Matching Tasks and Workers under Known Arrival Distributions: Online Task Assignment with Two-sided Arrivals

JOHN P. DICKERSON, University of Maryland, College Park, MD, College Park, USA
KARTHIK A. SANKARARAMAN, Meta, Menlo Park, CA, Austin, USA
ARAVIND SRINIVASAN, University of Maryland, College Park, MD, College Park, USA
PAN XU, New Jersey Institute of Technology, Newark, NJ, Newark, USA
YIFAN XU, Southeast University, Nanjing, Nanjing, Jiangsu, China

Efficient allocation of tasks to workers is a central problem in crowdsourcing. In this article, we consider a setting inspired by spatial crowdsourcing platforms, where both workers and tasks arrive at different times, and each worker-task assignment yields a given reward. The key challenge is to address the uncertainty in the stochastic arrivals from both workers and the tasks. In this work, we consider a ubiquitous scenario where the arrival patterns of worker "types" and task "types" are not erratic but can be predicted from historical data. Specifically, we consider a finite time horizon T and assume that in each time-step the arrival of a worker and a task can be seen as an independent sample from two (different) distributions.

Our model, called *Online Task Assignment with Two-sided Arrival* (OTA-TSA), is a significant generalization of the classical online task assignment problem when all the tasks are statically available. For the general case of OTA-TSA, we present an optimal non-adaptive algorithm (NADAP), which achieves a competitive ratio (CR) of at least 0.295. For a special case of OTA-TSA when the reward depends only on the worker type, we present two adaptive algorithms, which achieve CRs of at least 0.343 and 0.355, respectively. On the hardness side, we show that (1) no non-adaptive can achieve a CR larger than that of NADAP, establishing the optimality of NADAP among all non-adaptive algorithms; and (2) no (adaptive) algorithm can achieve a CR better than 0.581 (unconditionally) or 0.423 (conditionally on the benchmark linear program), respectively. All aforementioned negative results apply to even unweighted OTA-TSA when every assignment yields a uniform reward. At the heart of our analysis is a new technical tool, called *two-stage birth-death process*, which is a refined notion of the classical birth-death process. We believe it may be of independent interest. Finally, we perform extensive numerical experiments on a real-world rideshare dataset collected in Chicago and a synthetic dataset, and results demonstrate the effectiveness of our proposed algorithms in practice.

Part of the work was done when K.A.S. was a Ph.D. student at University of Maryland, College Park. There, K.A.S. was supported in part by NSF Awards No. CNS 1010789 and No. CCF 1422569. P.X. was partially supported by NSF CRII Award No. IIS-1948157. A.S. was supported in part by NSF Awards No. CNS-1010789, No. CCF-1422569, No. CCF-1749864, and No. CCF-1918749, and by research awards from Adobe, Inc., Amazon, Inc., and Google Inc. J.D. was supported in part by NSF CAREER Award No. IIS-1846237, NIST MSE Award No. 20126334, DARPA GARD No. HR00112020007, DARPA SI3-CMD No. S4761, DoD WHS Award No. HQ003420F0035, and a Google Faculty Research Award. A preliminary version of this article appeared in the International Conference on Autonomous Agents and Multiagent Systems (AAMAS, 2018).
Authors' addresses: J. P. Dickerson and A. Srinivasan, Brendan Iribe Center for Computer Science and Engineering, 8123 Paint Branch Drive, University of Maryland, College Park, MD 20742; e-mails: john@cs.umd.edu, asriniv1@umd.edu; K. A. Sankararaman, 1 Hacker Way, Meta, Menlo Park, CA 94025; e-mail: karthikabinavs@gmail.com; P. Xu, New Jersey Institute of Technology, 323 Dr Martin Luther King Jr Blvd, Newark, NJ 07102; e-mail: pxu@njit.edu; Y. Xu, Southeast University, 2 Sipailou, Nanjing, China 210096; e-mail: xyf@seu.edu.cn.

ACM 2167-8375/2024/06-ART6
https://doi.org/10.1145/3652021

CCS Concepts: • **Theory of computation → Online algorithms**

Additional Key Words and Phrases: Online matching, crowdsourcing markets, randomized algorithms, matching markets

ACM Reference Format:
John P. Dickerson, Karthik A. Sankararaman, Aravind Srinivasan, Pan Xu, and Yifan Xu. 2024. Matching Tasks and Workers under Known Arrival Distributions: Online Task Assignment with Two-sided Arrivals. *ACM Trans. Econ. Comput.* 12, 2, Article 6 (June 2024), 28 pages. https://doi.org/10.1145/3652021

1 INTRODUCTION

Assigning workers to tasks is a central challenge in various crowdsourcing platforms. For example, in mobile crowd-sensing [50, 51], a central platform allocates mobile users to complex data collection and analysis tasks; in joint crowdsourcing [8, 29], workers answer small questions with varying difficulties; and in spatial crowdsourcing [44, 45], workers and tasks are matched in the context of a spatial metric space. Another type of such assignment, motivated by the COVID-19 pandemic, involves matching medical volunteers and workers to patients and medical tasks during an emergency.

More recently, a special class of worker-task assignment, called ***online task assignment*** **(OTA)**, became popular in the literature. The basic setting is as follows: We have a set of offline tasks (known to us beforehand) and a set of online worker types. Each time a worker of a certain type arrives, an immediate and irrevocable decision is required: either reject the worker or assign her a task she shows interest in. Each assignment yields a known profit (uniform across all assignments or non-uniform); the goal is to design an allocation policy such that the (expected) total profit is maximized subject to various practical constraints such as the total budget for payments for workers, deadlines of tasks, and so on (see, e.g., Assadi et al. [6]). There are three common arriving assumptions for the online workers; see the list below.

- **Adversarial Order (AO):** the arrival sequence is completely unknown but fixed by an adversary;
- **Random Arrival Order (RAO):** the arrival sequence is sampled uniformly at random from the set of all permutations over the unknown set of workers;
- **Known Independent Identical distributions (KIID):** a random worker is chosen with replacement, from a known distribution each time.

Ho and Vaughan [21] considered OTA under RAO, where they assume the profit for each assignment has to be learnt. Assadi et al. [6] studied a budgeted version of OTA under AO and RAO; in the budgeted version, we have a global total budget and each assignment incurs a cost, which is the amount we need to pay the worker (this is equal to the bid the worker submitted for the task after arrival). The budgeted version of OTA and its generalizations have been systematically studied in the context of truthful mechanism design, where the goal is to elicit truthful bids from the online workers (see, e.g., References [17, 18, 40–42, 53]). In particular, Singer and Mittal [41] and Singla and Krause [42] considered the KIID setting, while Zhang et al. [53] and Subramanian et al. [43] considered the RAO setting.

In OTA, the main limiting assumption is that tasks are static (known in advance). This fails to capture various applications where the tasks are not all available at once but come at different times. Hassan and Curry [20] considered a variant of the worker-task assignment problem under the converse setting to that of the OTA, where the spatial tasks arrive dynamically over time while the workers are all available beforehand: the worker has to travel to the specific location of

the task to finish it. Tong et al. [45] studied a generalized setting where both workers and tasks arrive dynamically: This was motivated by a spatial crowdsourcing platform on university campuses, where anyone on campus can both post micro-tasks (e.g., buying drinks or collecting a package), and perform tasks as a worker. They assumed that the arrival sequence of workers and tasks form a random permutation over the set of all possible workers and tasks, which is unknown to the algorithm (i.e., RAO). They tested their algorithms on two real-world crowdsourcing datasets, namely, gMission [11] and EverySender.

Inspired by the above works, we propose a model, called **Online Task Assignment with *Two-sided* Arrival (OTA-TSA)**, where both workers and tasks arrive online each from the KIID arrival setting. A known bipartite graph $G = (U, V, E)$ is given as input (this graph is also called the compatibility graph throughout this article), where U and V represent the respective sets of worker-types and task-types and each type has a specific location attribute, and E represents whether a worker-type has expertise in a task-type or not. We have a finite time horizon T, in which vertices in U and V *both* arrive sequentially in each time-step. We make the common distributional assumptions on the arrival sequence. in each round (for a total of T rounds), a worker of type u is sampled from a known distribution over U, while simultaneously a task of type v is sampled from another known distribution over V independently. Once a task arrives, it has to be instantaneously and irrevocably be matched to one of the workers (from among the neighbors of the task in G) who have arrived before this time instance; we assume that once a worker joins the system they do not leave until matched to a task or the end of the time-horizon, whichever occurs first. Every assignment $f = (u, v)$ yields a given profit w_f, and our goal is to design an allocation policy such that the expected total profit is maximized (the expectation is over both the random online arrivals of workers and tasks and any internal randomization of the algorithm).

We now motivate the key assumptions in OTA-TSA.

Known independent and identical arrival distributions (KIID). In many crowdsourcing platforms, one collects meta-data about the tasks and workers. This data is used to predict both the performance and the arrival patterns of workers and tasks (e.g., References [13, 37, 39, 52, 54]). Hence the underlying compatibility graph G and the arrival sequence of tasks and workers is far from being *adversarial*. We can exploit the rich historical data to predict both the compatibility graph and the arrival distributions of workers and tasks. This motivates us to consider the KIID model, in which we assume the arrivals of tasks and workers follow two separate distributions that are known, identical and independent across the online phase. The KIID arrival setting has been adopted in many previous works (e.g., References [41, 42, 46]).

Retention in the system: workers versus tasks. In our model OTA-TSA, we assume that (a) once a task arrives, it has to be instantaneously and irrevocably assigned to one of the neighboring workers who has arrived so far or get rejected; and (b) once a worker arrives, it will stay in the system until being assigned. OTA-TSA is inspired by applications in *market-based* recommender systems where the two sides of the market are lopsided with more tasks than workers. One example is ride-hailing during peak hours, where drivers (workers) can be far outnumbered by riders (tasks) [38]. Any time a worker joins the system, they have an incentive to stay, since the lopsided numbers of workers and tasks imply that eventually a worker will be assigned to some task.

Adaptive versus non-adaptive algorithms. Non-adaptive algorithms are those whose strategies do not *adapt* to the (random) outcomes observed during previous rounds such as arrivals of online agents and realizations of random seeds used in algorithms. Otherwise, they are adaptive. Thus, a non-adaptive algorithm is a function mapping the time-step to an action, while adaptive algorithms map the history at the given time-step to an action.

Table 1. Summary of Lower and Upper Bounds on Competitive Ratios for OTA-TSA Presented in this Article

	Un-weighted	LHS Vertex-weighted	Edge-weighted
NADAP (Section 4)	≥0.295	≥0.295	≈0.295
GREEDY (Section 6.1)	≥0.250	—	—
ADAP (Section 6.2)	≥0.343	≥0.343	—
SCALED (Section 6.4)	≥0.355	≥0.355	—
Hardness among Non-Adaptive (Section 4)	≤0.295		
Hardness among Adaptive w.r.t. LP Equation (6) (Section 7.1)	≤0.423		
Hardness among Adaptive (Section 7.2)	≤0.581		

Competitive ratio. Consider an input instance of OTA-TSA, denoted by $\mathcal{I} = \{G = (U, V, E), \mathcal{P}_u, \mathcal{Q}_v, \{w_f | f \in E\}\}$, where $\mathcal{P}_u$ and $\mathcal{P}_v$ represent arrival distributions of workers and tasks, respectively. Note that all information in $\mathcal{I}$ is accessible to any algorithm in advance. Consider an offline optimal (OPT), also known as "clairvoyant optimal," which has the privilege of optimizing matching decisions *after* observing the full arrival sequence of online workers and tasks. In contrast, any (online) algorithm ALG must adhere to the instantaneous and irrevocable decision requirement upon the arrival of each task. Let $\mathbb{E}[\text{OPT}(\mathcal{I})]$ and $\mathbb{E}[\text{ALG}(\mathcal{I})]$ denote the expected performance of an offline optimal OPT and any given algorithm ALG on instance $\mathcal{I}$, respectively, where both expectation is taken over randomness in the arrival sequences of workers and tasks and possible random bits in OPT or ALG itself. The competitive ratio of a maximization program, as studied in this article, is defined as $\inf_{\mathcal{I}} \frac{\mathbb{E}[\text{ALG}(\mathcal{I})]}{\mathbb{E}[\text{OPT}(\mathcal{I})]}$ [7]. Thus, when we say ALG achieves a ratio at least $\alpha \in [0, 1]$, it means that for any instance of the problem, the expected profit obtained by ALG is at least an α fraction of the offline optimal. Note that in our definition, the offline optimal can always get a profit w_e with $e = (u, v)$ when matching worker u to task v, irrespective of whether u comes before or after v. This is in sharp contrast with the definition in Reference [47], where they assume both the offline and the online algorithm adopt the same weight function $w(u, v, t, s)$, which denotes the profit obtained when matching a supply agent u arriving at time t to a demand agent v arriving at time s. They set $w(u, v, t, s) = 0$ for all $s < t$, which ensures that the benchmark (i.e., offline optimal) and the online algorithm have the same set of available actions. This, however, makes the offline optimal defined in Reference [47] weaker (less powerful) than ours.

Our contributions. First, we propose a theoretical model, OTA-TSA, where both workers and tasks arrive online. We consider the arrival setting of KIID and assume that the distributions can be learned from historical data.

Second, we present a **non-adaptive algorithm (NADAP)** for the OTA-TSA, which is *optimal* among all possible non-adaptive algorithms (Section 4). We show that NADAP achieves a ratio of almost 0.3, which is larger than the 1/4 achieved by an adaptive algorithm shown in Reference [45] for the same problem but under the arrival setting of RAO. This is theoretical evidence showing the advantage of using historical data to predict the arrival distributions. Our main approach is to construct and solve an appropriate **linear program (LP)** and use that LP solution to guide online actions.

