leftarrow$ init mMVNN ▷ Weissteiner et al. (2023, S.3.2)
2 **for** $t = 0$ *to* $T-1$ **do**
3 **for** $r = 1$ *to* K **do** ▷ Demand responses for prices
4 Solve $\hat{x}_i^*(p^r) \in \operatorname{argmax}_{x \in \mathcal{X}} \mathcal{M}_i^{\theta_t}(x) - \langle p^r, x \rangle$
5 **if** $\hat{x}_i^*(p^r) \neq x_i^*(p^r)$ **then** ▷ mMVNN is wrong
6 $L(\theta_t) \leftarrow (\mathcal{M}_i^{\theta_t}(\hat{x}_i^*(p^r)) - \langle p^r, \hat{x}_i^*(p^r) \rangle) - (\mathcal{M}_i^{\theta_t}(x_i^*(p^r)) - \langle p^r, x_i^*(p^r) \rangle)$ ▷ Add predicted utility difference to loss
7 $\theta_{t+1} \leftarrow \theta_t - \gamma(\nabla_\theta L(\theta))_{\theta=\theta_t}$ ▷ SGD step
8 **return** *Trained parameters* θ_T *of the mMVNN* $\mathcal{M}_i^{\theta_T}$

simulations, the supplementary round is parametrized by the assumed bidder behaviour in this phase, i.e., which bundle-value pairs they choose to report. As in (Brero, Lubin, and Seuken 2021), we consider the following heuristics when simulating bidder behaviour:

- **Clock Bids:** Corresponds to having no supplementary round. Thus, the final allocation is determined based only on the inferred bids of the clock phase (Equation (2)).
- **Raised Clock Bids**: The bidders also provide their true value for all bundles they bid on during the clock phase.
- **Profit Max:** Bidders provide their true value for all bundles that they bid on in the clock phase, and additionally submit their true value for the $Q^{\text{P-Max}}$ bundles earning them the highest utility at the prices of the final clock phase.

3 Training on Demand Query Observations

In this section, we first propose a new version of MVNNs that are applicable to multiset domains $\mathcal{X}$ and extend the universality proof of classical MVNNs. Finally, we present our demand-query training algorithm.

3.1 Multiset MVNNs

MVNNs (Weissteiner et al. 2022a) are a recently introduced class of NNs specifically designed to represent *monotone combinatorial* valuations. We introduce an adapted version of MVNNs, which we call *multiset MVNNs (mMVNNs)*. Compared to MVNNs, mMVNNs have an added linear normalization layer D after the input layer. We add this normalization since the input (i.e., a bundle) $x \in \mathcal{X}$ is a positive integer vector instead of a binary vector as in the classic case of indivisible items with capacities $c_j = 1$ for all $j \in M$. This normalization ensures that $Dx \in [0,1]$ and thus we can use the weight initialization scheme from (Weissteiner et al. 2023). Unlike MVNNs, mMVNNs incorporate at a structural level the prior information that some items are identical and consequently significantly reduce the dimensionality of the input space. This improves the sample efficiency of mMVNNs, which is especially important in applications with a limited number of samples such as auctions. For more details on mMVNNs and their advantages, please see Appendix B.

Definition 1 (Multiset MVNN)**.** *An mMVNN* $\mathcal{M}_i^\theta : \mathcal{X} \to \mathbb{R}_{\geq 0}$ *for bidder* $i \in N$ *is defined as*

$$\mathcal{M}_i^\theta(x) \coloneqq W^{i,K_i} \varphi_{0,t^{i,K_i-1}} \left(\dots \varphi_{0,t^{i,1}}(W^{i,1}(Dx) + b^{i,1}) \dots \right) \tag{3}$$

- $K_i + 2 \in \mathbb{N}$ *is the number of layers (K_i hidden layers),*
- $\{\varphi_{0,t^{i,k}}\}_{k=1}^{K_i-1}$ *are the MVNN-specific activation functions with cutoff* $t^{i,k} > 0$*, called* bounded ReLU (bReLU)*:*

$$\varphi_{0,t^{i,k}}(\cdot) \coloneqq \min(t^{i,k}, \max(0, \cdot)) \tag{4}$$

- $W^i \coloneqq (W^{i,k})_{k=1}^{K_i}$ *with* $W^{i,k} \geq 0$ *and* $b^i \coloneqq (b^{i,k})_{k=1}^{K_i-1}$ *with* $b^{i,k} \leq 0$ *are the* non-negative *weights and* non-positive *biases of dimensions* $d^{i,k} \times d^{i,k-1}$ *and* $d^{i,k}$*, whose parameters are stored in* $\theta = (W^i, b^i)$.
- $D \coloneqq \operatorname{diag}(1/c_1, \dots, 1/c_m)$ *is the linear normalization layer that ensures* $Dx \in [0,1]$ *and is not trainable.*

