Advancing Differentiable Mechanism Design: Neural Architectures for Combinatorial Auctions

Anonymous Author(s)
Affiliation
Address
email

Abstract

Designing optimal auctions has major real-world impact, but remains notoriously difficult to solve analytically, often intractable in strategic or high-dimensional settings. Neural networks have recently approximated optimal mechanisms in multi-item auctions, yet extending them to combinatorial auctions (CAs) is harder. The main challenge is enforcing combinatorial feasibility, as bundle allocations involve non-convex, binary, and overlapping constraints beyond standard neural architectures. This paper introduces a novel combinatorial constraint enforcement technique for deep learning, applied to a fully connected network (CANet) and a transformer-based network (CAFormer). We also present CAGraph, a graph attention network (GAT) model that formulates the winner determination problem as set packing and captures interdependencies between bidders and bundles. Our approach yields three key results. First, our models consistently outperform heuristic baselines and prior learning-based methods—including RegretNet—across diverse synthetic combinatorial auction settings. Second, in real-world airport slot auctions, they maintain low regret while flexibly balancing welfare and revenue. Third, in a cyber defense case study, a defensive agent uses the auction's output to allocate limited resources to vulnerable network hosts, demonstrating the practical versatility of our framework. Together, these results demonstrate the flexibility, scalability, and effectiveness of differentiable, constraint-aware neural architectures for combinatorial mechanism design.

1 Introduction

Mechanism design, a field of economics, focuses on creating incentives and interaction rules among self-interested agents to achieve specific objectives for the group. It differs from traditional game theory by starting with a desired outcome and designing the rules so that agents acting in their self-interest naturally lead to that outcome. A key application of mechanism design is to create auction rules, which play a significant role in economic activities such as the fine art market, advertising on search engines or e-commerce platforms [6, 21].

The Vickrey-Clarke-Groves (VCG) mechanism is optimal and strategy-proof when the goal is to maximize total bidder welfare [47, 11, 25]. However, revenue maximization poses a greater challenge. [33] resolved the revenue-maximizing strategy-proof auction for a single item, later extended by [32] to multiple copies of a single item. The general revenue-maximizing auction for multiple items remains unsolved decades later, with only specific two-item cases addressed [4, 5].

The lack of theoretical progress in revenue-maximizing multi-item auctions inspired the development of Automated Mechanism Design (AMD) [42]. AMD uses computational methods to tailor mechanisms to specific problem instances, with the use of heuristic approaches [41, 43], but suffers from the curse of dimensionality in large-scale settings. More recently, leveraging the power of deep

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

learning, differentiable economics uses deep neural networks as function approximators to learn optimal auctions. This approach, introduced with RegretNet [20], offers a scalable alternative to traditional AMD methods.

Since its introduction, RegretNet has inspired a variety of extensions and improvements [35, 34, 26]. The complexity of multi-item auctions is further amplified in combinatorial auctions, where bidders express valuations for bundles rather than individual items, capturing complementarities or substitutabilities. Unlike traditional auctions, the interdependencies between items in combinatorial auctions introduce additional challenges. One such challenge is winner determination, a problem that is well known to be NP-hard, which involves allocating bundles to agents while ensuring that none of the bundles overlap. Although the later part of the RegretNet study discusses the 2-item, 2-bidder auction in a combinatorial setting [20], the problem is still wide open, primarily due to the lack of methodologies to effectively constrain the allocation space to satisfy combinatorial feasibility.

This paper builds on recent advances in auction theory and machine learning to design combinatorial revenue-optimal auctions. It makes the following contributions:

1. It provides an empirical method to find **near-optimal solutions to revenue-maximizing CA**, addressing problems where analytical solutions are not available. This method is not limited by assumptions about allowable bundle structures or bidder valuations. Similar to the previous work in differentiable economics [20], the method is approximately strategy-proof.
2. Unlike previous CA studies, [43, 45, 27], our models identify **randomized (lottery) mechanisms**, extending the RegretNet family to CA problems. It is well known that randomization can increase revenue in CAs. Our empirical experiments show substantially higher performance than deterministic mechanisms.
3. It introduces new architectures: **CANet** and **CAFormer**, extending prior designs from non-combinatorial auctions to a CA setting, and **CAGraph**, a novel GAT-based model learning from both valuation and the relational structures of bundles. While CANet is sensitive to the order and structure of the input, CAFormer is permutation-equivariant, offering better generalization in scenarios with varying input configurations. CAGraph, our new architecture, however, consistently outperformed the benchmarks, demonstrating the potential of using GNNs to handle combinatorial combinatorial problems. In addition, it implements several techniques to improve stability in training and tackle the vanishing gradient problem.
4. It provides an example of successful use of **gradient based methods** for **constrained optimization problems**, which are known to be challenging. In our particular case, the constraints are related to combinatorial feasibility. This approach can be extended to other differentiable combinatorial optimization problems.
5. To our knowledge, this is the first work to evaluate differentiable auction mechanisms in **real-world case studies**, including airport slot allocation and cyber defense, demonstrating their practical versatility and policy relevance.

2 Problem Statement

We study a combinatorial auction (CA) with one seller, n bidders, and m items. Each bidder $i \in N$ has a private valuation v_i over bundles $S \subseteq M$, with $K = 2^m - 1$ possible non-empty bundles. Valuations v_i are drawn from distributions F_i, which may be symmetric ($F_i = F$) or asymmetric, and utilities are quasi-linear:

$$u_i(v_i; b) = v_i(g_i(b)) - p_i(b),$$

where g_i is the allocation and p_i the payment.

A mechanism is DSIC if truth-telling is optimal:

$$u_i(v_i; (v_i, b_{-i})) \geq u_i(v_i; (b_i, b_{-i})) \quad \forall i, b_i, \tag{1}$$

and ex-post IR if

$$u_i(v_i; (v_i, b_{-i})) \geq 0 \quad \forall i. \tag{2}$$

Violations of DSIC are measured via expected ex-post regret:

$$\text{rgt}_i = \mathbb{E}\left[\max_{v_i'} u_i(v_i; (v_i', v_{-i})) - u_i(v_i; (v_i, v_{-i}))\right].$$

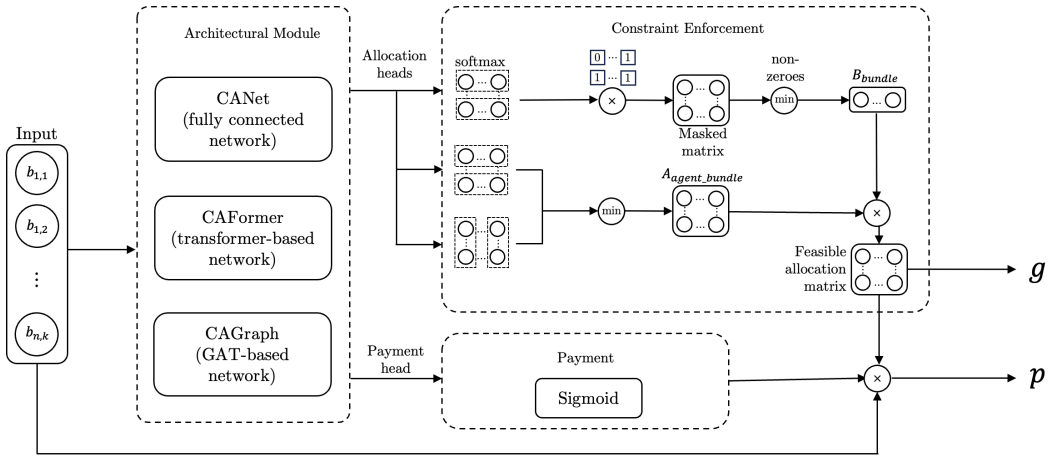

Figure 1: Mechanism overview. Bids flow through one of three architecture modules CANet, CAFormer or CAGraph, then a shared constraint-enforcement block yields a feasible allocation $\mathbf{Z}$ and a pricing network ensures IR, producing (g, p). Detailed schematics for each architecture appear in Appendix D.1–D.3.

Combinatorial feasibility. Let z_{iS} indicate whether bidder i receives bundle S. We require

$$\sum_{i \in N} \sum_{S \subseteq M:\, j \in S} z_{iS} \leq 1 \quad \forall j \in M, \qquad \sum_{S \in K} z_{iS} \leq 1 \quad \forall i \in N, \tag{3}$$

with the integrality constraint $z_{iS} \in \{0, 1\}$ in the discrete problem. For learning, we use the standard LP relaxation $z_{iS} \in [0, 1]$ and interpret z as *ex-ante* allocation probabilities. (Exact Birkhoff–von Neumann decompositions apply only in assignment or disjoint-bundle settings; implementability for general CAs is discussed in Appendix H.)

The auctioneer's goal is to maximize expected revenue,

$$\text{rev} = \sum_{i \in N} p_i(v),$$

subject to DSIC, IR, and feasibility.

We build on RegretNet [20], which encodes allocation and payment rules within neural networks and optimizes revenue subject to low regret. While the original work extends to combinatorial auctions, it is limited to a two-bidder, two-item setting. We specifically introduce a computational technique to enforce hard combinatorial constraints in broader combinatorial auction environments.

3 Proposed Method

After formulating CAs as a learning problem, we will propose models that facilitate end-to-end learning through gradient-based optimization techniques.

To ensure feasibility, we factorize the allocation matrix $\mathbf{Z} \in \mathbb{R}^{n \times k}$ as

$$\mathbf{Z} = \mathbf{B}^{\text{bundle}} \cdot \mathbf{A}^{\text{agent-bundle}},$$

corresponding to a two-step process: (i) items are assigned to bundles, and (ii) bundles are assigned to bidders.

The item-to-bundle matrix $\mathbf{B}^{\text{bundle}}$ is constructed from network outputs and the item–bundle incidence matrix, using per-item softmax normalization followed by a minimum operator to ensure that each bundle's probability respects item capacities. The bundle-to-agent matrix $\mathbf{A}^{\text{agent-bundle}}$ is obtained by combining agent- and bundle-wise softmax outputs:

$$\mathbf{A}^{\text{agent-bundle}} = \min(\text{softmax}_N(\mathbf{A}), \text{softmax}_K(\mathbf{A}')) \,.$$

The final allocation is then

$$z_{iS} = B_S \cdot A_{iS}^{\text{agent-bundle}},$$

which satisfies the allocation constraints from Section 2.

A full derivation of $\mathbf{B}^{\text{bundle}}$ and the proof of combinatorially feasibility are provided in Appendix C.

To instantiate our approach, we design three architectures. CANet, motivated by RegretNet, is a fully connected network with two modules: a deep allocation network that enforces feasibility constraints and a deep pricing network that ensures individual rationality. Together, these map bids to feasible allocations and corresponding payments (details in Appendix D.1). To enhance expressivity and capture permutation-equivariant structure in auctions, CAFormer combines exchangeable layers with self-attention blocks, enabling context-aware allocation decisions while remaining robust to bidder and bundle permutations (Appendix D.2). Finally, CAGraph frames winner determination as a set-packing task over a conflict graph; through graph attention layers, it learns bidder–bundle dependencies and suppresses overlapping allocations (Appendix D.3).