Third, we propose a few adaptive algorithms (see Section 6). The first one is a warmup algorithm, GREEDY. We show that it achieves a competitive ratio of at least 0.25 for the unweighted OTA-TSA; however the analysis is not tight. We conjecture that GREEDY obtains a ratio larger than 0.25, which is supported in part by the experimental analysis of Section 8. The second is an **adaptive algorithm (ADAP)** for OTA-TSA when all the edges incident to each worker—representing the set of all acceptable tasks for that worker—have the same weight, which is referred to as **left-hand-side (LHS)** vertex-weighted OTA-TSA. We show that ADAP achieves an improved ratio

of nearly 0.34. To accomplish this, we construct and solve a *stronger* LP than the one used for the non-adaptive algorithm and combine this with ideas previously used for other online matching problems. The last algorithm is called SCALED, which can be viewed as a boosted version of NADAP: it samples an available worker for each arriving task following a strengthened distribution extracted from the LP. We show that SCALED achieves a competitive ratio of at least 0.355 for LHS vertex-weighted OTA-TSA.

Fourth, we show conditional and unconditional negative results for OTA-TSA: no (adaptive) algorithm can achieve a ratio better than 0.423 using the benchmark LP and 0.581 unconditionally, even for the unweighted case (Section 7). Note that Brubach et al. [9] gave an adaptive algorithm, which yields a ratio of 0.729 for the classical online matching on an unweighted bipartite graph under KIID but with only one-sided arrival. This formally corroborates our intuition that the complexity increases significantly from one-sided arrival to two-sided arrival. Table 1 summarizes the main theoretical findings in this article.

Finally, we run numerical experiments to illustrate the various aspects of our algorithm. First, we run simulated experiments to show the behavior of the algorithm on various ranges of the input space. We illustrate when the proposed algorithm ADAP is better than GREEDY and when an algorithm like GREEDY suffices. Second, we use two real-world *crowdsourcing* datasets, namely, gMission [11] and EverySender [45] to show how to use this algorithm in practice. We find that despite having provable guarantees, we are able to obtain much better performance by using the GREEDY algorithm. Our experimental analysis also generalizes this model, where we assume that at each time-step a batch of workers and a batch of tasks arrive, respectively. Third, we use the Amazon product review dataset to simulate a modern recommender system and report the performance of our algorithm on this dataset.

In the analysis, we define and use a new technical tool, called *two-stage birth-death process*, which abstracts the random process into a general framework. This technical tool might be of independent interest to prove competitive ratios in other settings.

Closely-related works. Tong et al. [46] considered a similar online task assignment problem in spatial platforms, called **Flexible Two-sided Online task Assignment (FTOA)**. The two models are incomparable. The work of Reference [46] posits that in each round *a single vertex*, either a worker or a task, is sampled from a known distribution; in contrast, we assume both a worker and a task are sampled according to two respective distributions independently in each round. Additionally, the work of Reference [46] assumes: (1) unit weights on all assignments (unweighted); (2) the worker and task stay in the system until the deadline (known as input) after arrival; and (3) any available task-worker pair in the system can be matched (thus, a decision for the arriving vertex is not required to be *immediate*).

Recently, Truong and Wang [47] introduced a more general online bipartite matching model with two-sided arrival where the two sides are the supply and demand agents. Similar to our work, they assume that the supply and demand agents arrive in each time-step sampled independently from two known distributions. Also, they assume that once a demand agent arrives, it should either be *immediately and irrevocably* matched to an available supply agent or discarded. In their model, the arrival distributions of supply and demand agents can differ over time. Additionally, they assume that the supply agent departs from the system after a certain finite number of rounds. Finally, when matching a demand agent v arriving at time t to a supply agent u arriving at time s, they assume that the resulting (known) profit is jointly determined by the two matching agents and the difference in their arrival times. Under this general setting, they get lower and upper bounds on the competitive ratio of $1/4$ and $1/2$, respectively. Apart from the model, their work differs from ours in the benchmark as well. In particular, they have a weaker offline optimal benchmark compared to our work (see further discussion in the definition of competitive ratio).

Table 2. Summary of Related Works

	Un-weighted	LHS Vertex-weighted	Edge-weighted	One-sided Arrivals	Two-sided Arrivals	Adversarial	Random Order	Known Distributions
Feldman et al. [15]	✓			✓				✓
Tong et al. [46]	✓			✓				✓
Huang et al. [22]			✓		✓		✓	
Truong and Wang [47]			✓		✓			
Huang et al. [23]		✓		✓		✓		
Collina et al. [12]			✓		✓			✓
Aouad and Saritaç [2]			✓		✓		✓	
Huang and Zhang [25]		✓		✓			✓	
Huang et al. [24]			✓		✓		✓	
MacRury and Ma [32]			✓		✓		✓	
Our Work	✓	✓	✓		✓			✓

Collina et al. [12] considered a problem of two-sided arrival in a non-bipartite setting. They assumed both the arrival and departure of vertices are specified by a (specific) Poisson process and presented an online-matching algorithm that achieves a competitive ratio of 1/8. Recently, Aouad and Saritaç [2] considered a two-sided dynamic matching problem for both cost minimization and reward maximization. The key difference from our work is that in Reference [2], the performance of a policy is evaluated against an *optimal online policy* instead of an *optimal offline* as used here, where the former ratio is called *approximation ratio*, while the latter is called *competitive ratio*. The authors acknowledged that the benchmark LP there could have an arbitrarily large gap from the optimal offline performance. They cited that fact to justify their benchmark choice of an online optimal. A series of recent works ([22, 24, 25] have considered the two-sided arrival model (called *fully online matching*) in the adversarial and random order arrival for unweighted graphs. Since the arrival settings there are more general than the one considered in this article, while the unweighted-graph assumption is stronger than those considered in this article, the results are not directly comparable.

Other related works. We now briefly overview related research for the classical online matching problem; for a more in-depth review, we direct readers to the survey by Mehta [35]. Ever since the seminal work [28], online matching and related models have received significant attention during the last two decades, partly fueled by Internet advertising businesses such as Google and Meta. In the basic model, we are given a bipartite graph $G = (U, V, E)$ where U and V represent, respectively, the *offline* advertisers and *online* keywords (impressions). Each time a vertex $v \in V$ arrives, we have to make an instant and irrevocable decision: either reject it, or assign it to an unmatched neighbor $u \in U$ and obtain a profit w_e for the match $e = (u, v)$. The central problem is to design an online allocation policy such that the expected profit is maximized. Assumptions on the online arrival process leads to a landscape of different problems. Common arrival assumptions include adversarial order, random arrival order [23], and KIID [9, 15, 19, 26, 33]. Online (bipartite) matching models have wide applications beyond online advertising, e.g., online advance admission scheduling [48], kidney exchange and organ matching [3, 5, 34], social welfare maximization [1], efficient parking allocation [36], and dynamic recommendations [10, 30].

Departing from the traditional one-sided arrival setting on a bipartite graph, there are several theoretical models that consider a full arrival setting on general graphs. Wang and Wong [49] introduced a full arrival setting on general graphs with fractional matching. The basic setting is as follows: in each round a single vertex arrives (in an *adversarial* order) and all its incident edges to previously arrived vertices are revealed; once a vertex v arrives, an online action of either rejecting

v or (fractionally) matching it to its neighbors has to be chosen. Dutta and Sholley [14] considered a similar full arrival setting but assumed that each arriving vertex can be matched to only one of the d preceding vertices. Ashlagi et al. [4] studied another interesting variant of the full arrival setting where every agent stays in the system for at most d rounds after arrival and in any round we can match arbitrary pairs of agents currently present in the system. Huang et al. [22] considered a full arrival setting where an online action is required only at the deadline of the vertex. Table 2 provides an overview of the related works.

Roadmap. The rest of this article is organized as follows. We give a general statement of our problem in Section 2. We show our key technical tool, a **two-stage birth-death process (TS-BDP)**, in Section 3 and the benchmark linear program in Section 4. We present one non-adaptive and three adaptive algorithms and related online-competitive analysis in Sections 5 and 6 and offer conditional and unconditional negative results in Section 7. We describe our experimental results in Section 8, offer detailed proofs involved in TS-BDP in Section 9, and conclude our article in Section 10.

2 PROBLEM STATEMENT

Before describing our OTA-TSA model, we define the following terminologies. We group a set of similar tasks and call them "task types." Similarly, we group similar workers and call them "worker types." For example, in the context of spatial crowdsourcing, all workers present around a particular location belong to a single worker type.

Our model is as follows. We have a bipartite graph (known to the algorithm) $G = (U, V, E)$, where U and V represent the set of worker-types and task-types, respectively, and E represents the set of worker-task pairs that are "compatible," i.e., $(u, v) \in E$ iff any worker of type u shows interest toward tasks of type v. We have a finite time horizon T (known beforehand) and for each time $t \in [T] \doteq \{1, 2, \ldots, T\}$, a worker u from U and a task v from V is sampled (we also say u or v *arrives* or *comes* interchangeably) independently from *known* probability distributions $\mathcal{P}_u = \{p_u\}$ and $\mathcal{Q}_v = \{q_v\}$, respectively, with $\sum_u p_u = 1$ and $\sum_v q_v = 1$. The sampling process is independent across the whole online T time-steps (or rounds). Note that under the current setting, the total number of arrivals of workers is equal to that of tasks, and both are equal to T.

At each time $t \in [T]$, we first observe the online arrivals from U and V (in that order), say u and v. We then need to make an *instantaneous and irrevocable decision* to either reject v or assign v to one of its available compatible workers in U. For each $u \in U$, once it arrives, it will stay in the system until being assigned to some task.[1] In our model, we assume that each u has an integral arrival rate,[2] i.e., $T \cdot p_u$ is an integer for every u, and thus, we can further assume w.l.o.g. this integer to be 1 (by splitting each u into $T \cdot p_u$ copies).[3] Hence, we have that $|U| = T$ and $p_u = 1/T$ for all u. Similar to most of the theoretical works in online matching [9, 19, 26, 33], we assume $T \gg 1$ and some of our results are obtained when assuming T approaches infinity. Note that these two assumptions (i.e., $T \gg 1$ and $T \cdot p_u$ being an integer) are mild. When T is small, we can use a dynamic programming-based solution at each time-step to solve it to near optimality. And when $T \to \infty$, the quantity $T \cdot p_u$ is arbitrarily close to an integer. We associate a non-negative profit w_f with each assignment $f = (u, v)$. Let $r_v = T \cdot q_v$ denote the expected number of arrivals of v

[1] Here, we assume that each worker has the capacity to perform only one task. This is w.l.o.g., since if a worker type u can perform multiple tasks, we can split u into multiple copies each with a capacity to perform one task. This establishes the *matching constraint* for each worker.

[2] This is a common assumption in the classical online bipartite matching under known distributions, see Reference [15] for a discussion on the motivation and technical reasons for this assumption.

[3] This is without loss of generality for theoretical competitive-ratio analysis, though it may increase computational complexity when applied in practice.

Table 3. Glossary of Notation and Acronyms Used

$[T]$	Set of times that is equal to $\{1, 2, \ldots, T\}$ with T being the whole time horizon.
$G = (U, V, E)$	Input compatibility graph where U and V are sets of worker and task types, respectively.
E_u (E_v)	Set of edges incident to u (v).
p_u	Arrival probability of worker (of type) $u \in U$ in each round.
q_v	Arrival probability of task (of type) $v \in V$ in each round.
r_v	Expected number of arrivals of task v with $r_v = T \cdot q_v$.
$f = (u, v)$	Edge $f \in E$ that also refers to the match of u and v.
w_f	Non-negative profit on edge e.
e (Non-italic)	Natural base taking the value around 2.718.
KIID	Known independent and identical arrival distributions.
AO/ RAO	Adversarial Order/Random Arrival Order.
OTA	Online Task Assignment.
OTA-TSA	Online Task Assignment with Two-sided Arrival (our main problem); see Section 2.
TS-BDP	Two-Stage Birth-Death Process; see Definition 3.1 in Section 3.
$\kappa(q)$	Expected number of total deaths in TS-BDP$(1, q)$; see the paragraph below Definition 3.1 in Section 3.
NADAP	A non-adaptive; see Algorithm 1 in Section 4.
GREEDY	Greedy; see Algorithm 2 in Section 6.1.
ADAP	An adaptive; see Algorithm 3 in Section 6.2.
SCALED	An adaptive; see Algorithm 4 in Section 6.4.
UR	A heuristic that samples an available worker uniformly at random for each arriving task.

during the T rounds, which is referred to as the arrival rate of v. We assume this rate to be any number between $[0, 1]$ (upper bounding it by 1 is again w.l.o.g. via simple scaling). Our goal is to design an online-assignment policy such that the total expected profit of all assignments made is maximized.

Throughout this article, we use edge $f = (u, v)$ and the assignment of v to u interchangeably. Additionally, when we say at time $t \in [T]$, we mean we are *at the beginning of time t* either before or after observing the arrivals from U and/or V (clarified in the context) but definitely before the algorithm has made any online action. Table 3 offers a glossary of notation and acronyms used in this article.

3 TWO-STAGE BIRTH-DEATH PROCESS

We propose a new stochastic process, TS-BDP, and use it as a main technical tool to analyze our algorithms and derive negative results. Readers may wish to skip this section temporarily to get a coherent picture of how we design and analyze our algorithms: the technical tools presented here are solely for theoretical analysis purposes.

The process (described on random variables $\{X_t, Y_t\}$ and parameterized by values p and q) is described as follows. Consider a stochastic process with a time horizon T such that, (1) the process starts at $t = 1$ with $X_1 = 0$; (2) at every round t, first there is a birth event *followed by an independent death event*. For the birth event, we have $Y_t = X_t + 1$ with probability p/T and $Y_t = X_t$ with probability $1 - p/T$. For the death event, it has a left boundary point at 0; i.e., if $Y_t = 0$, then $X_{t+1} = Y_t$, else when $Y_t \geq 1$, we have $X_{t+1} = Y_t - 1$ with probability q/T and $X_{t+1} = Y_t$ with probability $1-q/T$. We refer to p and q as the *birth* and *death* rates of TS-BDP, respectively. TS-BDP differs from the classical **birth-death process (BDP)**, since in BDP we have that in every round, either a (random) birth or a death event occurs while in TS-BDP the two events occur *independently* in a *sequential* manner (the birth event is followed by the death event). Thus, TS-BDP is a special case of non-uniform BDP. (TS-BDP is a BDP with a time-horizon $2T$ where every odd step is a birth event and every even step is a death event.)

