Monotone-Value Neural Networks: Exploiting Preference Monotonicity in Combinatorial Assignment*

Jakob Weissteiner[1,3†], **Jakob Heiss**[2†], **Julien Siems**[1†] and **Sven Seuken**[1,3]
[1]University of Zurich
[2]ETH Zurich
[3]ETH AI Center
weissteiner@ifi.uzh.ch, jakob.heiss@math.ethz.ch, juliensiems@gmail.com, seuken@ifi.uzh.ch

Abstract

Many important resource allocation problems involve the *combinatorial assignment* of items, e.g., auctions or course allocation. Because the bundle space grows exponentially in the number of items, preference elicitation is a key challenge in these domains. Recently, researchers have proposed ML-based mechanisms that outperform traditional mechanisms while reducing preference elicitation costs for agents. However, one major shortcoming of the ML algorithms that were used is their disregard of important prior knowledge about agents' preferences. To address this, we introduce *monotone-value neural networks (MVNNs)*, which are designed to capture combinatorial valuations, while enforcing *monotonicity* and *normality*. On a technical level, we prove that our MVNNs are universal in the class of monotone and normalized value functions, and we provide a mixed-integer linear program (MILP) formulation to make solving MVNN-based winner determination problems (WDPs) practically feasible. We evaluate our MVNNs experimentally in spectrum auction domains. Our results show that MVNNs improve the prediction performance, they yield state-of-the-art allocative efficiency in the auction, and they also reduce the run-time of the WDPs. Our code is available on GitHub: https://github.com/marketdesignresearch/MVNN.

1 Introduction

Many important economic problems involve the *combinatorial assignment* of multiple indivisible items to multiple agents. In domains *with money*, prominent examples include *combinatorial auctions (CAs)* and *combinatorial exchanges (CEs)*. In CAs, heterogeneous items are allocated amongst a set of bidders, e.g. for the sale of spectrum licenses [Cramton, 2013]. In CEs, a set of items is allocated between multiple agents who can be sellers *and* buyers at the same time, e.g. for the reallocation of catch shares [Bichler *et al.*, 2019]. In domains *without money*, a popular example is *combinatorial course allocation*, where course seats are allocated to students at large business schools [Budish, 2011].

What all of these domains have in common is that the agents can report their values on *bundles* of items rather than only on individual items. This allows them to express more complex preferences, i.e., an agent's valuation of a bundle is not simply the sum of each individual item's value, but it can be more (*complementarity*) or less (*substitutability*). However, since the bundle space grows exponentially in the number of items, agents cannot report values for all bundles in settings with more than a modest number of items. Thus, parsimonious *preference elicitation* is key for the design of practical combinatorial assignment mechanisms.

In this paper, we present a new machine learning approach that exploits prior (structural) knowledge about agents' preferences and can be integrated well into iterative market mechanisms. Our contribution applies to any combinatorial assignment problem. However, we present our algorithms in the context of a CA specifically, to simplify the notation and because there exist well-studied preference generators for CAs that we can use for our experimental evaluation.

For CAs with general valuations, Nisan and Segal [2006] have shown that exponential communication in the number of items is needed in the worst case to find an optimal allocation of items to bidders, i.e., to ensure full efficiency. Thus, for general valuations, practical CAs cannot provide efficiency guarantees in large domains. In practice, *iterative combinatorial auctions (ICAs)* are employed, where the auctioneer interacts with bidders over multiple rounds, eliciting a *limited* amount of information, aiming to find a highly efficient allocation. ICAs are widely used; e.g., for the sale of licenses to build offshore wind farms [Ausubel and Cramton, 2011]. The provision of spectrum licenses via the *combinatorial clock auction (CCA) [Ausubel* et al.*, 2006]* has generated more than $20 billion in total revenue [Ausubel and Baranov, 2017]. Therefore, increasing the efficiency of such real-world ICAs by only 1% point translates into monetary gains of hundreds of millions of dollars.

1.1 ML-based Auction Design

In recent years, researchers have successfully integrated machine learning (ML) algorithms into the design of CAs to im-

*Full paper including appendix available at arXiv: https://arxiv.org/abs/2109.15117.

†These authors contributed equally to this paper.

prove their performance. Dütting *et al.* [2019] and Rahme *et al.* [2021] used neural networks (NNs) to learn entire auction mechanisms from data, following the automated mechanism design paradigm. Brero *et al.* [2019] studied a Bayesian ICA using probabilistic price updates to achieve faster convergence to an efficient allocation.

Most related to this paper is the work by Brero *et al.* [2018; 2021], who developed a value-query-based ICA that achieves even higher efficiency than the widely used CCA. In follow-up work, Weissteiner and Seuken [2020] extended their work by integrating Neural Networks (NN) in their mechanisms and could further increase the efficiency. Finally, Weissteiner *et al.* [2022] used Fourier transforms (FTs) to leverage different notions of sparsity of value functions in preference elicitation. However, despite these advances, it remains a challenging problem to find the efficient allocation while keeping the elicitation cost for bidders low. Even state-of-the-art approaches suffer from significant efficiency losses, highlighting the need for better preference elicitation algorithms.

We show in this paper that these limitations can be partially explained due to the usage of poor, non-informative ML-algorithms, which either do not include important prior domain knowledge or make too restrictive assumptions about the bidders' value functions. Brero *et al.* [2018; 2021] used support vector regressions (SVRs) with quadratic kernels which can only learn up to two way interactions between items and do not account for an important monotonicity property of bidders' value functions. While the fully-connected feed-forward NNs used by Weissteiner and Seuken [2020] are more expressive, they also do not account for this monotonicity property. In particular when operating with only a small number of data points (which is the standard in market mechanisms, because preference elicitation is costly), this can cause significant efficiency losses.

Over the last decade, major successes in ML were made by specialized NN architectures (e.g. Convolutional Neural Networks) that incorporate domain-specific prior knowledge to improve generalization [Bronstein *et al.*, 2017]. With this paper, we follow the same paradigm by incorporating prior knowledge about monotone preferences into a NN architecture to improve generalization, which is key for a well-functioning preference elicitation algorithm.