Our training follows the adversarial setup of [20], with an inner utility maximization (6) and an outer objective balancing revenue and regret (5). To stabilize this process, which is often sensitive to hyperparameters, we normalize task weights so that $w_{rev} + w_{rgt} = 1$ and apply a logarithmic transformation to the revenue term to prevent unbounded growth. The resulting outer loss is

$$\mathcal{L}_{\text{outer}} = -w_{rev}\ \log(1 + \text{rev}) + w_{rgt}\,\text{rgt} \tag{4}$$

where $w_{rev}, w_{rgt} \in [0, 1]$ balance the trade-off. Following [26], we also employ a regret budget, gradually reducing tolerated violations of DSIC during training. Weight updates for w_{rgt} are adapted dynamically based on current regret and revenue, using an Adam-style rule. Full update formulas and pseudocode are provided in Appendix E. For the revenue–welfare trade-off scenario in Section 4, we scalarize the performance term with a weight λ, using $\text{obj}_\lambda = \lambda\,\text{rev} + (1 - \lambda)\,\text{wel}$ and substituting obj_λ for rev in $\mathcal{L}_{\text{outer}}$ (the regret term is unchanged).

4 Experimental Results

4.1 Synthetic Data

Set up Our framework is implemented in PyTorch and trained for 50,000 or 100,000 outer optimization iterations, depending on the problem size. All networks used Glorot uniform initialization [23] and tanh activations. We sampled 640,000 valuation profiles offline for training and 10,000 profiles online for testing. Each outer optimization update included 50 inner steps during training and 1,000 during validation. Typical hyperparameters include: tanh scaler $\rho = 2$, softmax temperature $\theta \in \{10, 15, 25\}$, network learning rates $\{0.0004, 0.0007\}$, regret learning rate $\gamma \in \{0.01, 0.02\}$ and the target regret-revenue adjustment factor $\alpha \in \{0.5, 1\}$. CANet's allocation and pricing networks use $\{3, 6\}$ layers with 100 hidden nodes each, while CAFormer has a single attention layer with 2 heads and 64 hidden features. We initialized $w_{\text{rgt}} = 1$, exponentially annealling regret targets from $\bar{\text{rgt}}_{\text{start}} = 0.05$ to $\bar{\text{rgt}}_{\text{end}} \in \{0.0008, 0.001, 0.002, 0.003\}$ in $2/3$ of the training iterationw. Experiments are conducted on three scales: 2 bidders, 2 items (3 bundles); 2 bidders, 3 items (7 bundles); and 2 bidders, 5 items (31 bundles).

Baselines We compare against the VCG mechanism [47, 11, 25], with allocation found with an Interger Programming (IP) solver, RegretNet, Affine Maximizer Auction trained via grid search (AMA and VVCA), and local search methods (BLAMA, ABAMA, BBBVVCA) [43]. We re-implement VCG mechanism and the search algorithms [43]. The search algorithms are trained on 100 samples for 10 different random seeds, and evaluated on 1,000,000 samples.

Performance in combinatorial setting We compare the results in combinatorial setting. Let v_{ij}^{item} be the valuation of of bidder i for item j, individually. The valuation of bidder i for bundle S is drawn as $v_{iS} = \sum_{j \in S} v_{ij}^{\text{item}} + c_{iS}$ with $c_{iS} \sim U[-1, 1]$. For symmetric settings (B), $v_{ij}^{\text{item}} \sim U[1, 2], \forall i \in N$. For asymmetric settings (C), $v_{1j}^{\text{item}} \sim U[1, 2]$ and $v_{2j}^{\text{item}} \sim U[1, 5]$. Results are averaged over three runs; standard deviations are typically ≤ 0.1 for revenue, ≤ 0.001 for regret.

The results in Table 1 show that our models outperform heuristic benchmarks and RegretNet in all settings. As we specify the regret target as a proportion of revenue, the regret is higher in larger-scale

	Symmetric (B)			Asymmetric (C)		
Method	2×2	2×3	2×5	2×2	2×3	2×5
VCG	2.405 / 0	3.537 / 0	5.838 / 0	2.847 / 0	4.239 / 0	6.987 / 0
AMA	2.760 / 0	— / —	— / —	4.240 / 0	— / —	— / —
VVCA	2.770 / 0	— / —	— / —	4.240 / 0	— / —	— / —
BLAMA	2.630 / 0	4.125 / 0	7.280 / 0	4.080 / 0	4.812 / 0	8.164 / 0
ABAMA	2.630 / 0	4.166 / 0	7.145 / 0	4.010 / 0	5.916 / 0	11.140 / 0
BBBVVCA	2.620 / 0	4.105 / 0	7.156 / 0	4.010 / 0	5.898 / 0	11.135 / 0
RegretNet	2.871 / 1e-3	— / —	— / —	4.270 / 1e-3	— / —	— / —
CANet	2.904 / 1e-3	4.369 / 2e-3	7.304 / 7e-3	4.285 / 1e-3	6.556 / 1e-3	11.393 / 2e-3
CAFormer	2.919 / 1e-3	4.388 / 2e-3	7.318 / 3e-3	4.403 / 2e-3	6.693 / 1e-3	11.534 / 4e-3
CAGraph	**3.202** / 1e-3	**4.710** / 1e-3	**7.774** / 5e-3	**4.844** / 3e-3	**7.204** / 3e-3	**12.123** / 6e-3

Table 1: Combinatorial results (per cell: *Revenue* / *Regret*). Grid search methods (AMA and VVCA) become computationally infeasible in higher dimensions. RegretNet results are limited to 2×2 cases. CANet and CAFormer beat the benchmarks in all settings. All CANet, CAFormer and CAGraph achieve negligible regret. But CAGraph consistently outperforms the others in revenue. As a revenue upper bound for these instances (not a target for a truthful revenue mechanism), the optimal welfare-maximizing first-price outcome-computed via an IP solver for maximum independent set and pay bids-yields revenues of 3.548, 5.467, and 9.243 in the symmetric settings (2×2, 2×3, 2×5), and 6.184, 9.296, and 15.573 in the asymmetric settings, respectively.

settings. Both CAFormer and CANet achieve negligible regret in all experiments. However, CAGraph, with a greater expressivity especially when incorporating interdependencies, consistently outperforms CANet and CAFormer in revenue.

Performance in non-combinatorial settings. For completeness, we also evaluate additive valuation settings, where bundle values equal the sum of item values. CANet and CAFormer achieve comparable revenue to RegretNet and RegretFormer, both of which outperform heuristic designs, while CAGraph outperform the others. Full results are reported in Appendix G.1.

4.2 Case Study 1: Airport Slot Allocation

Airport slot divestitures (e.g., during mergers or under the 80% rule) are a natural application of combinatorial auctions: airlines derive value from bundles of slots that support schedules and connectivity. Currently, these slots are reallocated by Random Serial Dictatorship (RSD). We model slot allocation as a sealed-bid CA where valuations are derived from profit-maximizing schedules. This framework allows us to study explicit trade-offs between welfare and revenue, an important policy concern since efficiency ensures fair access while revenue compensates divesting airlines. We ground our analysis in the 2011 FAA auction at Reagan National Airport (DCA). Details of the scheduling formulation, datasets, and regulatory context are deferred to Appendix F.1.

Our mechanism exposes a revenue–welfare trade-off via a weight λ in the outer objective (4): λ=0 targets welfare, λ=1 targets revenue, and λ=0.5 balances both. Note that we report *ex-ante* welfare and revenue under the fractional allocation $Z \in [0,1]^{n\times k}$ (i.e., expectations under the randomized allocation), by contrast, VCG is deterministic and welfare-optimal among DSIC mechanisms; our learned mechanisms are approximately truthful (low regret) and optimize the λ-scalarized objective.

4.3 Case Study 2: Cyber Network Defense Auction Design

We illustrate the versatility of our framework in a cybersecurity setting, where defenders must allocate limited actions (e.g., Analyze, Remove, Restore) across vulnerable hosts. This problem is inherently combinatorial, as the value of actions depends on host type and synergies between defenses.

We ground our study in the CAGE Challenge 2 (CC2) environment, a high-fidelity cyber operations simulator. To construct valuations, we extract Q-values from trained reinforcement learning agents, interpreting them as empirical utilities over bundles of actions. These valuations feed into our combinatorial auction mechanism, enabling strategic planning under resource constraints.

	RSD	VCG	λ=0	λ=0.5	λ=1
Revenue	0	350,789	310,071	327,239	360,507
Welfare	300,009	422,768	619,106	607,864	582,730

Table 2: Case Study 1: Airport slot auction results. RSD yields baseline welfare but zero revenue, while VCG achieves higher welfare at the cost of low revenue. Our multi-objective mechanism outperforms both benchmarks, achieving substantial welfare improvements while maintaining competitive revenue, and regrets remain $\leq$ 5,000. Increasing λ shifts the balance toward revenue maximization, with $\lambda = 1$ surpassing VCG's revenue without sacrificing efficiency.

Agent	AUC (Reward vs. Steps)	t-stat	p-value
Original	$-27{,}219.5$	–	–
Shaped	$-25{,}266.5$	3.37	7.5×10^{-4}

Table 3: Case Study 2 (Cyber). Using the auction's allocation as a distributional reward-shaping target improves convergence and final performance (higher AUC is better); gains are statistically significant. Full setup and episodic curves are in the Appendix F.2

Our analysis highlights three findings. First, under truthful reporting, the mechanism concentrates on aggressive defenses (*Remove*, {*Analyze, Remove*}), while misreporting diffuses allocations but preserves host-level priorities, which is the evidence for robustness. Second, learned allocations align strongly with Blue-team activity in CC2, indicating robustness to underlying mission relevance. Third, using auction outputs as a reward-shaping signal improves RL training stability and convergence (statistically reported in Table 3), suggesting auctions can guide tactical agents toward more effective long-term strategies. Further details on environment setup, valuation construction, allocation comparisons, and statistical analyses are in Appendix F.2.

5 Discussion and Future Work

This work takes a first step toward extending differentiable economics to combinatorial auctions (CAs). We enforce combinatorial feasibility within end-to-end neural mechanisms and show that the approach can be instantiated with three architectures (CANet, CAFormer, CAGraph). While our methods deliver strong empirical revenue subject to low regret, it remains an open question whether they can represent the theoretically optimal mechanism in full generality.

Expressivity, scale, and training stability. Among our variants, CAGraph attains the highest revenue and benefits from graph inductive bias (permutation equivariance on the conflict graph, variable input size, and contextual features). Our scalarized training supports explicit revenue–welfare trade-offs. Nonetheless, min–max training is non-convex and lacks convergence guarantees; performance degrades if the inner adversary under-optimizes regret. Moreover, our experiments enumerate all bundles, which scales as 2^m and limits very large instances. Performance can degrade with weak inner optimization (underestimated misreports), in large bundle spaces (vanishing gradients, slow convergence), or with distribution shift. CAFormer may overfit at small scales; CAGraph can be sensitive to graph sparsity.