In Theorem 1, we extend the proof from Weissteiner et al. (2022a) and show that mMVNNs are *also universal* in the set of monotone value functions defined on a multiset domain $\mathcal{X}$. For this, we first define the following properties:

(M) Monotonicity (*"more items weakly increase value"*): For $a, b \in \mathcal{X}$: if $a \leq b$, i.e. $\forall k \in M : a_i \leq b_i$, it holds that $v_i(a) \leq v_i(b)$,

(N) Normalization (*"no value for empty bundle"*): $v_i(\emptyset) = v_i((0, \dots, 0)) \coloneqq 0$,

These properties are common assumptions and are satisfied in many market domains. We can now present the following universality result:

Theorem 1 (Multiset Universality)**.** Any *value function* $v_i : \mathcal{X} \to \mathbb{R}_+$ *that satisfies* ***(M)*** *and* ***(N)*** *can be represented exactly as an mMVNN* $\mathcal{M}_i^\theta$ *from Definition 1, i.e., for* $\mathcal{V} \coloneqq \{v_i : \mathcal{X} \to \mathbb{R}_+ \mid$ *satisfy* ***(M)*** *and* ***(N)***$\}$ *it holds that*

$$\mathcal{V} = \left\{ \mathcal{M}_i^{(W^i, b^i)} : W^i \geq 0, b^i \leq 0 \} \right\}. \tag{5}$$

Proof. Please, see Appendix B.2 for the proof. □

Furthermore, we can formulate maximization over mMVNNs, i.e., $\max_{x \in \mathcal{X}} \mathcal{M}_i^\theta(x) - \langle p, x \rangle$, as a *mixed integer linear program (MILP)* analogously to Weissteiner et al. (2022a), which will be key for our ML-powered clock phase.

3.2 Training Algorithm

In Algorithm 1, we describe how we train, for each bidder $i \in N$, a distinct mMVNN $\mathcal{M}_i^\theta$ on demand query data R_i. Our design choices regarding this training algorithm are motivated by the information that responses to demand queries provide. According to myopic best response bidding, at each round r, bidder i reports a utility-maximizing bundle $x_i^*(p^r) \in \mathcal{X}$ at current prices p^r. Formally, for all $x \in \mathcal{X}$:

$$v_i(x_i^*(p^r)) - \langle p^r, x_i^*(p^r) \rangle \geq v_i(x) - \langle p^r, x \rangle. \tag{6}$$

Notice that for any epoch t and round r, the loss $L(\theta_t)$ for that round calculated in Lines 4 to 6 is always non-negative, and can only be zero if the mMVNN $\mathcal{M}_i^\theta$ (instead of v_i) satisfies Equation (6). Thus, the loss for an epoch is zero iff the mMVNN $\mathcal{M}_i^\theta$ satisfies Equation (6) for all rounds,

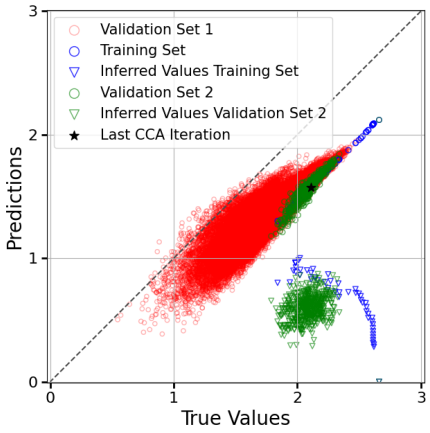

Figure 1: Scaled prediction vs. true plot of a trained mMVNN via Algorithm 1 for the national bidder in the MRVM domain (see Section 6).

and in that case the model has captured the full information provided by the demand query responses R_i of that bidder. Finally, note that Algorithm 1 can be applied to any MILP-formalizable ML model whose parameters can be efficiently updated via GD, such as MVNNs or ReLU-NNs.

In Figure 1, we present a prediction vs. true plot of an mMVNN, which we trained via Algorithm 1. We present the training set of 50 demand query data points R_i in blue circles, where the prices $\{p^r\}_{r=1}^{50}$ are generated according to the same rule as in CCA. Additionally, we mark the bundle $x^{\text{CCA}} \in \mathcal{X}$ from this last CCA iteration (i.e., the one resulting from p^{50}) with a black star. Moreover, we present two different validation sets on which we evaluate mMVNN configurations in our hyperparameter optimization (HPO): *Validation set 1* (red circles), which are $50,000$ uniformly at random sampled bundles $x \in \mathcal{X}$, and *validation set 2* (green circles), where we first sample 500 price vectors $\{p^r\}_{r=1}^{500}$ where the price of each item is drawn uniformly at random from the range of 0 to 3 times the average maximum value of an agent of that type for a single item, and then determine utility-maximizing bundles $x_i^*(p^r)$ (w.r.t. v_i) at those prices (cp. Equation (1)). While validation set 1 measures generalization performance in a classic sense over the whole bundle space, validation set 2 focuses on utility-maximizing bundles. We additionally demonstrate the inferred values of the bundles of the training set and validation set 2 using triangles of the same colour, i.e., $\{\langle p^r, x_i^*(p^r)\rangle\}_{r=1}^{50/500}$. These triangles highlight the only cardinal information that our mMVNNs have access to during training and are a lower bound of the true value. In Figure 1, we see that our mMVNN is able to learn at the training points (blue circles) the true value functions almost perfectly up to a constant shift κ, i.e., $\mathcal{M}_i^{\theta_T}(x) \approx v_i(x) + \kappa$. This is true even though the corresponding inferred values (blue triangles) are very far off from the true values $\langle p^r, x_i^*(p^r)\rangle \ll v_i(x_i^*(p^r))$. Moreover, the mMVNN generalizes well (up to the constant shift κ) on validation sets 1 and 2. Overall, this shows that Algorithm 1 indeed leads to mMVNNs $\mathcal{M}_i^{\theta_T}$ which are a good approximation of $v_i + \kappa$. Note that learning the true value function up to a constant shift suffices for our proposed demand query generation procedure presented in Section 4.