Several other approaches for incorporating monotonicity into NNs have previously been proposed. However, for these architectures, it is not known how the NN-based winner determination problem (WDP) could be solved quickly or they have other limitations. Sill [1998] proposes only a shallow architecture which violates the normalization property. You *et al.* [2017] proposes a non-standard architecture, where it is unknown if the corresponding MILP formulation of the WDP is computationally feasible. Wehenkel and Louppe [2019] implement monotonicity by representing the target function as an integral of a NN and thus the WDP would result in a computationally infeasible MILP. Liu *et al.* [2020] train NNs with successively higher regularization until a MILP based verification procedure guarantees monotonicity. The repeated retraining and verification lead to high computational cost.

In contrast, our approach is particularly well suited for combinatorial assignment, because (i) our NN-based WDP can be formulated as a succinct MILP and thus solved quickly and (ii) we propose a generic fully-connected feed-forward architecture with arbitrary number of hidden layers which can be trained efficiently.

1.2 Our Contribution

We make the following contributions:

1. We introduce *monotone-value neural networks (MVNNs)*, a new class of NNs with a carefully chosen activation function (bReLU) and enforced constraints on the parameters such that they are normalized and fulfill a monotonicity property (Section 3). These MVNNs are specifically suited to model monotone (combinatorial) value functions in economic settings.
2. On a technical level, we provide the following two theorems (Section 3.1): First, we prove that MVNNs are universal in the class of monotone and normalized combinatorial value functions, i.e., one can represent *any* value function with arbitrarily complex substitutabilities and complementarities exactly as an MVNN. Second, we show how to formulate the MVNN-based WDP as a MILP, which is key to efficiently calculate optimal allocations.
3. We experimentally evaluate the learning performance of MVNNs vs. NNs in four different spectrum CA domains and show that MVNNs are significantly better at modelling bidders' combinatorial value functions (Section 4).
4. Finally, we experimentally investigate the performance of MVNNs vs. plain NNs when integrated into an existing ML-based ICA (MLCA) and compare them also to the FT-based method by Weissteiner *et al.* [2022]. We show that MVNNs lead to substantially smaller efficiency losses than existing state-of-the-art mechanisms (Section 5).

2 Preliminaries

In this section, we present our formal model and review the ML-based ICA by Brero *et al.* [2021].

2.1 Formal Model for ICAs

We consider a CA with n bidders and m indivisible items. Let $N = \{1, \dots, n\}$ and $M = \{1, \dots, m\}$ denote the set of bidders and items, respectively. We denote with $x \in \mathcal{X} = \{0,1\}^m$ a bundle of items represented as an indicator vector, where $x_j = 1$ iff item $j \in M$ is contained in x. Bidders' true preferences over bundles are represented by their (private) value functions $v_i : \mathcal{X} \to \mathbb{R}_+,\ i \in N$, i.e., $v_i(x)$ represents bidder i's true value for bundle x.

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 $\mathcal{F} = \left\{a \in \mathcal{X}^n : \sum_{i \in N} a_{ij} \leq 1,\ \forall j \in M\right\}$. We assume that bidders' have quasilinear utilities u_i, i.e., for payments $p \in \mathbb{R}^n_+$, $u_i(a,p) = v_i(a_i) - p_i$. This implies that the (true) *social welfare* $V(a)$ of an allocation a is equal to the sum of all bidders' values, i.e., $V(a) = \sum_{i \in N} u_i(a_i, p_i) + u_{\text{auctioneer}}(p) = \sum_{i \in N} v_i(a_i) - p_i + \sum_{i \in N} p_i = \sum_{i \in N} v_i(a_i)$. We let $a^* \in \operatorname{argmax}_{a \in \mathcal{F}} V(a)$ be a social-welfare maximizing, i.e., *efficient*, allocation. The *efficiency* of any allocation $a \in \mathcal{F}$ is measured as $V(a)/V(a^*)$.

An ICA *mechanism* defines how the bidders interact with the auctioneer and how the final allocation and payments are determined. We denote a bidder's (possibly untruthful) reported value function by $\hat{v}_i : \mathcal{X} \to \mathbb{R}_+$. In this paper, we consider ICAs that ask bidders to iteratively report their values $\hat{v}_i(x)$ for particular bundles x selected by the mechanism. A set of $L \in \mathbb{N}$ reported bundle-value pairs of bidder i is denoted as $R_i = \left\{\left(x^{(l)}, \hat{v}_i(x^{(l)})\right)\right\}_{l=1}^{L}$, $x^{(l)} \in \mathcal{X}$. Let $R = (R_1, \ldots, R_n)$ denote the tuple of reported bundle-value pairs obtained from all bidders. We define the *reported social welfare* of an allocation a given R as $\widehat{V}(a|R) := \sum_{i \in N:\,(a_i, \hat{v}_i(a_i)) \in R_i} \hat{v}_i(a_i)$, where $(a_i, \hat{v}_i(a_i)) \in R_i$ ensures that only values for reported bundles contribute. The final optimal allocation $a^*_R \in \mathcal{F}$ and payments $p(R) \in \mathbb{R}^n_+$ are computed based on the elicited reports R *only*. More formally, the optimal allocation $a^*_R \in \mathcal{F}$ given the reports R is defined as

$$a^*_R \in \operatorname*{argmax}_{a \in \mathcal{F}} \widehat{V}(a|R). \tag{1}$$

As the auctioneer can generally only query each bidder i a limited number of bundles $|R_i| \leq Q^{\max}$ (e.g., $Q^{\max} = 100$), the mechanism needs a sophisticated preference elicitation algorithm, with the goal of finding a highly efficient final allocation a^*_R with a limited number of value queries.

2.2 A Machine Learning-powered ICA

We now review the *machine learning-powered combinatorial auction (MLCA)* by Brero *et al.* [2021]. Interested readers are referred to Appendix A, where we present MLCA in detail.

MLCA proceeds in rounds until a maximum number of value queries per bidder $Q^{\max}$ is reached. In each round, a generic ML algorithm $\mathcal{A}_i$ is trained for every bidder $i \in N$ on the bidder's reports R_i. Next, MLCA generates new value queries $q^{\text{new}} = (q_i^{\text{new}})_{i=1}^n$ with $q_i^{\text{new}} \in \mathcal{X} \setminus R_i$ by solving a ML-based WDP $q^{\text{new}} \in \operatorname*{argmax}_{a \in \mathcal{F}} \sum_{i \in N} \mathcal{A}_i(a_i)$.

