SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Bachelor's Thesis in Information Systems

Learning Methods for Iterative Combinatorial Auctions

Artem Melnychuk

SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Bachelor's Thesis in Information Systems

Learning Methods for Iterative Combinatorial Auctions

Lernmethoden für iterative kombinatorische Auktionen

Author:	Artem Melnychuk
Supervisor:	Prof. Dr. Martin Bichler
Advisor:	Eleni Batziou M.Sc.
Submission Date:	15.09.2023

I confirm that this bachelor’s thesis is my own work and I have documented all sources and material used.

Munich, 15.09.2023 Artem Melnychuk

Acknowledgments

First I would like to thank my advisor, Eleni Batziou, for her guidance and support throughout the process of writing this thesis. When I pitched an idea, she distilled its essence. When my experimental drafts were far from perfect, she patiently reviewed them and always gave me a hint of how to improve them. I am very grateful for her understanding, as well as her willingness to share her knowledge and invest her time in my work.

I have been fortunate to learn from Hayim Malkhasy, who has always provided unique perspectives and insightful comments. My thinking and work ethic was shaped by our thoughtful discussions. I am also grateful for his review and encouragement throughout the process of writing this thesis.

It was a great pleasure to discuss research ideas with Sarthak Parikh and see his passion for science. This also includes Kiril Ratarov and his uncanny ability to recognize pitfalls in various approaches to the problem.

A big thank to my brother who helped me with analyzing my experiments as well as always having my back. And to my parents, who sacrificed the comfort of staying in Ukraine to move to Germany allowing me to grasp new opportunities and chase my ambitions.

Abstract

The efficiency of combinatorial auctions is of high importance to multiple industries and organizations. Recent methods are incorporating machine learning and linear optimization into the process to improve the welfare of each participant and optimize allocations. Due to the nature of combinatorial auctions, bidders can only report a limited amount of preferences in the exponential space of possibilities. We explore the usage of active learning methods to improve the informativeness of each bundle to increase the efficiency of the allocation. We compare and test the efficiency and runtime of the auctions using Uniform Sampling, Greedy Farthest-First Traversal, Greedy Farthest-First Traversal Reverse, Greedy Active Learning on Input Values and Greedy Active Learning on Output Values. Our approach improves the efficiency of the iterative combinatorial auctions.

Contents

1 Introduction

1.1 Overview

Combinatorial auctions provide a sophisticated mechanism for allocating diverse sets of items to various bidders to attain an optimal solution. The optimal solution is defined as the allocation that maximizes the total utility of all bidders. This process faces significant **challenges:** limitation on the bundle submission, exponential growth of feasible bundles, computational restrictions and bidder reporting.

Contemporary models of combinatorial auctions impose constraints on bidders by restricting the number of bundles they can submit. Such restrictions can hinder the attainment of an efficient allocation.

The feasible set of bundles *expands exponentially* with the addition of each item. This exponential growth complicates the computational process, imposing a substantial barrier to the scalable allocation of goods, particularly in larger domains.

The inherent complexity of the exponential growth in the amount of bundles leads to computational constraints. Given a limited amount of reported bundles, these limitations restrict the auctioneer's ability to discern the true valuations of bidders' preferences. Thus, it is only possible to approximate each bidder's value function.

Bidders often find themselves restricted in their capacity to report the most informative bundles (Scheffel et al. 2012; Bichler, Shabalin, and Wolf 2013). This constraint directly impacts the maximization of allocation efficiency.

The underlying design of the auctions may sometimes provide incentives to bidders to withhold their true valuations rather than fully reveal them. Furthermore, in the context of real-world applications of combinatorial auctions, bidders may lack comprehensive insight into the underlying mechanisms governing the auction process. The reporting of bundles may be influenced more by personal preferences than by a strategic understanding of optimal allocation principles.

Progress: These challenges were thoroughly addressed by researchers, as evidenced by the work of Brero, Lubin, and Seuken (2021), serving as a platform to explore and evaluate diverse machine learning techniques in predicting user valuations. Those techniques tacked the problems of computational constraints and bidder reporting by using machine learning models to approximate the value functions of bidders.

A promising approach within this framework involved the integration of neural networks with Mixed-Integer Programming (MIP) (Weissteiner and Seuken 2020). Given the success of this approach, further methods have been developed to improve the efficiency of combinatorial auctions: Bayesian Optimization (Weissteiner et al. 2023) and Fourier based approach (Weissteiner et al. 2022). The results of those works have showed that the efficiency of combinatorial auctions can be improved by using machine learning techniques.

Remaining Challenges: Despite the progress achieved, significant challenges remain, especially in large domain auctions characterized by many bidders and items where each bidder's value function becomes more complex. This abundance of elements results in a multitude of potential combinations, complicating the analysis of such auctions. An illustration of these complexities can be found in the Canadian spectrum auctions (Taylor 2013), where at some point in time there were 2^{98} possible bundles, which approximates $3.16 \cdot 10^{29}$ possibilities. Meanwhile bidders could only report their true valuations for 500 bundles.

Outline of contributions: In this thesis, the focus is placed on the application and innovation of active learning techniques to enhance the efficiency of combinatorial auctions. This paper deals with the initial round of the auction process, where bidders must report their valuations of a limited number of bundles. Specifically, the work includes the implementation of two distinct methodologies: Greedy Active Learning on Input Values (GALI) and Greedy Active Learning on Output Values (GALO) (Estermann et al. 2023). Moreover, the thesis introduces another learning strategy designed for combinatorial auctions - Greedy Farthest-First Traversal (GFFT) and a similar method Greedy Farthest-First Traversal (reverse) (GFFT(reverse)). The results of these methods are compared to the baseline of Uniform Sampling (UF) in Deep Learning-powered Iterative Combinatorial Auctions (Weissteiner and Seuken 2020).

Motivations: The initial round of the auction dictates the efficiency and progress of the subsequent iterations of auctions. Iterative combinatorial auctions are used extensively in practice. From 2008 to 2014, in ten countries which were using spectral auctions, there were around fifteen instances of auctions creating $ 20 Billion in total revenue (Ausubel and Baranov 2017). A moderate 1-2% increase in efficiency will translate to millions in revenue. Therefore, it's evident that improving auction mechanisms can positively impact the economy making it a worthwhile endeavor. Having an interest in such pursuit, this thesis not only contribute to the market efficiency but also fosters economic growth, thereby serving as an accelerator for societal advancement.

Challenges of this approach: The approach of adding active learning techniques to iterative combinatorial auctions influences the subsequent iterations of the auction. The overall runtime can be increased significantly. The design of the algorithms is influenced by the computational complexity of the auction. The researcher has to decide between the tradeoff of better efficiency or shorter runtime. In this case study, the principles of the GFFT algorithms are leveraged to illustrate these dynamics, offering insights into fine-tuning active learning for speed and efficiency.

1.2 Related work

Combinatorial auctions have been the subject of extensive research over the past decades, with applications spanning various domains. Hershberger and Suri (2001) applied these auctions to network routing, while Caplice (1996) investigated their use in optimizing delivery routes. Additionally, Rassenti, Smith, and Bulfin (1982) studied combinatorial auctions in the context of airport slot allocations, and Brewer (1999) explored their application in railroad segments.

Among the most influential work for the preferences elicitation in combinatorial auctions are papers by Blum et al. (2003) and Lahaie and Parkes (2004). These introduce new elicitation learning algorithms and were further developed when the researchers introduced ML-based approaches to determine prices that guide bidders toward efficient allocations (Lahaie and Lubin 2019). Bayesian auctions were designed and built upon previous works to approximate competitive equilibrium prices in the works of Brero and Lahaie (2018) and Brero, Lahaie, and Seuken (2019).

Another line of research originated from the works of Conitzer and Sandholm (2002, 2004), focusing on formulating a space of possible solutions and then using a search problem to map bidders' preferences to bid profiles. More recently, Duetting et al. (2019) employed deep learning to enhance the design of revenue-maximizing auctions.

The work of Brero, Lubin, and Seuken (2021) and Weissteiner and Seuken (2020) are the most recent examples of using Support Vector Regression (SVR) and Deep Neural Network (DNN) respectively to approximate the value functions of bidders for generating new profiles of bundles.

As for active learning, while there is extensive work being done for classification problems, there is still a lack of research in the field of regression. A substantial amount of labeled samples are needed to train an accurate regression model. Instead of random selection, Wu, Lin, and Huang (2018) proposed to use geometric feature space of input and output of the model to increase diversity of the samples. By calculating expected model change for regression problems (Park and Kim 2020), it is possible to make choosing labels more robust by avoiding outliers, which do not contribute to the improvement of the model. Wu (2018) proposed to extend criteria for pool-based active learning, incorporating such things as informativeness, representativeness and diversity. These methods can also be applied to combinatorial auctions, as they are not restricted to a specific domain.

1.3 Contributions

The goal of this paper is to increase the efficiency of auctions by querying the bidder with most important initial bundles. By iteratively discovering the most informative bundles, true preferences of a bidder are successfully learned, allowing for efficient bundle allocation and ensuring high efficiency with low auction runtime.

Firstly, we extend the work of Brero, Lubin, and Seuken (2021) by incorporating active learning methods into the framework. This would allow future research in active learning in combinatorial auctions to be more flexible, as it is not restricted to only using UF for the initial bundles.

Secondly, this paper contributes by implementing GALI and GALO algorithms (Estermann et al. 2023). These algorithms are being tested on the Generalized Value Model (GSVM), Linearly Correlated Value Model (LSVM) and Multi-Region Value Model (MRVM) domains. The results are compared to the baseline of UF in Deep Learning-powered Iterative Combinatorial Auctions (Weissteiner and Seuken 2020). They show a considerable improvement in efficiency of the auctions.

Next, we propose GFFT and GFFT(reverse) algorithms, which are particularly useful for smaller auction domains. The results of GFFT and GFFT(reverse) are compared to the results of GALI and GALO algorithms. We show how GFFT and GFFT(reverse) perform even better in efficiency than other algorithms in smaller scale domains. This thesis contributes by showing that active learning methods can be used to improve the efficiency of combinatorial auctions.

1.4 Outline

This thesis is organized in the following structure. Chapter 1 provides an overview of the thesis, its contributions and the outline of the work. Chapter 2 introduces the preliminaries of understanding iterative combinatorial auctions, the work involved with optimizing them. It also introduces the important notation and definitions used throughout the thesis. Chapter 3 introduces active learning methods, describing the algorithms and providing their implementation. Chapter 4 presents the results of the experiments and the performances of the algorithms.

Given the findings, Chapter 5 concludes the thesis and provides an outlook on future research directions.