Outlook. The same recipe—parameterize a feasible region, differentiate the objective, and train with regret penalties—should extend to other combinatorial problems (e.g., routing, matching, partitioning) when constraints admit differentiable parameterizations. Promising directions include (i) alternative bidding languages and column generation, (ii) stronger inner optimizers and certified regret bounds, and (iii) tighter integration of graph structure and attention.

Reproducibility. Experiments were run on a local GPU server with *2× NVIDIA Quadro RTX 8000 (48 GB)* and *6× NVIDIA Quadro RTX 5000 (16 GB)*. All scripts, data and configurations are provided.

Societal impact. Airport slot allocation affects competition and consumer access; our multi-objective training surfaces explicit welfare–revenue trade-offs that can be aligned with policy goals. The cyber experiments use simulated environments; no real user data are involved. Risks include mis-specifying objectives and unequal access to algorithmic advantages.

References

[1] Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and J. Zico Kolter. Differentiable convex optimization layers, 2019.

[2] Tansu Alpcan and Tamer Basar. *Network Security: A Decision and Game-Theoretic Approach. Cambridge University Press*. Cambridge University Press, 2010.

[3] Brandon Amos and J. Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks, 2021.

[4] Mark Armstrong. Optimal multi-object auctions. *The Review of Economic Studies*, 67(3):455–481, 07 2000.

[5] Christopher Avery and Terrence Hendershott. Bundling and optimal auctions of multiple products. *The Review of Economic Studies*, 67(3):483–497, 2000.

[6] Patrick Bajari and Ali Hortaçsu. The winner's curse, reserve prices, and endogenous entry: Empirical insights from ebay auctions. *The RAND Journal of Economics*, 34(2):329–355, 2003.

[7] Michael O. Ball, Alexander S. Estes, Mark Hansen, and Yulin Liu. Quantity-contingent auctions and allocation of airport slots. *Transportation Science*, 54(4):858–881, 2020.

[8] Gianluca Brero, Benjamin Lubin, and Sven Seuken. Combinatorial auctions via machine learning-based preference elicitation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 128–136. International Joint Conferences on Artificial Intelligence Organization, 7 2018.

[9] Gianluca Brero, Benjamin Lubin, and Sven Seuken. Machine learning-powered iterative combinatorial auctions, 2021.

[10] Alan Carlin and R. E. Park. Marginal cost pricing of airport runway capacity. *The American Economic Review*, 60(3):310–319, 1970.

[11] Edward H. Clarke. Multipart pricing of public goods. *Public Choice*, 11:17–33, 1971.

[12] Jared Claypoole, Steven Cheung, Ashish Gehani, Vinod Yegneswaran, and Ahmad Ridley. Interpreting agent behaviors in reinforcement-learning-based cyber-battle simulation platforms, 2025.

[13] Gonzalo E. Constante-Flores, Hao Chen, and Can Li. Enforcing hard linear constraints in deep learning models with decision rules, 2025.

[14] Peter Cramton, Yoav Shoham, and Richard Steinberg. *Combinatorial Auctions*. The MIT Press, December 2005.

[15] Michael Curry, Tuomas Sandholm, and John Dickerson. Differentiable economics for randomized affine maximizer auctions, 2022.

[16] Michael Curry, Tuomas Sandholm, and John Dickerson. Differentiable economics for randomized affine maximizer auctions. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, IJCAI-2023, page 2633–2641. International Joint Conferences on Artificial Intelligence Organization, August 2023.

[17] Joseph I. Daniel. Congestion pricing and capacity of large hub airports: A bottleneck model with stochastic queues, 1995.

[18] Zhijian Duan, Haoran Sun, Yurong Chen, and Xiaotie Deng. A scalable neural network for dsic affine maximizer auction design, 2024.

[19] Zhijian Duan, Jingwu Tang, Yutong Yin, Zhe Feng, Xiang Yan, Manzil Zaheer, and Xiaotie Deng. A context-integrated transformer-based neural network for auction design. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 5609–5626. PMLR, 17–23 Jul 2022.

[20] Paul Duetting, Zhe Feng, Harikrishna Narasimhan, David Parkes, and Sai Srivatsa Ravindranath. Optimal auctions through deep learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1706–1715. PMLR, 09–15 Jun 2019.

[21] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *American Economic Review*, 97(1):242–259, March 2007.

[22] Zhe Feng, Harikrishna Narasimhan, and David C. Parkes. Deep learning for revenue-optimal auctions with budgets. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18, page 354–362, Richland, SC, 2018. International Foundation for Autonomous Agents and Multiagent Systems.

[23] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR.

[24] Noah Golowich, Harikrishna Narasimhan, and David C. Parkes. Deep learning for multi-facility location mechanism design. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, IJCAI-2018, page 261–267. International Joint Conferences on Artificial Intelligence Organization, July 2018.

[25] Theodore Groves. Incentives in teams. *Econometrica*, 41(4):617–631, 1973.

[26] Dmitry Ivanov, Iskander Safiulin, Igor Filippov, and Ksenia Balabaeva. Optimal-er auctions through attention, 2022.

[27] Arash Jamshidi, Seyed Mohammad Hosseini, Seyed Mahdi Noormousavi, and Mahdi Jafari Siavoshani. Differentially private machine learning-powered combinatorial auction design, 2024.

[28] Mitchell Kiely, David Bowman, Maxwell Standen, and Christopher Moir. On autonomous agents in a cyber defence environment. *arXiv preprint arXiv:2309.07388*, 2023.

[29] Kevin Kuo, Anthony Ostuni, Elizabeth Horishny, Michael J. Curry, Samuel Dooley, Ping yeh Chiang, Tom Goldstein, and John P. Dickerson. Proportionnet: Balancing fairness and revenue for auction design with deep learning. *ArXiv*, abs/2010.06398, 2020.

[30] Ron Lavi and Chaitanya Swamy. Truthful and near-optimal mechanism design via linear programming. *J. ACM*, 58(6), December 2011.

[31] Richard Lippmann and Kyle Ingols. An annotated review of past papers on attack graphs. *MIT Lincoln Laboratory Technical Report*, 2005.

[32] Eric Maskin, J. Riley, and F. Hahn. *Optimal Multi-Unit Auctions*, pages 312–335. Oxford University Press, 1989. Reprinted in P. Klemperer, The Economic Theory of Auctions, London: Edward Elgar, 2000.

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

[34] Jad Rahme, Samy Jelassi, Joan Bruna, and S. Matthew Weinberg. A permutation-equivariant neural network architecture for auction design, 2021.

[35] Jad Rahme, Samy Jelassi, and S. Matthew Weinberg. Auction learning as a two-player game, 2021.

[36] S. J. Rassenti, V. L. Smith, and R. L. Bulfin. A combinatorial auction mechanism for airport time slot allocation. *The Bell Journal of Economics*, 13(2):402–417, 1982.

[37] Sai Srivatsa Ravindranath, Zhe Feng, Shira Li, Jonathan Ma, Scott D. Kominers, and David C. Parkes. Deep learning for two-sided matching, 2023.

[38] Sai Srivatsa Ravindranath, Zhe Feng, Di Wang, Manzil Zaheer, Aranyak Mehta, and David C. Parkes. Deep reinforcement learning for sequential combinatorial auctions, 2024.

[39] Abhishek Ray, Mario Ventresca, and Karthik Kannan. A graph-based ant algorithm for the winner determination problem in combinatorial auctions. *Information Systems Research*, 32(4):1099–1114, December 2021.

[40] K. Roberts. The characterization of implementable choice rules. In J.-J. Laffont, editor, *Aggregation and Revelation of Preferences*, pages 321–348. North Holland Publishing, 1979.

[41] Tuomas Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence*, 135(1–2):1–54, February 2002.

[42] Tuomas Sandholm. Automated mechanism design: A new application area for search algorithms. In Francesca Rossi, editor, *Principles and Practice of Constraint Programming – CP 2003*, pages 19–36, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[43] Tuomas Sandholm and Anton Likhodedov. Automated design of revenue-maximizing combinatorial auctions. *Operations Research*, 63(5):1000–1025, October 2015.

[44] Aaron Schlenker, Haifeng Xu, Mina Guirguis, Christopher Kiekintveld, Arunesh Sinha, Milind Tambe, Solomon Sonya, Darryl Balderas, and Noah Dunstatter. Don't bury your head in warnings: A game-theoretic approach for intelligent allocation of cyber-security alerts. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 381–387, 2017.

[45] Andrea Tacchetti, DJ Strouse, Marta Garnelo, Thore Graepel, and Yoram Bachrach. Learning truthful, efficient, and welfare maximizing auction rules, 2022.

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

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

[48] Tonghan Wang, Yanchen Jiang, and David C. Parkes. Gemnet: Menu-based, strategy-proof multi-bidder auctions through deep learning, 2024.

[49] Qinghua Wu and Jin-Kao Hao. A clique-based exact method for optimal winner determination in combinatorial auctions. *Information Sciences*, 334–335:103–121, March 2016.

[50] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. Deep sets, 2018.

[51] Quanyan Zhu and Tamer Basar. Game-theoretic methods for robustness, security, and resilience of cyberphysical control systems: Games-in-games principle for optimal cross-layer resilient control systems. *IEEE Control Systems Magazine*, 35(1):46–65, 2015.

A Preliminary

RegretNet [20] provides a deep learning framework for optimal auction design, addressing multi-bidder, multi-item scenarios where analytical solutions are unknown. It encodes auction allocation and payment rules within neural network weights.

The architecture comprises two networks: the allocation network $\mathbf{A}^{\text{net}}$ and the payment network $\mathbf{P}^{\text{net}}$. Both process a bid vector $\mathbf{b}^{nm}$ through fully-connected layers. The Allocation Network maps bids to allocation probabilities $\mathbf{A}^{\text{net}}(\mathbf{b}^{nm}) = \mathbf{Z}^{nm}$, where $z_{ij} = \frac{e^{\tilde{z}_{ij}}}{\sum_{i=1}^{n+1} e^{\tilde{z}_{ij}}}$ and allows an item to remain unallocated by introducing a dummy participant. The payment network maps bids to payment allocations $\mathbf{P}^{\text{net}}(\mathbf{b}^{nm}) = \tilde{\mathbf{p}}^{n}$, where $\tilde{p}_i$ is scaled using sigmoid, and $p_i = \tilde{p}_i \sum_{j=1}^{m} z_{ij} b_{ij}$, adhering to IR constraints (2).