4 ML-powered Demand Query Generation

In this section, we show how we generate ML-powered demand queries and provide the theoretical foundation for our approach by extending a well-known connection between *clearing prices*, *efficiency* and a *clearing objective function*. First, we define indirect utility, revenue and clearing prices.

Definition 2 (Indirect Utility and Revenue)**.** *For linear prices $p \in \mathbb{R}^m_{\geq 0}$, a bidder's indirect utility U and the seller's indirect revenue R are defined as*

$$U(p, v_i) := \max_{x \in \mathcal{X}} \{v_i(x) - \langle p, x\rangle\} \text{ and} \tag{7}$$

$$R(p) := \max_{a \in \mathcal{F}} \left\{ \sum_{i \in N} \langle p, a_i \rangle \right\} \overset{1}{=} \sum_{j \in M} c_j p_j, \tag{8}$$

i.e., at prices p, Equations (7) *and* (8) *are the maximum utility a bidder can achieve for all $x \in \mathcal{X}$ and the maximum revenue the seller can achieve among all feasible allocations.*

Definition 3 (Clearing Prices)**.** *Prices $p \in \mathbb{R}^m_{\geq 0}$ are* clearing prices *if there exists an allocation $a(p) \in \mathcal{F}$ such that*

1. *for each bidder i, the bundle $a_i(p)$ maximizes her utility, i.e., $v_i(a_i(p)) - \langle p, a_i(p)\rangle = U(p, v_i), \forall i \in N$, and*
2. *the allocation $a(p) \in \mathcal{F}$ maximizes the sellers revenue, i.e., $\sum_{i \in N} \langle p, a_i(p)\rangle = R(p)$.*[2]

Next, we provide an important connection between *clearing prices*, *efficiency* and a *clearing objective W*. Theorem 2 extends Bikhchandani and Ostroy (2002, Theorem 3.1).

Theorem 2. *Consider the notation from Definitions 2 and 3 and the objective function $W(p, v) := R(p) + \sum_{i \in N} U(p, v_i)$. Then it holds that, if a linear clearing price vector exists, every price vector*

$$p' \in \operatorname*{argmin}_{\tilde{p} \in \mathbb{R}^m_{\geq 0}} W(\tilde{p}, v) \tag{9a}$$

$$\text{such that} \quad (x_i^*(\tilde{p}))_{i \in N} \in \mathcal{F} \tag{9b}$$

is a clearing price vector and the corresponding allocation $a(p') \in \mathcal{F}$ is efficient.[3]

Proof. Please, see Appendix C.1 for the proof. □

Theorem 2 does not claim the existence of *linear clearing prices (LCPs)* $p \in \mathbb{R}^m_{\geq 0}$. For general value functions v, LCPs may not exist (Bikhchandani and Ostroy 2002). However, in the case that LCPs do exist, Theorem 2 shows that *all* minimizers of (9) are LCPs and their corresponding allocation

[2] For linear prices, this maximum is achieved by selling every item, i.e., $\forall j \in M : \sum_{i \in N} (a_i)_j = c_j$ (see Appendix C.2).

[3] More precisely, constraint (9b) should be reformulated as

$$\exists\, (x_i^*(\tilde{p}))_{i \in N} \in \bigtimes_{i \in N} \mathcal{X}_i^*(\tilde{p}) : (x_i^*(\tilde{p}))_{i \in N} \in \mathcal{F},$$

where $\mathcal{X}_i^*(\tilde{p}) := \operatorname{argmax}_{x \in \mathcal{X}} \{v_i(x) - \langle \tilde{p}, x\rangle\}$, since in theory, $x_i^*(\tilde{p})$ does not always have to be unique.

is efficient. This is at the core of our ML-powered demand query generation approach, which we discuss next.

