An Improved Mechanism for Pricing Ride-Hailing Fares

Marek Adamczyk
Institute of Informatics,
University of Wroclaw
Wroclaw, Poland
marek.adamczyk@cs.uni.wroc.pl

Maurycy Borkowski
Institute of Informatics,
University of Wroclaw
Wroclaw, Poland
maurycy.borkowski@cs.uni.wroc.pl

Michał Pawłowski
University of Warsaw
Warsaw, Poland
Sapienza University of Rome
Rome, Italy
michal.pawlowski@mimuw.edu.pl

ABSTRACT

Pricing strategies in ride-hailing platforms present complex optimization challenges, attracting considerable research attention in computer science. Hikima et al. (AAAI 2021) introduced a model for this problem and achieved a 1/3-approximation for maximizing platform profit. This was later improved to a $(1-1/e)$-approximation by Brubach et al. (NeurIPS 2022). In this paper, we extend the problem to a more general and realistic setting.

Firstly, we consider an online stochastic model where customer requests arrive sequentially in a random order. This better reflects real-world scenarios than the offline assumption of known requests. Secondly, we frame the problem within the context of mechanism design, allowing us to benchmark our algorithm against the optimal Bayesian mechanism rather than the more restrictive posted-price mechanisms used in prior work.

Our main contributions include developing a $(1-1/e)$-approximation algorithm under these generalized settings, which we regard as stronger due to the comparison with a more powerful benchmark. The key technical innovation is a novel rounding procedure for fractional matchings. This allows us to devise a new Contention Resolution Scheme (CRS) for transversal matroids, leading to improved approximation guarantees for posted-price mechanisms in combinatorial environments. Specifically, we enhance the ratio from the previous $1/(k+1)$ to $(1-e^{-k})/k$ for the intersection of k transversal matroids.

KEYWORDS

Mechanism Design; Posted-Price Mechanism; Bipartite Matching; Contention Resolution Scheme

ACM Reference Format:
Marek Adamczyk, Maurycy Borkowski, and Michał Pawłowski. 2025. An Improved Mechanism for Pricing Ride-Hailing Fares. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 9 pages.

1 INTRODUCTION

In the past decade, digital platforms have changed the way users access services across many aspects of daily life. Among these, ride-hailing services such as Uber, LYFT, and Bolt have seen rapid growth, reshaping urban mobility. However, the unique demands of this industry often result in a larger number of metrics to optimize, along with greater complexity of the associated problems, compared to many other service-based platforms. Solving these challenges requires advanced methods from computational economics, machine learning, artificial intelligence, and combinatorial optimization. In this paper, we explore the model of dynamic client-to-cab matching for maximizing platform profit, as introduced by Hikima et al. [13], and build upon it to address key challenges within this area.

1.1 The Model of Hikima et al.

Consider how a typical ride-hailing application operates:

- A client opens the app on their phone, selects their destination, and may provide some additional details if needed;
- After some time between seconds and a minute, the client gets a proposed price for the chosen fare;
- The client then decides whether to accept or reject the offer;
- If the offer is accepted, the client waits for either an assignment to a cab or a notification that no cab is available.

But how does this process look like from the perspective of the ride-hailing provider?

- Every minute, the platform receives tens, if not hundreds, of ride requests from its clients;
- The platform has real-time information about available cabs, including which cabs can reach which clients within a reasonable time;
- It must determine the optimal prices to offer each client;
- After receiving clients' responses, the platform computes the optimal assignment of cabs to clients (potentially rejecting some requests) in order to maximize overall profit.

Hikima et al. [13] studied such a model, focusing on maximizing profit for the ride-hailing service. We would like to emphasize that their approach involves a two-stage process. In the first stage, the platform collects ride requests within a time window, typically around one minute, and offers each client a price. In the second stage, after receiving the clients' responses, the platform considers only those who accepted the offers and optimizes the assignment of cabs to clients to maximize overall profit. This process is thus made somewhat offline, as each client learns the outcome only at the end of this entire process. The specific details of their model are as follows.

Graph of Customers and Taxis. We are given two sets, C and T, representing customers and taxis, respectively. For simplicity, we assume that the set of customers is $C = \{1, 2, \ldots, n\}$. Additionally, for each customer-taxi pair, we are given a tuple $(c, t, w_{c,t})$, which indicates that customer c can be served by taxi t, with $w_{c,t}$ representing the cost of the trip. This relationship between customers

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

and taxis can be modeled as a bipartite graph $G = (C, T; E)$, where one set consists of customers, the other of taxis, and the edges represent feasible connections—meaning a taxi can reach a customer within a reasonable amount of time, such as 5 minutes.

Pricing Solution. A key assumption in Hikima's model is that for each client c, we know the probability distribution of their valuation v_c for the trip. The concept of a solution in this model is represented as a vector of prices $(\pi_c)c \in C$, where each πc is the price offered to client c. Based on the price π_c, client c will accept the offer if their valuation $v_c \geq \pi_c$. Given the distribution $\mathcal{V}_c$ of v_c, we can determine that client c accepts the price with probability $\mathbb{P}[v_c \geq \pi_c]$, or rejects it with probability $1 - \mathbb{P}[v_c \geq \pi_c]$.

If client c rejects the proposed price, we remove their corresponding vertex and the incident edges from the graph. Let $G_{ok} = (C_{ok}, T; E_{ok})$ be the graph that results after removing all customers who did not accept their price. From this reduced graph, we select a matching $M \subseteq E_{ok}$, a set of edges where no two share a common vertex. The value of the solution is the sum of the prices minus the trip costs, i.e., $\pi_c - w_{c,t}$, across the edges in the final matching. Here, the weights $w_{c,t}$ represent the cost of the fare, such as driver payments, fuel expenses, and other related costs.

Matching Under Uncertainty. Notice that once the prices $(\pi_c)_{c\in C}$ are set and we have learned the customers' decisions regarding their acceptance—i.e., the vector $(\mathbf{1}[v_c \geq \pi_c])\, c \in C$, or equivalently the set $C_{ok} = \{c \in C | \, v_c \geq \pi_c\}$—our problem reduces to the classic task of finding the maximum matching in a weighted bipartite graph. Let $f(\pi_1, \ldots, \pi_n; C_{ok})$ denote the solution to this maximization problem, which can be formulated as follows:

$$\begin{aligned} \max_{z_{C\times T}\in\{0,1\}^{C\times T}} \quad & \sum_{(c,t)\in E} (\pi_c - w_{c,t}) z_{c,t} \\ \text{s.t.} \quad & \sum_{t\in\delta(c)} z_{c,t} \leq 1[c \in C_{ok}] \quad \forall c \in C, \\ & \sum_{c\in\delta(t)} z_{c,t} \leq 1 \quad \forall t \in T, \\ & z_{c,t} \in \{0,1\} \quad \forall (c,t) \in E, \end{aligned}$$

where $\delta(c)$ denotes the set of taxis incident with customer c, and similarly, $\delta(t)$ denotes the set of customers incident with taxi t. Here, $z_{c,t} \in 0, 1$ indicates whether client c and taxi t are matched ($z_{c,t} = 1$) or not ($z_{c,t} = 0$). The first constraint ensures that only a customer c who has accepted the proposed price can be matched with a taxi, while the second constraint guarantees that each taxi t is matched with only one client.

