Allocation Problem in Remote Teleoperation: Online Matching with Offline Reusable Resources and Delayed Assignments

Osnat Ackerman Viden [*]
Bar-Ilan University
Ramat Gan, Israel
osnatackerman0376@gmail.com

Yohai Trabelsi [*]
Bar-Ilan University
Ramat Gan, Israel
yohai.trabelsi@gmail.com

Pan Xu
New Jersey Institute of Technology
Newark, NJ, USA
pxu@njit.edu

Karthik Abinav Sankararaman
Meta
Austin, TX, USA
karthikabinavs@gmail.com

Oleg Maksimov
Bar-Ilan University
Ramat Gan, Israel
oleg@maksimov.co.il

Sarit Kraus
Bar-Ilan University
Ramat Gan, Israel
sarit@cs.biu.edu

ABSTRACT

Many applications where tasks should be assigned to agents can be modeled as matching in bipartite graphs. In this paper, we consider applications where tasks arrive dynamically and rejection of a task may have significant adverse effects on the requester, therefore performing the task with some delay is preferred over complete rejection. The performance time of a task depends on the task, the agent, and the assignment, and only its distribution is known in advance. The actual time is known only after the task performance when the agent is available for a new assignment. We consider such applications to be one of two arrival types. With the first type, the arrival distribution is known in advance, while there is no assumption about the arrival times and order with the second type. For the first type, we present an LP-based online algorithm with a competitive ratio of 0.5. For the second type, we show no online algorithm with a constant competitive ratio. We run extensive experiments to evaluate our algorithm in a real-world dataset, demonstrating the advantages of the LP approach.

CCS CONCEPTS

• **Computing methodologies → Multi-agent planning**; • **Theory of computation → Online algorithms**.

KEYWORDS

Bipartite matching; Resource allocation; Teleoperation; Online matching

ACM Reference Format:
Osnat Ackerman Viden [*], Yohai Trabelsi [*], Pan Xu, Karthik Abinav Sankararaman, Oleg Maksimov, and Sarit Kraus. 2023. Allocation Problem in Remote Teleoperation: Online Matching with Offline Reusable Resources and Delayed Assignments. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 9 pages.

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom. © 2023 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

1 INTRODUCTION

In classic bipartite matching problems such as assigning classes to classrooms [28] or papers to reviewers [24], both sides of the matching are known in advance. However, in many other problems, one side of the matching is known, but the other side is dynamic. Such is the case in many two-sided marketplace problems where the available supplies (e.g., goods in a grocery store) are known in advance but the requirements (e.g., customers' demand) are dynamic.

In other applications, the product is rented out and will be available after the current customer finishes using it. An example of this is a cell phone application such as "Get Taxi" that assigns drivers to continuously arriving customers. Once a taxi finishes its journey it becomes available for the next customer. In some cases, we may assume that the usage time depends only on the user (e.g., when taxi drivers are very similar to one another). However, sometimes there are significant differences between them. Gong et al. [13] showed that, in the former case, there is no approximation algorithm with a reasonable ratio for the problem. In some situations, depending on the task properties and the available resources, it is possible that a request will not be fulfilled. If this occurs frequently, it may damage the usefulness and popularity of the application. In other applications, rejecting a request is disastrous. An example of such an application is assigning hospital resources like beds or doctors to patients (see, for example, [32]). In such applications, rejecting a patient or even serving her too late is not acceptable. At the beginning of the COVID-19 pandemic, many people who didn't get the required resources such as trained staff, beds, or respirators were affected severely or even died (see for example Emanuel et al. [9]).