The idea is the following: if $\mathcal{A}_i$ are good surrogate models of the bidders' true value functions then q^{new} should be a good proxy of the efficient allocation a^* and thus provide the auctioneer with valuable information. Additionally, in a real-world MLCA, bidders' are always allowed to report values of further bundles that they deem potentially useful ("push"-bids). This mitigates the risk of bidders not getting asked the right queries. In our experiments, MLCA achieves already state-of-the-art results without making use of any "push"-bids (mathematically additional "push"-bids can only improve the results further).

At the end of each round, MLCA receives reports R^{new} from all bidders for the newly generated queries q^{new}, and updates the overall elicited reports R. When $Q^{\max}$ is reached, MLCA computes an allocation a^*_R that maximizes the *reported* social welfare (see Equation (1)) and determines VCG payments $p(R)$ based on the reported values (see Appendix Definition B.1).

Remark 1 (IR, No-Deficit, and Incentives of MLCA)**.** *Brero* et al. *[2021] showed that MLCA satisfies* individual rationality (IR) *and* no-deficit*, with any ML algorithm $\mathcal{A}_i$. Furthermore, they studied the incentive properties of MLCA; this is important, given that opportunities for manipulations might lower efficiency. Like all deployed spectrum auctions (including the CCA [Ausubel and Baranov, 2017]) MLCA is not strategyproof. However, Brero* et al. *[2021] argued that it has good incentives in practice; and given two additional assumptions, bidding truthfully is an ex-post Nash equilibrium in MLCA. Their analyses apply to MLCA using any ML algorithm, and therefore also to an MVNN-based MLCA. We present a more detailed summary of their incentive analysis in Appendix B.*

3 Monotone-Value Neural Networks

In combinatorial assignment problems, value functions are used to model each agent's (reported) value for a bundle of items ($\hat{v}_i : \{0,1\}^m \to \mathbb{R}_+$). However, while the bundle space grows exponentially with the number of items, the agents' value functions often exhibit useful structure that can be exploited. A common assumption is monotonicity:

(M) Monotonicity (*"additional items increase value"*):
For $A, B \in 2^M$: if $A \subseteq B$ it holds that $\hat{v}_i(A) \leq \hat{v}_i(B)$.[1]

This property is satisfied in many market domains. For example, in many CAs, bidders can freely dispose of unwanted items; in combinatorial course allocation, students can just drop courses they have been assigned. However, prior work on ML-based market design [Weissteiner and Seuken, 2020; Weissteiner *et al.*, 2022; Brero *et al.*, 2021] has not taken this property into account, which negatively affects the performance (see Sections 4 and 5).

For ease of exposition, we additionally assume that the value functions are normalized:

(N) Normalization (*"no value for empty bundle"*):
$\hat{v}_i(\emptyset) = \hat{v}_i((0, \ldots, 0)) := 0$

Note that this property is not required by our method and can be easily adapted to be any other fixed value, or to be a learned parameter. In the following, we denote with

$$\mathcal{V} := \{\hat{v}_i : \mathcal{X} \to \mathbb{R}_+ | \text{ satisfy } \textbf{(N)} \text{ and } \textbf{(M)}\} \tag{2}$$

the set of all value functions, that satisfy the *normalization* and *monotonicity* property. Next, we introduce *Monotone-Value Neural Networks (MVNNs)* and show that they span the entire set $\mathcal{V}$. Thus, MVNNs are specifically suited to all applications with monotone value functions.

Definition 1 (MVNN)**.** *A MVNN $\mathcal{N}_i^\theta : \mathcal{X} \to \mathbb{R}_+$ for bidder $i \in N$ is defined as*

$$\mathcal{N}_i^\theta(x) = W^{i,K_i} \varphi_{0,t}\left(\ldots \varphi_{0,t}(W^{i,1}x + b^{i,1}) \ldots\right), \tag{3}$$

- *$K_i \in \mathbb{N}$ is the number of layers ($K_i - 2$ hidden layers),*
- *$\varphi_{0,t}$ is our MVNN-specific activation function with cutoff $t > 0$, which we call* bounded ReLU (bReLU)*.*[2]

$$\varphi_{0,t}(z) := \min(t, \max(0, z)) \tag{4}$$

[1] We slightly abused the notation here by using sets instead of their corresponding indicator vectors as arguments of $\hat{v}_i$.

[2] bReLUs have also been used in visual pattern recognition problems to enhance training stability (see Liew *et al.* [2016]).

- $W^i := (W^{i,k})_{k=1}^{K_i}$ *with* $W^{i,k} \geq 0$ *and* $b^i := (b^{i,k})_{k=1}^{K_i}$ *with* $b^{i,k} \leq 0$ *denote a tuple of 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* $\theta = (W^i, b^i)$.[3]

For implementation details of MVNNs we refer to Appendix C.6. Unless explicitly stated, we consider from now on a cutoff of $t = 1$ for the bReLU, i.e., $\varphi(z) := \varphi_{0,1}(z) = \min(1, \max(0, z))$. Next, we discuss the choice of bReLU.

Choice of bReLU (i) The constraints on the weights and biases enforce monotonicity of MVNNs (in fact for *any* monotone activation). (ii) For universality (see Theorem 1) we need a *bounded* monotone non-constant activation, i.e., with ReLUs and our constraints one cannot express substitutabilities. (iii) for the MILP (see Theorem 2), we need a *piecewise linear* activation (e.g., with sigmoids we could not formulate a MILP). Taking all together, bReLU is the simplest bounded, monotone, non-constant, piecewise-linear activation (see Appendix Remarks C.2 and C.3 for a detailed discussion).

Remark 2. *For applications where value functions are expected to be "almost" (but not completely) monotone, one can adapt MVNNs to only have* soft monotonicity constraints *by implementing the constraints on the weights and biases via regularization, e.g.,* $\sum_{i,k,j,l} \max(0, -W_{j,l}^{i,k})$*. This results in soft-MVNNs that can model non-monotone changes in some items if the data evidence is strong.*

3.1 Theoretical Analysis and MILP-Formulation

Next, we provide a theoretical analysis of MVNNs and present a MILP formulation of the MVNN-based WDP.

Lemma 1. *Let* $\mathcal{N}_i^\theta : \mathcal{X} \to \mathbb{R}_+$ *be a MVNN from Definition 1. Then it holds that* $\mathcal{N}_i^{(W^i,b^i)} \in \mathcal{V}$ *for all* $W^i \geq 0$ *and* $b^i \leq 0$*.*

We provide the proof for Lemma 1 in Appendix C.1. Next, we state our main theorem about MVNNs.