Definition 3.1. A two-stage birth-death process parameterized by (T, p, q) (time horizon, birth rate, death rate) refers to a sequence of random variables $\{X_t, Y_t | t \in [T]\} \cup \{X_{T+1}\}$, which satisfies

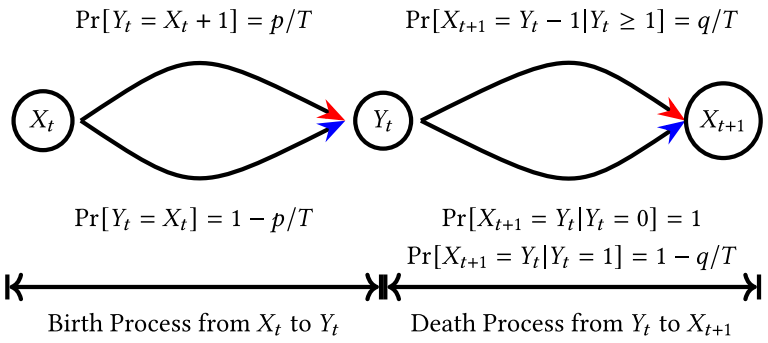


Fig. 1. The random process in a single round of the two-stage birth-death process (TS-BDP) parameterized by (T, p, q). TS-BDP starts with $X_1 = 0$ and consists of T rounds; in each round, it first goes through a birth process of rate $p \geq 0$ and then followed by another independent death process of rate $q \geq 0$.

(1) $X_1 = 0$ with probability 1; (2) For every $t \in [T]$, $Y_t = X_t + 1$ with probability p/T and $Y_t = X_t$ otherwise; (3) For every $t \in [T]$ after the completion of Step (2), if $Y_t = 0$, then $X_{t+1} = Y_t$ with probability 1; if $Y_t \geq 1$, then $X_{t+1} = Y_t - 1$ with probability q/T and $X_{t+1} = Y_t$ otherwise. Figure 1 offers a depiction of the random process in a single round.

In this article, we are particularly interested in the case when $p = 1$, $q \geq 0$ is a constant for a sufficiently large T ($T \to \infty$). We denote this specialization with TS-BDP$(1, q)$ (or TS-BDP(q) when the context is clear). For every $t \in [T]$, let $\Delta(t, T) := Y_t - X_{t+1}$ and $\Delta(T) := \sum_{t \in [T]} \Delta(t, T)$, the latter can be interpreted as the total number of death events in which Y_t gets decreased by 1. Let $\kappa(q) := \lim_{T \to \infty} \mathbb{E}[\Delta(T)]$. We now state some useful lemmas that we will use later. The detailed proofs are deferred to Section 9 and they involve computer-aided computations that were done on Mathematica 10. All numerical results are precise up to the third decimal place.

Lemma 1. *(1)* $\kappa(0) = 0$, *(2)* $\kappa'(0) = 1/\mathrm{e}$, *where* $\kappa'(0)$ *is the first (right) derivative of* $\kappa(q)$ *at* $q = 0$.

Lemma 2. *(1)* $0.295 \leq \kappa(1) \leq 0.302$; *(2)* $\kappa(1 + \frac{1}{\mathrm{e}(\mathrm{e}-1)}) \geq 0.343$; *and (3)* $\kappa(1 + \frac{1}{\mathrm{e}-1}) \leq 0.423$.

Lemma 3. $\kappa(q)$ *is non-decreasing and concave over* $q \in [0, \infty]$.

4 BENCHMARK LINEAR PROGRAM (LP)

As is common in this line of work, our algorithms use optimal solutions to LP constructed on the offline (expected) graph as a guide to the online algorithm. Additionally, this *benchmark* LP is used to upper bound the expected value of the optimal solution on a particular (offline) instance. Hence to compute a lower bound on the competitive ratio, it suffices to compute the ratio of the reward obtained by the algorithm to the optimal solution of this benchmark LP. We now describe the benchmark LP we use for our non-adaptive algorithm. Later, we show that this can further be strengthened based on some observations, which is used in our adaptive algorithm.

We associate a variable with every edge f in the graph. For each edge f, x_f denotes the expected number of matches in any offline optimal matching. For each u (resp. v), let E_u (resp E_v) be the set of its neighboring edges. Consider the following LP:

$$\max_{\{x_f : f \in E\}} \sum_{f \in E} w_f x_f, \tag{1}$$

$$\sum_{f \in E_v} x_f \leq r_v \qquad \forall v \in V, \tag{2}$$

$$\sum_{f \in E_u} x_f \leq 1 \qquad \forall u \in U, \tag{3}$$

$$x_f \geq 0 \qquad \forall f \in E. \tag{4}$$

Lemma 4.1. *The optimal value to* LP *Equation* (1) *is a valid upper bound for the offline optimal.*

Proof. Let $\mathbf{x} := (x_f)$ denote the optimal distribution over the edges for the offline optimal. We now argue that $\mathbf{x}$ is feasible for this linear program, which yields the lemma above. Constraint Equation (2) captures the fact that the expected total number of matches incident to a task v is no more than the expected number of arrivals of v. The same reasoning applies to Constraint Equation (3) but for workers. Constraint Equation (4) follows from the fact that the expected number of matches is non-negative. The objective function computes the expected reward obtained in the optimal offline solution. Thus, we claim that for any offline optimal, (x_f) should be feasible to the LP above. We therefore have that LP Equation (1) is a valid benchmark LP, i.e., its optimal value is an upper bound on the offline optimal. □

Throughout the article, we use (x_f^*) to denote an optimal solution of benchmark LP (or the stronger LP we define later, as appropriate).

5 NADAP: A BEST POSSIBLE NON-ADAPTIVE ALGORITHM

In this section, we present a non-adaptive algorithm, denoted by NADAP, which is *optimal* among all possible non-adaptive algorithms. Algorithm 1 describes our algorithm formally.

ALGORITHM 1: An optimal non-adaptive algorithm (NADAP)

1 Let v be the task arriving at time $t \in [T]$. Recall that E_v be the set of edges incident to v in the input graph.
2 Sample an edge $f = (u, v) \in E_v$ with probability x_f^*/r_v. If worker u is available, then assign v to u; otherwise, skip v.

Constraint $\sum_{f \in E_v} x_f^*/r_v \leq 1$ in LP Equation (2) implies that line 2 in NADAP always forms a distribution.

Theorem 5.1. *The non-adaptive algorithm* NADAP *achieves a competitive ratio of* $\kappa(1) \geq 0.295$ *for the* OTA-TSA.

Proof. Consider a given u. Let X_t and Y_t be the number of copies of u before and after the arrival process for U at time t, respectively. From the assumption that u arrives with probability $1/T$ in each round, we have $Y_t = X_t + 1$ with probability $1/T$ and $Y_t = X_t$ with probability $1 - 1/T$. From NADAP, we have that if $Y_t \geq 1$, then it decreases by 1 with probability $\frac{x_u^*}{T} \doteq \sum_{f \in E_u} \frac{x_f^*}{r_v} \frac{r_v}{T} \leq \frac{1}{T}$ and it remains unchanged with the remaining probability (here $x_u^* = \sum_{f \in E_u} x_f^*$). From the definition of TS-BDP in 3.1, we have that $\{X_t, Y_t\}$ is a TS-BDP$(1, x_u^*)$ with a time horizon of T.

Let A_f be the (random) number of matches for $f = (u, v)$ in NADAP across the T online rounds. Thus, we have

$$\begin{aligned}\mathbb{E}[A_f] &= \sum_{t\in[T]} \frac{r_v}{T}\frac{x^*_f}{r_v} \Pr[Y_t \geq 1] = \sum_{t\in[T]} \frac{x^*_f}{T} \Pr[Y_t \geq 1] = \frac{x^*_f}{x^*_u} \sum_{t\in[T]} \frac{x^*_u}{T} \Pr[Y_t \geq 1] \\ &= \frac{x^*_f}{x^*_u} \cdot \sum_{t\in[T]} \mathbb{E}[\Delta_t] \text{ (by the definition of } \Delta_t) \\ &= x^*_f \cdot \frac{\kappa(x^*_u)}{x^*_u} \text{ (by taking } T \to \infty \text{ and by definition, } \kappa(x^*_u) = \lim_{T\to\infty} \textstyle\sum_{t\in[T]} \mathbb{E}[\Delta_t]).\end{aligned}$$

From Lemma 3, we have that κ is non-decreasing and concave over $[0, 1]$. Thus, $\frac{\kappa(x)-\kappa(0)}{x-0}$ should be non-increasing over $x \in [0, 1]$. We also have that $\kappa(0) = 0$. Therefore,

$$\frac{\kappa(x^*_u)}{x^*_u} = \frac{\kappa(x^*_u) - \kappa(0)}{x^*_u - 0} \geq \frac{\kappa(1) - \kappa(0)}{1 - 0} = \kappa(1).$$

This implies that we have that $\mathbb{E}[A_f] \geq x^*_f \cdot \kappa(1)$. Since LP Equation (1) is a valid upper bound on the performance of an optimal offline, by linearity of expectation, we have that NADAP achieves a competitive ratio of $\kappa(1)$. □

Negative results for non-adaptive algorithms. We now show that any algorithm that is non-adaptive, cannot achieve a ratio better than $\kappa(1)$. In particular, we prove the following lemma.

Theorem 5.2. *No non-adaptive algorithm can achieve a competitive ratio better than* $\kappa(1)$ *even for the unweighted* OTA-TSA.

Proof. Consider an unweighted complete bipartite graph $G = (U, V, E)$ where $|U| = T, |V| = T^2$. Set $p_u = 1/T$ for each u and $p_v = 1/T^2$ for each v. In other words, each u has an integral arrival rate of 1 while each v has an arrival rate of $r_v = 1/T$.

Let OPT-A and OPT-B be the offline optimal algorithm and online optimal non-adaptive algorithm, respectively. With a slight abuse of notation, we use OPT-A and OPT-B to denote the corresponding performance as well. Observe that OPT-A $= T$ for every offline instance. Now, we analyze OPT-B. Since the probability that any v comes at least twice across the T rounds is arbitrarily small (when $T \to \infty$), hence it suffices for OPT-B to specify the online matching policy only for the case when v comes for the first time. Let the policy in OPT-B be $\{x_{u,v} | u \in U, v \in V\}$ such that $x_v \doteq \sum_{u\in U} x_{u,v} \leq 1$. Let $x_u \doteq \sum_{v\in V} x_{u,v}$ for each u. Notice that $\sum_u x_u = \sum_v x_v \leq T^2$.

Consider a given u. Let Z_u be the expected number of matches of u in OPT-B across the T online rounds. The number of copies of u at t before and after an arrival from U can be captured by a TS-BDP with a birth rate of 1 and a death rate of x_u/T, i.e., the corresponding probabilities of a birth and death events are $1/T$ and x_u/T^2, respectively. Thus, by definition, we have $\mathbb{E}[Z_u] = \kappa(x_u/T)$. The performance of OPT-B is $\sum_u \mathbb{E}[Z_u] = \sum_u \kappa(x_u/T)$. Define $\alpha_u := x_u/T$. We have $\sum_u \alpha_u \leq T$ due to the fact that $\sum_u x_u \leq T^2$. Consequently, The (optimal) performance of OPT-B can be obtained by solving a maximization program as follows:

$$\left\{ \max_{\{\alpha_u : u\in U\}} \sum_{u\in U} \kappa(\alpha_u) : \sum_{u\in U} \alpha_u \leq T,\ 0 \leq \alpha_u,\ \forall u \in U \right\}. \tag{5}$$

From Lemma 3, we have that $\kappa(q)$ is non-decreasing and concave when $q \in [0, \infty]$. This implies that the maximization Program (5) is maximized when $\alpha_u = 1$ for every u, and the resulting optimal

value is $T \cdot \kappa(1)$. Thus, we have that OPT-B achieves an online competitive ratio of $\kappa(1)$ on this example. □

6 ADAPTIVE ALGORITHMS

6.1 Warmup: Greedy

Our greedy algorithm is formally stated in Algorithm 2 as follows.

ALGORITHM 2: Greedy Algorithm (GREEDY)

1 Let v be the task arriving at time $t \in [T]$.
2 Choose an assignment $f = (u, v)$ such that f has the largest weight among all available assignments (i.e., u is available) at time t and assign v to u (break ties arbitrarily). Skip v if none is available.

Note that for the unweighted case (i.e., when all assignments have unit weights), GREEDY will choose an arbitrary available worker u when a task v arrives. We show that for the unweight case, GREEDY achieves an online competitive ratio at least 1/4.

Theorem 6.1. GREEDY *achieves a competitive ratio of at least* 0.25 *for the unweighted* OTA-TSA.

Proof. The arrival setting of **random arrival order (RAO)** is stricter than that of KIID as studied here. In the RAO, an adversary (with no information of the algorithm) selects two independent uniform random permutations over U and V, σ and π, respectively. At each time-step $t \in [T]$, the algorithm observes a worker $\sigma(t)$ and a task $\pi(t)$. As shown in Reference [27], a lower bound on the competitive ratio of any algorithm under RAO is also one for the KIID model. Thus, it suffices to show that GREEDY achieves a competitive ratio of at least 0.25 for the unweighted OTA-TSA under RAO.

Consider a graph $G = (U, V, E)$ and assume w.l.o.g. that $|U| = |V| = T$. At time t, the algorithm sees a neighbor u of $\pi(t)$ such that $\sigma^{-1}(u) \leq t$. Let $G(\sigma, \pi)$ be the graph of G under (σ, π) such that all vertices in U and V are re-ordered under σ and π, respectively, and where an edge $f = (u, v) \in E$ exists iff $\sigma^{-1}(u) \leq \pi^{-1}(v)$. Note that GREEDY computes a maximal matching for $G(\sigma, \pi)$. Let GREEDY(σ, π) be the total number of matched edges and OPT(σ, π) be the size of the maximum matching. Since GREEDY computes a maximal matching, this implies GREEDY$(\sigma, \pi) \geq$ OPT$(\sigma, \pi)/2$.

$$\text{GREEDY} = \mathbb{E}_{\sigma,\pi}[\text{GREEDY}(\sigma, \pi)] \geq \mathbb{E}_{\sigma,\pi}[\text{OPT}(\sigma, \pi)]/2 \geq \text{OPT}/4,$$

where OPT refers to the size of the maximum matching on G.