The key idea to generate ML-powered demand queries is as follows: As an approximation for the true value function v_i, we use for each bidder a distinct mMVNN $\mathcal{M}_i^\theta : \mathcal{X} \to \mathbb{R}_{\geq 0}$ that has been trained on the bidder's elicited demand query data R_i (see Section 3). Motivated by Theorem 2, we then try to find the demand query $p \in \mathbb{R}^m_{\geq 0}$ minimizing $W(p, \left(\mathcal{M}_i^\theta\right)_{i=1}^n)$ subject to the feasibility constraint (9b). This way, we find demand queries $p \in \mathbb{R}^m_{\geq 0}$ which, given the already observed demand responses R, have high clearing potential. Note that unlike the CCA, this process does not result in monotone prices.[4]

Remark 1 (Constraint (9b))**.** *An important economic insight is that minimizing $W(\cdot, (\mathcal{M}_i^\theta)_{i=1}^n)$ is optimal, when LCPs exist (also without constraint* (9b) *as shown in Lemma 2 in Appendix C.1). If however LCPs do not exist, it is favourable to minimize W under the constraint of having no predicted over-demand for any items (see Appendix D.9 for an empirical comparison of minimizing W with and without constraint* (9b)*). This is because in case the market does not clear, our ML-CCA (see Section 5), just like the CCA, will have to combine the clock bids of the agents to produce a* feasible *allocation with the highest inferred social welfare according to Equation* (2)*. See Appendix D.6 for details.*

Note that (9) is a hard, bi-level optimization problem. We minimize (9) via gradient descent (GD), since Theorem 3 gives us the gradient and convexity of $W(\cdot, \left(\mathcal{M}_i^\theta\right)_{i=1}^n)$.

Theorem 3. *Let $\left(\mathcal{M}_i^\theta\right)_{i=1}^n$ be a tuple of trained mMVNNs and let $\hat{x}_i^*(p) \in \operatorname{argmax}_{x \in \mathcal{X}} \left\{\mathcal{M}_i^\theta(x) - \langle p, x\rangle\right\}$ denote each bidder's predicted utility maximizing bundle w.r.t. $\mathcal{M}_i^\theta$. Then it holds that $p \mapsto W(p, \left(\mathcal{M}_i^\theta\right)_{i=1}^n)$ is* convex*,* Lipschitz-continuous *and* a.e. differentiable*. Moreover,*

$$c - \sum_{i \in N} \hat{x}_i^*(p) \in \nabla_p^{\text{sub}} W(p, \left(\mathcal{M}_i^\theta\right)_{i=1}^n) \tag{10}$$

is always a sub-gradient and a.e. a classical gradient.

Proof. In Appendix C.2 we provide the full proof. Concretely, Lemmas 3 and 4 prove the Lipschitz-continuity and the convexity. In the following, we provide a sketch of how the (sub-)gradients are derived. First, since $\mathcal{X}$ is finite, it is intuitive that $\hat{x}_i^*(p)$ is a piece-wise constant function and thus $\partial_p \hat{x}_i^*(p) \overset{\text{a.e.}}{=} 0$ (as intuitively argued by Pogančić et al. (2020) and proven by us in Lemma 6). Then we can compute the gradient a.e. as if $\hat{x}_i^*(p)$ was a constant:

$$\begin{aligned}
\nabla_p W\left(p, \left(\mathcal{M}_i^\theta\right)_{i=1}^n\right) &= \nabla_p \left(R(p) + \sum_{i \in N} U(p, \mathcal{M}_i^\theta)\right) \\
&= \nabla_p \left(\sum_{j \in M} c_j p_j + \sum_{i \in N} \left(\mathcal{M}_i^\theta(\hat{x}_i^*(p)) - \langle p, \hat{x}_i^*(p)\rangle\right)\right) \\
&\overset{\text{a.e.}}{=} c + \sum_{i \in N} (0 - \nabla_p \langle p, \hat{x}_i^*(p)\rangle) = c - \sum_{i \in N} \hat{x}_i^*(p).
\end{aligned}$$

For a mathematically rigorous derivation of sub-gradients and a.e. differentiability see Lemmas 5 and 6. □

With Theorem 3, we obtain the following update rule of classical GD $p_j^{\text{new}} \overset{a.e.}{=} p_j - \gamma(c_j - \sum_{i \in N} (\hat{x}_i^*(p))_j)$, $\forall j \in M$. Interestingly, this equation has an intuitive economic interpretation. If the j^{th} item is over/under-demanded based on the predicted utility-maximizing bundles $\hat{x}_i^*(p)$, then its new price p_j^{new} is increased/decreased by the learning rate times its over/under-demand. However, to enforce constraint (9b) in GD, we asymmetrically increase the prices $1 + \mu \in \mathbb{R}_{\geq 0}$ times more in case of over-demand than we decrease them in case of under-demand. This leads to our final update rule (see Item 1 in Appendix D.6 for more details):

$$p_j^{\text{new}} \overset{a.e.}{=} p_j - \tilde{\gamma}_j (c_j - \sum_{i \in N} (\hat{x}_i^*(p))_j), \forall j \in M, \tag{11a}$$

$$\tilde{\gamma}_j \coloneqq \begin{cases} \gamma \cdot (1 + \mu) & , c_j < \sum_{i \in N} (\hat{x}_i^*(p))_j \\ \gamma & , \text{else} \end{cases} \tag{11b}$$