Theorem 1 (Universality)**.** Any *value function* $\hat{v}_i : \mathcal{X} \to \mathbb{R}_+$ *that satisfies* ***(N)*** *and* ***(M)*** *can be represented exactly as a MVNN* $\mathcal{N}_i^\theta$ *from Definition 1, i.e.,*

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

We present a constructive proof for Theorem 1 in Appendix C.3. In the proof, we consider an arbitrary $(\hat{v}_i(x))_{x \in \mathcal{X}} \in \mathcal{V}$ for which we construct a two hidden layer MVNN $\mathcal{N}_i^\theta$ of dimensions $[m, 2^m - 1, 2^m - 1, 1]$ with parameters $\theta = (W_{\hat{v}_i}^i, b_{\hat{v}_i}^i)$ such that $\mathcal{N}_i^\theta(x) = \hat{v}_i(x) \, \forall x \in \mathcal{X}$.

Note that for q queries, it is always possible to construct MVNN of size $[m, q, q, 1]$ that perfectly fits through the q data-points (see Appendix Corollary C.1). Thus, the worst-case required architecture size grows only linearly with the number of queries. In our experiments we already achieve very good performance with even smaller architectures. Example C.1 in the Appendix nicely illustrates how exactly MVNNs capture complementarities, substitutabilities and independent items.

A key step in combinatorial assignment mechanisms is finding the social welfare-maximizing allocation, i.e., solving the *Winner Determination Problem* (WDP). To use MVNNs in such mechanisms, we need to be able to efficiently solve MVNN-based WDPs. To this end, we present a MILP formulation of the MVNN-based WDP. The key idea is to rewrite the bReLU $\varphi(z)$ as $-\max(-1, -\max(0, z))$ and encode for each bidder i, hidden layer k and neuron j both $\max(\cdot, \cdot)$ operators with two binary decision variables $y_j^{i,k}, \mu_j^{i,k}$. First, we show how to encode one *single* hidden layer of an MVNN as multiple linear constraints. We provide the proof in Appendix C.4.

Lemma 2. *Fix bidder* $i \in N$*, let* $k \in \{2, \dots, K_i - 1\}$ *and denote the pre-activated output of the* k^{th} *hidden layer as* $o^{i,k} := W^{i,k-1} z^{i,k-1} + b^{i,k-1}$ *with* $W^{i,k-1} \in \mathbb{R}^{d_k^i \times d_{k-1}^i}, b^{i,k-1} \in \mathbb{R}^{d_k^i}$*. Then the output of the* k^{th} *hidden layer* $z^{i,k} := \varphi(o^{i,k}) = \min(1, \max(0, o^{i,k})) = -\max(-1, -\eta^{i,k})$*, with* $\eta^{i,k} := \max(0, o^{i,k})$ *can be equivalently expressed by the following linear constraints:*

$$o^{i,k} \leq \eta^{i,k} \leq o^{i,k} + y^{i,k} \cdot L_1^{i,k} \tag{6}$$

$$0 \leq \eta^{i,k} \leq (1 - y^{i,k}) \cdot L_2^{i,k} \tag{7}$$

$$\eta^{i,k} - \mu^{i,k} \cdot L_3^{i,k} \leq z^{i,k} \leq \eta^{i,k} \tag{8}$$

$$1 - (1 - \mu^{i,k}) \cdot L_4^{i,k} \leq z^{i,k} \leq 1 \tag{9}$$

$$y^{i,k} \in \{0,1\}^{d_k^i}, \quad \mu^{i,k} \in \{0,1\}^{d_k^i}, \tag{10}$$

where $L_1^{i,k}, L_2^{i,k}, L_3^{i,k}, L_4^{i,k} \in \mathbb{R}_+$ *are large enough constants for the respective* big-M *constraints.*[4]

Finally, we formulate the MVNN-based WDP as a MILP.

Theorem 2 (MILP)**.** *The MVNN-based WDP* $\max_{a \in \mathcal{F}} \sum_{i \in N} \mathcal{N}_i^{(W^i,b^i)}(a_i)$ *can be equivalently formulated as the following MILP:*[5]

$$\max_{a \in \mathcal{F}, z^{i,k}, \mu^{i,k}, \eta^{i,k}, y^{i,k}} \left\{ \sum_{i \in N} W^{i,K_i} z^{i,K_i - 1} \right\} \tag{11}$$

s.t. for $i \in N$ *and* $k \in \{2, \dots, K_i - 1\}$

$$z^{i,1} = a_i \tag{12}$$

$$W^{i,k-1} z^{i,k-1} + b^{i,k-1} \leq \eta^{i,k} \tag{13}$$

$$\eta^{i,k} \leq W^{i,k-1} z^{i,k-1} + b^{i,k-1} + y^{i,k} \cdot L_1^{i,k} \tag{14}$$

$$0 \leq \eta^{i,k} \leq (1 - y^{i,k}) \cdot L_2^{i,k} \tag{15}$$

$$\eta^{i,k} - \mu^{i,k} \cdot L_3^{i,k} \leq z^{i,k} \leq \eta^{i,k} \tag{16}$$

$$1 - (1 - \mu^{i,k}) \cdot L_4^{i,k} \leq z^{i,k} \leq 1 \tag{17}$$

$$y^{i,k} \in \{0,1\}^{d_k^i}, \qquad \mu^{i,k} \in \{0,1\}^{d_k^i} \tag{18}$$

[3] We apply a linear readout map to the last hidden layer, i.e, no $\varphi_{0,t}$ and $b^{i,K_i} := 0$. By setting $b^{i,K_i} \neq 0$ with *trainable=False*, the MVNN can model any other value than zero in the normalization property. By not restricting $b^{i,k} \leq 0$ and setting $b^{i,K_i} \neq 0$ with *trainable=True* one can also learn the value for the empty bundle.

[4] To account for a general cutoff $t \neq 1$ in the bReLU, one needs to adjust (9) by replacing the left- and rightmost 1 with t.

[5] To account for a general cutoff $t \neq 1$ in the bReLU, one needs to adjust (17) by replacing the left- and rightmost 1 with t.