2 Preliminaries

2.1 Combinatorial Auctions

Auction is a process of buying or selling goods or services by bidding. Generally, the item is sold to the highest bidder and is bought from the lowest bidder. Combinatorial auctions are a type of auction that allows bidders to submit bids on a combination of items. Those combinations are called bundles and having a different combination of items produces a complementary effect for bidders. Meaning that the combinations of some items in the bundle are more or less valued according to different types of bidders. Notation is defined to support the claims:

Let $N = \{1, 2, \ldots, n\}$ denote the set of bidders participating in an auction, and let $M = \{1, 2, \ldots, m\}$ represent the set of indivisible items that are being auctioned off. Within this context, the set N is referred to as the *main economy*, whereas any subset $N \setminus \{i\}$ is termed a *marginal economy*.

A *bundle* of items is represented as a vector X of $\{0, 1\}^m$, where each element in the vector indicates the presence or absence of an item in the bundle.

The preferences of each bidder i are encapsulated by a valuation function $u_i : X \to \mathbb{R}_{\geq 0}$. In this function, $u_i(x)$ quantifies the true value that bidder i assigns to obtaining bundle x. For the entire profile of bidders, the valuation vector is denoted as $\mathbf{u} = \{u_1, u_2, \ldots, u_n\}$.

Important parameters for this work are Q_{max}, P_{max} and Q_{init}. Q_{max} is the maximum amount of queries per bidder, P_{max}, is the maximum amount of push bids per bidder and Q_{init}, is the amount of initial queries. Those parameters are influencing the amount of rounds in Machine Learning Combinatorial Auction (MLCA) algorithm

An *allocation* for all bidders is represented by the vector $\mathbf{a} = \{a_1, a_2, \ldots, a_n\}$, where a_i denotes the allocation of items to bidder i. An allocation is termed *feasible* if each item in M is allocated to at most one bidder.

Payments made by the bidders are captured by the payment vector $\mathbf{p} = \{p_1, p_2, \ldots, p_n\}$, where p_i represents the payment charged to bidder i. The utility function for each bidder i is defined as a quasi-linear function $u_i(a, p) = u_i(a_i) - p_i$.

The cumulative true values of bidders for an allocation constitute its social welfare, which is defined as $V(a) = \sum_{i \in N} u_i(a)$. The efficiency of any given allocation $\mathbf{a}$ is denoted by $\text{Eff}(\mathbf{a})$, and is defined as the ratio of the total valuation for allocation $\mathbf{a}$ to the total valuation for the optimal allocation $\mathbf{a}^*$, i.e.,

$$\text{Eff}(\boldsymbol{a}) = \frac{V(\boldsymbol{a})}{V(\boldsymbol{a}^*)} \tag{2.1}$$

2.1.1 Vickrey-Clarke-Groves Auction

This section introduces the concept of Vickrey-Clarke-Groves auction (VCG) (Vickrey 1961; Clarke 1971; Groves 1973), a class of auctions known for their strategy-proof properties:

The *allocation rule* is given by $a_{\text{VCG}} \in \arg\max_{a \in \mathcal{F}} \sum_{i \in N} u_i(a_i)$, whereas the *payment rule* is defined as:

$$p^i_{\text{VCG}} = \sum_{j \in N \setminus \{i\}} u_j(a_j^{-i}) - \sum_{j \in N \setminus \{i\}} u_j(a_{\text{j}}^{VCG}) \tag{2.2}$$

,

where an optimal item allocation a^{-i} is defined as:

$$a^{-i} \in \arg\max_{a \in \mathcal{F}} \sum_{j \in N \setminus \{i\}} u_j(a_j) \tag{2.3}$$

VCG is characterized by incentive compatibility, which encourages participants to reveal their true valuations. Formally, the mechanism is *strategy-proof*, implying that each bidder maximizes their utility by truthfully disclosing their valuation, irrespective of the strategies adopted by others.

For the VCG to be efficient, several conditions must be met:

1. *Rationality*: It is assumed that all bidders are rational agents who aim to maximize their individual utilities.

2. *Absence of Bid Collusion*: The mechanism assumes that no collusive behavior exists among the bidders.

3. *Unique Identity*: A single bidder is not permitted to submit bids under multiple identities.

If the conditions are satisfied, then truthful reporting of valuations creates a weakly dominant strategy, effectively making any alternative strategy profiles suboptimal.

However, the VCG is not viable for combinatorial auctions as it requires a full valuation profile of each bidder as it is written in Equation 2.2. Mishra and Parkes (2007) and Vries, Schummer, and Vohra (2007) have introduced iterative VCG mechanism. These iterative mechanisms are designed to engage with bidders by systematically eliciting information across multiple rounds of interaction.

2.2 Machine Learning-based Combinatorial Auction

The MLCA is an auction mechanism that utilizes iterative rounds and incorporates machine learning algorithms to approximate the valuation functions of participating bidders. During each round, bidders receive new queries concerning both their marginal and main economic preferences.

For each bidder $i \in N$ there is a unique algorithm J_i, which processes the reported bundle-value pairs R_i. In this specific implementation (Weissteiner and Seuken 2020), J_i corresponds to a Deep Neural Network (DNN). Each DNN, trained on R_i, learns an estimated valuation function v_i that can predict the value of any potential bundle for the corresponding bidder i.

The primary objectives of MLCA diverge from those of traditional VCG auctions in several key aspects. Firstly, MLCA aims to impose constraints on the volume of submitted bids, as opposed to the unconstrained bidding typically observed in VCG auctions. This constraint serves to accommodate realistic combinatorial auction scenarios, where bidders are often limited in their capacity to submit bids. In this implementation, for each bidder, the number of bids is limited to $Q^{\max}$ - $P^{\max}$.

Secondly, MLCA is designed to establish a robust framework explicitly tailored for iterative combinatorial auctions. Such a framework provides a foundational bedrock, upon which comparable baselines can be constructed for future research endeavors in this domain.

2.2.1 Machine Learning-powered Query Module

A dedicated query module powered by machine learning is employed to calculate the Winner Determination Problem (WDP). The query module accepts a parameter set, comprising a profile of machine learning algorithms for each bidder, an array of economies, and a profile consisting of bundles alongside their reported values. Afterwards, the module trains separate algorithms on the bundle-value pairs, finding valuation functions u_i Mathematically, the social welfare is denoted as $V(a) = \sum_i u_i$, where u_i represents the individual welfare attributable to a given bidder i. It is used as a objective function to be maximized by the ML-based WDP. All valuation functions v_i are fed into ML-based WDP to calculate the allocation a_{MLCA}, where queries are being generated for a specific bidder to achieve this allocation. In the Algorithm 1 the query module is represented as NextQueries function.

2.2.2 Design Elements

There exist four design elements that dictate the functionality of the MLCA:

1. Allocation is based on the reported bundle-value pairs, not on the learned valuation functions.

2. VCG payments are based on bundle-value pairs.

3. MLCA queries marginal economies of each individual bidder.

4. Bidders can submit or 'push' their bids.

Changing the underlying VCG mechanism to an iterative form no longer allows for the assumption that the mechanism is immune to manipulations. In their study, Brero, Lubin, and Seuken (2021) argues that these design choices promote good incentives and make any possible manipulation strategies impossible to execute successfully. No approximation is involved in calculating bundle-value pairs, meaning the allocation is based on the truthful and constant reporting of bidders. Thus, the allocations are always reproducible and do not allow randomness. When a bidder's marginal economy is independent of his reports, the only way to increase his utility is by enhancing the social welfare of the main economy through truthful value reporting.

MLCA framework offers a dynamic and adaptable system designed to make auction processes more flexible. Bidders can push bids that they deem informative to the auction mechanism. There is a specified limit on the number of bids each bidder can submit, preventing potential abuse or congestion. The auctioneer retains control over the number of rounds during an auction. This control is facilitated through a variable, denoted as Q_{round}, which controls the number of queries asked each round. The default payment rule is VCG. This VCG module can be easily exchanged for alternative payment rules, thereby accommodating diverse auction requirements and objectives.

2.2.3 MLCA Mechanism

Combining these details results in a fully-fledged model, described step by step in Algorithm 1.

- Initialization phase (Line 1-8), bidders push bids and are asked to report their values for randomly chosen bundles.

- Iteration phase (Line 9 - 25), for every round t choose a sample of bidders and generate new queries for them using NextQueries. Calculate query profile for main economy. Get reported values for bids from bidders.

- Final allocation phase (Line 26 - 30), compute final allocation and payments.

The initialization phase is tackled in this work. In line 4-5, there is a process of generating random initial bids for Q_{init}. As this process influences all following iterations, it is essential to optimize it. Brero, Lubin, and Seuken (2021) argue that randomizing initial samples is one of the design choices of the MLCA to promote the strategy-proof properties as the bidders cannot predict which bundles did other participants get. To support this phenomenon, uniform sampling is still employed in some of the algorithms.

2.2.4 Deep Learning-powered Iterative Combinatorial Auctions

The initial formulation of the MLCA mechanism employed SVR as the underlying machine learning model. In contrast, the current method leverages DNN (Weissteiner and Seuken 2020) with RELUs turned into MIP for resolving the WDP to generate the subsequent query profile. The integration of a machine learning algorithm into the MLCA framework primarily influences two facets: the *estimation step* and the *optimization step*. During the *estimation* phase, the chosen machine learning model predicts the bidders' valuations for the next auction round. Afterwards, in the *optimization* phase, the model is tasked with solving the WDP. Weissteiner and Seuken (2020) argued that SVR have limited expressiveness and cannot be effectively used in large auction domains.