The last inequality is obtained as follows. Let $\mathcal{M}$ be a maximum matching of G with OPT $= |\mathcal{M}|$. For each edge $f \in \mathcal{M}$, we have $\Pr[f \in G(\sigma, \pi)] = \Pr[\sigma^{-1}(u) \leq \pi^{-1}(v)] \geq 1/2$. Let $\mathcal{M}(\sigma, \pi) \subseteq \mathcal{M}$ be the set of edges in $\mathcal{M} \cap G(\sigma, \pi)$. We have $\mathbb{E}[\text{OPT}(\sigma, \pi)] \geq \mathbb{E}[|\mathcal{M}(\sigma, \pi)|] = \sum_{f \in \mathcal{M}} \Pr[f \in G(\sigma, \pi)] \geq |\mathcal{M}|/2 = \text{OPT}/2$. □

6.2 Adaptive Algorithm ADAP for the LHS Vertex-weighted Case

In this section, we consider a *relaxed* version of the problem where for any $u \in U$, all edges in E_u have the same weight $w_u \geq 0$. We refer to this relaxed version as OTA-TSA with LHS vertex-weighted. For this relaxation, one can *strengthen* the benchmark LP Equation (1) due to the following observation: the probability that an edge can be matched is at most that both the worker and the task is present at least once in the arrival sequence. This boils down to computing the expected value of the minimum of two *i.i.d.* Poisson random variables with mean upper bounded

by 1. We later show that this expected value is at most $(1-1/\mathrm{e})\cdot r_v$ and hence adding this stronger constraint, we obtain the following stronger LP Equation (6). As a side note, this constraint is also valid for the general version of edge-weighted OTA-TSA, but the simpler LP Equation (1) suffices for an optimal non-adaptive algorithm.

$$\max_{(x_f : f\in E)} \sum_{u\in U} w_u \sum_{f\in E_u} x_f, \tag{6}$$

$$\sum_{f\in E_v} x_f \le r_v \qquad \forall v \in V, \tag{7}$$

$$\sum_{f\in E_u} x_f \le 1 \qquad \forall u \in U, \tag{8}$$

$$0 \le x_f \le \left(1-\frac{1}{\mathrm{e}}\right)\cdot r_v \qquad \forall f \in E_v, \forall v \in V. \tag{9}$$

Lemma 6.2. *The optimal value to* LP *Equation* (6) *is an upper bound of the offline optimal for the* OTA-TSA *with LHS vertex-weighted.*

Proof. Consider a given edge $f=(u,v)$. For any offline instance, let X_u and X_v be the respective number of arrivals of u and v and X_f be the corresponding number of matches for edge e in an offline optimal. We have that $X_f \le \min(X_u, X_v)$, which implies $x_f = \mathbb{E}[X_f] \le \mathbb{E}[\min(X_u, X_v)]$. From our arrival assumption, we have that X_u is the sum of T *i.i.d.* Bernoulli random variables each with mean $1/T$. Thus, $X_u \sim \mathrm{Pois}(1)$ when $T \to \infty$. Similarly, we have that $X_v \sim \mathrm{Pois}(r_v)$. Hence, we have the following:

$$\mathbb{E}[\min(X_u, X_v)] = \sum_{k=1}^{\infty} \Pr[X_u \ge k]\cdot\Pr[X_v \ge k] = \sum_{k=1}^{\infty} \Pr[\mathrm{Pois}(1)\ge k]\cdot\Pr[\mathrm{Pois}(r_v)\ge k] \tag{10}$$

$$\le \left(1-\frac{1}{\mathrm{e}}\right)\sum_{k=1}^{\infty}\Pr[\mathrm{Pois}(r_v)\ge k] = \left(1-\frac{1}{\mathrm{e}}\right)\cdot r_v. \tag{11}$$

The inequality Equation (11) on the second line is due to the following fact: for any given integral $k \ge 1$, we have $\Pr[\mathrm{Pois}(1) \ge k] \le 1 - 1/\mathrm{e}$. The inequality Equation (11) becomes tight when $r_v \sim 0$. □

Our adaptive algorithm is inspired by the work [33]. Let $\{x_f^*\}$ be an optimal solution to LP Equation (6). Every time when a task v arrives, we generate a random *ordered* list $\mathcal{L}$ of two choices from E_v such that it satisfies the below two properties.

(P1): $\Pr[\mathcal{L}(1)=f] = \frac{x_f^*}{r_v}$ for each $f \in E_v$,

(P2): $\Pr[\mathcal{L}(2)=f \wedge \mathcal{L}(1)\ne f] \ge \frac{x_f^*}{r_v}\frac{1}{\mathrm{e}-1}$ for each $f\in E_v$,

where $\mathcal{L}(1)$ and $\mathcal{L}(2)$ denote the first and second choices on this list $\mathcal{L}$, respectively.

Later in this section, we will describe how to efficiently generate a random list satisfying Properties **(P1)** and **(P2)**. Note that Property **(P2)** relies critically on the new constraint Equation (9) added into LP Equation (6). The main idea of ADAP is as follows: Upon the arrival of an online task v at time t, we first generate random list $\mathcal{L}$ with respect to v, and then check the availability of the first choice $\mathcal{L}(1)$ and the second choice $\mathcal{L}(2)$ sequentially; make it if it is available. Compared with NADAP, ADAP gives every edge f a second chance to be potentially matched. Property **(P1)** ensures that the marginal distribution is same as that of the optimal LP solution for the first choice; Property **(P2)** gives a lower bound on the event that every f is tried as a second choice—a

high-enough lower bound ensured by Property **(P2)** is the reason we have an improvement on the final ratio over the NADAP. Algorithm 3 formally describes ADAP.

ALGORITHM 3: An adaptive algorithm (ADAP)

1 Let v be a task arriving at time $t \in [T]$.
2 Generate a random list $\mathcal{L}$ satisfying Properties **(P1)** and **(P2)**.
3 If the first choice $\mathcal{L}(1)$ is available, then assign v to $\mathcal{L}(1)$; else, if the second choice $\mathcal{L}(2)$ is available, then assign v to $\mathcal{L}(2)$; otherwise, skip v.

Theorem 6.3. *The adaptive algorithm* ADAP *achieves a competitive ratio of at least* $\kappa\left(1+\frac{1}{\mathrm{e}(\mathrm{e}-1)}\right) \geq 0.343$ *for the* OTA-TSA *with LHS vertex-weighted.*

Proof. Consider a worker u. Let X_t and Y_t be the number of copies of u at time t before and after observing an arrival from U. From the problem assumption u arrives with probability $1/T$ in each round and thus we have $Y_t = X_t + 1$ with probability $1/T$ and $Y_t = X_t$ with probability $1 - 1/T$.

Consider the case when $Y_t \geq 1$ and a compatible task v of u arrives at t. Let $\mathcal{L}$ be the random list that is generated for v at t. From ADAP, we have that Y_t decreases by 1 iff either (1) the assignment $f = (u, v)$ is made as a first choice ($\mathcal{L}(1) = f$) or (2) the assignment $f = (u, v)$ is made as a second choice ($\mathcal{L}(2) = f$) and the first choice $\mathcal{L}(1)$ is unavailable. Thus, we have the following:

$$\begin{aligned}
&\Pr[X_{t+1} = Y_t - 1 | v \text{ comes at } t\ ] \\
&\quad = \Pr[\mathcal{L}(1) = f] + \Pr[\mathcal{L}(2) = f \wedge \mathcal{L}(1) \neq f] \cdot \Pr[\mathcal{L}(1) \text{ is not available}] \\
&\quad \geq \frac{x^*_f}{r_v} + \frac{x^*_f}{r_v}\frac{1}{\mathrm{e}-1}\Pr[\mathcal{L}(1) \text{ is not available}] \\
&\quad \geq \frac{x^*_f}{r_v} + \frac{x^*_f}{r_v}\frac{1}{\mathrm{e}-1}\frac{1}{\mathrm{e}} = \frac{x^*_f}{r_v}\left(1 + \frac{1}{\mathrm{e}-1}\frac{1}{\mathrm{e}}\right).
\end{aligned}$$

The inequality on the second line follows from Properties **(P1)** and **(P2)**. The inequality on the third line is due to the fact that any given $\mathcal{L}(1) = (u', v)$ will be unavailable with probability at least $(1 - 1/T)^t \geq 1/\mathrm{e}$ (this refers to the probability that u' never comes in the first t time-steps). Thus, summing over all the neighbors of u, we have

$$\Pr[X_{t+1} = Y_t - 1] \geq \sum_{f=(u,v)\in E_u} \frac{r_v}{T}\frac{x^*_f}{r_v}\left(1 + \frac{1}{\mathrm{e}-1}\frac{1}{\mathrm{e}}\right) = \frac{\sum_{f\in E_u} x^*_f}{T}\left(1 + \frac{1}{\mathrm{e}-1}\frac{1}{\mathrm{e}}\right).$$

We have that $\{X_t, Y_t\}$ is a TS-BDP with death rate at least $x^*_u \cdot q$, where $x^*_u = \sum_{f\in E_u} x^*_f$ and $q = (1 + \frac{1}{\mathrm{e}-1}\frac{1}{\mathrm{e}})$. From the definition of the function κ, we have that $\kappa(x^*_u \cdot q)$ is equal to the expected number of matches for worker u. Note that x^*_u is the expected number of matches for u in the benchmark LP Equation (6). Thus, the resultant ratio is at least

$$\frac{\kappa(x^*_u \cdot q)}{x^*_u} = q \cdot \frac{\kappa(x^*_u \cdot q) - \kappa(0)}{x^*_u q - 0} \geq q \cdot \frac{\kappa(q)}{q} = \kappa(q) = \kappa\left(1 + \frac{1}{\mathrm{e}-1}\frac{1}{\mathrm{e}}\right).$$

The inequality above is due to the fact that $\kappa(q)$ is a concave function when $q \in [0, \infty]$, and that $x^*_u \leq 1$ due to Constraint on u in LP Equation (6). □

6.3 Generating a Random List $\mathcal{L}$ Satisfying Properties (P1) and (P2)

The list satisfying the two properties is primarily generated via a randomized procedure that introduces negative correlations. In what follows, we describe two different approaches, both satisfying the required properties.

The first approach. This idea is due to Manshadi et al. [33]. For every $f \in E_v$, let $y_f = x^*_f/r_v$; we have that $\sum_{f\in E_v} y_f \le 1$. Add a dummy edge $f' = (u', v)$ with $y_{f'} = 1 - \sum_{f\in E_v} y_f$ (the edge $f' = (u', v)$ has the meaning that we do nothing when v comes). Create two unit intervals, $\mathcal{I}_1$ and $\mathcal{I}_2$ as follows: (1) Sort $\{y_f | f \in E_v\} \cup \{y_{f'}\}$ in an increasing order; let $y_{f_1} \le y_{f_2} \le \ldots \le y_{f_n}$ be this order; (2) Let $\mathcal{S}_i$ be a segment of length y_{f_i} with a label f_i for each $i \in [n]$. Let $\mathcal{I}_1$ be the unit interval formed by *ordered* segements $(\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \ldots, \mathcal{S}_n)$ and let $\mathcal{I}_2$ be another unit interval formed by $(\mathcal{S}_n, \mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_{n-1})$. The random list $\mathcal{L}$ based on $(\mathcal{I}_1, \mathcal{I}_2)$ as follows: (1) Choose a value $x \in [0, 1]$ uniformly at random; (2) Let $\mathcal{I}_1(x)$ and $\mathcal{I}_2(x)$ be the labels of the segments where x falls in on the two unit intervals $\mathcal{I}_1$ and $\mathcal{I}_2$, respectively; Set $\mathcal{L}(1) = \mathcal{I}_1(x)$ and $\mathcal{L}(2) = \mathcal{I}_2(x)$.

Lemma 6.4. *The random list $\mathcal{L}$ generated by the above procedure satisfies Properties **(P1)** and **(P2)**.*

Proof. To prove Property **(P1)**, notice that x takes a value in $[0, 1]$ uniformly at random. Thus each $f \in E_v$, x falls in the segment labelled by f in $\mathcal{I}_1$ with probability $y_f = x^*_f/r_v$. This suggests that $\Pr[\mathcal{L}(1) = f] = y_f = x^*_f/r_v$.

Observe that for any $f \in E_v$ with $y_f \le 1/2$, we have $\Pr[\mathcal{L}(1) = \mathcal{L}(2) = f] = 0$. Thus, $\Pr[\mathcal{L}(2) = f \wedge \mathcal{L}(1) \ne f] = \Pr[\mathcal{L}(2) = f] = y_f > (x^*_f/r_v) \cdot (1/(\mathrm{e} - 1))$. Consider the other case $y_f > 1/2$. The event $(\mathcal{L}(2) = f \wedge \mathcal{L}(1) \ne f)$ occurs only when the random value x falls in the segment labelled by f in $\mathcal{I}_2$ and x does not fall in the segment labelled by f in $\mathcal{I}_1$. Therefore,

$$\Pr[\mathcal{L}(2) = f \wedge \mathcal{L}(1) \ne f] = y_f - (2y_f - 1) = y_f \cdot \left(\frac{1}{y_f} - 1\right) \ge y_f \cdot \left(\frac{1}{1 - 1/\mathrm{e}} - 1\right) = \frac{y_f}{\mathrm{e} - 1}.$$

The last inequality follows from $y_f = x^*_f/r_v \le 1 - 1/\mathrm{e}$ for every $f \in E_v$ that is due to Constraint Equation (9). □

A second approach. This idea is based on a modified version of ***dependent rounding*** **(DR)** [16] as shown in Reference [9], which accepts a more efficient implementation in practice. First, create a star graph with edge set $E_v \cup \{f' = (u', v)\}$ where $y_f = x^*_f/r_v$ for each $f \in E_v$ and $y_{f'} = 1 - \sum_{f\in E_v} x^*_f$. Second, multiply each value y_f by 2 and apply the dependent rounding technique [16] to this modified vector $\mathbf{y} = (2y_f | f \in E_v \cup \{f'\})$. We will get a random integral vector $\mathbf{Y} = (Y_f)$ with $Y_f \in \{0, 1, 2\}$ such that (i) $\mathbb{E}[Y_f] = 2 \cdot y_f$ and (ii) with probability 1, we have $\sum_{f\in E_v} Y_f + Y_{f'} = 2$. Let $\{f_a, f_b\}$ be the two edges rounded, i.e., $Y_{f_a} = Y_{f_b} = 1$ (in the case when some $Y_f = 2$, we set $f_a = f_b = f$). Set $\mathcal{L} = (f_a, f_b)$ and $\mathcal{L} = (f_b, f_a)$ with the respective probabilities of $1/2$ and $1/2$. We justify that $\mathcal{L}$ satisfies Properties **(P1)** and **(P2)** as follows.