The Ride-Hailing Problem. Function f represents the weight of the maximum matching for a given set of prices $(\pi_c)_{c\in C}$, based on which customers accept the offered prices. However, before setting the prices $(\pi_c)_{c\in C}$, we do not yet know the realization of the acceptance decisions $(1[v_c \geq \pi_c])_{c\in C}$. Our objective, then, is to find a vector of prices $(\pi_c)_{c\in C}$ that maximizes the expected value of the matching weight, where the expectation is taken over all possible realizations of the acceptance vector $(1[v_c \geq \pi_c])_{c\in C}$.

In other words, our goal is to maximize the following function of the price vector $(\pi_c)_{c\in C}$:

$$\max_{(\pi_c)_{c\in C}\in\mathbb{R}^+} \mathbb{E}_{(v_c)_{c\in C}} \left[f(\pi_1, \ldots, \pi_n; \{c \in C \mid v_c \geq \pi_c\}) \right].$$

We will refer to this problem as the (RH-PROBLEM).

The expected value above can be expressed more intuitively as a weighted sum of the maximum matchings' weights over all possible subsets of clients $C_{ok} \subseteq C$. Specifically, for each subset C_{ok}, its contribution to the sum is given by:

$$\prod_{c\in C_{ok}} \mathbb{P}(v_c \geq \pi_c) \cdot \prod_{c\in C\setminus C_{ok}} (1 - \mathbb{P}(v_c \geq \pi_c)) \cdot f(\pi_1, \ldots, \pi_n; C_{ok}).$$

1.2 Posted Price Mechanisms

Note that the (RH-PROBLEM) directly models the price optimization problem, and as we solve it, we obtain optimal prices for the given model. However, one can observe that the model, as stated, dictates a specific way of interacting with the client: it enforces a take-it-or-leave-it offer at a given price. These types of mechanisms are the so-called Posted Price Mechanisms.

Posted price mechanisms have a rich literature, starting with the paper by Chawla et al. [7]. Very interesting research questions arise around them because, for obvious reasons, they are suboptimal, making it incredibly important to determine the extent of this suboptimality. Despite their inherent limitations, they owe their popularity to their ease of implementation and the simplicity and transparency they provide from the client's perspective. In some cases, such as the ride-hailing problem we consider, they are even the only feasible mechanisms.

It would be hard to imagine implementing an optimal Bayesian mechanism where a client, who wants to get a cab as quickly as possible, would need to first submit a bid, wait for all other clients to submit theirs, and then wait again for the service to calculate the final assignment.

1.3 Room for Improvement

The model of Hikima et al. can be made more realistic and competitive in two key ways: by moving from an offline to an online setting and by comparing it against a stronger benchmark.

Going From Offline to Online Setting. One of the main limitations of Hikima's model is its two-stage procedure, where the platform must wait for all clients to make their decisions before computing the optimal assignment. This offline process has been improved by Brubach et al. [5], who addressed Problem A as proposed by Hikima in Section 7 of [13]. Their approach essentially views the problem as if the prices were already set, allowing the probability function $a(x_c)$ to be defined as $\mathbb{P}(v_c \geq x_c)$. In this problem, they consider an undirected bipartite graph $G = (C, T, E)$, where each node $c \in C$ has a price $x_c \in \mathbb{R}$ given upfront. The process follows two steps that repeat until either C or T becomes empty:

(1) Choose some $c \in C$ and $t \in T$, and attempt to match them, a process known as probing. The probing succeeds with probability $a_c(x_c)$ and fails with probability $1 - a_c(x_c)$;
(2) If the probing succeeds, both c and t are removed from C and T, respectively, and a profit of $(x_c + w_{ct})$ is obtained. If the probing fails, no profit is made, and c is removed from C.

Brubach et al.'s approach introduced a new probing method, enabling the problem to be solved in an online manner, where client requests are handled as they arrive rather than all at once. Their work improves the approximation ratio from the 1/3 achieved by Hikima to $1 - 1/e$. However, despite this improvement, the model's limitation persists in that it only benchmarks performance against other posted price mechanisms.

Stronger Benchmark. In the broader context of mechanism design, the approaches of both Hikima et al. and Brubach et al. leave an important question unanswered: can we design a posted price mechanism that remains competitive against an optimal Bayesian mechanism?

1.4 Our Contribution

In this paper, we build on the problem posed by Hikima et al., linking it to broader research on stochastic combinatorial optimization and contention resolution schemes, which have rich applications in mechanism design [7, 9, 11, 17, 18]. We improve on the known results by presenting a $(1 - 1/e)$-approximation algorithm for the problem of pricing ride-hailing fares.

Our contribution strengthens the work of Brubach et al., who compared their algorithm against an optimal posted-price mechanism. In contrast, we measure our performance against an optimal Bayesian mechanism, which provides the most effective pricing solution in this setup [15], outperforming any posted-price mechanism. This broader comparison sets a stronger benchmark, as posted-price mechanisms are a subset of Bayesian mechanisms.

Algorithm for Ride-Hailing. We introduce an online algorithm that adheres to the posted-price mechanism model of Hikima et al. while also evaluating its performance against an optimal Bayesian mechanism. Our approach begins by formulating a new mathematical program, which we use to upper bound the performance of the optimal Bayesian mechanism. We then present a posted-price mechanism that approximates the optimal solution of this program by a factor of $1 - 1/e$.

This leads to the main theorem of the paper:

Theorem 1.1. *There exists a posted price mechanism for the ride-hailing model of Hikima et al. that provides a $(1 - 1/e)$-approximation of an optimal Bayesian mechanism.*

The proof of this theorem is divided into two parts. In Section 2, we introduce the mathematical program that provides a pricing solution fitting the model of Hikima et al. In Section 4, we demonstrate that there exists a posted-price mechanism that achieves the $(1 - 1/e)$ approximation of its optimal value. Additionally, in the appendix in the full version, we show that the concave program we utilize is stronger than the (RH-PROBLEM) presented earlier.