RegretNet minimizes empirical negative revenue plus the regret penalty over profiles V:

$$\min_{\mathbf{w},\lambda,\rho} \left(-\sum_{i \in N} \mathbb{E}[P_i(v; \mathbf{w})] + \lambda_i \text{rgt}_i + \frac{\rho}{2} \text{rgt}_i^2 \right) \tag{5}$$

where regret $\text{rgt}_i(v_i', v; \mathbf{w}) = \max_{v_i'}(u(v_i, (v_i', v_{-i}); \mathbf{w}) - u(v_i, v; \mathbf{w}))$. Regret is iteratively minimized using augmented Lagrangian and gradient descent:

$$\mathcal{L}_{\text{inner}}(v_i') = -u(v_i, (v_i', v_{-i}); \mathbf{w}) \tag{6}$$

and the Lagrange multipliers λ_i and ρ are updated as $\lambda_i \leftarrow \lambda_i + \rho \tilde{R}_i$, $\rho \leftarrow \rho + \rho\Delta$.

The results of RegretNet are further extended to address combinatorial auctions in a two-item, two-bidder setting by adding the constraint:

$$\forall i, \quad z_{i,\{1\}} + z_{i,\{2\}} \leq 1 - \sum_{i=1}^{n} z_{i,\{1,2\}}$$

To accommodate this constraint, two softmax layers are used; one outputs a set of bidder scores, and the other a set of item scores, denoted by $\bar{s}_{i,S}$ and $\bar{s}^{(j)}_{i,S}$. The set of item scores is then used to compute a normalized set of scores for all i and S, given by

$$\bar{\bar{s}}_{iS} = \begin{cases} \bar{s}^{(i)}_{iS} & S = \{1\}, \{2\} \\ \min\{\bar{s}^{(k)}_{iS}\} & S = \{1, 2\} \end{cases}$$

and the final allocation $z_{i,S}$ is determined by the minimum of the bidder score and normalized item score.

$$z_{iS} = \min\left(\bar{s}_{iS}, \bar{\bar{s}}_{iS}\right)$$

B Related Work

The seminal works of [47] introduced auction mechanisms for single items, which were later extended to multi-item settings by developing the Vickrey-Clarke-Groves (VCG) mechanism [47, 11, 25]. Although VCG achieves efficiency in dominant strategies, it often fails to maximize revenue. [33] laid the theoretical foundation for revenue-maximizing auctions.

The study of combinatorial auctions originated from the need to allocate resources among bidders with complex preferences over bundles of items. [14] provide a detailed review of combinatorial auctions. Optimal combinatorial auctions are NP-hard because of the exponential number of possible bundles and the Incentive Compatibility (IC) and IR constraints. Heuristic and exact methods have been studied to solve the winner determination problem as a combinatorial optimization problem [41, 49, 39]. In the early work, both the winner determination and pricing problems were tackled using integer programming and approximation algorithms. [30] developed polynomial-time algorithms for approximately optimizing VCG-based mechanisms in certain structured settings. Another direction, extending the results of [40], is known as Affine Maximizer Auctions (AMAs). These are variations of the VCG mechanism that adjust the allocation process by assigning positive weights to each bidder's welfare and incorporating boosts for different allocations, potentially leading to higher revenue than the VCG mechanism [43, 16].

Recent advances in differentiable economics have extended its applications to various domains, reflecting the growing synergy between machine learning and economic theory. These works leverage neural architectures and differentiable approaches to address classical and emerging problems in auction design and beyond. Several papers have extended the ideas introduced in RegretNet. EquivariantNet [34] enhances generalization in symmetric auctions by enforcing permutation equivariance in its network structure, improving performance in settings with indistinguishable bidders and items. ALGnet [35] replaces RegretNet's augmented Lagrangian optimization with a GAN-based approach, framing the auctioneer-misreporter interaction as a two-player game to improve training efficiency. RegretFormer [26] introduces self-attention layers, leveraging permutation-equivariant representations and incorporating a regret budget to balance incentive compatibility violations and revenue. AMenuNet [18] restricts the search space to affine maximizer auctions, ensuring dominant strategy incentive compatibility (DSIC) and individual rationality (IR) while employing permutation-equivariant neural networks for improved generalization. CITransNet [19] extends RegretNet to contextual auctions by integrating public context information through transformer-based architectures, maintaining permutation equivariance without being limited to symmetric auctions. GemNet [48] further broadens this space by introducing a menu-based, strategy-proof auction framework for multi-bidder settings, achieving exact DSIC through menu compatibility constraints enforced during training and post-training price transformations via MILP optimization. [15] propose a differentiable framework for randomized affine maximizer auctions that achieves exact strategyproofness in multi-item, multi-bidder settings with additive valuations. While some existing studies refer to their setting as "combinatorial" due to its discrete structure, their work does not address the core challenge of bundle-based preferences in combinatorial auctions—where feasibility constraints are inherently non-convex and allocations must account for overlapping item bundles. These advancements demonstrate diverse extensions of RegretNet, each enhancing scalability, efficiency, or incentive properties in

auction design. Others have applied this framework to auctions with fairness or budget constraints [29, 22], or adapted similar methods to tackle broader mechanism design challenges [24, 37]. Several papers leverage machine learning-based approaches in combinatorial auctions. [8] used a machine learning-based elicitation algorithm to identify which values to query from the bidders. [45] propose a network architecture to learn Groves payment rules in combinatorial auctions with certain bidding languages. [27] present a machine learning-powered combinatorial auction based on the principles of differential privacy. [9] present a machine learning-powered iterative combinatorial auction. [38] use deep Reinforcement Learning for sequential combinatorial auctions.

In parallel, several works have explored constraint-aware neural networks. There is a line of work that embeds optimization problems as differentiable layers, such as OptNet [3] and DiffOpt [1]. [13] combines a standard prediction network with a "safety" component trained via decision rules to guarantee feasibility across all inputs. Our work builds on this line by proposing a constraint enforcement framework specifically tailored to combinatorial auctions, capable of being integrated with various neural architectures—including GNN-based approaches like CAGraphNet.

C Constraint Enforcement

Recall from Section 2 that the allocation must satisfy constraints in (3). We decompose the allocation matrix $\mathbf{Z} \in \mathbb{R}^{n \times k}$ into the product of two matrices:

$$\mathbf{Z} = \mathbf{B}^{\text{bundle}} \cdot \mathbf{A}^{\text{agent-bundle}},$$

where $\mathbf{A}^{\text{agent-bundle}}$ and $\mathbf{B}^{\text{bundle}}$ are defined below. This approach is motivated by a two-step allocation process: first, the allocation of items to the bundles, satisfying the item-wise constraint, and then the allocation of the bundles to the bidders, satisfying the other constraints. Each entry in the final allocation matrix represents the probability of allocating a bundle to a bidder, which is calculated by the product of the probability that the bundle is allocated B_S^{bundle} and the probability that the bidder receives the bundle, given that the bundle is allocated $A_{iS}^{\text{agent-bundle}}$.

The item-to-bundle allocation matrix $\mathbf{B}^{\text{bundle}}$ is derived from the output of the Allocation Network, normalized to represent item-wise probabilities. To construct $\mathbf{B}^{\text{bundle}}$, we define the incident matrix $\mathbf{I} \in \mathbb{R}^{m \times k}$, where:

$$I_{iS} = \begin{cases} 1 & \text{if item } i \text{ is included in bundle } S, \\ 0 & \text{otherwise.} \end{cases}$$

The initial bundle allocation matrix $\mathbf{B} \in \mathbb{R}^{m \times k}$ is adjusted by mapping bundles to items with a softmax function across m items:

$$\mathbf{B}^{\text{adjusted}} = \text{softmax}_M(\mathbf{B} \cdot \mathbf{I}).$$

To ensure valid item-to-bundle allocations, non-positive values in $\mathbf{B}^{\text{adjusted}}$ are replaced with a large constant M:

$$\mathbf{B}^{\text{masked}} = \begin{cases} \mathbf{B}^{\text{adjusted}}, & \text{if } \mathbf{B}^{\text{adjusted}} > 0, \\ M, & \text{otherwise.} \end{cases}$$

For each item i, we compute the minimum value across all bundles and reshape $\mathbf{B}_{\text{min}}$ to include the agent dimension:

$$\mathbf{B}^{\text{bundle}} = \text{Unsqueeze}(\min_S \mathbf{B}^{\text{masked}}),$$

The bundle-to-agent allocation matrix $\mathbf{A}_{\text{agent-bundle}}$ is computed from the allocation network outputs $\mathbf{A}$ and $\mathbf{A}'$ using the softmax function along the agent and the bundle dimension.

$$\mathbf{A}^{\text{agent}} = \text{softmax}_N(\mathbf{A}), \quad \mathbf{A}^{\text{bundle}} = \text{softmax}_K(\mathbf{A}')$$

then taking element-wise minimum of the two matrices

$$\mathbf{A}^{\text{agent-bundle}} = \min(\mathbf{A}^{\text{agent}}, \mathbf{A}^{\text{bundle}})$$

The final allocation matrix $\mathbf{Z}$ is given by:

$$\mathbf{Z} = \mathbf{B}^{\text{bundle}} \cdot \mathbf{A}^{\text{agent-bundle}},$$

where each entry z_{iS} represents the probability of allocating bundle S to agent i.

We will now prove Matrix $\mathbf{Z}$ is combinatorially feasible.

The matrix $\mathbf{A}^{\text{agent-bundle}}$ represents the bundle-to-agent allocation. Each entry $A_{iS}^{\text{agent-bundle}}$ is computed as:

$$A_{iS}^{\text{agent-bundle}} = \min\left(\frac{e^{a_{iS}/\theta}}{\sum_{i'} e^{a_{i'S}/\theta}}, \frac{e^{a'_{jS}/\theta}}{\sum_{S'} e^{a'_{iS'}/\theta}}\right).^{1}$$

This ensures that

$$\sum_{i} A_{iS}^{\text{agent-bundle}} \leq 1, \forall S \in K \tag{7}$$

$$\sum_{S} A_{iS}^{\text{agent-bundle}} \leq 1, \forall i \in N \tag{8}$$

From the definition of z_{iS}, the allocation of any bundle S is limited by its least-available item:

$$\sum_{i}\sum_{S\ni j} z_{iS} \leq \sum_{i}\sum_{S\ni j} \min_{j\in S}\{B_{jS}^{\text{adjusted}}\} \cdot A_{iS}^{\text{agent-bundle}}.$$

Since B^{adjusted} is normalized with softmax along item-wise dimension to ensure that no item j is allocated more than once:

$$\sum_{S\ni j} \min_{j\in S}\{B_{jS}^{\text{adjusted}}\} \leq \sum_{S\in K} \min_{j\in S}\{B_{jS}^{\text{adjusted}}\} \leq 1, \quad \forall j \in M.$$

Thus,

$$\sum_{i}\sum_{S\ni j} z_{iS} \leq \sum_{i}\sum_{S\ni j} A_{iS}^{\text{agent-bundle}}.$$

For a fixed item j, we can write:

$$\sum_{i}\sum_{S\ni j} A_{iS}^{\text{agent-bundle}} \leq \sum_{S\ni j}\sum_{i} A_{iS}^{\text{agent-bundle}}.$$

From (7), we have:

$$\sum_{i}\sum_{S\ni j} A_{iS}^{\text{agent-bundle}} \leq \sum_{S\ni j} 1.$$

From (8), the number of bundles S containing j is at most the total number of bundles k, constraint (3) is satisfied:

$$\sum_{i}\sum_{S\ni j} A_{iS}^{\text{agent-bundle}} \leq 1.$$

Thus, $\mathbf{Z}$ is combinatorially feasible.