Lemma 6.5. *The random list $\mathcal{L}$ generated by the above DR-based procedure satisfies Properties **(P1)** and **(P2)**.*

Proof. We prove Property **(P1)** first. Consider a given edge $f \in E_v$ with $y_f \le 1/2$. In this case, we have $Y_f \in \{0, 1\}$. Since marginal distribution is maintained in DR [16], we see that $\Pr[\mathcal{L}(1) = f] = \frac{1}{2} \cdot \Pr[Y_f = 1] = y_f = x^*_f/r_v$. For the other case $y_f > 1/2$, we see that $Y_f = 1$ with probability $2 - 2y_f$ and $Y_f = 2$ with probability $2y_f - 1$. Thus,

$$\Pr[\mathcal{L}(1) = f] = \Pr[Y_f = 2] + \frac{1}{2}\Pr[Y_f = 1] = 2y_f - 1 + (2 - 2y_f) \cdot \frac{1}{2} = y_f = x^*_f/r_v.$$

Now, we show Property **(P2)**. Consider a given edge $f \in E_v$ with $y_f \le 1/2$. Observe that in this case $Y_f \in \{0, 1\}$ and thus, $\Pr[\mathcal{L}(1) \neq f | \mathcal{L}(2) = f] = 1$. Therefore,

$$\Pr[\mathcal{L}(2) = f \wedge \mathcal{L}(1) \neq f] = \Pr[\mathcal{L}(2) = f] \cdot \Pr[\mathcal{L}(1) \neq f | \mathcal{L}(2) = f] = \frac{1}{2} \cdot (2y_f) = \frac{x^*_f}{r_v} \ge \frac{x^*_f}{r_v} \cdot \frac{1}{\mathrm{e}-1}.$$

As for the case $y_f > 1/2$: We see that with probability $2 - 2y_f$, $Y_f = 1$. Thus,

$$\Pr[\mathcal{L}(2) = f \wedge \mathcal{L}(1) \neq f] = \frac{1}{2} \cdot \Pr[Y_f = 1] = 1 - y_f = y_f \cdot \left(\frac{1}{y_f} - 1\right) \ge \frac{y_f}{\mathrm{e}-1} = \frac{x^*_f}{r_v} \cdot \frac{1}{\mathrm{e}-1}.$$

The above analysis is similar to that in the proof of Lemma 6.4. □

6.4 Another Adaptive Algorithm SCALED for the LHS Vertex-weighted OTA-TSA

Recall that LP Equation (6) is a valid benchmark LP for the LHS vertex-weighted OTA-TSA. Let $(x^*_f | f \in E)$ be an optimal solution to LP Equation (6). The adaptive algorithm SCALED presented below can be viewed as a boosted version of NADAP. The main idea is as follows. Suppose a task v arrives at some time $t \in [T]$, and let $E_{v,t}$ be the set of *available* edges incident to v at time t after observing the arrival from U. If $E_{v,t}$ is empty, then do nothing; otherwise, sample an edge $f \in E_{v,t}$ with probability $x^*_f / \sum_{f \in E_{v,t}}$. Note that (1) $E_{v,t}$ is *not* a multi-set: some neighbor u of v may have multiple copies at time t after the arrival from U, and in this case, we add only one copy $f = (u, v)$ into $E_{v,t}$; (2) E_v is deterministic (specified by the input compatible bipartite graph G), while $E_{v,t}$ is stochastic, whose outcome is determined jointly by the arrival sequences from the two sides and the random actions chosen by the algorithm up to time t.

ALGORITHM 4: An adaptive algorithm for OTA-TSA: SCALED.

1 Let v be a task arriving at time $t \in [T]$ and $E_{v,t}$ be the set of *available* assignments with respect to v at t after observing the arrival from U.
2 If $E_{v,t} = \emptyset$, then skip v; otherwise, sample an edge $f = (u, v) \in E_{v,t}$ with probability $x^*_f / \sum_{f' \in E_{v,t}} x^*_{f'}$.

Theorem 6.6. *Algorithm 4,* SCALED*, achieves a competitive ratio at least 0.355 for the LHS vertex-weighted* OTA-TSA.

Proof. Consider a given worker u. For each given $t \in [T]$, let $A_t = 1$ indicate that worker u arrives at time t and gets matched at some later time $t' \ge t, t' \le T$. Thus, $\mathbb{E}[A]$ with $A = \sum_{t \in [T]} A_t$ denotes the expected number of times that u gets matched. In the following, we lower bound $\mathbb{E}[A]/\tau$ that yields a competitive ratio for SCALED.

Assume u arrives at some time $t \in [T]$ with probability $1/T$ (denoted by $X_{u,t} = 1$). For each given $t' \in \{t, t+1, \ldots, T\}$, let $\alpha_{t'}$ be the probability that u gets matched at time t' assuming $\bar{u}$ remains unmatched at t'. For each $u' \in U$ and $t \le t' \le T$, let $\chi_{u',t'} = 1$ indicate that u' is available at time t' (after the arrival of U and before any actions of the algorithm). Observe the following chain of reasoning, where the first inequality follows from Jensen's inequality applied to the function $z \mapsto 1/z$: specifically, we use the fact that if the λ_i's are non-negative constants and the Z_i's are non-negative random variables, then

$$\mathbb{E}\left[\frac{1}{\lambda_0 + \sum_{i=1}^{\ell} \lambda_i Z_i} \,\middle|\, Z_0 = 1\right] \ge \frac{1}{\lambda_0 + \sum_{i=1}^{\ell} \lambda_i \mathbb{E}[Z_i \mid Z_0 = 1]}.$$

$$\alpha_{t'} = \mathbb{E}\left[\sum_{f=(u,v)\in E_u} \frac{r_v}{T}\cdot\frac{x_f^*}{x_f^* + \sum_{f'=(u',v)\in E_v, f'\neq f} x_{f'}\cdot\chi_{u',t'}} \;\middle|\; X_{u,t}=1\right]$$

$$\geq \sum_{f=(u,v)\in E_u} \frac{r_v}{T}\cdot\frac{x_f^*}{x_f^* + \sum_{f'=(u',v)\in E_v, f'\neq f} x_{f'}\cdot\mathbb{E}[\chi_{u',t'}\mid X_{u,t}=1]}$$

$$\geq \sum_{f=(u,v)\in E_u} \frac{r_v}{T}\cdot\frac{x_f^*}{x_f^* + \sum_{f'=(u',v)\in E_v, f'\neq f} x_{f'}\cdot\left(1-(1-1/T)^{t'-1}\right)}, \quad \text{since } \mathbb{E}[\chi_{u',t'}|X_{u,t}=1] \leq 1-\left(1-\frac{1}{T}\right)^{t'-1}$$

$$\geq \sum_{f=(u,v)\in E_u} \frac{r_v}{T}\cdot\frac{x_f^*}{x_f^* + (r_v - x_f^*)\cdot(1-\mathrm{e}^{-t'/T+o(t'/T)})} \quad \text{from Constraint Equation (7): } \sum_{f\in E_v} x_f^* \leq r_v$$

$$= \sum_{f=(u,v)\in E_u} \frac{1}{T}\cdot\frac{x_f^*}{x_f^*/r_v + (1-x_f^*/r_v)\cdot(1-\mathrm{e}^{-t'/T+o(t'/T)})}$$

$$\geq \sum_{f=(u,v)\in E_u} \frac{1}{T}\cdot\frac{x_f^*}{(1-1/\mathrm{e}) + (1/\mathrm{e})\cdot\left(1-\mathrm{e}^{-t'/T+o(t'/T)}\right)} \quad \text{from Constraint Equation (9): } x_f^* \leq r_v\cdot(1-1/\mathrm{e})$$

$$= \frac{1}{T}\cdot\frac{x_u^*}{(1-1/\mathrm{e}) + (1/\mathrm{e})\cdot\left(1-\mathrm{e}^{-t'/T+o(t'/T)}\right)}.$$

Thus, the probability that an arrival of u at time t gets matched should be

$$\mathbb{E}[A_t] = \frac{1}{T}\left(1-\prod_{t'=t}^{T}(1-\alpha_{t'})\right) = \frac{1}{T}\left(1-\exp\left[\sum_{t'=t}^{T}\ln(1-\alpha_{t'})\right]\right)$$
$$= \frac{1}{T}\left(1-\exp\left[\left(-\sum_{t'=t}^{T}\alpha_{t'}\right)\cdot(1+o(1))\right]\right),$$

where the "$o(1)$" is a vanishing term when $T\to\infty$. Summarizing the analysis thus far, we have

$$\mathbb{E}[A] = \sum_{t=1}^{T}\mathbb{E}[A_t] = \sum_{t=1}^{T}\frac{1}{T}\left(1-\exp\left[\left(-\sum_{t'=t}^{T}\alpha_{t'}\right)\cdot(1+o(1))\right]\right)$$
$$\geq \sum_{t=1}^{T}\frac{1}{T}\left(1-\exp\left[\left(-\sum_{t'=t}^{T}\frac{1}{T}\cdot\frac{x_u^*}{(1-1/\mathrm{e})+(1/\mathrm{e})\cdot\left(1-\mathrm{e}^{-t'/T+o(t'/T)}\right)}\right)\cdot(1+o(1))\right]\right)$$
$$= \int_0^1 d\mu\int_\mu^1 d\nu\left(1-\exp\left[\left(-\frac{x_u^*}{(1-1/\mathrm{e})+(1/\mathrm{e})\cdot\left(1-\mathrm{e}^{-\nu}\right)}\right)\right]\right) := g(x_u^*), \text{ by taking } T\to\infty.$$

We can numerically verify that $g(x_u^*)/x_u^*$ is decreasing over $x_u^* \in [0,1]$ with $g(x_u^*)/x_u^* \geq g(1) \sim 0.355$. We thus establish a lower bound on the competitive ratio of SCALED for the LHS vertex-weighted case. □

7 NEGATIVE RESULTS

We now show some negative results, i.e., upper bounds on the competitive ratio that any algorithm can achieve. All our results hold even in the simpler case of unweighted OTA-TSA when all edges take a uniform weight.

7.1 Conditional Negative Results

Theorem 7.1. *No algorithm can achieve a competitive ratio better than* $\kappa(1 + 1/(\mathrm{e}-1)) \sim 0.423$ *based on* LP *Equation* (6) *even for the unweighted* OTA-TSA.

Proof. Consider an unweighted star graph, where we have one single worker u with $p_u = 1/T$, which is connected to T tasks each having $r_v = (1/T)/(1-1/\mathrm{e})$. We can verify LP Equation (6) has an optimal solution such that $x_f^* = 1/T$ for all the T edges and an optimal LP value equal to 1. Now consider an optimal online algorithm, denoted by OPT. We see that every time upon the arrival of any v, OPT will match it to u as long as there is one copy of u available then. Note that in each round, u arrives with probability $1/T$, while one of the T tasks arrives with probability $\sum_v r_v/T = (1/T)/(1-1/\mathrm{e})$. By the definition of TS-BDP in 3.1, OPT has an expected number of matches equal to $\kappa(1/(1-\mathrm{e}))$. Thus, we claim that on this example, OPT achieves a competitive ratio no more than $\kappa(1/(1-\mathrm{e})) \sim 0.423$ by Lemma 2. □

7.2 Unconditional Negative Results

We now present an unconditional negative result that is independent of the choice of the benchmark LP. The result stems implicitly from the nature of the online process and can be viewed as the *online-offline stochastic gap*. In particular, we have the following theorem:

Theorem 7.2. *No algorithm can achieve a competitive ratio better than* $\frac{\kappa'(0)}{1-1/\mathrm{e}} = \frac{1}{\mathrm{e}-1} \sim 0.581$ *even for the unweighted* OTA-TSA.

Proof. Consider an unweighted bipartite graph $G = (U, V, E)$, where $|U| = |V| = T$ and $|E| = T$, which consists of a perfect matching. Let the arrival rates for every u be 1 with $p_u = 1/T$ and let every v have an arrival rate of ϵ (where ϵ is arbitrarily close to 0) with $p_v = \epsilon/T$. We assume that there exists a dummy node v' such that $p_{v'} = 1-\epsilon$ with v' having no neighbors.

Consider a given $f = (u, v)$. Let OPT-A and OPT-B be the respective offline and online optimal algorithms. Let X_f be the number of matches of f in OPT-A after the T rounds. Let X_u and X_v be the respective number of arrivals of u and v in an offline instance. We have that $X_f = \min(X_u, X_v)$. Observe that $X_u \sim \mathrm{Pois}(1)$ and $X_v \sim \mathrm{Pois}(\epsilon)$. Thus, we have

$$\mathbb{E}[\min(X_u, X_v)] = \sum_{k=1}^{\infty} \Pr[X_u \geq k] \cdot \Pr[X_v \geq k] = \sum_{k=1}^{\infty} \Pr[\mathrm{Pois}(1) \geq k] \cdot \Pr[\mathrm{Pois}(\epsilon) \geq k]$$

$$= \left(1 - \frac{1}{\mathrm{e}}\right)(1 - \mathrm{e}^{-\epsilon}) + \left(1 - \frac{2}{\mathrm{e}}\right)(1 - \mathrm{e}^{-\epsilon} - \epsilon\mathrm{e}^{-\epsilon}) + \cdots = \left(1 - \frac{1}{\mathrm{e}}\right) \cdot \epsilon + o(\epsilon).$$

Hence, we have that $\mathbb{E}[X_f] = (1 - \frac{1}{\mathrm{e}}) \cdot \epsilon + o(\epsilon)$.