Novelty of the Approach. Although our result builds upon prior work in contention resolution schemes (CRS), it introduces new insights. On the one hand, we know that for the type of convex program described in Section 2, a $(1 - 1/e)$-balanced CRS exists in matroid environments [14]. However, since our problem involves matching rather than matroid optimization, a custom scheme is necessary. On the other hand, the CRS from Brubach et al. cannot be directly applied in our approach, as the scheme we aim for needs to be, in a way, exactly $(1 - 1/e)$-balanced. By this, we mean that when an edge e has a probability p_e of appearing in the optimal matching, we have to include it in the created solution with precisely $(1 - 1/e)\, p_e$ probability. This stricter requirement arises because the convex program's objective contains both positive and negative terms that depend on p_e.

While this modification could also be applied to the matching CRS of Brubach et al., our approach is self-contained and does not rely on external procedures (i.e., dependent rounding technique by Gandhi et al. [10]). Instead, we employ differential equations to calculate the exact probabilities of key events, such as the availability of a taxi when clients arrive. This method also provides deeper insights into the relationships between elements in the system, especially when processing one element influences another.

Finally, our approach not only addresses the matching environment in the ride-hailing problem but also leads to two new results for transversal matroids and stochastic k-set packing (see the appendix in the full version).

Contention Resolution Scheme for Transversal Matroids. The technique used to prove Theorem 1.1 builds on the theory of matroid optimization under uncertainty, particularly in the context of contention resolution schemes. In addition to solving the matching environment in the problem of Hikima et al., our method generalizes to transversal matroids and their intersections. Previous work achieved a $1/(k+1)$-balanced CRS for the intersection of k transversal matroids, but our approach improves on this result:

Theorem 1.2. *There exists a $(1 - e^{-k})/k$-balanced contention resolution scheme for the intersection of k transversal matroids.*

2 NEW BENCHMARK

To introduce our new benchmark, we first need to present the ride-hailing problem of Hikima et al. within the framework of mechanism design.

Consider the classical Bayesian single-parameter mechanism design scenario, as outlined by Chawla et al. [7]. In this setting, we have a single seller offering a single service to a set C of clients, each $c \in C$ interested in being served. Each client's valuation for receiving the service is represented by a nonnegative random variable v_c, with the valuations being independent of each other and their distributions known upfront. The problem the seller faces is defined by a down-closed family $\mathcal{F} \subseteq 2^C$, which limits the subsets $\bar{C} \subseteq C$ that can be served. In our case, feasibility means finding a matching between $\bar{C}$ and the set T (the taxis), where the size of the matching equals $|\bar{C}|$. This is called a "single-parameter" setting because the seller offers only one type of service. The objective is to design a truthful mechanism that maximizes expected profit.

This problem is well understood and is optimally solved by Myerson's mechanism [15]. However, while Myerson's mechanism is theoretically optimal, it is impractical in real-world settings. In the context of ride-hailing, the mechanism would require collecting bids from clients, computing optimal allocations and prices, and then informing the clients whether they will be served and at what price. Implementing such a system on a ride-hailing platform is unrealistic. A posted-price mechanism, where clients receive a take-it-or-leave-it price, is a far more practical solution.

As mentioned in the previous section, the study of posted-price mechanisms in theoretical computer science began with the seminal work of Chawla et al. [7]. This opened the door to a significant body of research. Notable contributions that are most relevant to our work include those of Yan [18] and Feldman et al. [9].

2.1 Convex Program

The convex program we use in our algorithm is based on the one used by Yan [18]. We refer readers to [18] for a deeper discussion on the so-called ironing of the valuation function, which allows us to assume that the objective function is concave.

Consider a random set of clients $C^* \subseteq C$, $C^* \in \mathcal{F}$ served by Myerson's mechanism, i.e., an optimal truthful mechanism, and let $p_c^* = \mathbb{P}[c \in C^*]$ be the probability that client c gets served. Moreover, let $p_{c,t}^*$ be the probability that the optimal mechanism serves client c with taxi t. Since the mechanism always serves at most one client to at most one taxi, it follows that $(p_{c,t}^*)_{(c,t)\in C\times T}$ is a fractional matching in the bipartite graph between C and T.

Now fix a client c with its probability p_c^* of being served. Imagine for the time being that we focus only on them and we forget about other clients and feasibility constraints between them. We know their valuation distribution $\mathcal{V}_c$ and hence also its cumulative distribution function F_c can be easily determined.

If we would propose to client c price $F_c^{-1}(1-p_c^*)$, then c would accept the price with probability exactly p_c^* and upon acceptance we would earn $F_c^{-1}(1-p_c^*)$. This means that the expected profit would be $F_c^{-1}(1-p_c^*)\cdot p_c^*$. If it would be the case that function $p \mapsto F_c^{-1}(1-p)\cdot p$ was concave, then $F_c^{-1}(1-p_c^*)$ would be the optimal price of serving c under the constraint that we want to serve them with probability p_c^*. However, it may happen that $p \mapsto F_c^{-1}(1-p)\cdot p$ is not concave. In such a case, the optimum expected profit from a single client c is proven to be obtained by proposing to this client a random price for the service.

Based on results by Myerson [15], it follows that the optimal price distribution can be chosen to be a two-price distribution, which can be found by the so-called ironing technique, see Yan [18] for details. We denote by $\overline{R_c}(p_c^*)$ the expected profit of this optimal distribution, which can be shown to be concave in p_c^*. Intuitively, one should think of $p \mapsto \overline{R_c}(p)$ as being a convex closure of the $p \mapsto F_c^{-1}(1-p)\cdot p$ function.

Returning to the general case with multiple clients C, the mechanism design problems for individual clients (which are independent of each other) are less constrained than the original mechanism design problem, where the set of served clients must belong to $\mathcal{F}$. Therefore, the total expected profit, $\sum_{c\in C}\overline{R_c}(p_c^*)$, serves as an upper bound on the expected profit of the optimal mechanism for the ride-hailing problem.

This discussion is summarized in the following Lemma that can be found in Yan [18]:

Lemma 2.1 (Myerson). *Consider client c and their valuation distribution $\mathcal{V}_c$ described by its cumulative distribution function F_c. Suppose we are given a target probability p_c with which we want to serve client c. Then, the price distribution $\mathcal{D}_c$ that maximizes*

$$\mathbb{E}_{\pi\sim\mathcal{D}_c}[\pi\cdot(1-F(\pi))]$$

subject to the constraint that $\mathbb{E}_{\pi\sim\mathcal{D}}[1-F(\pi)] = p_c$ is a two-price distribution, where this distribution as well as the profit $\overline{R}(p_c)$ it gives can be determined from F. Moreover, $\overline{R}(p_c)$ is a concave function.