Our motivating application is teleoperating autonomous cars (driving them remotely) [2, 15, 16, 35] that have special characterizations. In many situations, autonomous cars get stuck and cannot continue their trip without human intervention. For example – consider an object permanently blocking the vehicle's way. A reasonable human driver would bypass this object even if they had to cross a continuous dividing line to do so. That decision cannot be made by an autonomous vehicle. The establishment of dedicated centers of human teleoperators, whose job is to deal with such cases will be required [40] and already exists (*e.g.,* autonomous mining trucks [34]). Furthermore, in many countries, a process has

[*]Equal contribution

been started to define regulations for such solutions [23]. When an autonomous vehicle encounters a scenario that it doesn't know to deal with, it will call for help from human teleoperators sitting in the centers. The chosen teleoperator from the center will take control of the car and drive it remotely. Once the situation is resolved, control will be given back to the car for continued autonomous driving. In this domain, the efficiency (i.e., task waiting time) of the allocation of a request for intervention (tasks) to human operators (agents) is extremely important since waiting in a stuck car is extremely annoying and may become dangerous (*e.g.,* if the car is blocking a road). Therefore, in this case, it is not acceptable to refuse a task or have a long delay. Another characterization of this application is that the time duration of the performance of a request for intervention (task) depends on the operator, the type of request for the intervention itself, and the assignment time. The dependency on the assignment time stems from the traffic load changes during the day. Furthermore, the arrival of a specific type of request for intervention also depends on the time and the type of request.

In this study, we address delayed online matching problems where a long delay or rejection is costly or unacceptable and one of two settings below applies. In the first setting, the expected arrival rate is known to the algorithm. We develop a novel online algorithm with a competitive ratio of 0.5 (a ratio between the expected yield of the online algorithm and an optimal offline algorithm) and demonstrate its effectiveness through extensive experiments on a dataset based on both simulations and real-world data. For settings where the arrival rate is unknown, we have the second category. In this category, there is no knowledge about the arrival distribution of tasks and therefore we consider an adversarial model for the sequence of arriving tasks. We adopt a proof of Gong et al. [13] and show that there is no online algorithm with a competitive ratio better than $O(\log T/T)$.

2 RELATED WORK

Online and offline bipartite matching is a rich area of study with many theoretical works. We are specifically interested in weighted matching problems, where each edge is assigned a weight and our goal is to maximize the total weight of the matching edges. The bipartite online matching problem was introduced by Karp et al. [21]. In this problem, tasks arrive over time and must be *performed as they arrive.* On the other hand, the agents serving these tasks are stationary. In teleoperation environments, we consider rejecting tasks problematic, and a short delay in providing the service is allowed.

Following the paper by Wang et al. [37], work on online matching can be divided into two categories. The first refers to problems that use the adversarial model. In this model, there is no prior knowledge about the incoming tasks and the order in which they arrive. The goal of the work in this category is to provide a bound for the worst-case scenario. The second category assumes some knowledge about the underlying distribution of incoming tasks. Work that belongs to this category usually attempts to optimize the expected utilization or revenue. In this paper, we address both categories.

Another branch of the problem is one where resources can be reused. Most similar to our work is that of Dickerson et al. [7], who considered the online matching problem when resources are reusable and the distribution of customer types is known. They considered cab dispatching and ride-sharing applications where stationary drivers are dynamically assigned to incoming tasks. They proposed a simulation- and LP-based approach and presented an algorithm with an online competitive ratio of $1/2 - \epsilon$ for any given $\epsilon > 0$. However, in their case tasks cannot be delayed and are rejected if they cannot be assigned immediately upon arrival. For the adversarial setting, Goyal et al. [14] discussed the case of online allocation of reusable resources and the uncertainty about their usage duration when their capacity is large or even infinite. Reusable resources have also been considered in the context of e-commerce (e.g., Feng et al. [10]) and crowdsourcing (e.g., Manshadi and Rodilitz [25]).

Righter [30] considers the problem of allocating resources to activities where the activities are known in advance and the resources arrive dynamically and delays are allowed. However, the arrival time or the waiting time does not affect the reward and therefore it is not useful in our application.

Dervovic et al. [6] assumes a Poisson process for the arrival of jobs and has presented an algorithm with a performance guarantee for the expected revenue. Recently, Aouad and Sarıtaç [1] consider a dynamic matching setting, where the algorithm can choose to handle the tasks in batches and serve them. However, unlike our setting, they do not consider reusable resources and assume that we can match more than one edge at a given time step.