Proof. The proof follows by iteratively applying Lemma 2 for each bidder and all her respective hidden MVNN layers. □

Fact 1. *One can significantly reduce the solve time for the MILP by tightening the bounds of each neuron. In Appendix C.5, we present bound tightening via* interval arithmetic (IA) *[Tjeng* et al.*, 2019] for MVNNs. For a plain ReLU NN, these IA bounds are not tight and calculating tighter bounds in a computationally efficient manner is very challenging. In contrast, the MVNN-IA bounds are always* perfectly tight*, because of their encoded monotonicity property. The upper and lower bound of a neuron is the value the neuron outputs for the input* $(1, \dots, 1)$ *and* $(0, \dots, 0)$*. This is a big advantage of MVNNs compared to plain (ReLU) NNs.*

Remark 3 (MVNNs as Preference Generator)**.** *To experimentally evaluate a new mechanism one needs to generate many different possible value functions. Preference generators like the Spectrum Auction Test Suite [Weiss* et al.*, 2017] are carefully designed to capture the essential properties of spectrum auctions, but they are not available for every application. Instead of using such a domain-specific preference generator, one can also use MVNNs with randomly initialized weights to generate possible value functions. An advantage of random MVNNs is that they are universal (see Theorem 1) and hence come with a diversity rich enough to sample any possible monotone value function with arbitrarily complex substitutabilities and complementarities (the distribution of supplements and complements can be controlled via the cutoff* t*, where the smaller/larger* t *the more substitutabilities/complementarities). These types of generative test beds become increasingly important to avoid overfitting on specific simulation engines and/or real data sets [Osband* et al.*, 2021].*

4 Prediction Performance of MVNNs

In this section, we show that in all considered CA domains, MVNNs are significantly better at capturing bidders' complex value functions than plain (ReLU) NNs, which allows them to extrapolate much better in the bundle space.

4.1 Experimental Setup - Prediction Performance

CA Domains In our experiments we use simulated data from the Spectrum Auction Test Suite (SATS) version 0.7.0 [Weiss *et al.*, 2017]. We consider the following four domains:

- **Global Synergy Value Model (GSVM)** [Goeree and Holt, 2010] has 18 items, 6 *regional* and 1 *national bidder*.
- **Local Synergy Value Model (LSVM)** [Scheffel *et al.*, 2012] has 18 items, 5 *regional* and 1 *national bidder*. Complementarities arise from spatial proximity of items.
- **Single-Region Value Model (SRVM)** [Weiss *et al.*, 2017] has 29 items and 7 bidders (categorized as *local*, *high frequency regional*, or *national*) and models large UK 4G spectrum auctions.
- **Multi-Region Value Model (MRVM)** [Weiss *et al.*, 2017] has 98 items and 10 bidders (*local*, *regional*, or *national*) and models large Canadian 4G spectrum auctions.

When simulating bidders, we follow prior work (e.g., [Brero *et al.*, 2021]) and assume truthful bidding (i.e., $\hat{v}_i = v_i$). Details on how we collect the data and the train/val/test split can be found in Appendix D.1.

HPO To efficiently optimize the hyperparameters and fairly compare MVNNs and plain NNs for best generalization across different instances of each SATS domain, we frame the *hyperparameter optimization (HPO)* problem as an algorithm configuration problem and use the well-established *sequential model-based algorithm configuration (SMAC)* [Hutter *et al.*, 2011]. SMAC quickly discards hyperparameters which already perform poorly on a few SATS instances and proposes more promising ones via Bayesian optimization. It is flexible enough for the parameterization of NNs as it naturally handles a mixture of categorical, integer and float hyperparameters. Further details on the setting including hyperparameter ranges can be found in Appendix D.2.

4.2 Prediction Performance Results

For ease of exposition, we only present our results for the MVNN-RELU-PROJECTED implementation of our MVNNs (termed MVNN in the following). Results for other MVNN implementations can be found in the Appendix D.3. In Table 1, we compare the prediction performance of the winning models that the HPO found for different amounts of training data (T) on the test data. We see that, compared to plain

			$R^2 \uparrow$		KT ↑	
DOMAIN	T	BIDDER	MVNN	NN	MVNN	NN
GSVM	10	NAT	0.686 ±0.061	0.534 ±0.040	0.668 ±0.027	0.583 ±0.021
		REG	0.618 ±0.068	0.504 ±0.062	0.633 ±0.038	0.557 ±0.033
	20	NAT	0.923 ±0.016	0.818 ±0.032	0.849 ±0.017	0.752 ±0.029
		REG	0.940 ±0.018	0.880 ±0.022	0.882 ±0.020	0.815 ±0.021
	50	NAT	0.992 ±0.001	0.988 ±0.001	0.962 ±0.003	0.953 ±0.003
		REG	0.997 ±0.001	0.988 ±0.001	0.974 ±0.002	0.953 ±0.003
LSVM	10	NAT	0.248 ±0.069	0.137 ±0.031	0.693 ±0.011	0.710 ±0.023
		REG	0.563 ±0.049	0.348 ±0.067	0.605 ±0.031	0.504 ±0.025
	50	NAT	0.616 ±0.020	0.199 ±0.031	0.753 ±0.009	0.678 ±0.035
		REG	0.921 ±0.015	0.872 ±0.012	0.860 ±0.017	0.812 ±0.013
	100	NAT	0.677 ±0.014	0.396 ±0.033	0.813 ±0.005	0.706 ±0.018
		REG	0.965 ±0.010	0.936 ±0.010	0.918 ±0.015	0.857 ±0.012
SRVM	10	H.F.	0.538 ±0.044	-2.123 ±0.268	0.626 ±0.020	0.607 ±0.012
		LO	0.381 ±0.045	0.267 ±0.042	0.559 ±0.030	0.489 ±0.032
		NAT	0.389 ±0.063	0.341 ±0.038	0.560 ±0.026	0.535 ±0.012
		REG	0.422 ±0.051	0.372 ±0.036	0.562 ±0.023	0.544 ±0.014
	50	H.F.	0.860 ±0.015	0.773 ±0.034	0.853 ±0.013	0.803 ±0.020
		LO	0.895 ±0.020	0.588 ±0.031	0.902 ±0.000	0.771 ±0.030
		NAT	0.988 ±0.004	0.828 ±0.015	0.918 ±0.005	0.801 ±0.009
		REG	0.989 ±0.004	0.872 ±0.047	0.931 ±0.004	0.823 ±0.022
	100	H.F.	0.911 ±0.008	0.849 ±0.011	0.908 ±0.006	0.896 ±0.006
		LO	0.948 ±0.014	0.723 ±0.005	0.903 ±0.000	0.900 ±0.002
		NAT	0.998 ±0.000	0.913 ±0.008	0.952 ±0.003	0.841 ±0.008
		REG	0.996 ±0.001	0.945 ±0.004	0.948 ±0.021	0.895 ±0.012
MRVM	10	LO	-0.055 ±0.058	-0.018 ±0.050	0.262 ±0.017	0.200 ±0.015
		NAT	0.182 ±0.045	-0.556 ±1.355	0.365 ±0.018	0.414 ±0.008
		REG	0.036 ±0.085	-0.048 ±0.092	0.322 ±0.022	0.255 ±0.038
	100	LOCAL	0.831 ±0.023	0.493 ±0.027	0.786 ±0.019	0.545 ±0.012
		NAT	0.778 ±0.022	0.560 ±0.019	0.726 ±0.014	0.581 ±0.010
		REG	0.832 ±0.028	0.447 ±0.053	0.779 ±0.027	0.572 ±0.018
	300	LOCAL	0.944 ±0.006	0.871 ±0.009	0.883 ±0.010	0.819 ±0.009
		NAT	0.868 ±0.016	0.855 ±0.029	0.814 ±0.009	0.808 ±0.013
		REG	0.917 ±0.017	0.847 ±0.010	0.851 ±0.019	0.809 ±0.012