Hence, the following is a concave relaxation of the original ride-hailing problem.

$$\begin{aligned}\max_{p_{C\times T}\in\mathbb{R}_{\geq 0}^{C\times T}} \quad & \sum_{c\in C}\overline{R_c}(p_c) - \sum_{(c,t)\in E} p_{c,t}\cdot w_{c,t} \\ \text{s.t.} \quad & \sum_{t\in\delta(c)} p_{c,t} \leq 1 \quad \forall_{c\in C}, \\ & \sum_{c\in\delta(t)} p_{c,t} \leq 1 \quad \forall_{t\in T}, \\ & p_c = \sum_{t\in\delta(c)} p_{c,t} \quad \forall_{c\in C}.\end{aligned} \tag{1}$$

The negative sum in the objective function models the expected cost of each fare. The concavity of the objective function allows us to solve the program in polynomial time.

2.2 Discussion

The programming relaxation we build on is similar to that of Yan [18] and Feldman et al. [9]. Yan provided a beautiful argument demonstrating that for any combinatorial environment, a posted price mechanism can approximate the optimal Bayesian mechanism with the same ratio as the correlation gap. This achieves a $1-1/e$ approximation for matroids.

For the matching environment, Yan's approach also provides a constant approximation. However, since the correlation gap for bipartite matchings is between 0.509 and 0.544 [6, 16], one might ask whether this suffices in our model.

It does not. Yan's reduction relies on Chawla et al.'s [7] greedy procedure, which repeatedly probes edges. Here, this would mean offering a client different cabs sequentially, each with a new take-it-or-leave-it price—an infeasible approach in ride-hailing. Moreover, our environment is not a matroid, requiring a tailored solution.

As discussed in Section 1, the convex program's objective function includes both positive and negative terms dependent on p_e. Thus, we design a contention resolution scheme (CRS) that selects elements with carefully chosen probabilities. In Section 4, we present a posted price mechanism using the p_c vector to ensure each client is served with probability *exactly* $(1-1/e)\,p_c$, capturing *exactly* $(1-1/e)$ of the total objective function value.

3 RELATED WORK

The model we use in this paper was first proposed by Hikima et al. [13], who presented a 1/3-approximation to the (RH-PROBLEM). They approached the problem using the following relaxation:

$$\begin{aligned}\max_{z_{C\times T}\in\mathbb{R}_{\geq 0}^{C\times T}} \quad & \sum_{(c,t)\in E}(\pi_c - w_{c,t})z_{c,t} \\ \text{s.t.} \quad & \sum_{t\in\delta(c)} z_{c,t} \leq \mathbb{P}[v_c \geq \pi_c] \quad \forall c\in C, \\ & \sum_{c\in\delta(t)} z_{c,t} \leq 1 \quad \forall t\in T, \\ & z_{c,t} \geq 0 \quad \forall (c,t)\in E.\end{aligned}$$

After solving this program using concave min-cost max-flow methods, they obtained their final solution. They shown that this relaxation relates to the well-known Stochastic Matching Problem [2, 4] and that their algorithm offers a $1/3$-approximation, referencing [4]. However, if they had referred to [2], they could have claimed a stronger $1/2.5$-approximation.

Brubach et al. [5] explored the fact that the relaxation proposed by Hikima et al. is equivalent to a relaxation of a special case of the Stochastic Matching problem. For this particular relaxation, they achieved a $(1 - 1/e)$-approximation.

In this paper, we offer a different perspective by incorporating posted pricing theory, which was initiated by Chawla et al. [7]. We utilize a relaxation technique introduced by Yan [18] and further developed by Feldman et al. [9]. We chose this concave relaxation for its conceptual simplicity, though a linear programming relaxation, as proposed by Gupta and Nagarajan [11], could also be used to achieve similar results.

Our approach to solving the mathematical program is grounded in the broader work on contention resolution schemes. This is closely related to the work of Feldman et al. [9], Adamczyk and Włodarczyk [3], Lee and Singla [14], and Pollner et al. [17], who have developed new contention resolution schemes in the context of mechanism design. The work of Brubach et al. [5] also fits into this line of research.

4 NEW ROUNDING ALGORITHM FOR MATCHING

Before presenting the main result of this paper, we will first discuss a simpler problem that introduces the tools we will use later.

4.1 A Toy Example

In this subsection, we assume that there are n clients sending requests and a single cab available to pick up any of them. As mentioned earlier, the first step in our solution is to estimate the probabilities of each client being served by the optimal mechanism and to set the prices accordingly. Suppose that after following these steps, we have the set $(p_u)_{u \in U}$, where p_u represents the probability that client u gets served. Here, we change the notation for the sets of clients and taxis to U and V, respectively, to introduce more natural notation when working with bipartite graphs.

Our goal is to find a procedure that, for each $u \in U$, selects client u with an overall probability of at least $b \cdot p_u$ for some positive constant b. It is important to note that, for simplicity, we relax the strict condition on the serving probability described in Section 2, which required this probability to be exactly $b \cdot p_u$. This relaxation helps make the analysis more comprehensible. Finally, let us denote the random set of clients who accepted their proposed prices by $A(p)$. Additionally, for the remainder of this paper, we will stop using the terms *clients* and *taxis* when referring to the elements of sets U and V.

4.1.1 The Algorithm. Here, we present an algorithm satisfying that for each $u \in U$ it chooses this element with probability $1 - 1/e$ when conditioned on u being active. Notice that in the appendix in the full version, we analyze a naive approach for this problem that traverses the elements of U in random order and selects the first one that is *active*, i.e., belongs to $A(p)$. Although it has a worse probability of $1/2$ of choosing each element, the proof steps are easier and may make for a good introduction to what follows in this subsection.

To decide in which order to process the elements of U, we assign each $u \in U$ an independent exponential random variable $Y_u \sim \text{Exp}(1)$. We interpret its realization as the arrival time of u. It is easy to notice that such an ordering corresponds to a random permutation as $(Y_u)_{u \in U}$ are equally distributed. Now, to improve the performance of the naive strategy, we add a dumping factor represented by flipping an asymmetric coin whenever an active element arrives. If the toss is successful (heads), we return this element as the solution. Otherwise, we skip it and wait for the next element to arrive. The main difficulty with this approach is to find the right relation between the values of p_u, Y_u and the probability that the coin toss for u results in heads. We decided to use the probability present in the pseudocode in Algorithm 1 (the dumping factor for u is defined using a binary variable D_u). We present an example run of this algorithm in Figure 1.

Algorithm 1 One Item Selection With a Coin Tossing

Given: $p = (p_u)_{u \in U}$ s.t. $\sum_{u \in U} p_u \leq 1$

1: **for each** $u \in U$ **do**
2: generate an exponential random variable Y_u distributed as
$$\mathbb{P}[Y_u \leq t] = 1 - e^{-t}$$
3: **for each** $u \in U$ in increasing order of Y_u **do**
4: **if** u is not active **then** continue
5: generate a binary random variable D_u distributed as
$$\mathbb{P}[D_u = 1] = e^{-p_u(1-e^{-Y_u})}$$
6: **if** $D_u = 1$ **then** return u

4.1.2 The Analysis. To be able to get the exact probability of element z being able to block u from time t to $t + dt$,[1] we introduce the notation of

$$F_u^S(t) = \mathbb{P}\big[u \text{ not blocked until } t \text{ and elements of } S \text{ do not arrive before } t \mid Y_u = t\big]$$

for each $u \in U$ and $S \subseteq U \setminus \{u\}$. The crucial observation is the following.