Wang et al. [37] use a reinforcement learning(RL) approach to solve online matching with delays, where both sides of bipartite graphs arrive dynamically over time. However, as in the previous work, the value of an assignment depends only on the agent and the task and not on other important features such as the arrival time and the waiting time. In addition, workers arrive dynamically and therefore we cannot use the available knowledge about the current workers to optimize the reward obtained. In both works, the only drawback associated with a longer waiting time is the higher probability that workers or jobs will disappear before they are assigned. Qin et al. [29] presents an RL approach to balance the cost of delaying tasks with the benefit of improved allocation efficiency. However, they assume a static Poisson process for arrivals, which is different from our more realistic arrival probabilities. Moreover, in the last couple of works, the decision about the time steps for assigning a request (or a driver) is made in batches, which reduces the complexity of the problem but does not optimize the utility of the assignment. Recent work by Li et al. [22] also addresses the problem of solving online matching with delays, where both sides of bipartite graphs arrive dynamically. They propose an LP-based approach with a constant approximation ratio. However, in their setting, the delay is not penalized but only capped.

Jintao et al. [20] presented a combined approach of RL and combinatorial optimization for cases where the decision is made for each passenger individually. Their model attempts to optimize some factors, such as the number of rejected tasks and the time from request arrival to matching. However, like many other RL works, they have not obtained theoretical results regarding the optimality of their approach compared to the performance of an optimal offline solution.

Another area that overlaps with online matching is online assortment optimization. In these problems, the decision maker's objective is to select a subset of products from the available resources to offer to the user, who in turn chooses to buy one of them, in order to maximize the expected reward of the decision maker, given the user's selection. For any given subset of the products offered, the user's choice depends on his probabilistic preference for the set of products, including the option to buy nothing or to drop out. The optimal selection of a subset is similar to the selection of the optimal single resource in a matching problem, which is considered an assortment problem where the resources are reusable and the execution time is uncertain.

In the literature, areas such as crowdsourcing, ride-sharing, ride-hailing and internet advertising have been studied in depth. For example, Ho and Vaughan [17] have studied the problem of assigning heterogeneous tasks to agents with different unknown skills in crowdsourcing markets such as Amazon Mechanical Turk. They present a two-stage task assignment algorithm and empirically evaluate this algorithm using data collected on Mechanical Turk. They show that this algorithm performs better than random assignments or greedy algorithms. In addition, Tong et al. [36] identified a practical micro-task allocation problem called *"the Global Online Micro-task Allocation in spatial crowdsourcing"* (GOMA). They considered the average performance of online algorithms, also known as the online random-order model, and demonstrated the effectiveness and efficiency of the proposed methods through extensive experiments on real and synthetic datasets.

Another family of problems involves multi-class queues where each task has its own characteristics, such as its urgency, the time it takes to be completed, and its arrival rate. Yoon and Lewis [39] address the problem of finding an optimal admission policy for queues with multiple classes. However, they have only one class of tasks. Rigter et al. [31] consider multiple classes of tasks. However, they only consider cases where all servers are identical. This assumption is unacceptable in our setting since there are very large differences between different teleoperators.

To the best of our knowledge, there is no research in the literature that pertains to our problems of online arrival of tasks during a finite time period, where the delay is allowed but is costly and resources are reusable. Furthermore, the duration time of the performance of a task depends on the arrival time, the resource, and the type of the task. When the arrival time duration is known, the arrival probability depends on the arrival time. Finally, we aim to bind the competitive ratio of the proposed solution.