To turn this soft constraint into a hard constraint, we increase this asymmetry via μ iteratively until we achieve feasibility and in the end we select the GD step with the lowest W value *within* those steps that were feasible. Based on the final update rule from (11), we propose NEXTPRICE (Algorithm 3 in Appendix D.6), an algorithm that generates demand queries with high clearing potential, which additionally induce utility-maximizing bundles that are predicted to be feasible (see Appendix D.6 for all details).

5 ML-powered Combinatorial Clock Auction

In this section, we describe our *ML-powered combinatorial clock auction (ML-CCA)*, which is based on our proposed new training algorithm from Section 3 as well as our new demand query generation procedure from Section 4.

We present ML-CCA in Algorithm 2. In Lines 2 to 5, we draw the first Q^{init} price vectors using some initial demand query method F_{init} and receive the bidders' demand responses to those price vectors. Concretely, in Section 6, we report results using the same price update rule as the CCA for F_{init}. In each of the next up to $Q^{\max} - Q^{\text{init}}$ ML-powered rounds, we first train, for each bidder, an mMVNN on her demand responses using Algorithm 1 (Line 7). Next, in Line 9, we call NEXTPRICE to generate the next demand query p based on the agents' trained mMVNNs (see Section 4). If, based on the agents' responses to the demand query (Line 11), our algorithm has found market-clearing

[4] We see no reason why non-monotone prices would introduce additional complexity for the bidders. With our approach, the prices quickly converge to the final prices, and then only change very little, as shown in Figures 5 to 8 of Appendix D.7. For this reason, one could even argue that round-over-round optimizations for the bidders may be *easier* in our auction: given that prices are close to each other round-over-round, the optimal bundle from the last round is still close to optimal (in terms of utility) in the next round.

	GSVM				LSVM				SRVM				MRVM			
MECHANISM	E_{CLOCK}	E_{RAISE}	E_{PROFIT}	CLE.	E_{CLOCK}	E_{RAISE}	E_{PROFIT}	CLE.	E_{CLOCK}	E_{RAISED}	E_{PROFIT}	CLE.	E_{CLOCK}	E_{RAISE}	E_{PROFIT}	CLE.
ML-CCA	98.23	98.93	100.00	56	91.64	96.39	99.95	26	99.59	99.93	100.00	13	93.04	93.31	93.68	0
CCA	90.40	93.59	100.00	3	82.56	91.60	99.76	0	99.63	99.81	100.00	8	92.44	92.62	93.18	0

Table 1: ML-CCA vs CCA. Shown are averages over a test set of 100 synthetic CA instances of the following metrics: efficiency in % for clock bids (E_{CLOCK}), raised clock bids (E_{RAISE}) and raised clock bids plus 100 profit-max bids (E_{PROFIT}) and percentage of instances where clearing prices were found (CLE.). Winners based on a paired t-test with $\alpha = 5\%$ are marked in grey.

Algorithm 2: ML-CCA($Q^{\text{init}}, Q^{\text{max}}, F_{\text{init}}$)

Parameters: $Q^{\text{init}}, Q^{\text{max}}$ with $Q^{\text{init}} \leq Q^{\text{max}}$ and F_{init}

1 $R \leftarrow (\{\})_{i=1}^{N}$
2 **for** $r = 1, ..., Q^{init}$ **do** ▷ Draw Q^{init} initial prices
3 $p^r \leftarrow F_{\text{init}}(R)$
4 **foreach** $i \in N$ **do** ▷ Initial demand query responses
5 $R_i \leftarrow R_i \cup \{(x_i^*(p^r), p^r)\}$
6 **for** $r = Q^{init} + 1, ..., Q^{max}$ **do** ▷ ML-powered rounds
7 **foreach** $i \in N$ **do**
8 $\mathcal{M}_i^\theta \leftarrow$ TRAINONDQS(R_i) ▷ Algorithm 1
9 $p^r \leftarrow$ NEXTPRICE($(\mathcal{M}_i^\theta)_{i=1}^n$) ▷ Algorithm 3
10 **foreach** $i \in N$ **do** ▷ Demand query responses for p^r
11 $R_i \leftarrow R_i \cup \{(x_i^*(p^r), p^r)\}$
12 **if** $\sum_{i=1}^{n}(x_i^*(p^k))_j = c_j \; \forall j \in M$ **then** ▷ Market-clearing
13 Set final allocation $a^*(R) \leftarrow (x_i^*(p^r))_{i=1}^n$
14 Calculate payments $\pi(R) \leftarrow (\pi_i(R))_{i=1}^n$
15 **return** $a^*(R)$ *and* $\pi(R)$
16 **foreach** $i \in N$ **do**
17 $R_i \leftarrow R_i \cup B_i$ ▷ Optional Push bids
18 Calculate final allocation $a^*(R)$ as in Equation (2)
19 Calculate payments $\pi(R)$ ▷ E.g., VCG (Appendix A)
20 **return** $a^*(R)$ *and* $\pi(R)$