Lemma 4.1. *Let $S_u = S \cup \{u\}$ for $S \subseteq U \setminus \{u\}$. Moreover, let $p(S) = \sum_{w \in S} p_w$. Then, it holds that*

$$F_u^S(t) = e^{-p(U \setminus S_u)(1-e^{-t}) - |S|t}. \tag{2}$$

Proof. We proceed by induction on the decreasing cardinality of $S \subseteq U \setminus \{u\}$. For the base case, let us consider $S = U \setminus \{u\}$. It is easy to notice that it satisfies the following

$$\begin{aligned} F_u^{U \setminus \{u\}}(t) &= \mathbb{P}\big[u \text{ not blocked until } t \text{ and elements of } U \setminus \{u\} \text{ do not arrive before } t \mid Y_u = t\big] \\ &= \mathbb{P}\big[\text{elements of } U \setminus \{u\} \text{ do not arrive before } t \mid Y_u = t\big] \end{aligned}$$

[1] We say that element z blocked u whenever z is selected by the algorithm, forbidding u to get selected as well. We present the exact conditions that need to be satisfied for an element to block another one in the proof of Lemma 4.1.

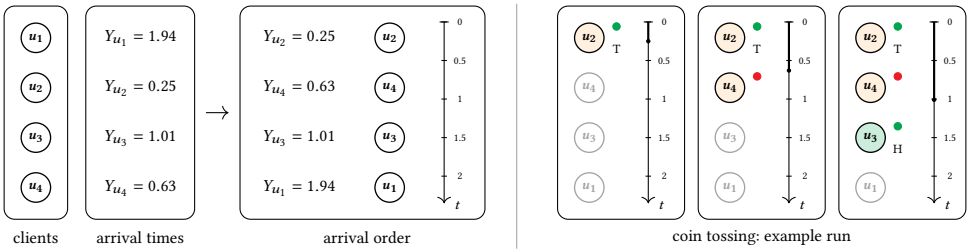

Figure 1: Example on how the arrival order is determined and how the coin tossing algorithm works. Here, we mark active elements by putting a green dot next to them. Otherwise, we place a red dot. Elements rejected by the algorithm are coloured orange, while the selected element is coloured light green. To denote the results of these flips, we write H (heads) and T (tails) next to the element.

since the last event implies u being available at t. Furthermore, if we recall that the minimum of independent exponential variables is exponential with the rate being the sum of components rates [8], we get

$$F_u^{U\setminus\{u\}}(t) \;=\; \mathbb{P}\left[\min_{w\in U\setminus\{u\}} Y_w < t\right] \;=\; e^{-|U\setminus\{u\}|t}.$$

Let us move to the induction step now. We may assume that the hypothesis holds for all sets of cardinality at least k for some $k \in \mathbb{N}_+$. Our goal is to prove that all $(k-1)$-element subsets of $U \setminus \{u\}$ satisfy (2). Let S be any of them.

Suppose that we are at some moment t, item u is still available, and the elements of S have not arrived yet. In order to determine the dependency between $F_u^S(t+td)$ and $F_u^S(t)$, we need to examine the two things that can happen from t to $t+dt$ and affect $F_u^S(t)$. First, an element $w \in S$ may arrive during this time. Such an event happens with probability

$$\begin{aligned}\mathbb{P}\big[Y_w < t+dt \mid Y_w \geq t\big]\cdot F_u^S(t) \;&=\; \mathbb{P}\big[Y_w < dt\big]\cdot F_u^S(t)\\ &=\; \left(1-e^{-dt}\right)F_u^S(t).\end{aligned} \tag{3}$$

Second, if some other element $z \in U \setminus S_u$ has not arrived yet, it can *block* u before $t+dt$. We say that u gets *blocked* by $z \in U$ if we choose z before approaching u, i.e., it holds that: 1) $Y_z < Y_u$, 2) z is active, 3) no element was returned before Y_z, 4) z is not dumped in the coin tossing phase. The probability of this event is

$$\mathbb{P}\big[Y_z < t+dt \mid Y_z \geq t\big]\cdot\mathbb{P}\big[z\in A(p)\big]\cdot\mathbb{P}\big[D_z=1\big]\cdot F_u^{S\cup\{z\}}(t),$$

which simplifies to

$$\mathbb{P}\big[Y_z < dt\big]\cdot\mathbb{P}\big[z\in A(p)\big]\cdot\mathbb{P}\big[D_z=1\big]\cdot F_u^{S\cup\{z\}}(t) \tag{4}$$

as we apply the memoryless property. By the definition of the exponential distribution, we get $\mathbb{P}[Y_z < dt] = 1-e^{-dt}$. We know that each element z is active with probability p_z independently of the other elements, which means that $\mathbb{P}[z \in A(p)] = p_z$. Finally, $\mathbb{P}[D_z = 1] = e^{-p_z(1-e^{-t})}$ by the definition, and $F_u^{S\cup\{z\}}(t)$ can be expressed using the induction hypothesis. Hence, (4) is equal to

$$\left(1-e^{-dt}\right)\cdot p_z\cdot e^{-p_z(1-e^{-t})}\cdot e^{-p(U\setminus(S_u\cup\{z\}))(1-e^{-t})-|S\cup\{z\}|t}.$$

After some modifications, we obtain the following formula

$$\left(1-e^{-dt}\right)\cdot p_z\cdot e^{-t}\cdot e^{-p(U\setminus S_u)(1-e^{-t})-|S|t}.$$

Thus, the probability of an unwanted event happening from t to $t+dt$ is equal to the sum of (3) over elements $w \in S$ plus the sum of (4) over elements $z \in U \setminus S_u$. It holds since the probability of more than one bad event happening during this interval is negligible as $dt \to 0$. After regrouping the components, we obtain

$$\left(1-e^{-dt}\right)\left(|S|F_u^S(t) + p\left(U\setminus S_u\right)\cdot e^{-t}\cdot e^{-p(U\setminus S_u)(1-e^{-t})-|S|t}\right).$$

Hence, we can write that the following equation holds

$$\begin{aligned}F_u^S(t+td) \;=\; F_u^S(t) - \left(1-e^{-dt}\right)\Big(&|S|F_u^S(t) + p\left(U\setminus S_u\right)\\ &\cdot e^{-t}\cdot e^{-p(U\setminus S_u)(1-e^{-t})-|S|t}\Big).\end{aligned}$$

As we move $F_u^S(t)$ to the left-hand side of the equation, divide both sides by dt and take the limit $dt \to 0$, we obtain that $(F_u^S)'(t)$ equals

$$-\left(|S|F_u^S(t) + p\left(E\setminus S_u\right)\cdot e^{-t}\cdot e^{-p(U\setminus S_u)(1-e^{-t})-|S|t}\right).$$

Together with the initial condition $F_u^S(0) = 1$, the differential equation above describes the function given by Formula (2), which concludes the proof. □

Corollary 4.2. *For $S = \emptyset$, we obtain*

$$\mathbb{P}\big[u \text{ not blocked at } t \mid Y_u = t\big] \;=\; F_u^{\emptyset}(t) \;=\; e^{-p(U\setminus\{u\})(1-e^{-t})}.$$

This allows us to prove the following theorem.