3 PROBLEM DEFINITION: ONLINE MATCHING WITH DELAYED ASSIGNMENTS (OMDA)

Following works by [18, 27, 38], we use a bipartite graph $G = (I, J, E)$ to model the network between offline agents I (human operators) and online agent types J (task types), where an edge $e = (i, j)$ indicates the feasibility of matching between agent i and agent of type j due to practical constraints. We assume here by default that the offline agents are all static, while the online agents arrive dynamically. The online process is as follows: we have a time horizon of T rounds. During each round (interchangeably time) $t \in [T] := \{1, 2, \ldots, T\}$, a task of type $\hat{j}$ will be sampled from J such that $\Pr[\hat{j} = j] = p_{j,t}$ with $\sum_{j \in J} p_{j,t} = 1$[1]. In this case, we say that a task of the sampled type, $\hat{j}$, arrived at time t. For each task of type j arriving at time t, we can assign it to any offline neighbor i with $(i, j) \in E$ at any time $t' \geq t$ as long as i is available (*i.e.,* unmatched) at t', which is referred to as an assignment $\lambda = (j, t, i, t')$. Let $\Lambda = \{\lambda = (j, t, i, t') | (i, j) \in E, t' \geq t\}$ be the collection of all valid assignments. For each valid assignment $\lambda = (j, t, i, t') \in \Lambda$, it is associated with a positive reward w_λ gained by the system and a usage duration $C_\lambda \in [T]$, which denotes the (random) number of rounds during which i will be occupied by the assignment of the task of type j arriving at t that is scheduled on i at t'. Note that our setting is general enough to allow both reward (w_λ) and the occupation distribution (C_λ) to be sensitive to all of the four elements involved in the assignment, *i.e.,* the task type, the arriving time, the assigned agent, and the scheduling time. We assume the distributions of $\{C_\lambda\}$ are all accessible to the algorithm. In addition to that, all information of $\{G = (I, J, E), T, \{w_\lambda\}\}$ is known as part of the input. We consider two variants of the problem. In the first case, the arriving probabilities $\{p_{j,t}\}$ are also known to the algorithm, which is referred to as OMDA-KD; while in the second case, they are unknown but fixed by an adversary (referred to as OMDA-UKD). *Our goal is to design online policies (or algorithms) such that the expected total rewards are maximized.* Throughout this paper, we denote $[n] = \{1, 2, \ldots, n\}$ for a generic positive integer n; we use OPT to denote both a clairvoyant optimal policy and the corresponding performance, and the same applies to ALG, which denotes both a generic algorithm and its performance.

4 FIRST CASE: KNOWN ARRIVAL DISTRIBUTIONS

In this section, we describe the case where the arrival probabilities $\{p_{j,t}\}$ of the tasks are known. First, we define the competitive ratio used to evaluate our algorithm. Then, we define a benchmark linear program used by the algorithm. Finally, we define our algorithm and give a theoretical guarantee for its performance.

4.1 Competitive Ratio (CR)

The CR is a commonly-used metric to evaluate the performance of online algorithms. Consider a given online algorithm (or policy) ALG and a clairvoyant optimal OPT. In the context when the arrival distributions are known in advance, ALG observes task arrivals sequentially, while OPT can access all arrivals of tasks at the very beginning (including their arriving time). Neither ALG nor OPT has the information of the exact realized values of $\{C_\lambda\}$, but both can access their distributions in advance. The CR is defined as $\mathbb{E}[\text{ALG}]/\mathbb{E}[\text{OPT}]$.

4.2 Benchmark Linear Program (LP)

For each assignment λ, let x_λ be the probability that λ is made in OPT. Recall that $\Lambda = \{\lambda = (j, t, i, t') | (i, j) \in E, t' \geq t\}$ is the collection of all valid assignments. For each given pair $j \in J$ and $t \in [T]$, let $\Lambda_{j,t}$ denote the collection of all valid assignments

[1]We can always make this equal by creating a dummy node whose arrival simulates the case of no arrival at time t.

involving task of type j arriving at t. Similarly, let $\Lambda_{i,t'}$ be that of all valid assignments that are scheduled to be matched on i at t'.

$$\max \sum_{\lambda \in \Lambda} x_\lambda \cdot w_\lambda, \tag{1}$$

$$\sum_{\lambda \in \Lambda_{j,t}} x_\lambda \le p_{j,t}, \qquad \forall j \in J, \forall t \in [T] \tag{2}$$

$$\sum_{t' \le t} \sum_{\lambda \in \Lambda_{i,t'}} x_\lambda \Pr[C_\lambda > t - t'] \le 1, \forall i \in I, \forall t \in [T] \tag{3}$$

$$0 \le x_\lambda \le 1, \qquad \forall \lambda \in \Lambda. \tag{4}$$

Throughout this paper, we refer to the LP above simply as LP (1) that has Constraints (2) to (4) by default.