Let Y_f be the number of matches of f in OPT-B. Similar to the proof in Theorem 5.1, we have $\mathbb{E}[Y_f] = \kappa(\epsilon)$. Thus, the ratio on the above instance is

$$\frac{\mathbb{E}[Y_f]}{\mathbb{E}[X_f]} = \frac{\kappa(\epsilon)}{\left(1 - \frac{1}{\mathrm{e}}\right) \cdot \epsilon + o(\epsilon)}.$$

Taking $\epsilon \to 0$, we have that the above value is

$$\lim_{\epsilon \to 0} \frac{\kappa(\epsilon)}{\left(1 - \frac{1}{\mathrm{e}}\right) \cdot \epsilon + o(\epsilon)} = \lim_{\epsilon \to 0} \frac{\kappa(\epsilon)}{\epsilon} \frac{1}{1 - 1/\mathrm{e}} = \frac{\kappa'(0)}{1 - 1/\mathrm{e}}.$$

From Lemma 1, $\kappa'(0) = 1/\mathrm{e}$; thus, we get our claim. □

8 EXPERIMENTS

In this section, we describe the experimental results in this article. We implement the adaptive and non-adaptive algorithms on both a simulated setup and a public ridesharing dataset collected in the city of Chicago.[4]

Preprocessing of a real-world dataset. The dataset has more than 169 million trips since November 2018. Each trip record includes the following information: departure and arrival timestamps, pick-up and drop-off locations, and some other information like the total fare. Following the idea in Reference [31], we use the community area to divide the Chicago into 77 regions and categorize all trips according to the predefined community areas. In the context of ridesharing, we assume that driver-types and request-types represent the worker-types and task-types in the OTA-TSA model, respectively. We group the drivers into a "type" if they share the same community area. Similarly, we group the requests into a "type" if they share the same starting and ending community areas. In this way, we end up with 77 driver-types and $77 \times 77 = 5929$ request-types.

To precisely construct the compatibility graph between the drivers and requests, we focus on the rush hour from 18:00 to 19:00 between August 1 and August 5 in 2022, and we have 36,907 trips in total. By categorizing these trips according to the 77 predefined community areas, we have 2,661 trip types, each associated a capacity of c. Inspired by the setting in Reference [38], we continue to filter out those trip types with capacities no larger than 10 and sample 200 trip types randomly, and thus, we get the set of driver-types U with $|U| = 52$ and the set of request-types V with $|V| = 200$. Here, we construct the arriving distribution among requests by setting $q_v = c_v / \sum_{v \in V} c_v$ with $\sum_v q_v = 1$. Similarly, the arriving probability for each driver-type $u \in U$ is computed as $p_u = c_u / \sum_{u \in U} c_u$. We add an edge between a driver and a request type if the community area of driver type is the same as the starting community area of request type. In the edge-weighted case, for each edge $f = (u, v)$, we use the averaged total fare of request-type v as the final weight of w_f; while in the LHS vertex-weighted case, we use the averaged total fare of driver-type u instead. The detailed setting of all other parameters can be found in Table 4.

A synthetic dataset. We generate the bipartite graph by setting the parameters of $|U|$, $|V|$ and T as shown in Table 4. Note that we set the number of copy for each driver-type as 1, such that the total number of drivers arriving is the same as $|U|$. We further assume that all driver-types have the same arriving probability $p_u = 1/T$ for every $u \in U$. Hence, with probability $1 - |U|/T$, no drivers will arrive in each round $t \in [T]$. We generate the request-type arrival probabilities by choosing a random vector $\{q_v\}$ such that each q_v is uniformly distributed over $[0, 1]$ conditioning on $\sum_v q_v = 1$. For each pair of u and v, we add an edge between them with probability 0.2. In the edge-weighted case, for each edge $f = (u, v)$, we choose a value from $[0, 1]$ uniformly at random

[4] https://data.cityofchicago.org/Transportation/Transportation-Network-Providers-Trips/m6dm-c72p

Table 4. Setting of Parameters in the Experiments, Where Default Values Are Marked as Bold

Dataset	Real-world ridesharing dataset		Synthetic dataset	
OTA-TSA version	edge-weighted	LHS vertex-weighted	edge-weighted	LHS vertex-weighted
# of driver-types |U|	52		{20, 40, …, **100**, …, 140}	{10, 50, **100**, 150, 200}
# of request-types |V|	200		100	
# of arriving drivers	{50, 100, …, **200**, …, 350}	{10, 50, 100, **200**, 300}	{20, 40, …, **100**, …, 140}	{10, 50, **100**, 150, 200}
# of rounds (T)	{250, 500, …, **1,000**, …, 2,000}	{200, 500,**1,000**, 1,500, 2,000}	{100, 200, …, **500**, …, 900}	{100, 500,**1,000**, 1,500, 2,000}

and set it as w_f. While in the LHS vertex-weighted case, for each driver-type u, we generate w_u from $[0, 1]$ uniformly at random, and then set $w_f = w_u$ for all edge incident to u.

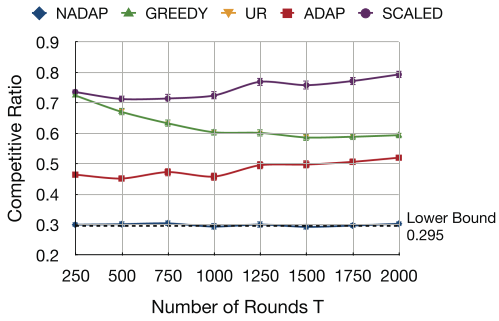
Algorithms. We implement all the algorithms analyzed in the article, namely, NADAP, GREEDY, ADAP, and SCALED. Additionally, we test one heuristic, denoted by UR, which selects one of the available drivers uniformly at random upon every arrival of rider. Observe that both GREEDY and UR are agnostic to the underlying LP.

Results and discussion. We run two types of experiments by varying either the total number of rounds T (i.e., the total number of requests' arrivals) or the total number of drivers's arrivals, while keep all other parameters fixed, as shown in Table 4. For each instance, we run all algorithms 100 independent trials and take the average as the final performance. The competitive ratios are computed as the ratio of the averaged performance to the optimal value of the corresponding LPs.[5] We also report the 95% confidence intervals to keep track of the robustness.

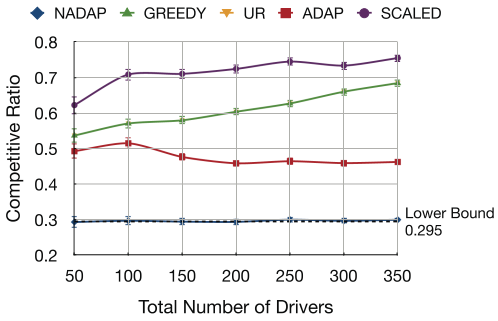
Figure 2 shows that the proposed NADAP algorithm always stays above its theoretical lower bound of 0.295, as suggested in Theorem 5.1. This suggests the tightness of our theoretical analysis. Additionally, we see that SCALED is the clear winner in almost all the instances for the edge-weighted case except when T is extremely small, as shown in Figures 2(a) and 2(c). Observe that when the total number of rounds (T) is small, each arriving request has a limited number of drivers to choose from, since the number of drivers who have arrived and are compatible is small. In this case, the advantage of greedily matching an available driver outweighs the potential loss from a mismatch. However, when T increases and each arriving request has more options to choose from, the guidance from the LP becomes increasingly crucial, since it takes the future arrivals into consideration (in expectation). This also explains why ADAP beats GREEDY as T increases in the synthetic dataset. Figure 2(b) shows that as the total number of drivers increases, the competitive ratios of GREEDY and UR (the lines for these two algorithms are overlapped) both increase. This is because more available drivers make it less necessary to optimize the matching process. Note that ADAP outperforms GREEDY in the synthetic dataset, but not the real one. This is partially due to that we have less driver-types in the real dataset compared with the synthetic. Thus, we have lower density for the driver arrivals and less need to reserve drivers as ADAP does.

Figure 3 shows that, for all tested instances, the competitive ratios achieved by ADAP and SCALED always stay above their corresponding theoretical lower bounds, i.e., 0.343 (Theorem 6.3) and 0.355 (Theorem 6.6), respectively. We notice that the three algorithms, GREEDY, SCALED and UR, all have a similar performance and outperform the rest two. This is expected, since the LHS vertex-weighted setting reduces variations among edge weights, which gives favor to LP-agnostic algorithms.

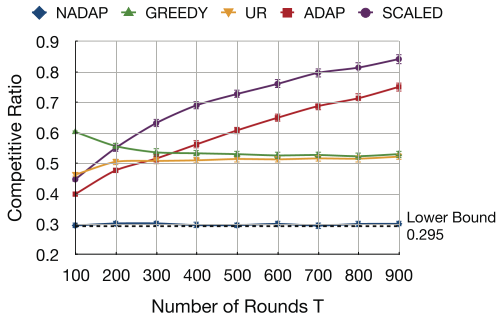
[5] LP Equation (1) for the edge weighted case and LP Equation (6) for the LHS vertex-weighted case.



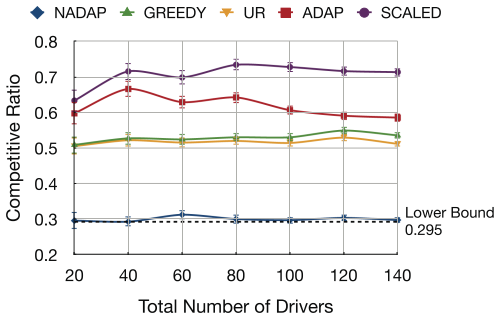
(a) Real dataset: results when varying the number of rounds T.



(b) Real dataset: results when varying the total number of drivers.



(c) Synthetic dataset: results when varying the number of rounds T.



(d) Synthetic dataset: results when varying the total number of drivers.

Fig. 2. Experiments results on the synthetic and real datasets for the edge-weighted case.

9 PROOFS OF KEY LEMMAS IN THE BIRTH-DEATH PROCESS

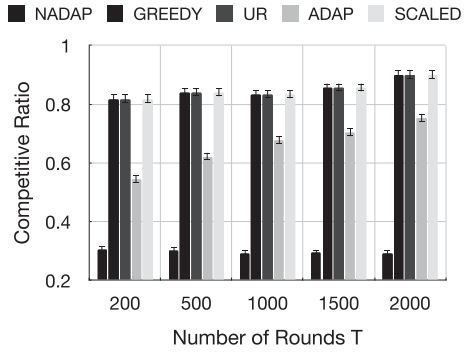
Consider a two-stage birth-death process $\{X_t, Y_t | t \in [T]\} \cup \{X_{T+1}\}$ with a birth rate of 1, a death rate of $q \geq 0$, and a finite-time horizon T (denoted by TS-BDP(q)). Recall that $\Delta(T)$ is the (random) number of death events (i.e., Y_t gets decreased by 1) in the T rounds, and $\kappa(q) = \lim_{T\to\infty} \mathbb{E}[\Delta(T)]$. The random process of TS-BDP(q) can be viewed equivalently as follows. First, we generate a random binary string $\mathcal{S}$ of length T such that each $\mathcal{S}(t) \sim \text{Ber}(1/T)$ with $t \in [T]$, i.e., each $\mathcal{S}(t)$ is an independent Bernoulli random variable with mean $1/T$. We call $\mathcal{S}$ the "birth" string. Second, we apply a "death" process to the random birth string $\mathcal{S} = S \in \{0,1\}^T$ as follows. Set two counters $C = D = 0$. At each time $t = 1, 2, \ldots, T$: (1) set $C \leftarrow C + S(t)$; (2) If $C \geq 1$, then update $C \leftarrow C - 1, D \leftarrow D + 1$ with probability q/T; else, do nothing. Let $\Delta_S(T) = \mathbb{E}[D|S]$ be the expected value of D after T rounds given $\mathcal{S} = S$. Then, we have the following:

$$\mathbb{E}[\Delta(T)] = \sum_{S\in\{0,1\}^T} \Pr[\mathcal{S} = S]\Delta_S(T) = \mathbb{E}_{\mathcal{S}}\Big[\Delta_{\mathcal{S}}(T)\Big].$$

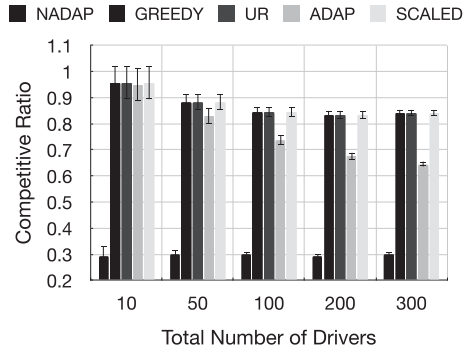
For a given T and $\mathcal{S} = S$, we refer to the above death process applied to S with death rate q as DP(T, S, q).

9.1 Proof of Lemma 1

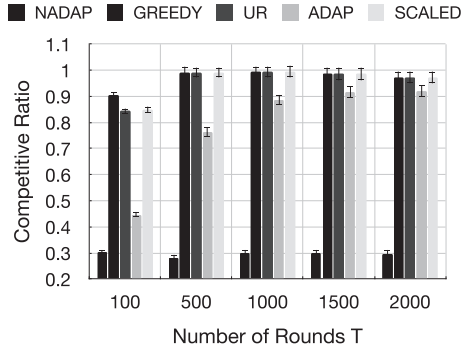
Lemma 1. *(1)* $\kappa(0) = 0$, *(2)* $\kappa'(0) = 1/\mathrm{e}$, *where* $\kappa'(0)$ *is the first (right) derivative of* $\kappa(q)$ *at* $q = 0$.



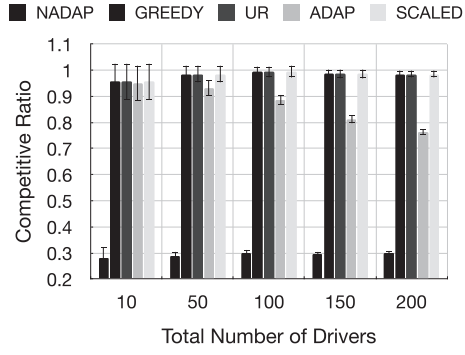
(a) Real dataset: results when varying the number of rounds T.