Theorem 4.3. *The probability that Algorithm 1 returns element u is $(1-1/e)p_u$ for each $u \in U$. In other words, u is chosen with probability $1-1/e$ when conditioned on being active.*

Proof. Algorithm 1 returns element $u \in U$ whenever it is active, not blocked, and the coin toss for u results in heads (i.e., $D_u = 1$). Thus, we obtain that $\mathbb{P}[u \text{ returned}]$ is equal to

$$\begin{aligned}&\mathbb{E}_{Y_u}\Big[\mathbb{1}\big[u\in A(p)\big]\cdot\mathbb{P}\big[u \text{ not blocked} \mid Y_u\big]\cdot\mathbb{1}\big[D_u=1\big]\Big]\\ &=\; p_u\cdot\mathbb{E}_{Y_u}\Big[e^{-p(U\setminus\{u\})(1-e^{-Y_u})}\cdot e^{-p_u(1-e^{-Y_u})}\Big]\\ &=\; p_u\cdot\int_0^\infty e^{-p(U\setminus\{u\})(1-e^{-t})}\cdot e^{-p_u(1-e^{-t})}\cdot e^{-t}dt\\ &=\; p_u\cdot\int_0^\infty e^{-p(U)(1-e^{-t})-t}dt \;\geq\; p_u\cdot\int_0^\infty e^{-(1-e^{-t})-t}dt\\ &=\; p_u\cdot\left(-e^{-(1-e^{-t})}\Big|_0^\infty\right) \;=\; \left(1-e^{-1}\right)p_u,\end{aligned}$$

which concludes the proof. □

4.2 The Actual Algorithm for Matchings

With all the tools and insights introduced for the toy example, we are ready to tackle the problem of rounding a fractional matching $(p_e)_{e\in E}$ obtained from program (1). Our goal here is to find a procedure that returns a matching M in a bipartite graph $G = (U, V; E)$ for which all its endpoints in U belong to the set $A(p)$ of active elements. At the same time, we want to guarantee that for each edge $e \in E$, the overall probability of adding e to the final matching *equals* $(1 - 1/e)p_e$.

The idea is to first define a procedure that pairs each vertex $u \in U$ with one of its neighbours in V and then for each vertex $v \in V$ choose one of the vertices in U that paired with it (if such exists). Then, the edges whose endpoints selected each other will form the matching in question. We decide to use the controller mechanism described below to select a pairing for each $u \in U$ and an adaptation of Algorithm 1 for the vertices of V.

4.2.1 Preprocessing. Unlike in the previous problem, this one requires us to obtain the exact probability of $1 - 1/e$ of a given edge being added to the final matching. Thus, instead of relying on the inequality $\sum_{u\in\delta(v)} p_{u,v} \leq 1$ that holds for each $v \in V$, we want to find a way to round this sum up to one. For this purpose, for each $v \in V$ we add a new vertex v' to U, connect it with vertex v and define this new edge to be active with probability $p_{v'} = p_{v',v} = 1-\sum_{u\in\delta(v)} p_{u,v}$. Such a procedure gives us a graph $G' = (U', V; E')$, where U' is the set of initial vertices extended by the vertices v' and E' is obtained from E by adding all edges of form $\{v', v\}$ for $v \in V$.

4.2.2 Controller Mechanism. By the definition of program (1), it holds that

$$\forall_{u\in U} \;\; p_u \;=\; \sum_{v\in\delta(u)} p_{u,v} \leq 1. \tag{5}$$

Thus, for each $u \in U$ we can make a randomized choice of a *controller* $c(u) \in V$ following the distribution

$$\forall_{v\in V} \;\; \mathbb{P}\left[c(u) = v\right] \;=\; \frac{p_{u,v}}{p_u}. \tag{6}$$

In other words, we make a weighted choice of an edge to represent u in the matching that we aim to build. From property (5), we have that the probabilities of all possible controller choices for every vertex u sum up to 1. Hence, the procedure is defined correctly. We visualize both the controller mechanism and preprocessing procedure in Figure 2.

4.2.3 The Algorithm. Finally, we are able to introduce the rounding algorithm for the fractional solution $(p_e)_{e\in E}$ obtained from program (1). We use the following approach.

The first step is to transform graph G into G' as described before. Then, to decide in which order to serve the elements of U', we once again use the exponential variables. Next, at the moment of arrival of an active element $u \in U'$, we choose its controller and flip an asymmetric coin to determine the fate of $e = \{u, c(u)\}$. If the toss is successful, we set the value of a helper binary variable $d_{u,c(u)}$ to 1 to inform that the edge is ready to be included in the matching. Then, we check whether there exists another edge $\{z, c(u)\} \in E$ that satisfies $d_{z,c(u)} = 1$. If the answer is positive, such an edge can already be in the matching forcing us to skip e. Otherwise, we add e to the matching. Formally, we obtain the following algorithm.

Algorithm 2 Rounding Fractional Matchings in Bipartite Graphs

Given: bipartite graph $G = (U, V; E)$; fractional matching $p = (p_{u,v})_{\{u,v\}\in E}$ in G
Structures: $d_{u,v}$ for each $\{u, v\} \in E$; $M = \emptyset$

1: transform G to G' as described before
2: choose controllers according to (6)
3: **for each** $u \in U'$ **do**
4: generate an exponential random variable Y_u distributed as
$$\mathbb{P}\left[Y_u \leq t\right] = 1 - e^{-t}$$
5: **for each** $u \in U'$ in increasing order of Y_u **do**
6: **if** u is not active **then** continue
7: generate a binary random variable $D_{u,c(u)}$ s.t.
$$\mathbb{P}\left[D_{u,c(u)} = 1\right] = e^{-p_{u,v}\left(1-e^{-Y_u}\right)}$$
8: **if** $D_{u,c(u)} = 1$ **then**
9: $d_{u,c(u)} \leftarrow 1$
10: **if** $u \in U$ **and** $\forall_{z\in\delta(c(u)),z\neq u}\; d_{z,c(u)} = 0$ **then**
11: add $\{u, c(u)\}$ to M
12: **return** M

4.2.4 The Analysis. First, we have to justify that the resulting set M returned by the algorithm above is a matching. By definition, the controller mechanism guarantees that each $u \in U$ is covered by at most one edge from M. Furthermore, the condition presented in line 10. guarantees that the same holds for each $v \in V$.