Data: Profile of ML algorithms M, maximum amount of queries per bidder Q_{max}, amount of initial queries $Q_{\text{init}} \leq Q_{\text{max}}$, amount of queries per round Q_{round}, maximum amount of push bids per bidder P_{max}
Result: Final allocation a_{MLCA} and payments p_{MLCA}

for *each bidder* $i \in N$ **do**
  receive $P_i \leq P_{\text{max}}$ push bids;
end
for *each bidder* $i \in N$ **do**
  ask bidder to report his value for Q_{init} randomly chosen bundles;
end
Let $R = (R_1, \ldots, R_N)$ denote the initial report profile;
Let $T = \left\lfloor \frac{Q_{\text{max}} - Q_{\text{init}}}{Q_{\text{round}}} \right\rfloor$ and $t = 1$ denote the number of auction rounds and the current round, respectively;
while $t \leq T$ **do**
  Let $S = (S_1, \ldots, S_n)$ denote the profile of queries for this auction round;
  for *each bidder* $i \in N$ **do**
    Sample a set of bidders $N' \subseteq N \setminus \{i\}$ with $|N'| = Q_{\text{round}} - 1$;
    for *each* $i \in N'$ **do**
      Generate query profile $q_i := \text{NextQueries}^J(N' \setminus \{i\}, R'_{-i}, S'_{-i})$ for bidder i;
      $S_i = S_i \cup \{q_i\}$;
    end
  end
  Generate query profile $q := \text{NextQueries}^J(N, R, S)$;
  for *each bidder* $i \in N$ **do**
    $S_i = S_i \cup \{q_i\}$;
    send new queries S_i to bidder i and wait for reports;
    receive bundle-value reports R'_i and add them to R_i;
  end
  $t = t + 1$;
end
Let v_i denote bidder i's report function capturing bundle-value reports $R_i, \forall i \in N$;
Compute final allocation $a_{\text{MLCA}} = \arg\max_{a \in F} \sum_{i \in N} v_i(a_i)$;
for *each bidder* $i \in N$ **do**
  $p_i^{\text{MLCA}} = \sum_{j \in N \setminus \{i\}} v_j(a_j^{-i}) - \sum_{j \in N} v_j(a_j^{\text{MLCA}})$, where $a^{-i} \in \arg\max \sum_{j \in N \setminus \{i\}} v_j(a_j)$;
end

Algorithm 1: Machine Learning-powered Combinatorial Auction (MLCA)

3 Active Learning

3.1 Overview

A profound idea behind a concept of active learning is to reduce the amount of data needed to train a model. There are cases in which unlabeled data is abundant but requires labeling. In such scenarios, active learning algorithms can be used to reduce the amount of data to be labeled by selecting the most informative data points. The model has the ability to query the oracle (usually a human annotator) for labels of data instances that it finds most "informative". Here is a breakdown of the major types of active learning:

- Pool-Based Sampling: the model evaluates each instance and picks the ones where it's most uncertain. The model queries instances for which it is most uncertain, typically those where the predicted probabilities are closest to 0.5 or the entropy is highest.
- Stream-Based Selective Sampling: instances come in a stream, one at a time. Your model has to make an immediate decision whether to query a label for each instance. There is a threshold for uncertainty, and if the model is uncertain beyond that threshold, it queries the label.
- Query-By-Committee: multiple models are trained on the same data. They 'vote' on each unlabeled instance. If there's discord, i.e., high variance in predictions, the instance is queried.
- Expected Error Reduction: the model computes the expected error reduction for each unlabeled instance. It queries the instance that is expected to reduce the error the most.
- Variance Reduction: the model computes the variance of the model parameters for each unlabeled instance. It queries the instance that is expected to reduce the variance the most.
- Bayesian Active Learning: the model computes the posterior probability of each unlabeled instance. It queries the instance that is expected to maximize the posterior probability.

- Reinforcement Learning based: the model learns a policy to select the most informative instances to query.

This paper focuses on the Pool-Based Sampling approach, as it is the most applicable to the MLCA domain.
Wu (2018) proposes three criteria to be considered in pool-based sampling:

- Informativeness: samples, which are selected, must contain rich information. This is measured in entropy, distance to the decision binary, expected model change etc.
- Representativeness: number of samples, which represent the whole variance.
- Diversity: how samples are scattered around the full space.

The main idea behind pool-based approach is to select the most informative bundles from the available pool of bundles. The metrics in this paper are defined differently per active learning algorithms. However, the main idea is to query a bundle with the largest "distance" from the already selected bundles. For GFFT, GFFT(reverse) and GALI the distance is calculated between the input values (bundles), and for GALO the distances is calculated between the output values (values of bundles). Increasing the information gain per bundle reduces the number of bundles that need to be sampled. We enhance informativeness by choosing bundles that have the greatest distance from the bundles already selected. In our experiments, representativeness is a constant number, which means every algorithm has a limited number of samples available. We measure the diversity parameter using the distance between bundles. In other words, the more distant the selected bundles are, the more diverse the sample becomes.
Main problem in active learning for auctions is that the majority of the research in this area is being done in the context of text classification, image classification, etc. Those methods are usually not applicable to combinatorial auctions, because those methods are used for classification problems, while combinatorial auctions require active learning solutions to regression problems. In contrast to the diversity of image or text data, combinatorial auctions feature a set of bundles that are monotonous and predictable.
The core mechanism of the MLCA utilizes a UF technique for selecting an initial subset of bundles from a considerably large set. UF is a probabilistic method used to extract a representative subset of data elements, denoted as S, from a larger data population $\mathcal{X}$. The underlying principle behind this sampling technique is that each element $x \in \mathcal{X}$ has an equal probability, $p(x)$, of being selected. This probability for UF can be mathematically represented as follows:

$$p(x) = \frac{1}{|\mathcal{X}|}, \quad \forall x \in \mathcal{X} \tag{3.1}$$

The computational complexity of generating a sample S is generally $O(n)$, assuming random access to $\mathcal{X}$. UF is widely applied in various domains for its simplicity and efficiency. However, it might not be suitable for populations with high variance, such as the MRVM auction domain discussed in this thesis.

The cardinality of this subset, denoted as $|S|$, usually spans between 30 and 100 per auction model in the implementation. This range helps to achieve increased efficiency without significantly impacting runtime, even through this quantity might seem trivial when compared to the totality of available bundles. As $|S|$ grows, the computational complexity of preference allocation escalates exponentially. In real-world scenarios, handling such complexity becomes infeasible. Therefore, it is essential to devise methodologies to minimize the number of bundles to be sampled, ensuring optimal efficiency without compromising runtime.

3.2 GFFT

A Greedy Farthest-First Traversal (GFFT) algorithm is developed as a preliminary method for computing the distance between specific clusters of bundles and the rest of the bundles in the given set. The intent of this approach is to maximize entropy, aiming to explore previously not encountered combinations of items. This algorithm proves particularly useful in performance evaluation within smaller domains.

The computational complexity of the algorithm is expressed in $O(n^2)$ steps and $O(n^2)$ distance computations, making it suitable for smaller auction settings.

Formal Definition: Let $\mathcal{S}$ be a selected subset of bundles, and $\mathcal{X} \setminus \mathcal{S}$ be the universal set containing all available bundles. The subsequent bundle to be incorporated into S, denoted as x^*, is determined as:

$$x^* = \arg\max_{x \in \mathcal{X}} ||x - \mathcal{S}|| \quad \forall x \in X \tag{3.2}$$

This formulation facilitates the selection of the bundle that maximizes its distance from the existing set $\mathcal{S}$. This GFFT is employed to benchmark both runtime and efficiency for smaller domains. The distance between $\mathcal{X}$ and $\mathcal{S}$ is computed by employing Euclidean metrics.

The details for the algorithm are provided in Algorithm 2.

Although this algorithms

Algorithm 2: GFFT

Data: Amount of bundles n
Result: Matrix S containing selected bundles and their values
X $\leftarrow$ generate available bundles
$b_0 \leftarrow$ random element from X
$S \leftarrow \varnothing$
$S[0] \leftarrow b_0$
for i *from* 1 *to* $n-1$ **do**
  dist$_x$ $\leftarrow$ calculate Euclidean distance between X and S
  $x^* = \arg\max_{x \in \mathcal{X}} dist_x$
  $S[i] \leftarrow x^*$
end
$S \leftarrow$ append values for each bundle provided by bidder
return S

3.3 GFFT: Reverse

In contrast to the original greedy algorithm, the present variant employs a similar heuristic but introduces an optimization to mitigate the computational burden of distance calculations in exponential time. By leveraging the inherent data characteristics, vector bundles can be transformed to reflect the frequency of individual items.

Formal Definition: Let $\mathcal{S}$ denote a selected subset of bundles, where $s_i \in \mathcal{S}$. A rounding function Θ is introduced and defined as:

$$\Theta(x) = \begin{cases} 0, & \text{if } x > 0.5 \\ 1, & \text{otherwise} \end{cases} \tag{3.3}$$

This function is designed to introduce an equilibrium within the bundle set $\mathcal{S}$ through a process of bit inversion. Initially, the mean vector of $\mathcal{S}$ is computed. Thereafter, the subsequent bundle to be appended to $\mathcal{S}$ is identified by rounding the components of the mean vector to their closest bit values and subsequently inverting these bits. This means that an item in the bundle is **not** included if it is present in more than half of the bundles in $\mathcal{S}$.

$$(x^*)_j = \Theta\left(\frac{1}{i}\sum_{s \in S}(s_i)_j\right) \quad \text{for } j = 1, 2, \ldots, n \tag{3.4}$$

Using Algorithm 3, an average occurrence of all items in the bundle set $\mathcal{S}$ is achieved.

Algorithm 3: GFFT(reverse)

Data: Amount of bundles n, Initial Sample Size k
Result: Matrix S containing selected bundles and their values
$b_k \leftarrow$ generate k bundles and values using uniform sampling
$S \leftarrow \varnothing$
$S \leftarrow b_k$
$count \leftarrow 1$
while $count \leq n - k$ **do**
  $m \leftarrow$ mean of all bundles in S
  $x^* \leftarrow \Theta(m)$
  $S[k + count] \leftarrow x^*$
  count $\leftarrow$ count $+ 1$
end
$S \leftarrow$ append value from bidder
return S

3.4 Greedy Active Learning on Input Values

The GALI (Yu and Kim 2010) was originally conceived as a technique for maximizing diversity among data points in the input space. Given a set S consisting of bundles, the objective is to identify the bundle x^* that is at the maximum distance from all the bundles in S. This active learning method explores the geometric solution, without the need to learn a regression function. As a result, sampling becomes more efficient and stable, as it is not dependent on iterations of learning and validating a regression function. Wu, Lin, and Huang (2018) propose an algorithm to iterate over all unlabeled bundles in the available pool, the size of the bundle space in large auctions is growing exponentially, presenting a challenge. To address this, Estermann et al. (2023) modified the method to be used in combinatorial auctions and used Integer Linear Programming (ILP) for calculating the distance in pseudo-polynomial time.

Formal Definition: Given a bundle space $\mathcal{X}$ and a set of sampled bundles S, where $s_i \in S$, the objective is to identify the bundle x^* that is at the maximum distance from all the bundles in S. For each already sampled bundle $s \in S$, a subset $H \subseteq X \setminus S$ is defined, whose elements are at maximal distance from s.