Table 1: Prediction performance measured via R-squared (R^2) and Kendall tau (KT) with a 95%-CI in four SATS domains with corresponding bidder types: high frequency (H.F.), local (LO), regional (REG) and national (NAT), averaged over 30 auction instances. Both MVNNs and plain NNs are trained on T and evaluated on $209\,715 - T$ random bundles. Winners are marked in grey.

Domain	Q^{MAX}	Efficiency Loss in % ↓ MVNN	NN	FT	RS	T-Test for Efficiency: $\mathcal{H}_0 : \mu_{\text{NN}} \leq \mu_{\text{MVNN}}$	$\mathcal{H}_0 : \mu_{\text{FT}} \leq \mu_{\text{MVNN}}$
GSVM	100	00.00 ± 0.00	00.00 ± 0.00	01.77± 0.96	30.34 ± 1.61		$p_{\text{VAL}} = 3\text{e}{-6}$
LSVM	100	00.70 ± 0.40	02.91 ± 1.44	01.54± 0.65	31.73 ± 2.15	$p_{\text{VAL}} = 2\text{e}{-03}$	$p_{\text{VAL}} = 5\text{e}{-3}$
SRVM	100	00.23 ± 0.06	01.13 ± 0.22	00.72± 0.16	28.56 ± 1.74	$p_{\text{VAL}} = 5\text{e}{-10}$	$p_{\text{VAL}} = 2\text{e}{-8}$
MRVM	100	08.16 ± 0.41	09.05 ± 0.53	10.37± 0.57	48.79 ± 1.13	$p_{\text{VAL}} = 9\text{e}{-03}$	$p_{\text{VAL}} = 1\text{e}{-7}$

Table 2: Efficiency loss of MVNN vs plain NNs, the Fourier Transform (FT) benchmark and random search (RS). Shown are averages and a 95% CI on a test set of 50 CA instances. Winners based on a (paired) t-test with significance level of 1% are marked in grey.

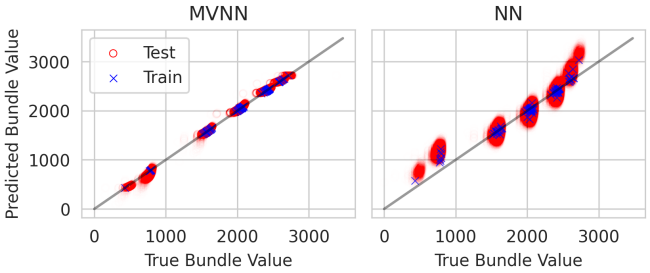

Figure 1: Prediction performance of MVNNs vs plain NNs in SRVM (national bidder). The identity is shown in grey.

NNs, MVNNs provide both a significantly better fit in terms of R-squared $\boldsymbol{R^2}$ as well as a better Kendall Tau rank correlation **KT** (i.e., a better ordinal ranking). Thus, enforcing the monotonicity property in MVNNs significantly improves the learning performance.

Figure 1 illustrates our findings by providing a visual comparison of the prediction performance for the highly non-unimodal SRVM.[6] We see that the MVNN fits the training data exactly (blue crosses), although the HPO only optimized generalization performance on the validation data. This is a strong indication that MVNNs correspond to a more realistic prior, since for a realistic prior, it is optimal to exactly fit the training data in noiseless settings [Heiss *et al.*, 2021, Proposition D.2.a]. In contrast, the HPO has selected hyperparameters for the plain NNs that result in a worse fit of the training data (otherwise generalization to unseen data would be even worse). Moreover, the plain NNs show a particularly bad fit on the less frequent lower and higher valued bundles.

5 MVNN-powered Iterative CA

To evaluate the performance of MVNNs when used inside a combinatorial market mechanism, we have integrated MVNNs into MLCA (see Section 2.2), yielding an *MVNN-powered Iterative CA*. In this section, we compare the economic efficiency of our MVNN-based MLCA against the previously proposed NN-based MLCA. For solving the MVNN-based WDPs in MLCA, we use our MILP from Theorem 2.

5.1 Experimental Setup - MLCA

To generate synthetic CA instances, we use the same four SATS domains as in Section 4. SATS also gives us access to the true optimal allocation a^*, which we use to measure the *efficiency loss*, i.e., $1 - V(a^*_R)/V(a^*)$ and *relative revenue* $\sum_{i \in N} p(R)_i / V(a^*)$ of an allocation $a^*_R \in \mathcal{F}$ and payments $p(R) \in \mathbb{R}^n_+$ determined by MLCA when eliciting reports R. Due to the long run-time of a single evaluation of MLCA , we perform a restricted HPO, which, for each domain, only uses the winners of the prediction performance experiment (Table 1) for the three amounts of training data T. This is a reasonable choice, because in the prediction performance we optimize the generalization performance for bidders' value functions that is also key in MLCA.