(b) Real dataset: results when varying the total number of drivers.



(c) Synthetic dataset: results when varying the number of rounds T.



(d) Synthetic dataset: results when varying the total number of drivers.

Fig. 3. Experiments results on the synthetic and real datasets for the LHS vertex-weighted case.

Proof. This Lemma is proved via the following two observations.

— When $q = 0$, for any given $\mathcal{S} = S \in \{0,1\}^T$, we have $\Delta_S(T) = 0$, since $\Delta(T) = 0$ and thus $\kappa(q) = 0$.

— First, note that we have $\kappa'(0) = \lim_{q\to 0} \frac{\kappa(q)}{q}$. Additionally, when $T \to \infty$, and S is the all-ones string, we have $D \sim \text{Pois}(q)$. This implies that

$$\lim_{T\to\infty} \Pr[\Delta(T) \geq 2] \leq \Pr[\text{Pois}(q) \geq 2] = 1 - \mathrm{e}^{-q}(1+q) = o(q).$$

Moreover, we have that

$$\lim_{T\to\infty} \sum_{k\geq 2} \Pr[\Delta(T) \geq k] \leq \sum_{k\geq 2} \Pr[\text{Pois}(q) \geq k] = q - (1 - \mathrm{e}^{-q}) = o(q).$$

Thus, we have

$$\kappa(q) = \lim_{T\to\infty} \mathbb{E}[\Delta(T)] = \lim_{T\to\infty} \sum_{k\geq 1} \Pr[\Delta(T) \geq k] = \lim_{T\to\infty} \Pr[\Delta(T) \geq 1] + o(q). \tag{12}$$

Thus, it remains to consider the quantity $\Pr[\Delta(T) \geq 1]$. Notice that $\Delta(T) \geq 1$ occurs iff (1) $\mathcal{S} = S$ where $|S| \geq 1$ (the number of ones in S is at least 1) and (2) a death event occurs at

least once in the time-steps $t = k, k+1, \ldots, T$, where $k \in [T]$ is the position of the first 1 in S. Putting these together, we have

$$\Pr[\Delta(T) \geq 1] = \sum_{k=1}^{n} \frac{1}{T}\left(1 - \frac{1}{T}\right)^{k-1}\left(1 - \left(1 - \frac{q}{T}\right)^{n-k+1}\right) \doteq F_1(T, q).$$

Plugging the above expression back into Equation (12), we have

$$\lim_{q\to 0} \frac{\kappa(q)}{q} = \lim_{q\to 0} \lim_{T\to\infty} \frac{F_1(T, q)}{q} = \frac{1}{\mathrm{e}}.$$

□

9.2 Proof of Lemma 2

Lemma 2. *(1)* $0.295 \leq \kappa(1) \leq 0.302$*; (2)* $\kappa(1 + \frac{1}{\mathrm{e}(\mathrm{e}-1)}) \geq 0.343$*; and (3)* $\kappa(1 + \frac{1}{\mathrm{e}-1}) \leq 0.423$.

Proof. We prove a stronger version of the Lemma, where we consider an arbitrary given $q \geq 0$. Recall that $\Delta(T)$ is the expected number of "death" events (i.e., Y_t decreases by 1) in the TS-BDP$(1, q)$. Hence,

$$\kappa(q) = \lim_{T\to\infty} \mathbb{E}[\Delta(T)] = \sum_{k\geq 1} \lim_{T\to\infty} \Pr[\Delta(T) \geq k]. \tag{13}$$

Consider the term $\lim_{T\to\infty} \Pr[\Delta(T) \geq 1]$. Notice that in the proof of Lemma 1, we have $\Pr[\Delta(T) \geq 1] = F_1(T, q)$ and, thus,

$$\lim_{T\to\infty} \Pr[\Delta(T) \geq 1] = \lim_{T\to\infty} F_1(T, q) = 1 - \frac{1}{\mathrm{e}} - \frac{1}{\mathrm{e}} \frac{\mathrm{e}^{1-q} - 1}{1 - q} := F_1(q).$$

Now consider the term $\lim_{T\to\infty} \Pr[\Delta(T) \geq 2]$. Notice that $\Delta(T) \geq 2$ occurs iff (1) $\mathcal{S} = S$ where S has at least two 1s with ℓ_1, ℓ_2 being the positions of the first and second 1s, respectively. (2) Two death events occur with either both occurring in the interval $[\ell_2, T] \doteq \{\ell_2, \ell_2 + 1, \ldots, T\}$ or one death event occurring in $\{\ell_1, \ldots, \ell_2 - 1\}$ and the other one in $[k_2, T]$. Therefore, we have that

$$\begin{aligned} F_2(T, q) &\doteq \Pr[\Delta(T) \geq 2] \\ &= \sum_{\ell_1=1}^{T} \sum_{\ell_1<\ell_2\leq T} \frac{1}{T^2}\left(1 - \frac{1}{T}\right)^{\ell_2-2} \\ &\quad \cdot \left\{ \sum_{t=\ell_1}^{\ell_2-1} \left(1 - \frac{q}{T}\right)^{t-\ell_1} \frac{q}{T} \sum_{t'=\ell_2}^{T} \frac{q}{T}\left(1 - \frac{q}{T}\right)^{t'-\ell_2} + \sum_{t=\ell_2+1} \left(\frac{q}{T}\right)^2 \left(1 - \frac{q}{T}\right)^{t-\ell_1-1} (t - \ell_2) \right\}. \end{aligned}$$

Using computer-aided proof, we have that

$$F_2(q) \doteq \lim_{T\to\infty} F_2(T, q) = \frac{\mathrm{e}^{-q-1}\left(\mathrm{e}^q(1 - 2q) + \mathrm{e}^{q+1}(q - 1) + q + \mathrm{e} - 1\right)}{q - 1}.$$

Finally, we have that

$$\sum_{k\geq 3} \lim_{T\to\infty} \Pr[\Delta(T) \geq k] \leq \sum_{k\geq 3} \Pr[\mathrm{Pois}(1) \geq k] \cdot \Pr[\mathrm{Pois}(q) \geq k].$$

Thus, from Equation (13), we have

$$F_1(q) + F_2(q) \leq \kappa(q) \leq F_1(q) + F_2(q) + \sum_{k\geq 3} \Pr[\mathrm{Pois}(1) \geq k] \cdot \Pr[\mathrm{Pois}(q) \geq k]. \tag{14}$$

The two statements in the Lemma follow from the above Inequality Equation (14) when $q = 1$ and $q = 1 + \frac{1}{e(e-1)}$, respectively. □

9.3 Proof of Lemma 3

Lemma 3. *$\kappa(q)$ is non-decreasing and concave over $q \in [0, \infty]$.*

We split it into the following two Claims.

Claim 1. *$\kappa(q)$ is non-decreasing when $q \in [0, \infty]$.*

Claim 2. *$\kappa(q)$ is concave when $q \in [0, \infty]$.*

Proof of Claim 1. Recall that $\kappa(q) = \lim_{T\to\infty} \mathbb{E}[\Delta(T)] = \lim_{T\to\infty} \mathbb{E}_{\mathcal{S}}[\Delta_S(T)]$. It would suffice to show that $\Delta_S(T)$ is non-decreasing for any given S and T.

Consider a given birth string S of length T. The random death process, $\mathrm{DP}(T, S, q)$, is as follows. First, we generate a random bit-string of length T (called *death string*), say S^*, such that $S^*(t) \sim \mathrm{Ber}(q/T)$. Second, (i) initialize $C = D = 0$; (ii) at each time $t = 1, 2, \ldots, T$: set $C \leftarrow C + S(t)$; if $S^*(t) = 0$, then proceed to $t + 1$; otherwise, if $C \geq 1$, update $C \leftarrow C - 1$ and $D \leftarrow D + 1$. In other words, time-steps t with $S^*(t) = 1$ can be viewed as points at which we invoke a "checking" operation (called "check points"). Let $\Delta_{S,S^*}(T)$ be the value of D after T steps, given S, S^*, and T. We have that $\Delta_S(T) = \mathbb{E}_{S^*}[\Delta_{S,S^*}(T)]$ for any given S and T.

Consider two different death rates p, q with $0 \leq p < q$. Let S_p^* and S_q^* be the random sequences we need to generate in the two processes, respectively. From the analysis above, we have that for each $t \in [T]$, $S_p^*(t) \sim \mathrm{Ber}(p/T)$ and $S_q^*(T) \sim \mathrm{Ber}(q/T)$. Thus, the random variable $S_p^*(t) = S_q^*(t) \cdot \mathrm{Ber}(p/q)$. For any given S and T, the generating process of S_p^* and S_q^* can be viewed as follows. First, generate S_q^* such that $S_q^*(t) \sim \mathrm{Ber}(q/T)$ for each t. Then, generate S_p^* by flipping every 1 in S_q^* to 0 with probability $1 - p/q$ independently. This implies that the set of ones in S_p^* will be a subset (or equal) of S_q^*, which further suggests that $\Delta_{S,S_p^*}(T) \leq \Delta_{S,S_q^*}(T)$ for each given S_q^*. Notice that $\Delta_S(T)$ is the expected value of $\Delta_{S,S^*}(T)$ over all possible S^* and thus, we claim that $\Delta_S(T)$ is non-decreasing for any given S and T. □

Proof of Claim 2. Consider a given birth string S with a fixed length T. Let $p = q/T \in [0, 1]$. Note that $\Delta_S(T)$ is a function of p in $[0, 1]$. It will suffice to show that $\Delta_S(T)$ is concave when $p \in [0, 1]$ for any given T and any given $S \in \{0, 1\}^T$.

Consider the following *generalized* death process on S. Initialize two counts $C = a \geq 0$ and $D = 0$, where a is a fixed nonnegative integer. Run the random process as before. Let $f_{a,S}(p)$ denote the expected value of D after T rounds. We prove the following statements via induction on the length of $t = |S|$ with $t = 1, 2, \ldots, T$.

(P1) For all $a \geq 0$ and all $S \in \{0, 1\}^t$, $f''_{a,S}(p) \leq 0$.
(P2) For all $a \geq 1$ and all $S \in \{0, 1\}^t$, $f'_{a,S}(p) \geq f'_{a-1,S}(p)$.

Note that $\Delta_S(T) = f_{0,S}$. Thus, the above two properties will imply the Claim 2. It is verify that Properties **(P1)** and **(P2)** hold when $t = 1$. Now, we show the inductive step from $t-1$ to t. Assume $S[1] = 0$ and let $S' = S[2, t]$ that denotes the subsequence of S after removing the first entry. We have that

$$\begin{aligned} f_{a,S}(p) &= (1-p) f_{a,S'}(p) + p\left(1 + f_{a-1,S'}(p)\right) && \text{if } a \geq 1 \\ &= f_{0,S'}(p) && \text{if } a = 0. \end{aligned}$$

Thus, we have

$$\begin{aligned} f'_{a,S} &= (1-p)f'_{a,S'} - f_{a,S'} + \left(1 + f_{a-1,S'}\right) + pf'_{a-1,S'} && \text{if } a \geq 1 \\ &= f'_{0,S'} && \text{if } a = 0. \end{aligned}$$

and

$$\begin{aligned} f''_{a,S} &= f''_{a,S'}(1-p) + f''_{a-1,S'}p + 2\left(f'_{a-1,S'} - f'_{a,S'}\right) && \text{if } a \geq 1 \\ &= f''_{0,S'} && \text{if } a = 0. \end{aligned}$$

Notice that $|S'| = t-1$ and therefore from induction, we have that $f''_{a,S} \leq 0$ for all a and $S \in \{0,1\}^t$. Therefore, we claim that Property **(P1)** holds for the case of t. Now, we show that Property **(P2)** holds for the case of t as well. We split the proof into two cases.

— Case 1: $a \geq 2$. In this case, we have that

$$f'_{a,S} - f'_{a-1,S} = (1-p)\left(f'_{a,S'} - f'_{a-1,S'}\right) + p\left(f'_{a-1,S'} - f'_{a-2,S'}\right) + f_{a-1,S'} - f_{a-2,S'} - (f_{a,S'} - f_{a-1,S'})$$

From the inductive hypothesis for Property **(P2)** and Claim 3, we have that $f'_{a,S} \geq f'_{a-1,S}$.

— Case 2: $a = 1$. In this case, we have

$$f'_{1,S} - f'_{0,S} = (1-p)\left(f'_{1,S'} - f'_{0,S'}\right) + 1 - (f_{1,S'} - f_{0,S'}).$$

From the inductive hypothesis for Property **(P2)** and Claim 3, we have that $f'_{1,S} \geq f'_{0,S}$.

Following a similar analysis to above, we can show the case when $S[0] = 1$. Therefore, we establish our claim. □

Claim 3. *For any S and $a \geq 2$, we have that (1) $f_{a-1,S} - f_{a-2,S} - (f_{a,S} - f_{a-1,S}) \geq 0$ and (2) $1 - (f_{1,S} - f_{0,S}) \geq 0$.*

Proof of Claim 3. Recall that $f_{a,S}$ with $|S| = T$ denotes the expected number of death events during the generalized death process with a initial count set as $a \geq 0$ and a fixed birth string $S \in \{0,1\}^T$. Consider a given death string $S^* \in \{0,1\}^T$, where $S^*(t) = 1$ indicates a checking point. The two-stage birth-death process with an initial count $a \geq 0$, a given birth string S, and a given death string S^*, denoted by TS-BDP(a, S, S^*), goes as follows: (i) Set $C = a$ and $D = 0$ at $t = 0$. (ii) For each time $t = 1, 2, \ldots, T$: (ii-a) Update $C \leftarrow C + S(t)$; (ii-b) If $S^*(t) = o$, then continue to $t+1$ (if $t < T$) or stop (if $t = T$); otherwise, update $C \leftarrow C - 1$ and $D \leftarrow 1$, provided that $C \geq 1$. Let f_{a,S,S^*} be the value of D (the number of death event) at the end. It will suffice to show that the two inequalities hold for f_{a,S,S^*} on any fixed $S^* \in \{0,1\}^T$.