The next bundle to query, where $x \in H$ is defined as:

$$x^* = \arg\max_{x \in H} \|x - s\| \quad \forall s \in S \tag{3.5}$$

Constructing the set H necessitates calculating distances between each element of S and X, leading to exponential complexity. To avoid this, the subsequent optimization problem is solved for each $s \in S$:

$$\begin{aligned} &\arg\max_{x \in X} \|x - s\| \\ &\quad \text{s.t.} \quad \|x - s\| \leq \|x - s'\| \quad \forall s' \in S, s' \neq s \end{aligned} \tag{3.6}$$

The core aim of this optimization is to identify a vector x that maximizes its distance to a given vector s while ensuring its proximity to all other vectors s' in the set S is minimized. Since the formulation is non-linear, it is reformulated into a linear optimization problem:

$$\begin{aligned} &\arg\max_{x \in X} \sum_{j=1}^{m} x_j + s_j - 2x_j s_j \\ \text{s.t.} \quad &\sum_{j=1}^{m} x_j + s_j - 2x_j s_j \leq \sum_{j=1}^{m} x_j + s'_j - 2x_j s'_j \quad \forall s' \in S, s' \neq s \end{aligned} \tag{3.7}$$

From the computed bundles, the ILP selects the bundle that maximizes the distance metric. The runtime is pseudo-polynomial, with the pseudo-code detailed in Algorithm 4.

Algorithm 4: GALI

Data: Amount of bundles n
Result: Matrix S containing selected bundles and their values
$b_0 \leftarrow$ generate random bundle
$S \leftarrow \varnothing$
$S[0] \leftarrow b_0$
$count \leftarrow 1$
while $count \leq n$ **do**
    $\text{dist}_x \leftarrow \varnothing$
    for i *from* 1 *to* $n - 1$ **do**
        $x^* = \arg\max_{x \in H} \|x - s\|$ compute using ILP Equation 3.7
        $\text{dist}_x \leftarrow x^*$
    end
    $S[count] \leftarrow \arg\max dist_x$
    count $\leftarrow$ count + 1
end
$S \leftarrow$ append values for each bundle provided by bidder
return S

3.5 Greedy Active Learning on Output Values

The GALO is based on the similar idea as GALI. However, instead of calculating distances between bundles (input values), GALO calculates distances between the reported values of bundles (output values).

Given a set S consisting of pairs of bundles and their respective values Y, the formulation is extended to include $u(x)$, a value function associated with a bundle x. The learning of this value function is performed separately from the linear optimization and requires a initial sampling strategy. UF was used for learning the output values.

The choice of machine learning algorithm to learn value function, be it linear or a fully-connected neural network, leads to variations in the implementation of the ILP due to the different representations of the value function $u(x)$.

The next bundle to query is determined by:

$$x^* = \arg\max_{x \in H} \|u(x) - s\| \quad \forall y \in Y \tag{3.8}$$

To circumvent the need to calculate this distance for every bundle in X, the following optimization problem is posed:

$$\begin{aligned} & \arg\max_{x \in X} |y - u(x)| \\ \text{s.t.} \quad & |y - u(x)| \leq |y' - u(x)| \quad \forall y' \in Y, y' \neq y \end{aligned} \tag{3.9}$$

This problem formulation requires adaption into a linear program due to the non-linear constraints imposed by the Euclidean norm and the value function $u(x)$. Subsequent sections discusses linear optimizations for both a Linear Model and a Neural Network Model. In this context, a parameter k is introduced, representing the number of bundles to initially sample using UF for learning the value function. The pseudocode is provided in Algorithm 5.

3.5.1 Greedy Active Learning on Output Values: Linear Model

In case of the Linear Model, the value function $u(x)$ takes the form $u(x) = w^T\mathbf{x} + w_0$, where w are the coefficients in $\mathbb{R}^m$ and w_0 is an intercept in $\mathbb{R}$. The input bundle of x remains a vector $\{0,1\}^m$

The corresponding ILP serves as a linearized implementation of the previously defined optimization problem. Additionally, C is introduced as a large constant larger than y or $u(x)$ values, which ensures that the constraints are satisfiable and equivalent to Equation 3.9: Different linear models, such as Ordinary Least Squares, Ridge Regression, Non-negative Least Squares, were tested without observing any significant performance difference. Ordinary Least Squares was selected as the default method for

Algorithm 5: GALO

Data: Amount of bundles n, Initial Sample Size k, Machine Learning Model M
Result: Matrix S containing selected bundles and their values
$b_k \leftarrow$ generate k bundles and values using uniform sampling
$S \leftarrow \varnothing$
$S \leftarrow b_k$
$count \leftarrow 1$
while $count \leq n - k$ **do**
    $u(i) \leftarrow M(S_x, S_y)$where S_x are bundle and S_y are values
    $\text{dist}_x \leftarrow \varnothing$
    iterations $= len(S)$
    for *i from* 0 *to iterations* **do**
        $x^* = \arg\max_{x \in H} \|x - s\|$ compute using ILP Equation 3.9
        $\text{dist}_x \leftarrow x^*$
    end
    $S[k + count] \leftarrow \arg\max dist_x$
    for bundle provided by $S[k + count] \leftarrow$ append value from bidder
    count $\leftarrow$ count $+ 1$
end
return S

Greedy Active Learning on Output Values (linear) (GALO(linear)) due to its simplicity and computational efficiency.

$$
\begin{aligned}
\arg\max_{x \in X}\ & r \\
\text{s.t.}\quad & y - u(x) \leq r, \\
& u(x) - y \leq r, \\
& y' - u(x) + C \cdot b_{y'} \geq r, \\
& u(x) - y' + C \cdot (1 - b_{y'}) \geq r, \\
& \forall y' \in Y, \\
& b_{y'} \in \{0, 1\} \quad \forall y' \in Y, \\
& r \in \mathbb{R}
\end{aligned}
\tag{3.10}
$$

3.5.2 Greedy Active Learning on Output Values: Neural Network

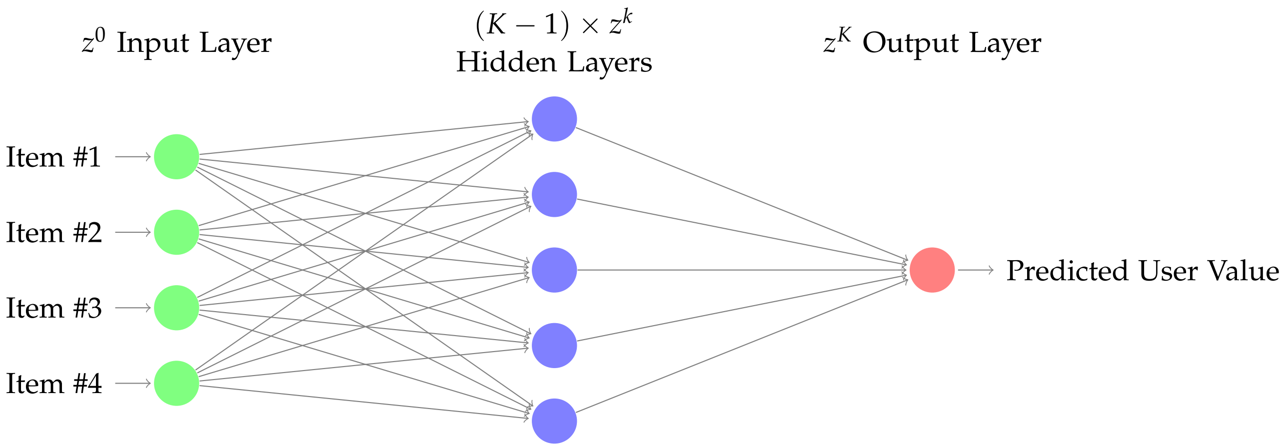
A neural network notation to approximate a value function $u(x)$ is defined as:

- $K = \{0, \ldots, K\}$ denotes the set of layers in the network(input and output layer included).
- k is an index representing a specific layer in K.
- d_k denotes the dimension of layer k.
- b_k is the bias term for layer k.
- W_k is the weight matrix for layer k.
- o_k is the output of layer k, defined as $o_k = ReLU(W_{k-1} \cdot o_{k-1} + b_{k-1})$.

Subsequently, a few constraint vectors are introduced to ensure that the network output is equal to the value function: $u(x)$: z^k, y^k, q^k, all in d_k dimension. z^k and q^k are continuous constrains, whereas y^k is a binary constraint. L is used as a large constant, so for every $z \in z^k, z < L$ and $q \in y^k, y < L$. The objective of the linear optimization is to minimize the distance between the output of the network and the value function $u(x)$ and achieve $z^0 = o^0 = x^*$

$$
\begin{aligned}
& \arg\max_{x \in X} r \\
\text{s.t.} \quad & y - z^K \leq r \\
& z^K - y \leq r \\
& y' - z^K + C \cdot b_{y'} \geq r \\
& z^K - y' + C \cdot (1 - b_{y'}) \geq r \\
& \forall y' \in Y, \quad b_{y'} \in \{0, 1\} \\
& r \in \mathbb{R} \\
& z_0 = x \\
& z^k - q^k = W^{k-1} \cdot z^{k-1} + b^{k-1} \\
& 0 \leq z^k \leq y_k \cdot L \\
& 0 \leq q^k \leq (1 - y^k) \cdot L \\
& y^k \in \{0, 1\}^{d_k} \\
& \forall k \in \{1, \ldots, K\}
\end{aligned}
\tag{3.11}
$$

The following picture provides a visual representation how the neural network is used in Greedy Active Learning on Output Values (Neural Network) (GALO(NN)).



While the initial experimental results for the GALO(NN) algorithm did not meet our performance benchmarks, the approach itself holds considerable promise. The core innovation behind GALO(NN) offers a new perspective on solving complex problems. Although the current implementation struggles with efficiency in MLCA, these challenges are likely surmountable with further research and optimization. The foundational principles of GALO(NN) offer a fertile ground for future improvements and adaptations, making it a candidate worth further investigation. For example, the

GALO(NN) algorithm could be adapted to use a different activation functions or deep neural networks with more layers or neurons.

The claims of this paper for the transformation of optimization problems of GALI and GALO into ILP are supported by proofs of Estermann et al. (2023).

4 Results

4.1 Experiment setup

The experimental evaluations were conducted on a computing cluster running the Ubuntu 20.04 operating system. The cluster's hardware specifications include an NVIDIA GeForce RTX 2080 Ti graphics card and an Intel(R) Xeon(R) Silver 4116 CPU that operates at a base speed of 2.10 GHz, complemented by 12 GB of RAM.

The implementation was done using Python version 3.6.13. For solving MIP and GALI optimization problems, we used IBM ILOG CPLEX Optimization Studio V12.8.0 (Cplex 2009). We interfaced with CPLEX using the docplex 2.4.61. For training the GALO models, we opted for Gurobi version 9.1.2 due to its superior parallelization capabilities. Linear regression models were implemented using with Scikit-learn version 0.24.2 and PyTorch version 1.8.0.