prices, then the corresponding allocation is efficient and is returned, along with payments $\pi(R)$ according to the deployed payment rule (Line 15). If, by the end of the ML-powered rounds, the market has not cleared, we optionally allow bidders to submit push bids, analogously to the supplementary round of the CCA (Line 17) and calculate the optimal allocation $a^*(R)$ and the payments $\pi(R)$ (Lines 18 and 19). Note that ML-CCA can be combined with various possible payment rules $\pi(R)$, such as VCG or VCG-nearest.

6 Experiments

In this section, we experimentally evaluate the performance of our proposed ML-CCA from Algorithm 2.

6.1 Experiment Setup.

To generate synthetic CA instances, we use the GSVM, LSVM, SRVM, and MRVM domains from the spectrum auction test suite (SATS) (Weiss, Lubin, and Seuken 2017) (see Appendix D.1 for details). We compare our ML-CCA with the original CCA. For both mechanisms, we allow a maximum of 100 clock rounds per instance, i.e., we set $Q^{\text{max}} = 100$. For CCA, we set the price increment to 5% as in (Industry Canada 2013) and optimized the initial reserve prices to maximize its efficiency. For ML-CCA, we create Q^{init} price vectors to generate the initial demand query data using the same price update rule as the CCA, with the price increment adjusted to accommodate for the reduced number of rounds following this price update rule. In GSVM, LSVM and SRVM we set $Q^{\text{init}} = 20$ for ML-CCA, while in MRVM we set $Q^{\text{init}} = 50$. After each clock round, we report efficiency according to the clock bids up to that round, as well as efficiency if those clock bids were raised (see Section 2.2). Finally, we report the efficiency if the last clock round was supplemented with $Q^{\text{P-Max}} = 100$ bids using the profit max heuristic. Note that this is a very unrealistic and cognitively expensive bidding heuristic in practice, as it requires the agents to both discover their top 100 most profitable bundles as well as report their exact values for them, and thus only adds theoretical value to gauge the difficulty of each domain.

6.2 Hyperparameter Optimization (HPO).

We optimized the hyperparameters (HPs) of the mMVNNs for each bidder type of each domain. Specifically, for each bidder type we trained an mMVNN on the demand responses of a bidder of that type on 50 CCA clock rounds and selected the HPs that resulted in the highest R^2 on validation set 2 as described in Section 3.1. For more details please see Appendix D.3.

6.3 Results.

All efficiency results are presented in Table 1, while in Figure 2 we present the efficiency after each clock round, as well as the efficiency if those clock bids were enhanced with the clock bids raised heuristic (for 95% CIs and p-values see Appendix D.7).

In GSVM, ML-CCA's clock phase exhibits over 7.8% points higher efficiency compared to the CCA, while if we add the clock bids raised heuristic to both mechanisms, ML-CCA still exhibits over 5.3% points higher efficiency. At the same time, ML-CCA is able to find clearing prices in 56% of the instances, as opposed to only 3% for the CCA.

The results for LSVM are qualitatively very similar; ML-CCA's clock phase increases efficiency compared to the CCA by over 9% points, while clearing the market in 26% of the cases as opposed to 0%. If we add the clock bids raised heuristic to both mechanisms, ML-CCA still increases efficiency by over 4.7% points.

The SRVM domain, as suggested by the existence of only 3 unique goods, is quite easy to solve. Thus, both mecha-

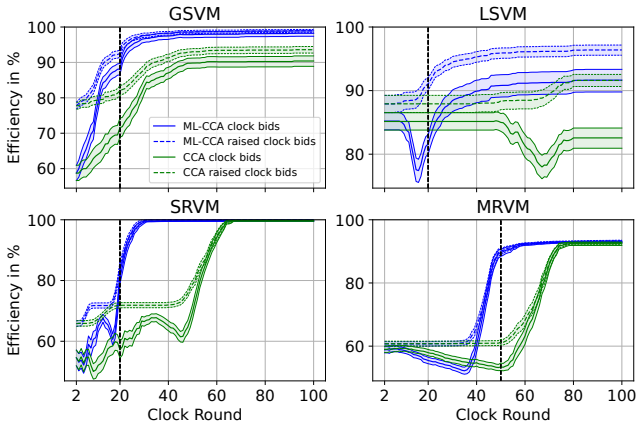

Figure 2: Efficiency path plots in SATS for ML-CCA and CCA both after clock bids (solid lines) and raised clock bids (dashed lines). Averaged over 100 runs including a 95% CI. The dashed black vertical line indicates the value of Q^{init}.

nisms can achieve almost 100% efficiency after their clock phase. For the clock bids raised heuristic our method reduces the efficiency loss by a factor of more than two (from 0.19% to 0.07%), and additionally, one can see from Figure 2 that our method reaches over 99% in less than 30 rounds.