Consider the process TS-BDP(a, S, S^*). Let $t_a \in [T]$ be the first checking point such that $C + S(t_a) = 0$ and $S^*(t_a) = 1$ and thus, we have no death event at t_a in TS-BDP(a, S, S^*). Set $t_a = T+1$ if such t_a doesnot exist in TS-BDP(a, S, S^*). When $t_a \leq T$: We have in TS-BDP$(a+1, S, S^*)$, $C + S(t_a) = 1$, and thus, we have have one extra death event at t_a and there will be only one, since the two processes TS-BDP(a, S, S^*) and TS-BDP$(a+1, S, S^*)$ will be the same when $t < t_a$ and $t > t_a$. When $t_a = T+1$ (t_a does not exist), the two process will have the same set of updates all the time. This analysis suggests that $f_{a+1,S,S^*} \leq f_{a,S,S^*} + 1$ for all $a \geq 0$. Thus, we prove the second inequality (which is $a = 0$). Note that $f_{a-1,S,S^*} - f_{a-2,S,S^*} = 1$ iff $t_{a-2} \leq T$ and $f_{a,S,S^*} - f_{a-1,S,S^*} = 1$ iff $t_{a-1} \leq T$. Since $t_{a-1} > t_{a-2}$, we have that $f_{a-1,S,S^*} - f_{a-2,S,S^*} \geq f_{a,S,S^*} - f_{a-1,S,S^*}$. Thus, we are done. □

10 CONCLUSION AND FUTURE DIRECTIONS

In this article, we present a mathematical model for crowdsourcing platforms where the number of workers is far fewer than the number of tasks (e.g., UberEats delivering food to customers). We propose an LP-based non-adaptive algorithm for the edge-weighted case, and GREEDY and two LP-based adaptive algorithms for the unweighted and vertex-weighted cases, respectively. These algorithms, as demonstrated in this article, can be theoretically analyzed to prove lower bounds on their competitive ratios. On the hardness side, we establish both conditional (based on benchmark LP) and unconditional hardness results. Finally, we implement all the algorithms on two datasets (a real-world and a synthetic) and evaluate their practical performance numerically.

Our work suggests several future directions. The primary open question is to formally analyze the performance of GREEDY for both the edge-weighted and vertex-weighted cases. In particular, it would be interesting to prove that GREEDY is optimal for the vertex-weighted case, as observed in the experiments. Another open direction is to propose an improved benchmark LP that closely approximates the best offline (expected) solution.

ACKNOWLEDGEMENTS

The authors thank the anonymous reviewers of AAMAS-2018 and the journal reviewers.

REFERENCES

[1] Elliot Anshelevich, Meenal Chhabra, Matthew Gerrior, and Sanmay Das. 2012. On the social welfare of mechanisms for repeated batch matching. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 3*. 1321–1322.

[2] Ali Aouad and Ömer Saritaç. 2020. Dynamic stochastic matching under limited time. In *Proceedings of the 21st ACM Conference on Economics and Computation*. 789–790.

[3] Itai Ashlagi, Maximilien Burq, Patrick Jaillet, and Vahideh Manshadi. 2019. On matching and thickness in heterogeneous dynamic markets. *Oper. Res.* 67, 4 (2019), 927–949.

[4] Itai Ashlagi, Maximilien Burq, Patrick Jaillet, and Amin Saberi. 2018. Maximizing efficiency in dynamic matching markets. Retrieved from https://arXiv:1803.01285

[5] Itai Ashlagi, Patrick Jaillet, and Vahideh H. Manshadi. 2013. Kidney exchange in dynamic sparse heterogenous pools. In *Proceedings of the ACM Conference on Economics and Computation*. 25–26.

[6] Sepehr Assadi, Justin Hsu, and Shahin Jabbari. 2015. Online assignment of heterogeneous tasks in crowdsourcing markets. In *Proceedings of the 3rd AAAI Conference on Human Computation and Crowdsourcing*.

[7] Allan Borodin and Ran El-Yaniv. 1998. *Online Computation and Competitive Analysis*. Cambridge University Press, New York, NY.

[8] Jonathan Bragg, Andrey Kolobov, Mausam Mausam, and Daniel S. Weld. 2014. Parallel task routing for crowdsourcing. In *Proceedings of the 2nd AAAI Conference on Human Computation and Crowdsourcing*.

[9] Brian Brubach, Karthik Abinav Sankararaman, Aravind Srinivasan, and Pan Xu. 2020. Online stochastic matching: New algorithms and bounds. *Algorithmica*, 82, 10 (2020), 2737–2783.

[10] Xi Chen, Will Ma, David Simchi-Levi, and Linwei Xin. 2016. Dynamic recommendation at checkout under inventory constraint. Retrieved from http://dx.doi.org/10.2139/ssrn.2853093

[11] Zhao Chen, Rui Fu, Ziyuan Zhao, Zheng Liu, Leihao Xia, Lei Chen, Peng Cheng, Caleb Chen Cao, Yongxin Tong, and Chen Jason Zhang. 2014. Gmission: A general spatial crowdsourcing platform. *Proc. VLDB Endow.* 7, 13 (2014).

[12] Natalie Collina, Nicole Immorlica, Kevin Leyton-Brown, Brendan Lucier, and Neil Newman. 2020. Dynamic weighted matching with heterogeneous arrival and departure rates. In *Proceedings of the International Conference on Web and Internet Economics*. Springer, 17–30.

[13] Djellel Eddine Difallah, Gianluca Demartini, and Philippe Cudré-Mauroux. 2013. Pick-a-crowd: Tell me what you like, and i'll tell you what to do. In *Proceedings of the 22nd International Conference on World Wide Web*. 367–374.

[14] Chinmoy Dutta and Chris Sholley. 2018. Online matching in a ride-sharing platform. Retrieved from https://arXiv:1806.10327

[15] Jon Feldman, Aranyak Mehta, Vahab Mirrokni, and S Muthukrishnan. 2009. Online stochastic matching: Beating 1-1/e. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS'09)*. IEEE, 117–126.

[16] Rajiv Gandhi, Samir Khuller, Srinivasan Parthasarathy, and Aravind Srinivasan. 2006. Dependent rounding and its applications to approximation algorithms. *J. ACM* 53, 3 (2006).

[17] Gagan Goel, Afshin Nikzad, and Adish Singla. 2013. Matching workers expertise with tasks: Incentives in heterogeneous crowdsourcing markets. In *Proceedings of the NIPS Workshop on Crowdsourcing*.
[18] Gagan Goel, Afshin Nikzad, and Adish Singla. 2014. Allocating tasks to workers with matching constraints: Truthful mechanisms for crowdsourcing markets. In *Proceedings of the 23rd International Conference on World Wide Web*. ACM, 279–280.
[19] Bernhard Haeupler, Vahab S. Mirrokni, and Morteza Zadimoghaddam. 2011. Online stochastic weighted matching: Improved approximation algorithms. In *Proceedings of the International Workshop on Internet and Network Economics*. Springer, 170–181.
[20] U. U. Hassan and E. Curry. 2014. A multi-armed bandit approach to online spatial task assignment. In *Proceedings of the 11th IEEE International Conference on Ubiquitous Intelligence and Computing (UIC'14)*. 212–219.
[21] Chien-Ju Ho and Jennifer Wortman Vaughan. 2012. Online task assignment in crowdsourcing markets. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*. 45–51.
[22] Zhiyi Huang, Ning Kang, Zhihao Gavin Tang, Xiaowei Wu, Yuhao Zhang, and Xue Zhu. 2018. How to match when all vertices arrive online. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, 17–29.
[23] Zhiyi Huang, Zhihao Gavin Tang, Xiaowei Wu, and Yuhao Zhang. 2019. Online vertex-weighted bipartite matching: Beating 1-1/e with random arrivals. *ACM Trans. Algor.* 15, 3 (2019), 1–15.
[24] Zhiyi Huang, Zhihao Gavin Tang, Xiaowei Wu, and Yuhao Zhang. 2020. Fully online matching ii: Beating ranking and water-filling. Retrieved from https://arXiv:2005.06311
[25] Zhiyi Huang and Qiankun Zhang. 2020. Online primal dual meets online matching with stochastic rewards: Configuration lp to the rescue. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*. 1153–1164.
[26] Patrick Jaillet and Xin Lu. 2013. Online stochastic matching: New algorithms with better bounds. *Math. Oper. Res.* 39, 3 (2013).
[27] Chinmay Karande, Aranyak Mehta, and Pushkar Tripathi. 2011. Online bipartite matching with unknown distributions. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing*. ACM, 587–596.
[28] Richard M. Karp, Umesh V. Vazirani, and Vijay V. Vazirani. 1990. An optimal algorithm for on-line bipartite matching. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*. 352–358.
[29] Andrey Kolobov, Daniel S. Weld et al. 2013. Joint crowdsourcing of multiple tasks. In *Proceedings of the 1st AAAI Conference on Human Computation and Crowdsourcing*.
[30] Will Ma and David Simchi-Levi. 2017. Online resource allocation under arbitrary arrivals: Optimal algorithms and tight competitive ratios. Retrieved from http://dx.doi.org/10.2139/ssrn.2989332
[31] Will Ma, Pan Xu, and Yifan Xu. 2023. Fairness maximization among offline agents in online-matching markets. *ACM Trans. Economics and Comput.* 10, 4 (2023), 1–27.
[32] Calum MacRury and Will Ma. 2023. Random-order contention resolution via continuous induction: Tightness for bipartite matching under vertex arrivals. Retrieved from https://arXiv:2310.10101
[33] Vahideh H. Manshadi, Shayan Oveis Gharan, and Amin Saberi. 2012. Online stochastic matching: Online actions based on offline statistics. *Math. Oper. Res.* 37, 4 (2012).
[34] Nicholas Mattei, Abdallah Saffidine, and Toby Walsh. 2017. Mechanisms for online organ matching. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 345–351.
[35] Aranyak Mehta. 2012. Online matching and ad allocation. *Theor. Comput. Sci.* 8, 4 (2012).
[36] Reshef Meir, Yiling Chen, and Michal Feldman. 2013. Efficient parking allocation as online bipartite matching with posted prices. In *Proceedings of the International Conference on Autonomous Agents and Multi-agent Systems*. 303–310.
[37] Luis Moreira-Matias, Joao Gama, Michel Ferreira, Joao Mendes-Moreira, and Luis Damas. 2013. Predicting taxi–passenger demand using streaming data. *IEEE Trans. Intell. Transport. Syst.* 14, 3 (2013), 1393–1402.
[38] Vedant Nanda, Pan Xu, Karthik Abhinav Sankararaman, John Dickerson, and Aravind Srinivasan. 2020. Balancing the tradeoff between profit and fairness in rideshare platforms during high-demand hours. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 2210–2217.
[39] Jeffrey M. Rzeszotarski and Aniket Kittur. 2011. Instrumenting the crowd: Using implicit behavioral measures to predict task performance. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*.
[40] Yaron Singer and Manas Mittal. 2011. Pricing tasks in online labor markets. In *Proceedings of the 11th AAAI Conference on Human Computation*. 55–60.
[41] Yaron Singer and Manas Mittal. 2013. Pricing mechanisms for crowdsourcing markets. In *Proceedings of the 22nd International Conference on World Wide Web*. 1157–1166.
[42] Adish Singla and Andreas Krause. 2013. Truthful incentives in crowdsourcing tasks using regret minimization mechanisms. In *Proceedings of the 22nd International Conference on World Wide Web*. 1167–1178.
[43] A. Subramanian, G. S. Kanth, S. Moharir, and R. Vaze. 2015. Online incentive mechanism design for smartphone crowd-sourcing. In *Proceedings of the 13th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt'15)*. 403–410.

[44] Yongxin Tong, Jieying She, Bolin Ding, Lei Chen, Tianyu Wo, and Ke Xu. 2016. Online minimum matching in real-time spatial data: Experiments and analysis. *Proc. VLDB Endow.* 9, 12 (2016).
[45] Y. Tong, J. She, B. Ding, L. Wang, and L. Chen. 2016. Online mobile micro-task allocation in spatial crowdsourcing. In *Proceedings of the IEEE 32nd International Conference on Data Engineering (ICDE)*. 49–60.
[46] Yongxin Tong, Libin Wang, Zimu Zhou, Bolin Ding, Lei Chen, Jieping Ye, and Ke Xu. 2017. Flexible online task assignment in real-time spatial data. *Proc. VLDB Endow.* 10, 11 (2017), 1334–1345.
[47] Van-Anh Truong and Xinshang Wang. 2019. Prophet inequality with correlated arrival probabilities, with application to two sided matchings. Retrieved from https://arXiv:1901.02552
[48] Xinshang Wang, Van-Anh Truong, and David Bank. 2018. Online advance admission scheduling for services with customer preferences. Retrieved from https://arXiv:1805.10412
[49] Yajun Wang and Sam Chiu-wai Wong. 2015. Two-sided online bipartite matching and vertex cover: Beating the greedy algorithm. In *Automata, Languages, and Programming. Part I.* Vol. 9134. 1070–1081.
[50] Mingjun Xiao, Jie Wu, Liusheng Huang, Ruhong Cheng, and Yunsheng Wang. 2017. Online task assignment for crowdsensing in predictable mobile social networks. *IEEE Trans. Mobile Comput.* 16, 8 (2017).
[51] Dejun Yang, Guoliang Xue, Xi Fang, and Jian Tang. 2012. Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing. In *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking*. 173–184.
[52] Huaxiu Yao, Fei Wu, Jintao Ke, Xianfeng Tang, Yitian Jia, Siyu Lu, Pinghua Gong, Jieping Ye, and Zhenhui Li. 2018. Deep multi-view spatial-temporal network for taxi demand prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.
[53] Qi Zhang, Yutian Wen, Xiaohua Tian, Xiaoying Gan, and Xinbing Wang. 2015. Incentivize crowd labeling under budget constraint. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM'15)*. IEEE, 2812–2820.
[54] Kai Zhao, Denis Khryashchev, Juliana Freire, Cláudio Silva, and Huy Vo. 2016. Predicting taxi demand at high spatial resolution: Approaching the limit of predictability. In *Proceedings of the IEEE International Conference on Big Data (Big Data'16)*. IEEE, 833–842.

Received 2 October 2021; accepted 7 March 2024