4.2 Spectrum Auction Test Suite

Combinatorial Auction Test Suite (CATS) (Leyton-Brown, Pearson, and Shoham 2000) stands as one of the seminal combinatorial auctions test suites for testing real-world scenarios. Despite its significant contributions to many research endeavors, CATS does not offer specific features for simulating spectrum auctions as they were not designed to do so. Moreover, CATS lacks specific features for research applications, such as a possibility to assign a value for a specific bundle.

On the other hand, Spectrum Auction Test Suite (SATS) v0.8.1 (Weiss, Lubin, and Seuken 2017) serves as the specialized suite for the combinatorial auctions, introducing a lifelike model of spectrum auctions. A notable feature is the integrated MIP formulation for the WDP, eliminating the need for additional overhead for this study. The MIP formulation encodes the values for a substantial speed up in performance for solving WDP. The suite differentiates bidders as local, regional or national players. National players generally target all regions, while regional players tend to prioritize their regions. Local bidders exclusively value items in their region, meaning they have zero utility for items outside their region. Additionally, SATS incorporates domain-specific knowledge such as bandwidth, population per region, quality of service metrics.

Three auction models from SATS — GSVM, LSVM, and MRVM — have been employed for the experiments. GSVM encompasses 6 regional bidders and 1 national bidder competing for bidding over 18 items with no synergistic values between item combinations. LSVM involves 5 regional bidders and 1 national bidder bidding for 18 items. For this type of auction, there is complementary effect in having combinations of items. MRVM is the most intricate, compromising 4 regional bidders, 3 national bidders and 3 local bidders, vying for 98 items. MRVM also captures the geographic and frequency parameters to determine utility for a specific bidder per unique bundle.

4.3 Hyperparameters

Table 4.1: Neural Network Architectures. Numbers inside brackets describe the number of neurons in hidden layers. The input layer is the same size as the number of items in the auction and the output is 1.

Bidder Type	**GSVM**	**LSVM**	**MRVM**
Local Bidder	N/A	N/A	[16,16]
Regional Bidder	[32,32]	[32,32]	[16,16]
National Bidder	[10,10]	[10,10,10]	[16,16]

Weissteiner and Seuken (2020) conducted an in-depth study that proposed and rigorously tested various deep neural network architectures tailored for different types of auctions. Although the optimization of network architecture is undeniably crucial for performance improvement, the primary focus of this study diverges from this aspect. Instead, the research is oriented toward the effective integration of active learning algorithms with existing architectures. Therefore, for the purposes of this study, the configurations and architectures validated in previous studies (Estermann et al. 2023) serve as the experimental baseline. Neural network architecture for the NextQueries is described in Table 4.1. Each auction domain has almost the same MIP and Neural Network (NN) hyperparameters, which are listed in Table 4.2. The only difference is in the MRVM domain, where the Scaler is set to MinMaxScaler(copy=True, feature_range=(0, 500)) and Regularization Type are $L1$ and $L2$.

Table 4.2: Hyperparameter Settings Auctions

Parameter	
NN Hyperparameter Settings	
Regularization Type	L2
Regularization Factor	1×10^{-5}
Learning Rate	0.01
Dropout	True
Dropout Rate	0.05
MIP Hyperparameters Settings	
L	3000
Mip_bounds_tightening	IA
sample_weight	False
Scaler	False
warm_start	False
Seed Instances	0 - 20

Table 4.3: Active Learning Hyperparameter Settings. k is the number of bundles to be sampled with UF, c_0 is the number of initial bundles sampled by active learning strategy and c_e is the number of bundles to be sampled in each iteration by NextQueries.

Model	Sampling Technique	k	c_0	c_e
GSVM	UF	N/A	30	20
	GFFT	N/A	30	20
	GFFT (reverse)	20	10	20
	GALI	N/A	30	20
	GALO (linear)	20	10	20
LSVM	UF	N/A	40	10
	GFFT	N/A	40	10
	GFFT (reverse)	30	10	10
	GALI	N/A	40	10
	GALO (linear)	30	10	10
MRVM	UF	N/A	40	30
	GFFT (reverse)	30	10	30
	GALI	N/A	40	30
	GALO (linear)	30	10	30

4.4 Comparison

The study compares the efficiency and runtime of MLCA based on DNN. UF, GFFT, GFFT(reverse), GALI and GALO(linear) are used for the GSVM and LSVM domains. GFFT is not applied to the MRVM domain due to a substantial amount of time to the computational intensity required to handle the 2^{98} possible bundles. This would require $O(n^2)$ computations and $O(n^2)$ distance calculations. GALO(NN) has deemed not effective in our experiments. From now on we refer to GALO(linear) as GALO, if not stated otherwise. Increasing the number of threads improves the runtime of the GALO and GALI algorithms by a few minutes, as Gurobi allows for parallelization. The runtime of each active learning technique is included into the total runtime. Initial bundles, however, still influence the consequent iterations of auction models as they are used for the MLCA WDP MIP solver. Throughout the process of collecting experiments, we have experienced a few problems in parallelizing lots of instances at the same time. The reason behind it is that the MLCA solver was designed to use all available threads. This means that the solver cannot be run in parallel, as it will cause substantial decrease of runtime. To showcase runtime we use boxplots: a *boxplot* is a graphical representation used for depicting the distribution and spread of a dataset. The box itself represents the interquartile range (IQR), showing the 25th percentile (Q1) and the 75th percentile (Q3). The horizontal line within the box indicates the median of the data. *Whiskers* extend from the box to show the range of the data. Outliers are plotted individually.

4.4.1 GSVM

First, we will discuss the outcomes obtained from applying active learning techniques to the GSVM auction domain. We observe a notable increase in efficiency for both GFFT(reverse) and GALI. Given that GSVM is a relatively straightforward auction, the efficiency rate closely approaches 100%. When compared with UF, the efficiency demonstrates an overall increment of approximately 2%. In the case of GALO, a modest improvement of around 0.5% is observed. UF and GFFT exhibit nearly identical efficiencies, as illustrated in Figure 4.1. Despite the high computational complexity associated with the GFFT method, it does not yield a significant improvement in efficiency for the selected hyperparameters.

It is evident that there is an overall increase in runtime for all active learning algorithms when compared to UF. This can be attributed to the initial computational overhead required by these algorithms. Based on the results obtained from the sampling process, GALI exhibits a notably low runtime. This can be ascribed to the algorithm's low computational complexity, as it does not necessitate the calculation of distances between sets of bundles. Upon examination of the data, GALI emerges as a superior active learning method for the GSVM auction domain.

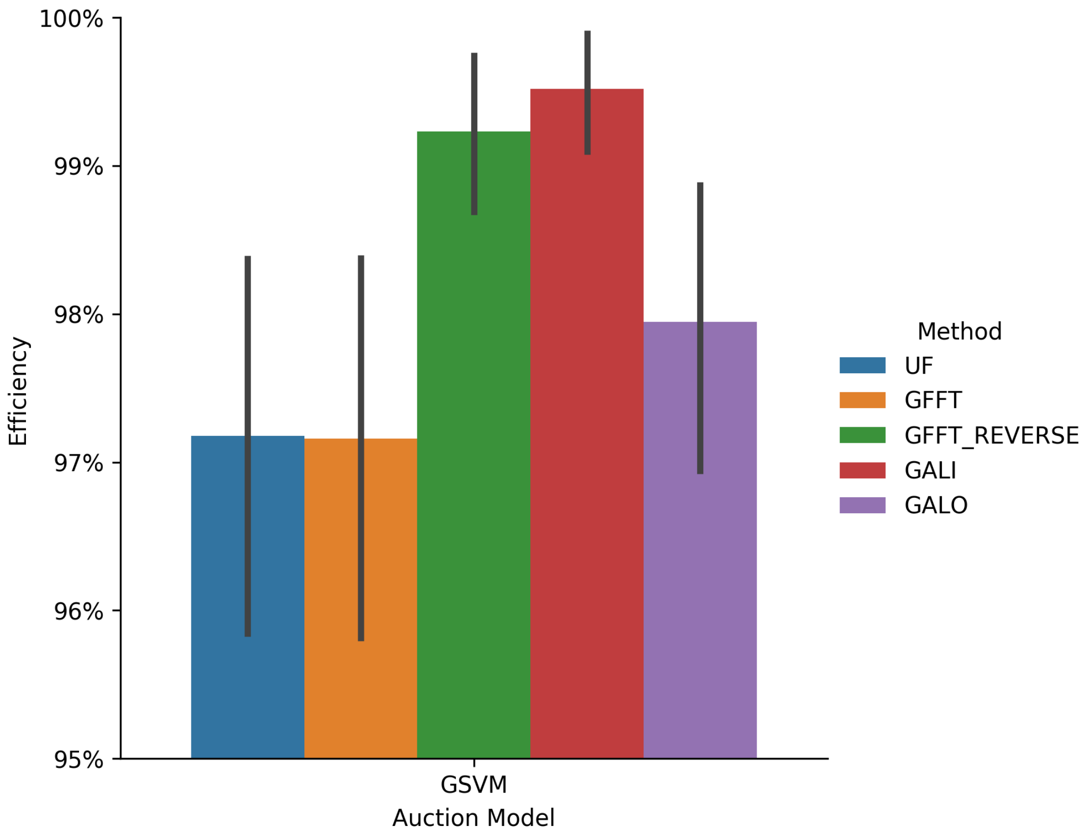


Figure 4.1: Efficiency of a GSVM auction

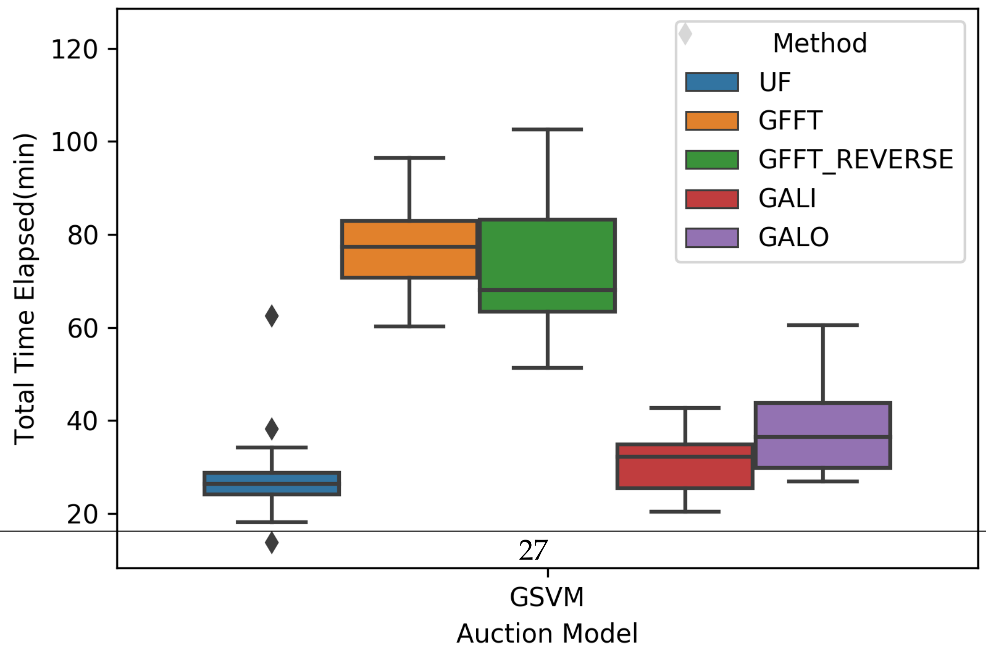


Figure 4.2: Runtime of a GSVM auction

Additionally, we discuss the efficiency per round for each active learning algorithm, a metric that facilitates the observation of efficiency rate progression for MLCA. This is crucial as the initial bundles exert influence on all subsequent auction mechanism iterations. Given the hyperparameters adopted in this paper, the GSVM auction domain undergoes three iterations. The initial efficiency of the first round is computed based on bundle profiles initialized by the active learning algorithms. Our observations indicate that GFFT(reverse) exhibits above-average efficiency in the initial round. The algorithm balances out the items not included in the initially sampled bundles and maintains high efficiency, only falling behind GALI in the final round. Although GALO commences with suboptimal efficiency, it experiences a significant uptick in subsequent rounds. Altering the model architecture to increase the number of rounds and employing GALO could yield promising results.