In MRVM, ML-CCA again achieves statistically significantly better results for all 3 bidding heuristics. Notably, the CCA needs both the clock bids raised heuristic *and* 38 profit max bids to reach the same efficiency as our ML-CCA clock phase, i.e., it needs up to 138 additional *value* queries per bidder (see Appendix D.7). In MRVM, LCPs never exist, thus neither ML-CCA nor the CCA can ever clear the market.

To put our efficiency improvements in perspective, in the GSVM, LSVM and MRVM domains, ML-CCA's clock phase achieves higher efficiency than the CCA enhanced with the clock bids raised heuristic, i.e., the CCA, even if it uses up to an additional 100 value queries per bidder, cannot match the efficiency of our ML-powered clock phase. In Figure 2, we see that our ML-CCA can (almost) reach the efficiency numbers of Table 1 in a significantly reduced number of clock rounds compared to the CCA, while if we attempt to "speed up" the CCA, then its efficiency can substantially drop, see Appendix D.8. In particular, in GSVM and LSVM, using 50 clock rounds, our ML-CCA can achieve higher efficiency than the CCA can in 100 clock rounds.

6.4 Computational Efficiency.

For our choice of hyperparameters, the computation time when using 8 CPU cores[5] (see Appendix D.2 for details on our compute infrastructure) for a single round of the ML-CCA averages under 45 minutes for all domains tested. Notably, for three out of four domains, it averages less than 10 minutes (see Table 7 in Appendix D.7). The overwhelming majority of this time is devoted to training the mMVNNs of all bidders using our Algorithm 1 and generating the next DQ using our Algorithm 3 detailed in Section 4. It is important to note that both of these algorithms can always be parallelized by up to the number of bidders N, further reducing the time required. In our implementation, while we parallelized the training of the N mMVNNs, we did not do so for the generation of the next DQ. In spectrum auctions, typically no more than 2 rounds are conducted per day. For the results presented in this paper, we ran the full auction for 400 instances. Given the estimated welfare improvements of over 25 million USD per auction, attributed to ML-CCA's efficiency gains, we consider ML-CCA's computational and time requirements to be negligible.

[5]Note that Algorithm 1 is not GPU-implementable, as it requires solving a MIP in each iteration, for every demand response by an agent

7 Conclusion

We have proposed a novel method for training MVNNs to approximate the bidders' value functions based on demand query observations. Additionally, we have framed the task of determining the price vector with the highest clearing potential as minimization of an objective function that we prove is convex, Lipschitz-continuous, a.e. differentiable, and whose gradient for linear prices has an intuitive economic interpretation: change the price of every good proportionally to its predicted under/over-demand at the current prices. The resulting mechanism (ML-CCA) from combining these two components exhibits significantly higher clearing potential than the CCA and can increase efficiency by up to 9% points while at the same time converging in a much smaller number of rounds. Thus, we have designed the first *practical* ML-powered auction that employs the same interaction paradigm as the CCA, i.e., demand queries instead of cognitively too complex value queries, yet is able to significantly outperform the CCA in terms of both efficiency and clearing potential in realistic domains.

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).

References

Ausubel, L. M.; and Baranov, O. 2017. A practical guide to the combinatorial clock auction. *Economic Journal*, 127(605): F334–F350.

Ausubel, L. M.; Cramton, P.; and Milgrom, P. 2006. The clock-proxy auction: A practical combinatorial auction design. In Cramton, P.; Shoham, Y.; and Steinberg, R., eds., *Combinatorial Auctions*, 115–138. MIT Press.

Balcan, M.-F.; Sandholm, T.; and Vitercik, E. 2023. Generalization Guarantees for Multi-item Profit Maximization: Pricing, Auctions, and Randomized Mechanisms. arXiv:1705.00243.

Bichler, M.; Hao, Z.; and Adomavicius, G. 2017. *Coalition-based pricing in ascending combinatorial auctions*, 493–528. Cambridge University Press. ISBN 9781107135345.

Bikhchandani, S.; and Ostroy, J. M. 2002. The package assignment model. *Journal of Economic theory*, 107(2): 377–406.

Blum, A.; Jackson, J.; Sandholm, T.; and Zinkevich, M. 2004. Preference elicitation and query learning. *Journal of Machine Learning Research*, 5: 649–667.

Brero, G.; and Lahaie, S. 2018. A Bayesian clearing mechanism for combinatorial auctions. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*.

Brero, G.; Lahaie, S.; and Seuken, S. 2019. Fast Iterative Combinatorial Auctions via Bayesian Learning. In *Proceedings of the 33rd AAAI Conference of Artificial Intelligence*.

Brero, G.; Lubin, B.; and Seuken, S. 2018. Combinatorial Auctions via Machine Learning-based Preference Elicitation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*.