D Model Architectures

D.1 CANet Architecture

Motivated by RegretNet [20], we present CANet, a deep neural network architecture designed for combinatorial auction mechanisms. The CANet architecture, visualized in Figure 2, comprises two modules: the Deep Allocation Network and the Deep Pricing Network. The input to CANet, denoted by $\mathbf{b}$, is a vector $\mathbf{b} \in \mathbb{R}^{n\cdot k}$, where $b_{iS} \sim F_i$ is the bid submitted by agent i for bundle S.

The Deep Allocation Network computes a feasible allocation matrix $\mathbf{Z} \in \mathbb{R}^{n\times k}$ ensuring that both individual item constraints and global allocation constraints are satisfied. The input $\mathbf{b}$ is passed through fully connected layers with non-linear activations. The network produces three output vectors $\mathbf{A} \in \mathbb{R}^{n\times k}$ and $\mathbf{A'} \in \mathbb{R}^{n\times k}$ representing an unnormalized agent-bundle allocation; and $\mathbf{B} \in \mathbb{R}^{m\times k}$

[1] In all of our implementations, the softmax function incorporates a *temperature* parameter $\theta \in \mathbb{R}^{+}$, inspired by the Boltzmann distribution in statistical mechanics, which controls the concentration of probability mass around the highest logit.

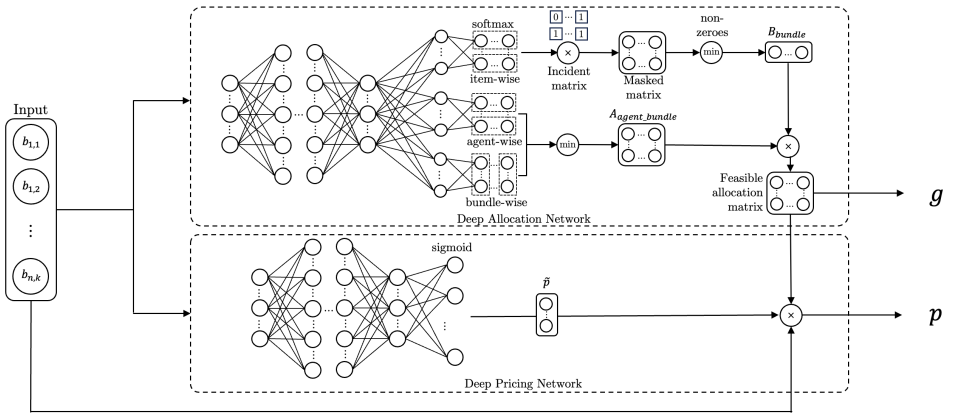

Figure 2: CANet Architecture

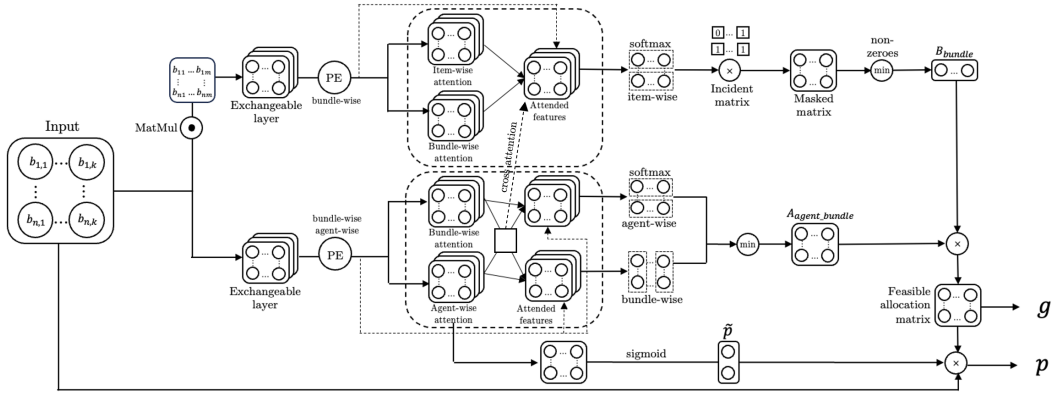

Figure 3: CAFormer Architectures.

representing an unnormalized bundle-item allocation. These outputs are combined following the process described in section C to form a feasible allocation matrix $\mathbf{Z}$.

The Deep Pricing Network computes the pricing vector $\tilde{\mathbf{p}} \in \mathbb{R}^n$, which represents a discount applied to the bids. The input $\mathbf{b}$ is processed through fully connected layers with sigmoid transformation, to ensure $\tilde{p}_i \in [0, 1]$. The final price is vector $\mathbf{p}$ with $p_i = \tilde{p}_i \sum_{S=1}^{k} z_{iS} b_{iS}$.

D.2 CAFormer Architecture

To enhance expressivity and ensure permutation-equivariance and context-awareness, we introduce CAFormer. This architecture combines exchangeable layers (proposed in [50] and used in EquivariantNet [34]) with stacked attention layers [46], as seen in RegretFormer [26]. The architecture is illustrated in Figure D. The attention mechanism is permutation-invariant, while exchangeable layers ensure permutation-equivariance—properties particularly desirable in symmetric auctions. For asymmetric settings or preserving input order across the bundle dimension, we apply agent-wise and bundle-wise positional encoding as presented in [46] after the exchangeable layers.

The input to CAFormer is a two-dimensional matrix $\mathbf{b} \in \mathbb{R}^{n \times k}$. Unlike CANet that can output an item-bundle allocation at the last layer, CAFormer must be designed properly to maintain insensitivity to the size of the problem. We achieve it by transforming the bid matrix into a representation of item-bundle allocation $\mathbf{b}' \in \mathbb{R}^{m \times k}$ through matrix multiplication with the individual item bid matrix $\mathbf{b}^{\text{item}} \in \mathbb{R}^{n \times m}$.

$$\mathbf{b}' = \text{MatMul}(\mathbf{b}^\top, \mathbf{b}^{\text{item}})^\top,$$

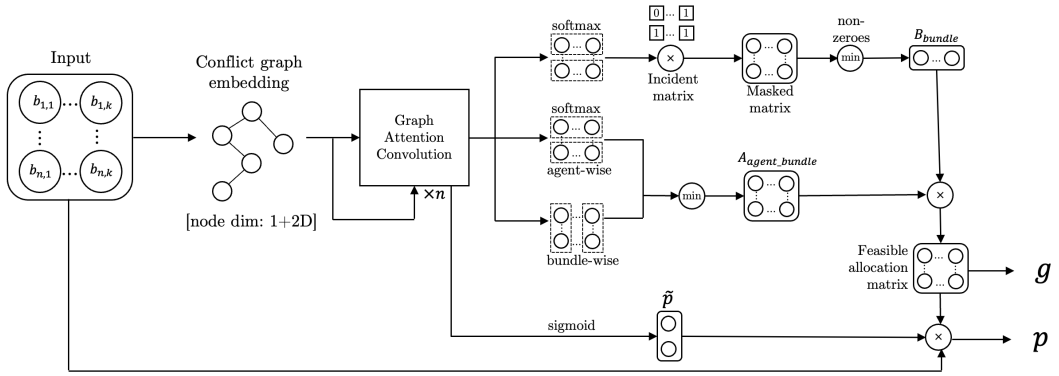

Figure 4: CAGraph Architecture

where $^\top$ denotes the transpose operation. Both $\mathbf{b}$ and $\mathbf{b}'$ are then expanded in a feature dimension that contain global information of bids with exchangeable layers, a key building block in architectures like EquivariantNet [34] and RegretFormer [26] which ensures permutation-equivariance. This layer processes a tensor $\mathbf{X} \in \mathbb{R}^{n\times k\times 1}$, into an output tensor $\mathbf{X}_{\text{ex}} \in \mathbb{R}^{n\times k\times d}$. The exchangeable layer computes the element of the output channels by aggregating input tensor B over elements, rows, columns, and globally, weighted by learnable parameters, and applies a non-linear activation σ. Details can be found in Deep Sets [50].

$$\mathbf{b}_{\text{ex}} = \text{Exchangeable}(\mathbf{b}), \quad \mathbf{b}'_{\text{ex}} = \text{Exchangeable}(\mathbf{b}')$$

After each exchangeable layer, we apply sequential attention-based blocks to the tensors $\mathbf{b}_{\text{ex}}$ and $\mathbf{b}'_{\text{ex}}$. Each block consists of multi-head self-attention layers with residual connections.

For each layer, the input tensor is reshaped to enable item-wise, bundle-wise, or agent-wise self-attention. Item-wise attention operates on reshaped input $\mathbf{b}'_{\text{item}} \in \mathbb{R}^{m\times d}$, bundle-wise attention operates on $\mathbf{b}_{\text{bundle}}$ and $\mathbf{b}'_{\text{bundle}} \in \mathbb{R}^{k\times d}$, and agent-wise attention operates on $\mathbf{b}_{\text{agent}} \in \mathbb{R}^{n\times d}$, where d is the feature dimensionality of the input. The outputs are concatenated, forming $\mathbf{b}'_{\text{concat}} \in \mathbb{R}^{m\times k\times d'_{\text{concat}}}$ and $\mathbf{b}_{\text{concat}} \in \mathbb{R}^{n\times k\times d_{\text{concat}}}$, where d_{concat} and d'_{concat} are the combined attended features. After the attention-blocks, we use fully-connected layers $f_{\text{fc}}^{\text{item}}$, $f_{\text{fc}}^{\text{bundle}}$, and $f_{\text{fc}}^{\text{agent}}$ to map the features back to the original dimensionality 1:

$$\mathbf{B} = f_{\text{fc}}^{\text{item}}(\mathbf{b}'_{\text{concat}}),$$

$$\mathbf{A} = f_{\text{fc}}^{\text{agent}}(\mathbf{b}_{\text{concat}}), \quad \mathbf{A}' = f_{\text{fc}}^{\text{bundle}}(\mathbf{b}_{\text{concat}})$$

Again, following the procedure outlined in Section C, we derive a feasible allocation matrix $\mathbf{Z}$. The pricing vector $\tilde{\mathbf{p}} \in \mathbb{R}^n$ is derived by applying a fully connected layer with sigmoid activation to the agent-wise attention output $\mathbf{b}_{\text{agent}}$, reducing its dimensionality to 1. The final prices are computed using the same method as in CANet.

D.3 CAGraph Architecture

Given that the winner determination problem in combinatorial auctions (CAs) is inherently a combinatorial optimization problem, and can be mapped to well-studied structures in graph theory, we propose a novel graph-based approach. In this section, we introduce CAGraph, a graph neural network architecture designed to approximate optimal allocation and payment rules in CAs. The architecture exploits the relational structure among bundles to reason over complex feasibility constraints and revenue objectives in a scalable and expressive way.