For each domain, we use $Q^{\text{init}} = 40$ initial random queries and set the query budget to $Q^{\text{max}} = 100$. We terminate MLCA in an intermediate iteration if it already found an efficient allocation (i.e., with 0 efficiency loss).

5.2 Efficiency Results

In Table 2, we present the efficiency results of MVNN-based MLCA and NN-based MLCA, averaged over 50 auction instances. We focus on efficiency rather than revenue, since spectrum auctions are government-run auctions with a mandate to maximize efficiency and not revenue [Cramton, 2013]. In Appendices E.2 and E.3 we also present and discuss detailed revenue results.

For each domain, we present results corresponding to the best MVNNs and NNs amongst the three incumbents obtained from the prediction performance experiments. We present results for all three incumbents in Appendix E.2. Overall, we see that the better prediction performance of MVNNs (Table 1) translates to smaller efficiency losses in MLCA. In LSVM and SRVM, MVNNs significantly outperform NNs and have a more than four times lower efficiency loss. In MRVM, MVNN's average efficiency loss is approximately 1% point smaller than the NN's loss. Given that in the 2014 Canadian 4G auction the total revenue was on the order of 5 billion USD [Ausubel and Baranov, 2017], an efficiency loss decrease of 1% point in MRVM can translate to welfare gains on the order of 50 million USD. Finally, in GSVM, the simplest domain, where bidders' value functions have at most two-way interactions between items, both MVNNs and plain NNs incur no efficiency loss. As further baselines, we evaluate the Fourier transform (FT) auction [Weissteiner *et al.*, 2022] using their proposed optimal hyperparameters and random search (RS). We do not compare to SVRs [Brero *et al.*, 2021] since they were already outperformed by plain NNs in [Weissteiner and Seuken, 2020]. We observe that RS incurs efficiency losses of 30–50% illustrating the need for smart preference elicitation. Moreover, we see that MVNNs also significantly outperform FTs in all domains.

[6] We provide corresponding plots for the other domains and bidder types in Appendix D.3; the results are qualitatively similar.

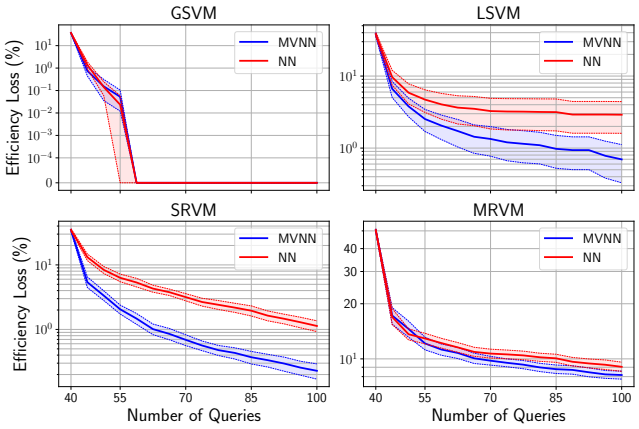

Figure 2: Efficiency loss paths of MLCA with MVNNs vs plain NNs. Shown are averages with 95% CIs over 50 auction instances.

In Figure 2, we present the efficiency loss path of MVNNs vs NNs (i.e., the regret curve) corresponding to Table 2. We see that in LSVM and SRVM, MVNNs lead to a smaller (average) efficiency loss for *every* number of queries. In MRVM, the same holds true for 50 and more queries. In GSVM, both networks have no efficiency loss in every instance after only 56 queries. Since a single query can be very costly in real-world CAs, it makes sense to ask few queries. Figure 2 shows that MVNNs consistently outperform plain NNs also in settings with a small number of queries (i.e., reduced $Q^{\max}$)

5.3 MILP Runtime Analysis

When integrating MVNNs into MLCA or another iterative combinatorial assignment mechanism, one needs to solve the MVNN-based WDP multiple times in one full run. Thus, the key computational challenge when integrating MVNNs in such mechanisms is to efficiently solve the MVNN-based WDP. In Theorem 2, we have shown how to encode the MVNN-based WDP as a succinct MILP, which can be solved efficiently for moderate architectures sizes. However, due to the bReLU activation function, i.e., the two cutoffs at 0 and $t > 0$, the MVNN-based MILP has twice the number of binary variables ($y^{i,k}$ and $\mu^{i,k}$) than the MILP encoding of a plain NN with ReLUs [Weissteiner and Seuken, 2020].

Figure 3 presents MILP runtimes of MVNNs vs plain ReLU NNs for selected architectures. We observe two effects: First, even though the MVNN-based MILPs have twice the number of binary variables, they can be solved faster than the plain NN-based MILPs for all architectures. Second, the deeper the architecture or the more neurons, the larger this difference becomes. One possible explanation for both effects is that MVNNs are more regular than plain NNs (due to their monotonicity property) and are thus easier to optimize.

6 Conclusion

In this paper, we have introduced MVNNs, a new class of NNs that is specifically designed to model normalized and monotone value functions in combinatorial assignment problems. We have experimentally evaluated the performance of MVNNs in four combinatorial spectrum auction domains and

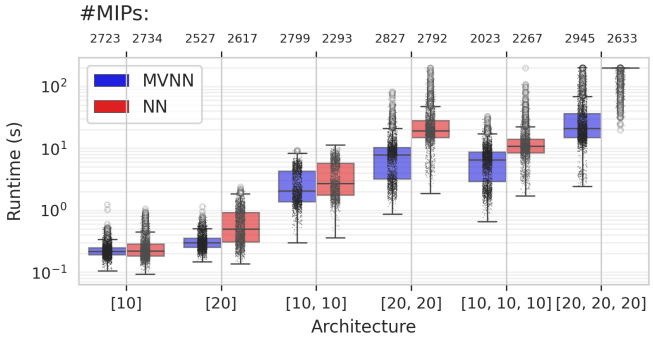

Figure 3: MILP runtime (200s time limit) of MVNNs vs plain NNs in MLCA on 10 LSVM instances for a selection of architectures.

shown that MVNNs outperform plain NNs with respect to prediction performance, economic efficiency, and runtime. Overall, our experiments suggest that MVNNs are the best currently available model for preference elicitation in combinatorial assignment (also compared to FTs and SVRs). Thus, incorporating important structural knowledge in the ML algorithm plays an important role in combinatorial assignment.