Before we prove the main theorem of this paper, let us introduce the notion of blocking between the edges. Let $e = \{u, v\}$ be any edge in E. We say that e gets blocked if we set $d_{z,v} = 1$ for some $z \neq u$, before approaching e, i.e., it holds that: 1) $Y_z < Y_u$, 2) $f = \{z, v\}$ is active, 3) v is chosen as the controller of z, 4) f is considered by the algorithm (dumping factor $D_{z,v}$ equals one). Hence, not blocking e is an event when there exists no such edge f. Thus, if we condition on item u arriving at a given time t, we can define a helper function

$$F^S_{u,v}(t) \;=\; \mathbb{P}\Big[\, \{u, v\} \text{ not blocked until } t \text{ and elements of } S \text{ do not arrive before } t \mid Y_u = t\Big]$$

for each $\{u, v\} \in E$ and $S \subseteq \delta(v) \setminus \{u\}$. The crucial observation is the following.

Lemma 4.4. *Let $\{u, v\} \in E$ be any edge in G, $S \subseteq \delta(v) \setminus \{u\}$, and $S_u = S \cup \{u\}$. Moreover, denote $\sum_{w\in S} p_{w,v}$ by $p(S)$. Then,*

$$F^S_{u,v}(t) \;=\; e^{-p(\delta(v)\setminus S_u)\left(1-e^{-t}\right)-|S|t}. \tag{7}$$

Due to space limitations, we have moved all the proofs from this subsection to the appendix in the full version.

Corollary 4.5. *For $S = \emptyset$, we look at $F^{\emptyset}_{u,v}(t)$ and obtain*

$$\mathbb{P}\left[\, \{u, v\} \text{ not blocked at } t \mid Y_u = t\right] \;=\; e^{-p(\delta(v)\setminus\{u\})\left(1-e^{-t}\right)},$$

This allows us to prove the following theorem.

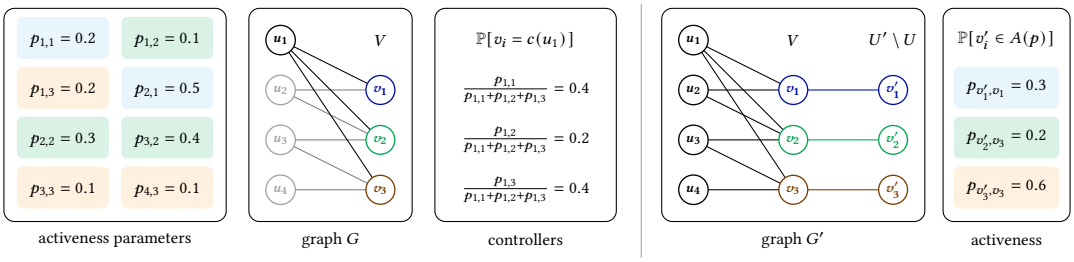

Figure 2: Left side of this figure shows the visualization for the controller selection. In this part, we use the simplified notation of $p_{i,j}$ to represent p_{u_i,v_j}. The graph on the right shows how the preprocessing procedure works. The colours used in this picture are only supposed to help distinguish between the elements of V and the edges incident to them.

Theorem 4.6. *The probability of Algorithm 2 adding edge $e = \{u, v\}$ to the final matching is $(1 - 1/e)p_{u,v}$ for each $e \in E$. In other words, e gets added to the matching with probability $1 - 1/e$ when conditioned on it being active.*

Together with the reasoning given in Section 2, it implies that Theorem 1.1 holds.

5 CONTENTION RESOLUTION SCHEME FOR TRANSVERSAL MATROIDS

In the previous section, we considered the constraints of one-element selection and building up a subset of edges that forms a matching in a given bipartite graph. Both of them could be described using matroids.

Definition 5.1 (Matroid). Consider a pair $\mathcal{M} = (E, \mathcal{I})$, where E is the universe of elements and $\mathcal{I} \subseteq 2^E$ is a family of *independent sets*. We say that $\mathcal{M}$ is a *matroid* if it holds that:

(1) the family $\mathcal{I}$ is down-closed, i.e. $\forall_{A \in \mathcal{I}} B \subset A \implies B \in \mathcal{I}$,
(2) the extension axiom is satisfied, i.e. $\forall_{A,B \in \mathcal{I}} |A| < |B| \implies \exists_{b \in B \setminus A} A \cup \{b\} \in \mathcal{I}$.

In this section, we focus on a more general subclass of matroids called transversal matroids. To work with them, we use the following characterization.

Proposition 5.2. *For any* transversal matroid $\mathcal{M} = (U, \mathcal{I})$ *there exists a bipartite graph $(U, V; E)$ such that $\mathcal{K} = \{S \subseteq U :$ there exists a matching that covers $S\}$ contains exactly the sets that are independent in $\mathcal{M}$, namely $\mathcal{K} = \mathcal{I}$. Moreover, this relation is bijective, meaning that for any bipartite graph $(U, V; E)$, the pair $(U, \mathcal{K})$ describes a transversal matroid.*

Now, let us recall that both problems in Section 4 came down to rounding a fractional solution to make it feasible under the given constraints. Such a problem is known by the name of the contention resolution scheme (CRS). To introduce it in a more general form, we need one more definition first.

Definition 5.3 (Matroid Polytope). For a given matroid $\mathcal{M} = (E, \mathcal{I})$, the convex hull of characteristic vectors of all independent sets in $\mathcal{M}$, namely $\left\{x \in \mathbb{R}^E_{\geq 0} \mid \forall_{I \in \mathcal{I}} \sum_{e \in I} x_e \leq |I|\right\}$, is called the *matroid polytope* $\mathcal{P}(\mathcal{M})$.

Definition 5.4 (Contention Resolution Scheme). Suppose that there is set of constraints $\mathcal{C}$ given as a collection of matroids $\{\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_k\}$ over E and a point $x \in \bigcap_{i=1}^k \mathcal{P}(\mathcal{M}_i)$. A contention resolution scheme is a procedure that takes point x and rounds each coordinate x_i to obtain an integer point that satisfies all the constraints. We say that a CRS is b-balanced if every element is selected with probability at least $b \cdot x_i$.

Finally, we can restate our main theorem regarding transversal matroids and contention resolution schemes.

Theorem 1.2. *There exists a $(1 - e^{-k})/k$-balanced contention resolution scheme for the intersection of k transversal matroids.*

Due to space constraints, we give the proof of this theorem in the appendix in the full version.