D.3.1 Set-packing Representation

We adopt a set-packing formulation of the winner determination problem, which is a classical representation in combinatorial optimization. In this representation, each node in the graph corresponds to a bidder-bundle pair (i, S), where $i \in N$ is a bidder and $S \subseteq M$ is a bundle of items. Each node is associated with a weight representing the bidder's valuation $v_i(S)$ for that bundle.

An edge exists between two nodes if their corresponding bundles have any item overlap, i.e., if $S \cap S' \neq \emptyset$, indicating that both allocations cannot simultaneously be part of a feasible solution.

The goal of the winner determination problem is then to select a subset of non-conflicting nodes (i.e., an independent set in the conflict graph) such that the total valuation (weight) of the selected nodes is maximized. Formally, this is equivalent to solving a differentiable relaxation of maximum weighted independent set problem over the constructed graph:

$$\max_{z \in [0,1]^{n \times k}} \sum_{(i,S)} v_i(S) z_{iS} \quad \text{subject to: } z_{iS} + z_{jS'} \leq 1 \text{ if } S \cap S' \neq \emptyset \quad \forall i \in N, j \in M.$$

Here, $z_{iS} \in [0, 1]$ is a variable indicating the probability that bundle S is allocated to bidder i.

This formulation naturally lends itself to GNN-based approximation, where node embeddings are learned through message passing over the conflict graph. The learned embeddings capture global auction structure and local conflicts, and are passed through a final scoring layer to estimate probabilities for each node.

D.3.2 CAGraph Details

Input and Embedding Dimensions. Recall that each node in the conflict graph corresponds to a bidder-bundle pair (i, S), and is represented by a feature vector that combines the bid value b_{iS}, a learnable embedding of the bidder i, and a learnable embedding of the bundle S. These embeddings are dense vectors of dimension D, initialized randomly and trained jointly with the rest of the network. The purpose of these embeddings is to encode semantic and structural information about bidders and bundles that may not be directly observable from bids alone. Unlike handcrafted features or static encodings, these embeddings are learned end-to-end through gradient descent, thus finding representations that are useful for allocation and pricing decisions. The final node feature vector is constructed by concatenating the scalar bid and the two embedding vectors, resulting in an input dimension of

$$\text{input\_dim} = 1 + 2D.$$

These node features are then passed through a linear projection layer followed by two graph convolution layers, which propagate and refine local information through the conflict edges between overlapping bundles.

Graph Attention Convolution. Following the input projection layer, CAGraph applies two stacked Graph Attention Network (GAT) layers to propagate structural information across the conflict graph. These layers enable nodes—each representing a bidder-bundle pair—to exchange information with their neighbors, which correspond to other conflicting pairs.

Given node features $\mathbf{X} \in \mathbb{R}^{N \times d_{\text{in}}}$, each GAT layer performs neighborhood aggregation with learned attention weights. For each node i, the updated feature vector is computed as:

$$\mathbf{X}'_i = \sigma\left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij} \mathbf{W} \mathbf{X}_j\right) + \text{Res}(\mathbf{X}_i),$$

where $\mathbf{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ is a shared linear projection, $\mathcal{N}(i)$ denotes the set of neighbors of node i, and σ is an activation function (ELU). The attention weights α_{ij} quantify the influence of node j on node i, and are computed as:

$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}(\mathbf{a}^\top [\mathbf{W}\mathbf{X}_i \| \mathbf{W}\mathbf{X}_j])\right)}{\sum_{k \in \mathcal{N}(i)} \exp\left(\text{LeakyReLU}(\mathbf{a}^\top [\mathbf{W}\mathbf{X}_i \| \mathbf{W}\mathbf{X}_k])\right)},$$

where $\mathbf{a} \in \mathbb{R}^{2d_{\text{out}}}$ is a learnable attention vector, and $\|$ denotes vector concatenation. The concatenated input to the attention mechanism has dimension $2d_{\text{out}}$, matching the parameterization of $\mathbf{a}$. These scores are normalized via softmax across each node's neighborhood.

This mechanism allows the network to assign greater weight to more relevant neighbors when aggregating information, effectively learning which conflicts are more important in determining feasible and high-value allocations. A residual connection $\text{Res}(\cdot)$ ensures stable optimization, mitigates over-smoothing in graph propagation and is implemented as linear projection.

By stacking two GAT layers, the model captures second-order interactions in the conflict graph. This enables bidder-bundle nodes to reason not just about direct conflicts, but also about indirect competitive effects, improving its ability to model allocation feasibility. The learned attention structure acts as a soft constraint mechanism, allowing the network to suppress overlapping bundles and prioritize mutually compatible ones during allocation inference.

Output Layers. After the GAT layers, the model compute final allocation outputs through a series of fully connected layers. These output layers handle bidder-bundle and item-level decisions and integrate with the constraint enforcement framework described in section C.

E Training Procedure

Similarly to RegretNet, our training process also includes an inner utility maximization (6) and an outer revenue maximization and regret minimization (5). For brevity, we refer the reader to the original paper by [20] for details of the adversarial training. Previous works [20, 34, 26] relies heavily on gradient-based Lagrangian optimization (see Section 2). This method is known to be sensitive to the choice of hyper-parameters. An example is in the outer loss (5), when λ_i becomes excessively large, the model disproportionately focuses on minimizing regret, and gradients for regret become negligible due to a scaling effect or poor numerical conditioning. To ensure a stable tradeoff, we constrain the loss weights such that their sum is 1. This simple balancing mechanism does not directly tackle the overemphasis on regret, but it ensures regret gradient does not shrink due to the excessive weight λ_i. Additionally, we apply a logarithmic transformation to the revenue term to avoid unbounded growth. This is in the spirit of multi-task learning, which seeks to optimize both objectives in the present of trade-offs. Our outer loss function is defined as:

$$\mathcal{L}_{\text{outer}} = -w_{rev} \log(1 + \text{rev}) + w_{rgt}\text{rgt},$$

where w_{rev} and w_{rgt} are task weights that represent the emphasis on revenue and regret, respectively.

Following [26], we also set a regret budget, allowing controlled violations of DSIC by gradually decreasing the budget during training. The update for w_{rgt} is explicitly computed as:

$$g_t = \log(\text{rgt}) - \log(\bar{\text{rgt}}) - \log(1 + \alpha\text{rev}) \tag{9}$$

where $\bar{\text{rgt}}$ is a target regret value. The parameter α is optional and adjusts the ratio between the target regret and revenue.

We provide the Adam-style update for w_{rgt} with coupling $w_{\text{rev}} = 1 - w_{\text{rgt}}$ in algorithm 1

F Case Study Details

F.1 Case Study 1: Airport Slot Allocation

F.1.1 Motivation and Setup

Airport congestion, particularly at high-traffic hubs, poses a complex resource allocation challenge that is inherently combinatorial in nature. Airlines derive value not from individual takeoff or landing slots in isolation, but from bundles of slots that support network connectivity, fleet rotation, and coordinated passenger itineraries. Traditional models—such as those developed by [10] and [17]—focused on congestion pricing as an economic tool to regulate demand and allocate runway capacity efficiently. Subsequent work introduced market-based mechanisms, most notably combinatorial auctions for slot allocation [36], with further policy refinements by [7]. While these approaches improve allocative efficiency by prioritizing agents with the highest valuations, they often neglect strategic behavior, and the possibility that airlines may misreport preferences to manipulate outcomes in their favor.

While earlier studies proposed large-scale auctions to alleviate congestion, this study focuses on the design of slot auctions for small-scale divestiture scenarios, such as those arising from regulatory

Algorithm 1: Adam-style update for w_{rgt} with coupling $w_{\text{rev}} = 1 - w_{\text{rgt}}$

Input: gradient g_t from Eq. (9), step size γ_w, $\beta_1, \beta_2 \in (0, 1)$, $\varepsilon > 0$, scale $\rho > 0$
Input: $t \leftarrow 0$, $m \leftarrow 0$, $v \leftarrow 0$, initialize $w_{\text{rgt}} \in [0, 1]$
while *training* **do**
  $t \leftarrow t + 1$
  // First/second moments
  $m \leftarrow \beta_1 m + (1 - \beta_1)\, g_t$
  $v \leftarrow \beta_2 v + (1 - \beta_2)\, g_t^2$
  // Bias correction
  $\hat{m} \leftarrow m/(1 - \beta_1^t)$
  $\hat{v} \leftarrow v/(1 - \beta_2^t)$
  // Adam step on $w_{\texttt{rgt}}$
  $w_{\text{rgt}} \leftarrow w_{\text{rgt}} + \gamma_w\, \hat{m}/\big(\sqrt{\hat{v}} + \varepsilon\big)$
  // Normalize to $[0, 1]$ and couple with $w_{\texttt{rev}}$
  $w_{\text{rgt}} \leftarrow \frac{1}{2}\Big(1 + \tanh\big(\frac{w_{\text{rgt}}}{\rho}\big)\Big)$
  $w_{\text{rev}} \leftarrow 1 - w_{\text{rgt}}$
end
return $w_{\text{rev}},\ w_{\text{rgt}}$

interventions during airline mergers, or underutilization under 80% rule. Our mechanism is proposed to substitute Random Serial Dictatorship, the random lottery of the order of slot selection, as formalized in 14 CFR §93.225. These settings are characterized by a limited number of bidders and discrete bundles of slots, which makes them tractable for strategic auction design and evaluation.

A welfare-maximizing mechanism, such as VCG auction, allocates slots to those who value them most — promoting efficient use of capacity and better outcomes for passengers and the network. However, VCG may yield low or zero revenue [33, 14], particularly in thin markets or when bidders' valuations are highly correlated. In contrast, revenue-maximizing mechanisms—such as first-price auctions or neural networks trained to optimize payments—can ensure that sellers (e.g., divesting airlines or regulatory agencies) are compensated. This is particularly important in divestiture settings, where the seller may be giving up valuable operational rights as part of a merger remedy. In such cases, generating sufficient revenue from the auction can help: offset financial losses to the divesting party, encourage participation in reallocation mechanisms, and reduce political resistance to mandatory slot redistribution. Yet revenue alone is not a sufficient design goal. Allocative efficiency remains central to the long-term effectiveness of slot policy, especially in regulated markets where public interest, competition, and consumer access are key concerns. For this reason, in the spirit of multi-objective optimization in section E, we propose and evaluate mechanisms that allow explicit trade-offs between revenue and welfare, adapting to policy priorities. Our mechanism provides a flexible framework for balancing these objectives: by adjusting the training loss, we can interpolate between welfare, revenue, or hybrid targets. This allows policymakers to weigh short-term fiscal outcomes (e.g., compensating a divesting airline) against long-term system performance and fairness. To evaluate these trade-offs under realistic airline behavior, we model the slot allocation problem as a sealed-bid combinatorial auction. We ground our case study in a real-world event: the 2011 FAA slot auction at Reagan National airports (DCA), in which 8 slot pairs were divested by Delta and US Airways under a DOT-imposed remedy. We simulate airline preferences by solving a profit-maximizing scheduling optimization for each carrier. Details about the slot allocation process at U.S. coordinated airports and historical precedent for slot auctions are provided in the Appendix.