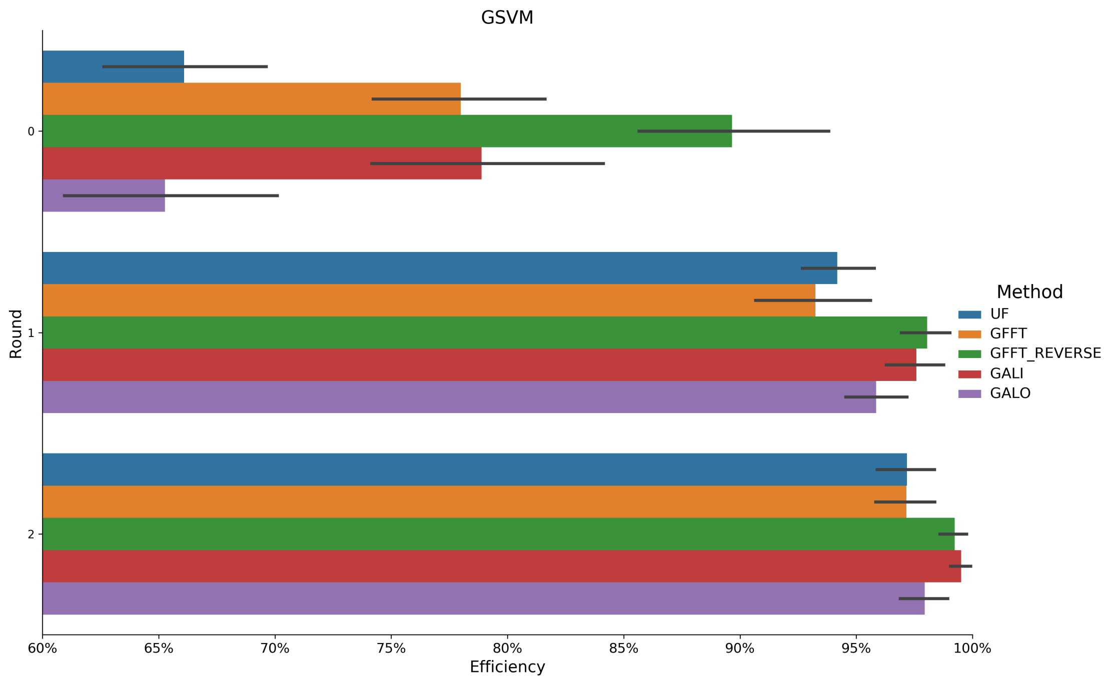


Figure 4.3: Efficiency per round of a GSVM auction

4.4.2 LSVM

In contrast to the GSVM auction domain, the LSVM domain presents greater complexity, resulting in divergent outcomes among the active learning algorithms. This auction undergoes only two rounds of iterations due to the presence of 40 initial instances and an additional 10 for the elicitation algorithm in MLCA. This characteristic makes the domain more reliant on the efficiency of the initial round and less so on subsequent rounds. As evident in Figure 4.4, all active learning algorithms surpass UF in performance. Both GFFT and GALI yield the highest efficiencies, with GFFT having a slight edge. GALO demonstrates a moderate efficiency increase, whereas GFFT(reverse) falls in the intermediate range. Interestingly, GFFT, despite its lackluster performance in the GSVM domain, exhibits a significant efficiency boost in the LSVM domain.

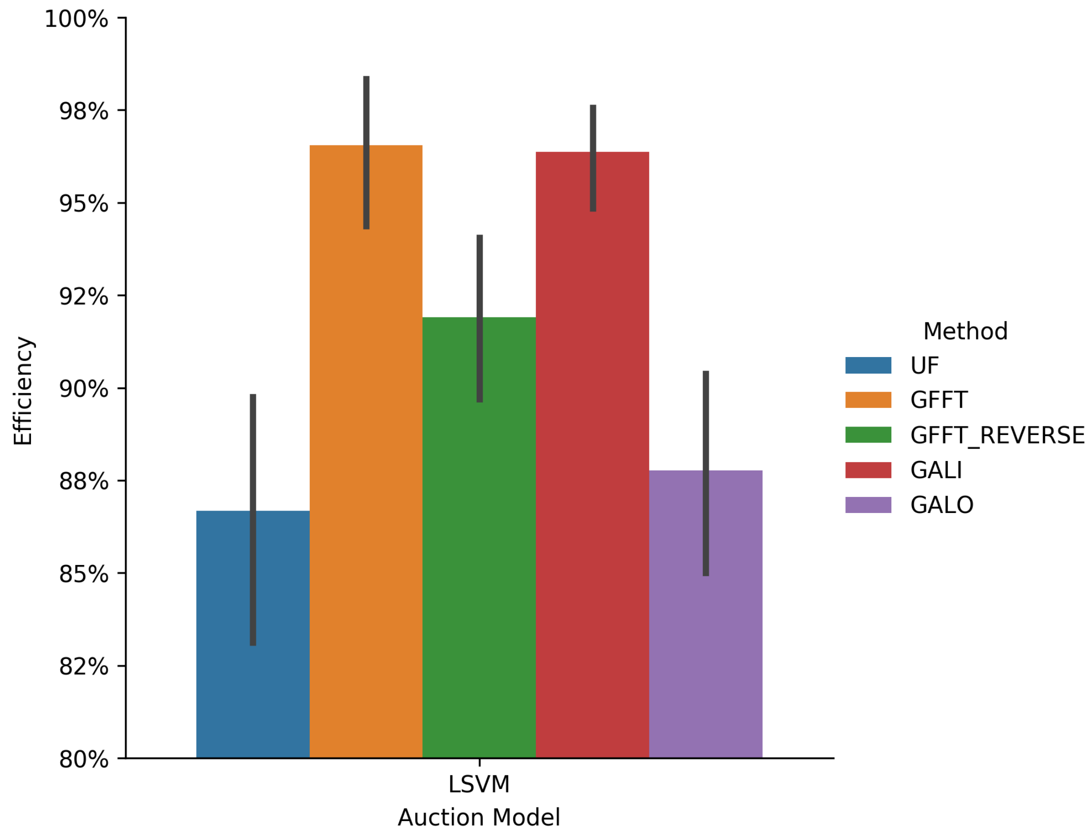


Figure 4.4: Efficiency of a LSVM auction

Figure 4.5 presents an empirical comparison of runtime across five distinct algorithms in the LSVM domain. The results are more uniform compared to those in the GSVM domain. All active learning algorithms exhibit slightly higher runtimes than the UF method. However, GFFT(reverse) emerges as the most time-efficient among the active learning algorithms, followed closely by the others. GALI presents some outliers that elevate its runtime, but its mean time remains lower than that of GALO. If runtime and efficiency over a limited number of rounds are priorities for the auction, GFFT(reverse) is the optimal choice. On the other hand, for auctions anticipated to run over numerous rounds, GALI or GALO are more suitable options, as evidenced by its performance in the GSVM domain.

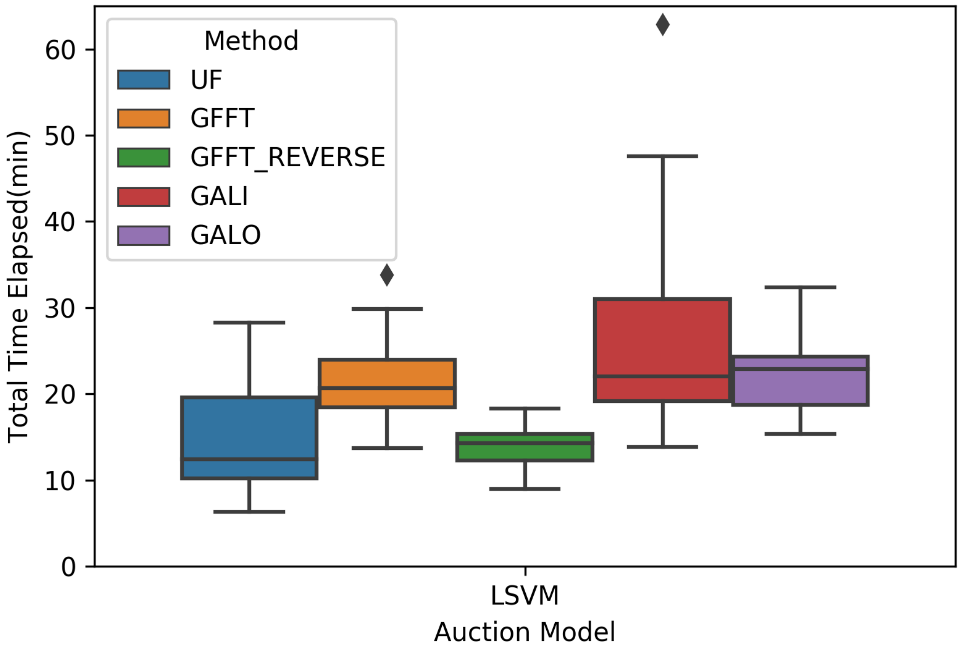


Figure 4.5: Runtime of a LSVM auction

Based on the empirical evaluations presented in Figure 4.6, GFFT(reverse) excels in performance during the initial round, surpassing all other algorithms. Conversely, both GALI and GFFT demonstrate their efficacy in the subsequent iteration, achieving the highest efficiency among the tested algorithms.