MVNNs enable us to incorporate an informative *prior* into a market mechanism. Future work could use such informative priors and enhance existing mechanisms (e.g., MLCA) by also using the *posterior* estimates in a more principled way than just the mean prediction. For example, one could frame an ICA as a (combinatorial) Bayesian optimization task and integrate a well-defined notion of posterior uncertainty to foster exploration [Heiss *et al.*, 2021]. Finally, it would be interesting to also evaluate the performance of MVNNs in other combinatorial assignment problems such as course allocation.

Acknowledgments

We thank the anonymous reviewers for helpful comments. 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 and Baranov, 2017] Lawrence M Ausubel and Oleg Baranov. A practical guide to the combinatorial clock auction. *Economic Journal*, 127(605):F334–F350, 2017. 1, 3, 6

[Ausubel and Cramton, 2011] Lawrence Ausubel and Peter Cramton. Auction design for wind rights. *Report to Bureau of Ocean Energy Management, Regulation and Enforcement*, 2011. 1

[Ausubel *et al.*, 2006] Lawrence M Ausubel, Peter Cramton, and Paul Milgrom. The clock-proxy auction: A practical combinatorial auction design. In Peter Cramton, Yoav Shoham, and Richard Steinberg, editors, *Combinatorial Auctions*, pages 115–138. MIT Press, 2006. 1

[Bichler *et al.*, 2019] Martin Bichler, Vladimir Fux, and Jacob K Goeree. Designing combinatorial exchanges for the

reallocation of resource rights. *Proceedings of the National Academy of Sciences*, 116(3):786–791, 2019. 1

[Brero *et al.*, 2018] Gianluca Brero, Benjamin Lubin, and Sven Seuken. Combinatorial auctions via machine learning-based preference elicitation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018. 2

[Brero *et al.*, 2019] Gianluca Brero, Sébastien Lahaie, and Sven Seuken. Fast iterative combinatorial auctions via bayesian learning. In *Proceedings of the 33rd AAAI Conference of Artificial Intelligence*, 2019. 2

[Brero *et al.*, 2021] Gianluca Brero, Benjamin Lubin, and Sven Seuken. Machine learning-powered iterative combinatorial auctions. *arXiv preprint arXiv:1911.08042*, Jan 2021. 2, 3, 5, 6

[Bronstein *et al.*, 2017] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017. 2

[Budish, 2011] Eric Budish. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy*, 119(6):1061–1103, 2011. 1

[Cramton, 2013] Peter Cramton. Spectrum auction design. *Review of Industrial Organization*, 42(2):161–190, 2013. 1, 6

[Dütting *et al.*, 2019] Paul Dütting, Zhe Feng, Harikrishna Narasimhan, David C Parkes, and Sai Srivatsa Ravindranath. Optimal auctions through deep learning. In *Proceedings of the 36th International Conference on Machine Learning*, 2019. 2

[Goeree and Holt, 2010] Jacob K Goeree and Charles A Holt. Hierarchical package bidding: A paper & pencil combinatorial auction. *Games and Economic Behavior*, 70(1):146–169, 2010. 5

[Heiss *et al.*, 2021] Jakob Heiss, Jakob Weissteiner, Hanna Wutte, Sven Seuken, and Josef Teichmann. Nomu: Neural optimization-based model uncertainty. *arXiv preprint arXiv:2102.13640*, 2021. 6, 7

[Hutter *et al.*, 2011] Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *International Conference on learning and intelligent optimization*, pages 507–523. Springer, 2011. 5

[Liew *et al.*, 2016] Shan Sung Liew, Mohamed Khalil-Hani, and Rabia Bakhteri. Bounded activation functions for enhanced training stability of deep neural networks on visual pattern recognition problems. *Neurocomputing*, 216:718–734, 2016. 3

[Liu *et al.*, 2020] Xingchao Liu, Xing Han, Na Zhang, and Qiang Liu. Certified monotonic neural networks. *Advances in Neural Information Processing Systems*, 33:15427–15438, 2020. 2

[Nisan and Segal, 2006] Noam Nisan and Ilya Segal. The communication requirements of efficient allocations and supporting prices. *Journal of Economic Theory*, 129(1):192–224, 2006. 1

[Osband *et al.*, 2021] Ian Osband, Zheng Wen, Mohammad Asghari, Morteza Ibrahimi, Xiyuan Lu, and Benjamin Van Roy. Epistemic neural networks. *arXiv preprint arXiv:2107.08924*, 2021. 5

[Rahme *et al.*, 2021] Jad Rahme, Samy Jelassi, Joan Bruno, and S. Matthew Weinberg. A permutation-equivariant neural network architecture for auction design. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, 2021. 2

[Scheffel *et al.*, 2012] Tobias Scheffel, Georg Ziegler, and Martin Bichler. On the impact of package selection in combinatorial auctions: an experimental study in the context of spectrum auction design. *Experimental Economics*, 15(4):667–692, 2012. 5

[Sill, 1998] Joseph Sill. Monotonic networks. *Advances in Neural Information Processing Systems*, 10:661–667, 1998. 2

[Tjeng *et al.*, 2019] Vincent Tjeng, Kai Y. Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming. In *International Conference on Learning Representations*, 2019. 5

[Wehenkel and Louppe, 2019] Antoine Wehenkel and Gilles Louppe. Unconstrained monotonic neural networks. *Advances in Neural Information Processing Systems*, 32:1545–1555, 2019. 2

[Weiss *et al.*, 2017] Michael Weiss, Benjamin Lubin, and Sven Seuken. Sats: A universal spectrum auction test suite. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pages 51–59, 2017. 5

[Weissteiner and Seuken, 2020] Jakob Weissteiner and Sven Seuken. Deep learning-powered iterative combinatorial auctions. In *Proceedings of the 34th AAAI Conference of Artificial Intelligence*, pages 2284–2293, 2020. 2, 3, 6, 7

[Weissteiner *et al.*, 2022] Jakob Weissteiner, Chris Wendler, Sven Seuken, Ben Lubin, and Markus Püschel. Fourier analysis-based iterative combinatorial auctions. In *Proceedings of the 31st International Joint Conference on Artificial Intelligence*, 2022. 2, 3, 6

[You *et al.*, 2017] Seungil You, David Ding, Kevin Canini, Jan Pfeifer, and Maya Gupta. Deep lattice networks and partial monotonic functions. *Advances in neural information processing systems*, 30, 2017. 2