F.1.2 Valuation Model

We aim to estimate the value (i.e., expected profit) that each airline assigns to specific airport slot bundles. These valuations are not directly observed but are inferred by solving a scheduling optimization problem for each airline-bundle pair. Itinerary-level fares and route information are derived from the Airline Origin and Destination Survey (DB1B) dataset. The seat capacities are obtained from the T-100 Domestic Segment (U.S. Carriers) dataset. All data are downloaded from https://www.transtats.bts.gov. The profit-maximizing assignment of flights to the available slots

determines the airline's valuation for the bundle. These values are later used as input to an auction solver, which assumes known bidder valuations.

We assume that all slots are at a *hub airport* (e.g., LGA or DCA), and each slot may be used either for an *arrival* or a *departure*, but not both. We model the assignment of directional flights to slots subject to feasibility and flow balance (i.e., aircraft arriving = aircraft departing).

The problem is formulated as follows

Given:

- F: set of feasible flights for the airline (each flight has a direction)
- T: set of time slots in the slot bundle
- $V(f,t)$: profit from assigning flight f to slot t
- $\mathrm{dir}(f) \in \{+1,-1\}$: direction of flight f, where $+1$ indicates a departure and -1 an arrival

Decision Variables:

- $z_{ft} \in \{0,1\}$: equals 1 if flight f is assigned to slot t
- $w_f \in \{0,1\}$: equals 1 if flight f is not operated

Objective:

$$\max_{z,w} \quad \sum_{f\in F}\sum_{t\in T} V(f,t)\cdot z_{ft} \tag{10}$$

Subject to:

$$\sum_{f\in F} z_{ft} \le 1, \quad \forall t \in T \tag{11}$$

$$\sum_{t\in T} z_{ft} + w_f = 1, \quad \forall f \in F \tag{12}$$

$$\sum_{f\in F}\sum_{t\in T} \mathrm{dir}(f)\cdot z_{ft} = 0 \tag{13}$$

$$z_{ft} \in \{0,1\}, \quad w_f \in \{0,1\}, \quad \forall f \in F, t \in T \tag{14}$$

Constraint (F.1.2) ensures that each slot is assigned to at most one flight. Constraint (F.1.2) ensures that each flight is either operated in exactly one slot or canceled. Constraint (F.1.2) enforces that the total number of arrivals equals the total number of departures at the hub airport. The objective function (F.1.2) computes total profit, which defines the valuation of the slot bundle for the airline.

F.2 Case study 2: Cyber Network Defense Auction Design

F.2.1 Motivation and Setup

Modern cybersecurity operations demand both reactive defenses and strategic planning under resource constraints. On the one hand, reactive tools like intrusion detection and incident response handle immediate threats. On the other, strategic planning involves anticipating attacks and proactively allocating resources, such as time, bandwidth, or analyst effort, across networked systems to minimize long-term risk.

In this context, strategic planning refers to decisions made before attack episodes begin, targeting hosts and services most critical to operational resilience. Meanwhile, resource constraints may stem from human limitations (e.g., analyst bandwidth), computational budgets (e.g., action frequency caps), or system costs (e.g., downtime from defensive interventions). Thus, defensive actions like

Analyze, Remove, and Restore must be allocated across a distributed enterprise network in a manner that balances operational cost and cyber risk.

Prior work has explored long-term cyber defense through game-theoretic models [2, 44], attack graphs [31], and stochastic control frameworks [51], but these often rely on centralized control and handcrafted utility functions, limiting their adaptability to real-world constraints and adversarial environments.

To overcome these limitations, we frame cyber defense as a decentralized resource allocation problem, where each host in the network acts as a self-interested agent with private valuations over bundles of defensive actions. This allows us to extend our combinatorial auction (CA) framework to reason about resource allocation in cyber planning, accounting for action synergies, private preferences, and dynamic threats. Note that it can also be seen as a virtual auction-based guidance mechanism to enhance the robustness of cyber defense strategies.

We ground this framework in the CAGE Challenge 2 (CC2) simulation environment [28], a high-fidelity autonomous cyber operations testbed. In CC2, a Blue agent defends a 13-host enterprise network against persistent Red-team adversaries. Defensive actions must be selected in anticipation of potential compromises, lateral movements, or disruptions. Each episode simulates 30 timesteps of adversarial interaction, capturing realistic operational dynamics. The CC2 environment is implemented on the CybORG platform and includes diverse host types (e.g., user workstations, enterprise services, defender nodes) with varying roles and vulnerabilities. Blue agents execute tactical decisions during each timestep, choosing actions based on host state and past observations. Red agents follow policy-driven strategies like BLine or Meander, simulating attacker behaviors ranging from direct exploits to stealthy lateral movement.

F.2.2 Valuation Modeling

While prior CC2 agents (including those trained via PPO) optimize at the tactical level, we propose to "lift" this information upstream for strategic planning. Specifically, we extract Q-values from trained RL agents, representing long-term expected utility for each action at each host. These Q-values serve as empirical proxies for private valuations, capturing both immediate and downstream consequences of defense.

To handle interactions between actions, we compute valuations over bundles of actions using curvature-based modeling (e.g., additive, submodular, supermodular forms), reflecting synergy or redundancy between actions. These valuations form the input to our auction-based planner, which computes strategic allocations subject to feasibility and incentive constraints.

F.2.3 Effect of Truthfulness on Allocation

.

We compare allocation behaviors under four conditions: truthful reporting, strategic misreporting, oracle, and greedy heuristic (Figure 5). The greedy allocation baseline assigns to each agent the single action bundle that yields the highest individual valuation, selecting the action with the highest bid per agent without considering global feasibility or incentive alignment. The oracle and greedy allocations are averaged across the samples. Interestingly, we observe that user-type hosts are frequently prioritized in learned allocation, often receiving aggressive actions such as *Remove* and {*Analyze, Remove*}. This may seem counterintuitive compared to static criticality rankings, where enterprises are often considered higher value. However, the Q-values driving our allocation reflect long-term strategic impact, suggesting that early disruption of User hosts may significantly hinder adversarial progress. This behavior aligns with the findings of the CC2 evaluation study [12], which shows that User hosts often serve as stepping stones toward more privileged targets. Thus, our mechanism implicitly learns to act preemptively, prioritizing early-stage containment.

F.2.4 Alignment with Cyber Objectives

To further assess whether the learned allocation mechanism prioritizes hosts involved in more cyber activity, we analyze the correlation between the aggregated allocation scores and the number of Red and Blue actions each host receives during the simulation. Allocation scores are aggregated by adding

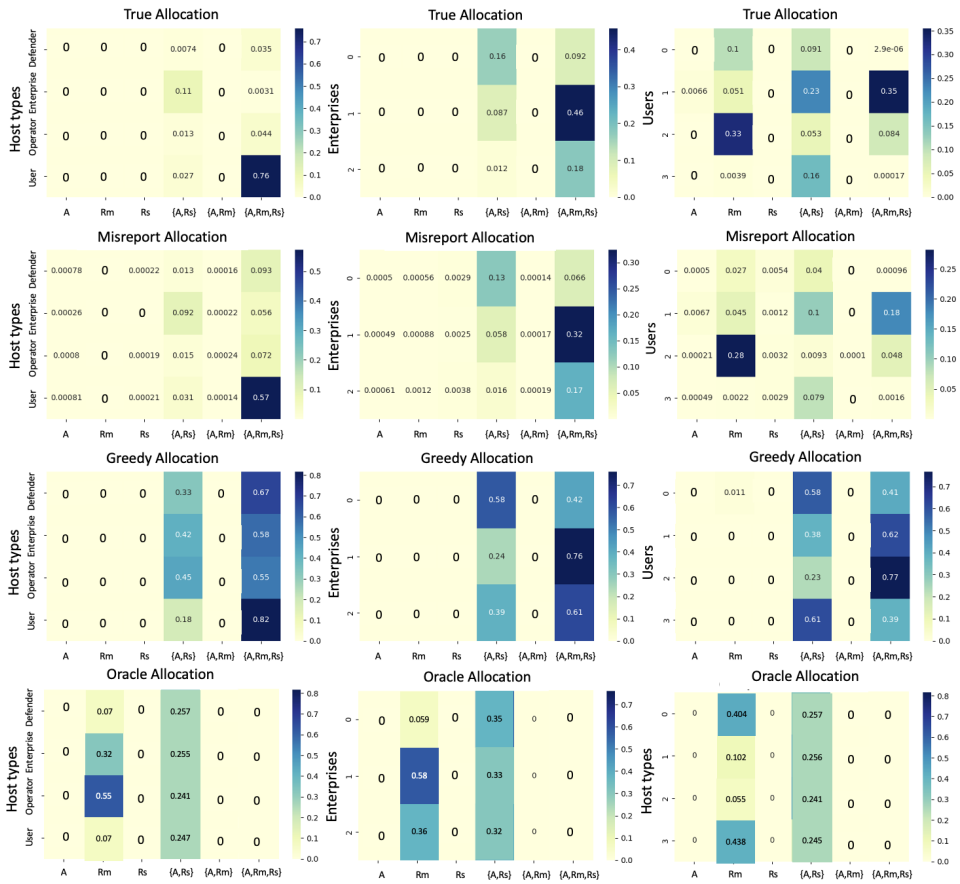

Figure 5: Allocation matrices under truthful vs. perturbed agent valuations. Under truthful reports, the learned mechanism concentrates most allocations on *Remove* and {*Analyze, Remove*} actions, indicating a consistent preference for aggressive defense strategies. When agents misreport, this concentration weakens—allocations become more diffused across action bundles, suggesting that the mechanism is sensitive to manipulation. However, the relative importance of hosts (the allocation order) remains mostly preserved, demonstrating structural robustness. The greedy baseline over-allocates to the most comprehensive bundle {*Analyze, Remove, Restore*}, particularly for user hosts, underscoring the inefficiency of naive valuation-based strategies. This confirms that while our mechanism is not fully strategyproof, it maintains coherent allocation priorities and performs more adaptively than naive methods. While CAFormer and Greedy tends to prefer the comprehensive bundle, the oracle chooses bundle 1 (*Remove*) more often because it can isolate the marginal value of just *Remove* in some contexts. It does not favor the largest bundle as strongly, because it is often not strictly better than the sum of smaller bundles, especially in additive settings.

Correlation	Pearson r (p)	Spearman ρ (p)
Red (all)	0.529 (0.47)	0.800 (0.20)
Blue (all)	**0.964** (0.04)	0.800 (0.20)
Red (enterprises)	0.351 (0.77)	0.500 (0.67)
Blue (enterprises)	0.753 (0.46)	**1.000** (0.00)
Red (users)	0.845 (0.15)	0.800 (0.20)
Blue (users)	0.769 (0.23)	0.600 (0.40)

Table 4: Correlation between allocation scores and counts of Red/Blue actions across hosts. Entries are coefficient (p-value).

the model output in all bundles per host. We compute Pearson's r and Spearman's ρ correlation coefficients.