6 CONCLUSIONS

In this paper, we provided a new perspective on the ride-hailing problem of Hikima et al. [13], grounding it in the microeconomic context of Myerson's optimal mechanism [15]. Utilizing tools from contention resolution schemes, we devised an algorithm that addresses the ride-hailing problem and generalizes to other combinatorial optimization settings.

Our approach offers both theoretical and practical benefits. Theoretically, it advances the understanding of posted-price mechanisms in complex matching environments, providing improved approximation guarantees. Practically, it is easily implementable using the reduction from Gupta and Nagarajan [11], converting the concave program into a linear one via one-hot encoding of prices.

To demonstrate practicality, we implemented our approach using the open-sourced code of Hikima et al. [12] for the sigmoid valuation model. The prices determined by our algorithm yield similar results to those of Hikima et al., with variations within roughly ±3% on the same dataset, depending on the validation subset.

In ongoing work, we leverage our approach to develop an online algorithm that does not require grouping requests, enabling real-time processing. We also provide additional implementations incorporating ideas from submodular optimization [1].

ACKNOWLEDGMENTS

This research was supported by the Polish National Science Centre (NCN) Grant 2019/35/D/ST6/03060.

REFERENCES

[1] Marek Adamczyk, Maurycy Borkowski, Michał Pawłowski, Mateusz Opala, and Michał Zobniów. 2024. Discovering Ride-Hailing Using Tools from Combinatorial Optimization under Uncertainty. Manuscript in preparation.

[2] Marek Adamczyk, Brian Brubach, Fabrizio Grandoni, Karthik Abinav Sankararaman, Aravind Srinivasan, and Pan Xu. 2020. Improved Approximation Algorithms for Stochastic-Matching Problems. *CoRR* abs/2010.08142 (2020). arXiv:2010.08142 https://arxiv.org/abs/2010.08142

[3] Marek Adamczyk and Michal Wlodarczyk. 2018. Random Order Contention Resolution Schemes. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, Mikkel Thorup (Ed.). IEEE Computer Society, 790–801. https://doi.org/10.1109/FOCS.2018.00080

[4] Nikhil Bansal, Anupam Gupta, Jian Li, Julián Mestre, Viswanath Nagarajan, and Atri Rudra. 2012. When LP Is the Cure for Your Matching Woes: Improved Bounds for Stochastic Matchings. *Algorithmica* 63, 4 (2012), 733–762. https://doi.org/10.1007/s00453-011-9511-8

[5] Brian Brubach, Nathaniel Grammel, Will Ma, and Aravind Srinivasan. 2021. Improved Guarantees for Offline Stochastic Matching via new Ordered Contention Resolution Schemes. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (Eds.). 27184–27195. https://proceedings.neurips.cc/paper/2021/hash/e43739bba7cdb577e9e3e4e42447f5a5-Abstract.html

[6] Simon Bruggmann and Rico Zenklusen. 2022. An optimal monotone contention resolution scheme for bipartite matchings via a polyhedral viewpoint. *Math. Program.* 191, 2 (2022), 795–845. https://doi.org/10.1007/s10107-020-01570-6

[7] Shuchi Chawla, Jason D. Hartline, David L. Malec, and Balasubramanian Sivan. 2010. Multi-parameter mechanism design and sequential posted pricing. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, Leonard J. Schulman (Ed.). ACM, 311–320. https://doi.org/10.1145/1806689.1806733

[8] Rick Durrett. 2019. *Probability: theory and examples*. Vol. 49. Cambridge university press.

[9] Moran Feldman, Ola Svensson, and Rico Zenklusen. 2015. Online Contention Resolution Schemes. *CoRR* abs/1508.00142 (2015). arXiv:1508.00142 http://arxiv.org/abs/1508.00142

[10] Rajiv Gandhi, Samir Khuller, Srinivasan Parthasarathy, and Aravind Srinivasan. 2006. Dependent rounding and its applications to approximation algorithms. *Journal of the ACM (JACM)* 53, 3 (2006), 324–360.

[11] Anupam Gupta and Viswanath Nagarajan. 2013. A Stochastic Probing Problem with Applications. In *Integer Programming and Combinatorial Optimization - 16th International Conference, IPCO 2013, Valparaíso, Chile, March 18-20, 2013. Proceedings (Lecture Notes in Computer Science, Vol. 7801)*, Michel X. Goemans and José R. Correa (Eds.). Springer, 205–216. https://doi.org/10.1007/978-3-642-36694-9_18

[12] Yuya Hikima. 2021. Integrated Optimization for Bipartite Matching and Its Stochastic Behavior. https://github.com/Yuya-Hikima/AAAI-2021-Integrated-Optimization-fot-Bipartite-Matching-and-Its-Stochastic-Behavior/tree/main/. Accessed: 2024-10-16.

[13] Yuya Hikima, Yasunori Akagi, Hideaki Kim, Masahiro Kohjima, Takeshi Kurashima, and Hiroyuki Toda. 2021. Integrated Optimization of Bipartite Matching and Its Stochastic Behavior: New Formulation and Approximation Algorithm via Min-cost Flow Optimization. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 3796–3805. https://ojs.aaai.org/index.php/AAAI/article/view/16497

[14] Euiwoong Lee and Sahil Singla. 2018. Optimal Online Contention Resolution Schemes via Ex-Ante Prophet Inequalities. In *26th Annual European Symposium on Algorithms, ESA 2018, August 20-22, 2018, Helsinki, Finland (LIPIcs, Vol. 112)*, Yossi Azar, Hannah Bast, and Grzegorz Herman (Eds.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 57:1–57:14. https://doi.org/10.4230/LIPIcs.ESA.2018.57

[15] Roger B. Myerson. 1981. Optimal Auction Design. *Mathematics of Operations Research* 6, 1 (1981), 58–73. http://www.jstor.org/stable/3689266

[16] Pranav Nuti and Jan Vondrák. 2022. Towards an Optimal Contention Resolution Scheme for Matchings. *CoRR* abs/2211.03599 (2022). https://doi.org/10.48550/arXiv.2211.03599 arXiv:2211.03599

[17] Tristan Pollner, Mohammad Roghani, Amin Saberi, and David Wajc. 2022. Improved Online Contention Resolution for Matchings and Applications to the Gig Economy. In *EC '22: The 23rd ACM Conference on Economics and Computation, Boulder, CO, USA, July 11 - 15, 2022*, David M. Pennock, Ilya Segal, and Sven Seuken (Eds.). ACM, 321–322. https://doi.org/10.1145/3490486.3538295

[18] Qiqi Yan. 2011. Mechanism Design via Correlation Gap. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, Dana Randall (Ed.). SIAM, 710–719. https://doi.org/10.1137/1.9781611973082.56