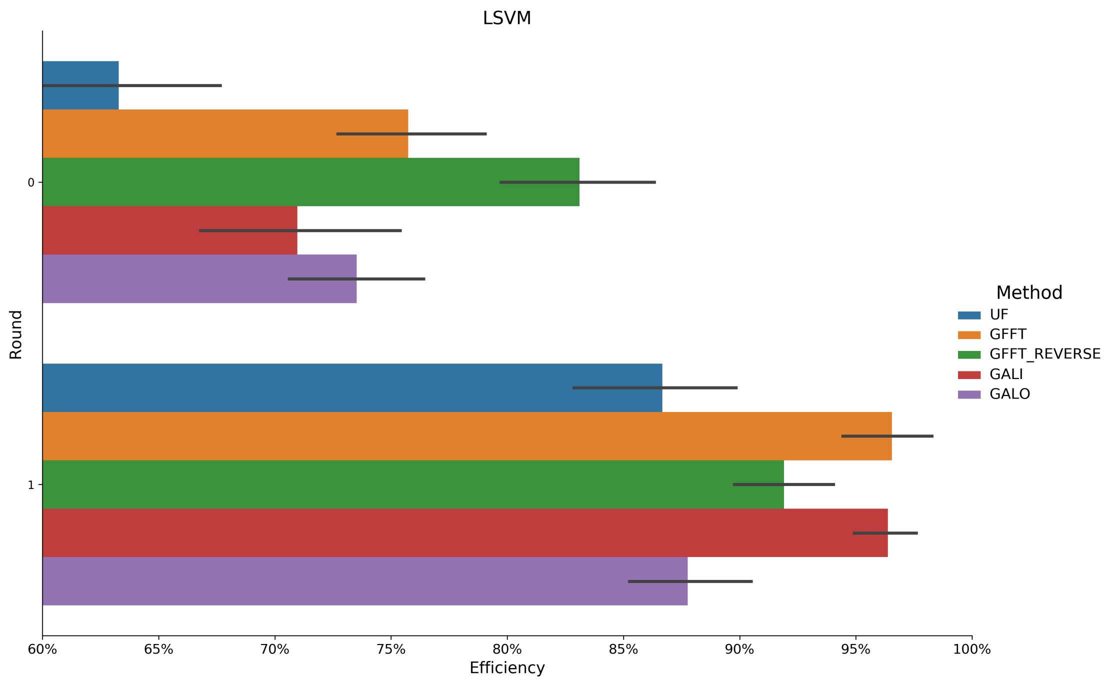


Figure 4.6: Efficiency per round of a LSVM auction

4.4.3 MRVM

For the largest domain under consideration, we evaluate both the efficiency and efficiency-per-round metrics. Due to the need for parallelization across numerous instances, the runtime results exhibit non-consistent behavior as we have used multiple parallelization tools for some instances. GFFT method is not used in this auction domain, as it has an exponential runtime. In the MRVM domain, the differences in efficiency are less pronounced; however, a modest improvement is still observable. GALO exhibits the highest efficiency, followed by GFFT(reverse) and GALI, while UF performs the worst. This can be attributed to the higher number of items and bidders in the MRVM domain, complicating the exploitation of geometric structures in output sets. One of the top-performing algorithms, GFFT(reverse), maintains their standings, consistent with outcomes from previous domains. Regarding efficiency per round, GFFT(reverse) starts strong, in line with its performance in other domains. GALI begins at a lower efficiency but manages to catch up in subsequent rounds. GALO exhibits a surprising result for a large auction domain. This can be attributed to the fact that the algorithm is not as reliant on the bundles as other algorithms. GALO is more dependent on the values of bundles, giving it an edge in domains, where preferences

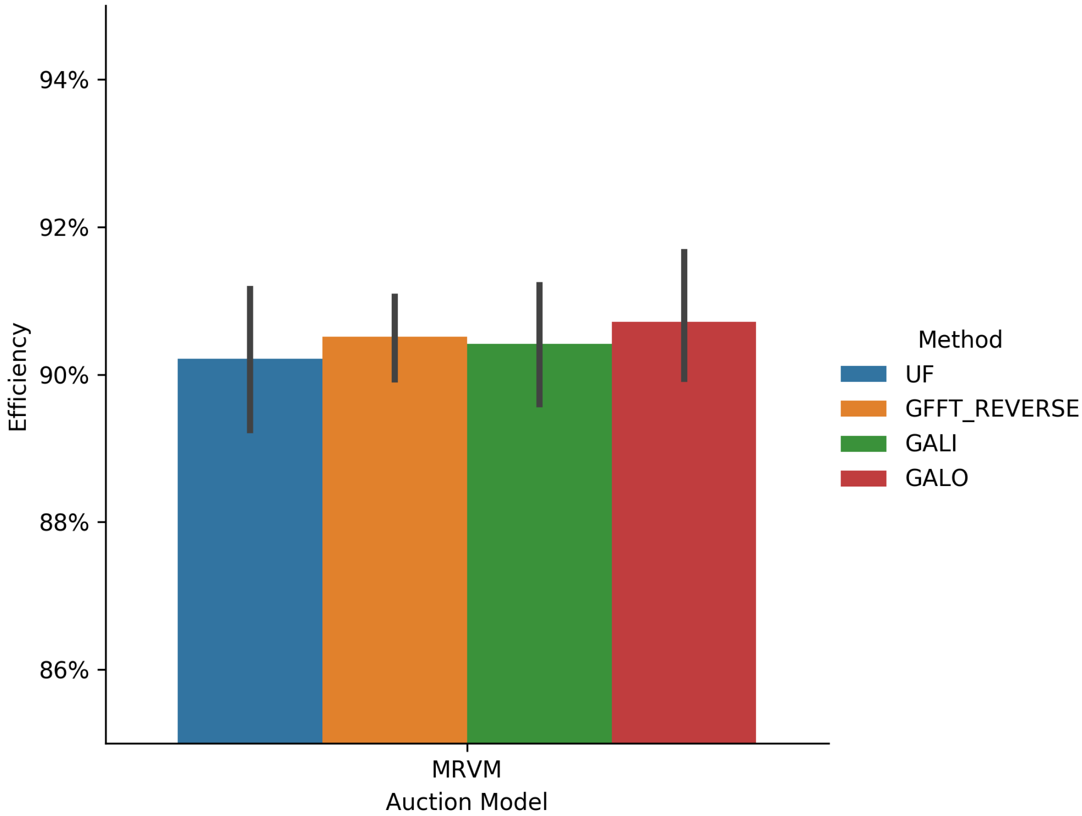


Figure 4.7: Efficiency of a MRVM auction

are hard to elicit. Other algorithms are using distance between groups of bundles to find the most informative bundles, which is not as effective in the MRVM domain.

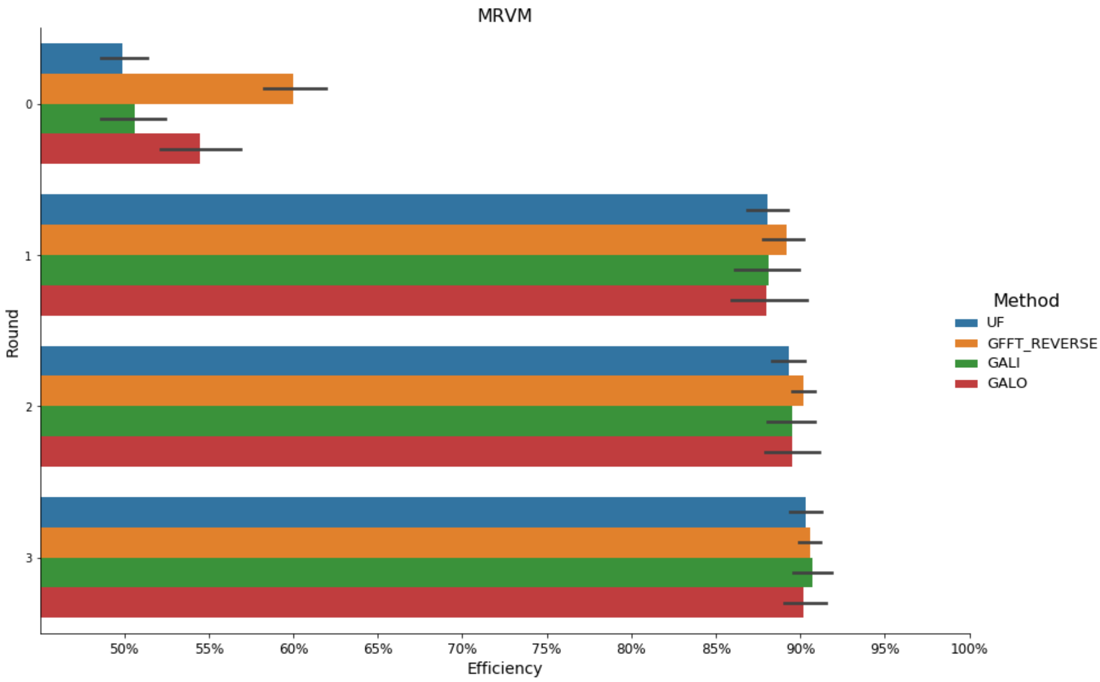


Figure 4.8: Efficiency per round of a MRVM auction

5 Conclusion

In this work, we have introduced a suite of active learning algorithms tailored for the MLCA framework and elucidated their impact on augmenting the efficiency of combinatorial auctions. Through experimental evaluation, we have furnished contributions to the field of combinatorial auction research. We improved the results of MLCA by using active learning algorithms. On the technical front, our implementation serves as a robust foundation for active learning methods aimed at enhancing auction efficiency.

Future Prospects and Extensions: Our research opens up multiple avenues for further inquiry and refinement in the active learning landscape:

- Given the extensible nature of our implementation, the platform is amenable to the integration and evaluation of a broad array of active learning techniques tailored for auction labeling tasks.
- Algorithms such as GALO(NN) and GALO(linear) hold significant promise. Future work could delve into varying architectures of neural networks and types of linear regressions to optimize these algorithms further.
- The hyperparameter space remains largely unexplored, offering opportunities for in-depth investigation, particularly concerning the sample size for each active learning algorithm.

Our findings lay the groundwork for subsequent research efforts, offering a robust methodology for enhancing the efficiency of complex auction mechanisms.