As shown in Table 4, we observe a strong positive correlation between allocation scores and the number of defensive actions by Blue in all hosts (Pearson $r = 0.964, p = 0.0361$), suggesting that the mechanism tends to focus attention on hosts where defenders are more active. The correlation with Red (attacker) activity is also positive, but weaker and not statistically significant. Within host-type subsets (e.g., Users or Enterprises), correlation trends are directionally similar, though only Blue (enterprises) reaches significance, likely due to the reduced sample sizes. These results indicate that, while the mechanism is not explicitly aware of the underlying mission-criticality metadata, its learned allocations align closely with observed operational activity in the environment.

F.2.5 Distributional Reward Shaping via Auction

We further explore the potential of using the auction-derived allocation output as a guiding signal for the RL agent. The allocation is computed from the Q-values of a converged policy across multiple episodes and 30 time steps, representing a desirable action distribution over host-action pairs. We interpret this as a *target distribution* that reflects strategic prioritization of defensive responses. During training, this pseudo-auction result is applied as a reward shaping signal—penalizing or rewarding the agent based on its alignment with the derived allocation. This approach allows us to inject domain knowledge into the learning process without altering the environment itself. The training curve is visualized in figure Empirical results show measurable improvements in convergence and overall policy performance, summarized in Table 3. The Area Under the Curve (AUC) is computed by integrating the episodic reward over training steps, while the t-statistic and p-value result from a two-sample t-test comparing the shaped and original reward distributions, excluding the initial 50 steps to avoid initialization bias. The shaped agent consistently outperforms the original agent across training episodes. The statistically significant result ($p < 0.001$) and a gain of over 1900 in AUC highlight the effectiveness of auction-guided reward shaping in accelerating convergence and enhancing policy quality.

This initial exploration highlights the promise of using virtual auctions—derived from the Q-values of a converged agent—as a structured reward shaping mechanism. While the current implementation statically applies the auction allocation as a fixed target distribution, future work can explore dynamic or online integration of this signal. Specifically, auction allocations could be periodically updated during training (e.g., every N episodes or steps) to better align with the agent's evolving policy and environmental dynamics. Such adaptive shaping could bridge the gap between offline expert policies and real-time learning. Furthermore, integrating this mechanism into a decentralized training framework—where hosts act as independent agents sharing soft allocation guidance—might provide scalability and robustness in complex, multi-agent environments. Investigating generalization to unseen scenarios, especially beyond the initial 30 evaluation steps, and analyzing robustness under non-stationary adversaries are also important directions.

.

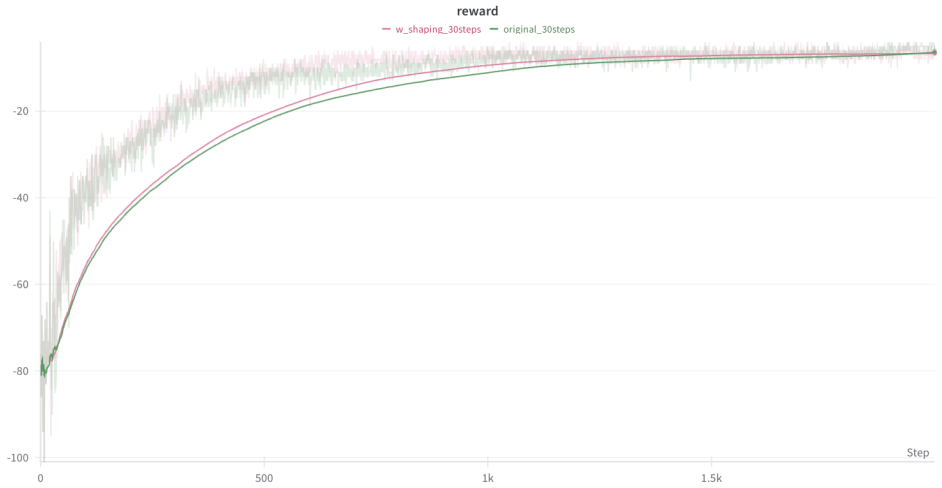

Figure 6: Total reward collected each steps with and without reward shaping, evaluated for 1,000 episodes. The agent learned with reward shaping climbs faster and collects more reward per training step. The difference is statistically significant.

	2×2 **(A)**	2×3 **(A)**	2×5 **(A)**
VCG	0.667 / 0	1.000 / 0	1.671 / 0
AMA	0.860 / 0	– / –	– / –
VVCA	0.866 / 0	– / –	– / –
BLAMA	0.786 / 0	1.222 / 0	2.231 / 0
ABAMA	0.786 / 0	1.255 / 0	2.251 / 0
BBBVVCA	0.776 / 0	1.238 / 0	2.242 / 0
RegretNet	0.878 / 1e−3	1.317 / 1e−3	2.339 / 1e−3
RegretFormer	0.908 / 1e−3	1.416 / 1e−3	2.453 / 1e−3
CANet	0.879 / 1e−3	1.317 / 1e−3	2.282 / 4e−3
CAFormer	0.891 / 1e−3	1.326 / 1e−3	2.329 / 5e−3
CAGraph	**1.111** / 8e−3	**1.640** / 14e−3	**2.671** / 10e−3

m

Table 5: Non-combinatorial results: Our revenue performance is comparable to that of RegretNet and RegretFormer, which outperform heuristic designs. In symmetric (B), the revenue upper bound attains 3.548, 5.467, 9.243 for 2×2 , 2×3 , 2×5; in asymmetric (C): 6.184, 9.296, 15.573.

G Additional Results

G.1 Performance in non-combinatorial setting

We compare the computational results for non-combinatorial setting, where bidders draw their value for each item from $U[0, 1]$ (setting A). In these experiments, we make additive valuation assumption. The revenue performance is reported in Table 5.

Our revenue performance is comparable to the machine learning-powered models, which outperform the heuristic designs. The slight underperformance in comparison to RegretNet and RegretFormer is due to the complex constrained optimization space. For instance, in the 2-agent, 5-item setting, RegretNet and RegretFormer optimize allocations at the agent-item level (2×5), while CANet and CAFormer optimize at the agent-bundle level (2×31), which introduces excessive complexity to non-combinatorial settings, preventing full convergence. Besides, regret estimation is less reliable in larger-scale settings, because adversarial optimization of the inner loss (6) might be inaccurate, which affects the convergence of optimal revenue. If the utility of the best misreport is underestimated, the revenue might be overestimated. Future work is encouraged to evaluate this approximation's accuracy.

H Implementing Randomized Allocations

From fractional Z to feasible outcomes. Our models output fractional allocations $Z \in [0,1]^{n \times k}$. Unlike assignment problems—where Birkhoff–von Neumann exactly decomposes a doubly stochastic matrix—general CA polytopes need not equal the convex hull of integral allocations, so an exact lottery may not exist for arbitrary Z. We therefore implement randomized outcomes with lightweight rounding or an explicit (approximate) mixture of feasible allocations.

(A) One-shot sampler with contention resolution (fast, default). For each bidder i, sample one bundle (or "no bundle") from the row distribution $\{z_{iS}\}_{S \in K} \cup \{1 - \sum_S z_{iS}\}$; sort sampled pairs (i, S_i) by a priority key (e.g., $v_i(S_i)$ or weighted random); traverse in order, accepting (i, S_i) iff S_i is disjoint from items already allocated and i has no prior assignment. This yields an integral, feasible allocation in $O(nk \log nk)$ and preserves "at-most-one-bundle-per-bidder" exactly; marginals are matched approximately.

(B) Dependent/contention–resolution rounding (better marginals). Independently activate (i, S) with probability $x_{iS} = \min\{1, \alpha z_{iS}\}$ for a tuning $\alpha \in (0, 1]$ that controls load, then apply a contention–resolution scheme: process activated pairs in random priority, accept if feasible (no item conflict; bidder unused), otherwise discard. This maintains feasibility and improves alignment with the row marginals.

(C) Column–generation lottery (explicit mixture). Construct a small mixture $\{\theta_t, A^{(t)}\}$ of feasible integral allocations by iteratively solving the pricing IP (winner–determination) on the current residual R: $A^\star \leftarrow \arg\max_{A \in \mathcal{F}} \langle R, A \rangle$, add $A^\star$ as a column, and refit $\{\theta_t\}$ to minimize $\left\| \sum_t \theta_t A^{(t)} - Z \right\|_1$ subject to $\theta_t \geq 0, \sum_t \theta_t = 1$. Sampling $A^{(t)}$ with probability θ_t implements an explicit lottery; exact decomposition is achieved when Z lies in $\mathrm{conv}(\mathcal{F})$.

Payments. If payments are trained as $p_i = \tilde{p}_i \sum_S z_{iS} b_{iS}$, we either (i) charge the expected payment (aligns ex-ante revenue with training) or (ii) recompute/adjust on the realized integral outcome (operationally natural but may shift objectives). In all cases feasibility is preserved and truthfulness remains approximate (low regret).

What we evaluate. For each method we track (i) feasibility rate (always 100% by construction), (ii) deviation of realized marginals from Z (row-wise TV distance), (iii) ex-ante vs. realized revenue/welfare gaps, and (iv) runtime. Due to space, full quantitative comparisons are deferred; we will report these ablations in the extended appendix/supplement.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract/intro state that we enforce combinatorial feasibility in differentiable auction mechanisms and instantiate CANet/CAFormer/CAGraph; Secs. 2, 3, and 4 present the setup and experiments (synthetic + two case studies) that support these claims.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: We discuss limits due to bundle enumeration, non-convex min–max training, randomized allocation implementation, and distribution shift; see Discussion/Conclusion.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: We do not introduce new theorems; our focus is algorithmic and empirical.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Architectures, loss, data generation, seeds, and training schedules are specified in appendix and supplement.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We include an anonymized code bundle with run scripts and instructions in the supplemental; public data sources (DB1B, T-100, CC2/CybORG) and preprocessing steps are documented in Appx. F.1 and F.2.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Training/test splits, regret budgets, temperatures, learning rates, network sizes, and iteration counts are given in Sec.4 (Synthetic Data/Case Studies) and Appx.E.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Results are averaged over three runs; standard deviations are typically ≤ 0.1 for revenue, ≤ 0.001 for regret.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Hardware and resources are reported in Discussion/Conclusion.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We rely on public datasets and simulated environments, no human subjects, and discuss potential impacts and safeguards in Discussion/Conclusion.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We analyze potential positive/negative impacts for airport slots and cyber defense, including policy trade-offs and risks of mis-specified objectives, Discussion/Conclusion.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We use original datasets/tools and credit licenses/terms in the paper.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We release synthetic generators, preprocessing, and training scripts with a README (usage, env, license); details in supplemental.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.

- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.