Brero, G.; Lubin, B.; and Seuken, S. 2021. Machine Learning-powered Iterative Combinatorial Auctions. *arXiv preprint arXiv:1911.08042*.

Cole, R.; and Roughgarden, T. 2014. The Sample Complexity of Revenue Maximization. In *Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing*, STOC '14, 243–252. New York, NY, USA: Association for Computing Machinery. ISBN 9781450327107.

Cramton, P. 2013. Spectrum auction design. *Review of Industrial Organization*, 42(2): 161–190.

Dütting, P.; Feng, Z.; Narasimhan, H.; Parkes, D. C.; and Ravindranath, S. S. 2019. Optimal auctions through deep learning. In *Proceedings of the 36th International Conference on Machine Learning*.

Dütting, P.; Fischer, F.; Jirapinyo, P.; Lai, J. K.; Lubin, B.; and Parkes, D. C. 2015. Payment rules through discriminant-based classifiers. *ACM Transactions on Economics and Computation*, 3(1): 5.

Goetzendorff, A.; Bichler, M.; Shabalin, P.; and Day, R. W. 2015. Compact Bid Languages and Core Pricing in Large Multi-item Auctions. *Management Science*, 61(7): 1684–1703.

Golowich, N.; Narasimhan, H.; and Parkes, D. C. 2018. Deep Learning for Multi-Facility Location Mechanism Design. In *Proceedings of the Twenty-seventh International Joint Conference on Artificial Intelligence and the Twenty-third European Conference on Artificial Intelligence*, 261–267.

Heiss, J. M.; Weissteiner, J.; Wutte, H. S.; Seuken, S.; and Teichmann, J. 2022. NOMU: Neural Optimization-based Model Uncertainty. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, 8708–8758. PMLR.

Industry Canada. 2013. "Responses to Clarification Questions on the Licensing Framework for Mobile Broadband Services (MBS) — 700 MHz Band".

Kwasnica, A. M.; Ledyard, J. O.; Porter, D.; and DeMartini, C. 2005. A New and Improved Design for Multiobject Iterative Auctions. *Management Science*, 51(3): 419–434.

Lahaie, S.; and Lubin, B. 2019. Adaptive-Price Combinatorial Auctions. In *Proceedings of the 2019 ACM Conference on Economics and Computation*, EC '19, 749–750. New York, NY, USA: Association for Computing Machinery. ISBN 9781450367929.

Lahaie, S. M.; and Parkes, D. C. 2004. Applying learning algorithms to preference elicitation. In *Proceedings of the 5th ACM Conference on Electronic Commerce*.

Milgrom, P.; and Segal, I. 2017. Designing the US incentive auction. *Handbook of spectrum auction design*, 803–812.

Morgenstern, J.; and Roughgarden, T. 2015. The Pseudo-Dimension of near-Optimal Auctions. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, 136–144. Cambridge, MA, USA: MIT Press.

Narasimhan, H.; Agarwal, S. B.; and Parkes, D. C. 2016. Automated mechanism design without money via machine learning. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*.

Nisan, N.; and Segal, I. 2006. The communication requirements of efficient allocations and supporting prices. *Journal of Economic Theory*, 129(1): 192–224.

Parkes, D. C.; and Ungar, L. H. 2000. Iterative combinatorial auctions: Theory and practice. In *Proceedings of the seventeenth national Conference on artificial intelligence*, 74–81.

Pogančić, M. V.; Paulus, A.; Musil, V.; Martius, G.; and Rolinek, M. 2020. Differentiation of blackbox combinatorial solvers. In *International Conference on Learning Representations*.

Rassenti, S. J.; Smith, V. L.; and Bulfin, R. L. 1982. A combinatorial auction mechanism for airport time slot allocation. *The Bell Journal of Economics*, 402–417.

Soumalias, E.; Zamanlooy, B.; Weissteiner, J.; and Seuken, S. 2023. Machine Learning-powered Course Allocation. *arXiv preprint arXiv:2210.00954*.

Weiss, M.; Lubin, B.; and Seuken, S. 2017. Sats: A universal spectrum auction test suite. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, 51–59.

Weissteiner, J.; Heiss, J.; Siems, J.; and Seuken, S. 2022a. Monotone-Value Neural Networks: Exploiting Preference Monotonicity in Combinatorial Assignment. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, 541–548. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Weissteiner, J.; Heiss, J.; Siems, J.; and Seuken, S. 2023. Bayesian Optimization-based Combinatorial Assignment. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37.

Weissteiner, J.; and Seuken, S. 2020. Deep Learning—Powered Iterative Combinatorial Auctions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(02): 2284–2293.

Weissteiner, J.; Wendler, C.; Seuken, S.; Lubin, B.; and Püschel, M. 2022b. Fourier Analysis-based Iterative Combinatorial Auctions. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, 549–556. International Joint Conferences on Artificial Intelligence Organization. Main Track.