Abbreviations

MRVM Multi-Region Value Model

LSVM Linearly Correlated Value Model

GSVM Generalized Value Model

GALI Greedy Active Learning on Input Values

GALO Greedy Active Learning on Output Values

GALO(linear) Greedy Active Learning on Output Values (linear)

GALO(NN) Greedy Active Learning on Output Values (Neural Network)

GFFT Greedy Farthest-First Traversal

GFFT(reverse) Greedy Farthest-First Traversal (reverse)

UF Uniform Sampling

MIP Mixed-Integer Programming

WDP Winner Determination Problem

NN Neural Network

VCG Vickrey-Clarke-Groves auction

MLCA Machine Learning Combinatorial Auction

SVR Support Vector Regression

DNN Deep Neural Network

ILP Integer Linear Programming

CATS Combinatorial Auction Test Suite

SATS Spectrum Auction Test Suite

List of Algorithms

List of Figures

List of Tables

Bibliography

Ausubel, Lawrence M., and Oleg Baranov. 2017. A Practical Guide to the Combinatorial Clock Auction. *The Economic Journal* 127 (605): F334–F350. ISSN: 0013-0133, accessed August 24, 2023. https://www.jstor.org/stable/45022990.

Bichler, Martin, Pasha Shabalin, and Jürgen Wolf. 2013. Do core-selecting Combinatorial Clock Auctions always lead to high efficiency? An experimental analysis of spectrum auction designs. *Experimental Economics* 16 (4): 511–545. ISSN: 1386-4157, accessed September 10, 2023. https://econpapers.repec.org/article/kapexpeco/v_3a16_3ay_3a2013_3ai_3a4_3ap_3a511-545.htm.

Blum, Avrim, Jeffrey Jackson, Tuomas Sandholm, and Martin Zinkevich. 2003. Preference Elicitation and Query Learning. *The Journal of Machine Learning Research* 5 (July): 649–667. ISSN: 978-3-540-40720-1. https://doi.org/10.1007/978-3-540-45167-9_3.

Brero, Gianluca, and Sébastien Lahaie. 2018. *A bayesian clearing mechanism for combinatorial auctions.* arXiv: 1712.05291 [cs.GT].

Brero, Gianluca, Sébastien Lahaie, and Sven Seuken. 2019. *Fast iterative combinatorial auctions via bayesian learning.* arXiv: 1809.05340 [cs.GT].

Brero, Gianluca, Benjamin Lubin, and Sven Seuken. 2021. *Machine learning-powered iterative combinatorial auctions.* arXiv: 1911.08042 [cs.GT].

Brewer, Paul J. 1999. Decentralized computation procurement and computational robustness in a smart market [in en]. *Economic Theory* 13, no. 1 (January): 41–92. ISSN: 1432-0479, accessed August 24, 2023. https://doi.org/10.1007/s001990050242. https://doi.org/10.1007/s001990050242.

Caplice, Christopher George. 1996. An optimization based bidding process : a new framework for shipper-carrier relationships [in eng]. PhD diss., Massachusetts Institute of Technology. Accessed August 24, 2023. https://dspace.mit.edu/handle/1721.1/10846.

Clarke, Edward. 1971. Multipart pricing of public goods. *Public Choice* 11 (1): 17–33. ISSN: 0048-5829, accessed August 30, 2023. https://econpapers.repec.org/article/kappubcho/v_3a11_3ay_3a1971_3ai_3a1_3ap_3a17-33.htm.

Conitzer, Vincent, and Tuomas Sandholm. 2002. *Complexity of mechanism design.* arXiv: cs/0205075 [cs.GT].

———. 2004. Self-interested automated mechanism design and implications for optimal combinatorial auctions. In *Proceedings of the 5th ACM conference on Electronic commerce,* 132–141. EC ’04. New York, NY, USA: Association for Computing Machinery, May. ISBN: 9781581137712, accessed September 13, 2023. https://doi.org/10.1145/988772.988793. https://doi.org/10.1145/988772.988793.

Cplex, IBM ILOG. 2009. V12. 1: user’s manual for cplex. *International Business Machines Corporation* 46 (53): 157.

Duetting, Paul, Zhe Feng, Harikrishna Narasimhan, David Parkes, and Sai Srivatsa Ravindranath. 2019. Optimal Auctions through Deep Learning [in en], 1706–1715. PMLR, May. Accessed September 13, 2023. https://proceedings.mlr.press/v97/duetting19a.html.

Estermann, Benjamin, Stefan Kramer, Roger Wattenhofer, and Ye Wang. 2023. Deep Learning-Powered Iterative Combinatorial Auctions with Active Learning. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems,* 2919–2921. AAMAS ’23. Richland, SC: International Foundation for Autonomous Agents / Multiagent Systems, May. ISBN: 9781450394321, accessed August 24, 2023.

Groves, Theodore. 1973. Incentives in Teams. *Econometrica* 41 (4): 617–631. ISSN: 0012-9682, accessed August 30, 2023. https://doi.org/10.2307/1914085. https://www.jstor.org/stable/1914085.

Hershberger, J., and S. Suri. 2001. Vickrey prices and shortest paths: what is an edge worth? In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science,* 252–259. ISSN: 1552-5244. October. https://doi.org/10.1109/SFCS.2001.959899.

Lahaie, Sébastien, and Benjamin Lubin. 2019. Adaptive-Price Combinatorial Auctions. In *Proceedings of the 2019 ACM Conference on Economics and Computation,* 749–750. EC ’19. New York, NY, USA: Association for Computing Machinery, June. ISBN: 9781450367929, accessed September 13, 2023. https://doi.org/10.1145/3328526.3329615. https://doi.org/10.1145/3328526.3329615.

Lahaie, Sébastien, and David C. Parkes. 2004. Applying learning algorithms to preference elicitation. In *Proceedings 5th ACM Conference on Electronic Commerce (EC-2004), New York, NY, USA, May 17-20, 2004,* edited by Jack S. Breese, Joan Feigenbaum, and Margo I. Seltzer, 180–188. ACM. https://doi.org/10.1145/988772.988800.

Leyton-Brown, Kevin, Mark Pearson, and Yoav Shoham. 2000. Towards a universal test suite for combinatorial auction algorithms. In *Proceedings of the 2nd ACM conference on Electronic commerce,* 66–76. EC '00. New York, NY, USA: Association for Computing Machinery, October. ISBN: 9781581132724, accessed August 23, 2023. https://doi.org/10.1145/352871.352879. https://dl.acm.org/doi/10.1145/352871.352879.

Mishra, Debasis, and David C. Parkes. 2007. Ascending price Vickrey auctions for general valuations. *Journal of Economic Theory* 132, no. 1 (January): 335–366. ISSN: 0022-0531, accessed August 30, 2023. https://doi.org/10.1016/j.jet.2005.09.004. https://www.sciencedirect.com/science/article/pii/S0022053105001985.

Park, Sung Ho, and Seoung Bum Kim. 2020. Robust expected model change for active learning in regression [in en]. *Applied Intelligence* 50, no. 2 (February): 296–313. ISSN: 0924-669X, 1573-7497, accessed August 24, 2023. https://doi.org/10.1007/s10489-019-01519-z. http://link.springer.com/10.1007/s10489-019-01519-z.

Rassenti, S. J., V. L. Smith, and R. L. Bulfin. 1982. A Combinatorial Auction Mechanism for Airport Time Slot Allocation. *The Bell Journal of Economics* 13 (2): 402–417. ISSN: 0361-915X, accessed August 24, 2023. https://doi.org/10.2307/3003463. https://www.jstor.org/stable/3003463.

Scheffel, Tobias, Georg Ziegler, Martin Bichler, and Riko Jacob. 2012. On the Impact of Cognitive Limits in Combinatorial Auctions: An Experimental Study in the Context of Spectrum Auction Design. *Experimental Economics* 15 (December). https://doi.org/10.1007/s10683-012-9321-0.

Taylor, Gregory. 2013. *Oil in the Ether: A Critical History of Spectrum Auctions in Canada* [in en]. Technical report 2473272. Rochester, NY, January. Accessed August 24, 2023. https://doi.org/10.2139/ssrn.2473272. https://papers.ssrn.com/abstract=2473272.

Vickrey, William. 1961. Counterspeculation, Auctions, and Competitive Sealed Tenders. *The Journal of Finance* 16 (1): 8–37. ISSN: 0022-1082, accessed August 30, 2023. https://doi.org/10.2307/2977633. https://www.jstor.org/stable/2977633.

Vries, Sven de, James Schummer, and Rakesh V. Vohra. 2007. On ascending Vickrey auctions for heterogeneous objects. *Journal of Economic Theory* 132 (1): 95–118. ISSN: 0022-0531, accessed August 30, 2023. https://econpapers.repec.org/article/eeejetheo/v_3a132_3ay_3a2007_3ai_3a1_3ap_3a95-118.htm.

Weiss, Michael, Benjamin Lubin, and Sven Seuken. 2017. SATS: A Universal Spectrum Auction Test Suite. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems,* 51–59. AAMAS '17. Richland, SC: International Foundation for Autonomous Agents / Multiagent Systems, May. Accessed August 23, 2023.

Weissteiner, Jakob, Jakob Heiss, Julien Siems, and Sven Seuken. 2023. Bayesian Optimization-Based Combinatorial Assignment [in en]. *Proceedings of the AAAI Conference on Artificial Intelligence* 37, no. 5 (June): 5858–5866. ISSN: 2374-3468, accessed August 24, 2023. https://doi.org/10.1609/aaai.v37i5.25726. https://ojs.aaai.org/index.php/AAAI/article/view/25726.

Weissteiner, Jakob, and Sven Seuken. 2020. Deep Learning-powered Iterative Combinatorial Auctions. ArXiv:1907.05771 [cs], *Proceedings of the AAAI Conference on Artificial Intelligence* 34, no. 02 (April): 2284–2293. ISSN: 2374-3468, 2159-5399, accessed August 23, 2023. https://doi.org/10.1609/aaai.v34i02.5606. http://arxiv.org/abs/1907.05771.

Weissteiner, Jakob, Chris Wendler, Sven Seuken, Ben Lubin, and Markus Püschel. 2022. Fourier Analysis-based Iterative Combinatorial Auctions. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence,* 549–556. ArXiv:2009.10749 [cs]. July. Accessed August 24, 2023. https://doi.org/10.24963/ijcai.2022/78. http://arxiv.org/abs/2009.10749.

Wu, Dongrui. 2018. *Pool-based sequential active learning for regression.* arXiv: 1805.04735 [cs.LG].

Wu, Dongrui, Chin-Teng Lin, and Jian Huang. 2018. *Active learning for regression using greedy sampling.* arXiv: 1808.04245 [cs.LG].

Yu, Hwanjo, and Sungchul Kim. 2010. Passive Sampling for Regression. In *2010 IEEE International Conference on Data Mining,* 1151–1156. ISSN: 2374-8486. December. https://doi.org/10.1109/ICDM.2010.9.