Lemma 1. *The optimal value of* LP (1) *is a valid upper bound of the total expected reward achieved by a clairvoyant optimal policy for* OMDA-KD.

Proof. Note that for each valid assignment $\lambda \in \Lambda$, x_λ denotes the probability that λ is made in clairvoyant optimal policy (OPT). We can verify that Objective (1) encodes the expected reward in OPT. Thus, to prove Lemma 1, it suffices to show that $\{x_\lambda\}$ is feasible to all constraints in LP (1). For Constraint (2): Observe that the left-handed side (LHS) value denotes the probability that a task of type j arriving at time t gets assigned in OPT. Thus, it should be no larger than the probability that j arrives at t, which is exactly equal to $p_{j,t}$. As for Constraint (3): consider a given i and t. The summation over $t' < t$ on the LHS represents the probability that i is occupied by some assignments made at some previous time $t' < t$, while the summation on $t = t'$ represents the probability that i is available at t since $\Pr[C_\lambda > 0] = 1$. Thus, the sum of these two parts of $t' < t$ and $t' = t$ should be no more than 1. Constraint (4) is trivial since x_λ is a probability value. Therefore, we establish the feasibility of $\{x_\lambda\}$ to all constraints in LP (1). □

4.3 A Sampling Policy with Attenuations

We present an LP-based sampling policy with attenuations in ALG-LP. First, the algorithm solves the LP (1). Then, during its online phase, it iterates through all the available unprocessed valid assignments and assigns them to an agent with some probability if the agent is available. Note that all unprocessed tasks that have arrived so far are considered candidates for the assignment.

Slightly abusing the notation, we use $\{x_\lambda\}$ to denote an optimal solution to LP (1).

Remarks on ALG-LP. (i) For each $t \in [T]$, let $\Lambda(t) \subseteq \Lambda$ be the set of all possible valid assignments that are scheduled at t. We can impose an *arbitrary* order π_t over $\Lambda(t)$, which then yields an order on $\mathcal{S}_t$. Note that $\mathcal{S}_t$ is a *random* set but surely is a subset of $\Lambda(t)$. Here is an example of π_t over $\Lambda(t)$. Consider two different assignments $\lambda = (j, t', i, t)$ and $\bar{\lambda} = (\bar{j}, \bar{t}', \bar{i}, t)$ in $\Lambda(t)$. We set $\lambda \prec \bar{\lambda}$ (λ falls before $\bar{\lambda}$ under π_t) if either $i < \bar{i}$[2] or $t' < \bar{t}'$ or $j < \bar{j}$. (ii) The value β_λ can be estimated at an arbitrary accuracy via simulating Steps (5) to (11) before ALG-LP checking λ for enough number of times.[3] Note that β_λ can be affected by the orders $\{\pi_t\}$ imposed on $\{\mathcal{S}_t\}$. (iii) The constant 0.5 in line 10 is the best possible we can choose. Let α be the constant such that we sample each valid available assignment x_λ from $\mathcal{S}_t$ with a probability $\alpha x_\lambda / \beta_\lambda$. On the one hand, we should set an α value as large as possible since it determines the final competitive ratio achieved by ALG-LP($\cdot$); on the other hand, to ensure ALG-LP($\cdot$) functions well, we can set the α value no more than 0.5 (see the proof of Theorem 1). That's how we get the final choice of 0.5.

Theorem 1. ALG-LP($\cdot$) *achieves a competitive ratio of* 0.5 *for* OMDA-KD.

Proof. We first show that the expected total weight of all assignments made by ALG-LP should be at least half of the optimal value of the benchmark LP-(1). This further suggests a competitive ratio of 0.5 since LP-(1) offers a valid upper bound for the clairvoyant optimal (by Lemma 1). For showing the ratio between the expected weight of assignments and the optimal solution of the LP-(1), we observe that in Step 10 of ALG-LP, we sample an assignment with probability $0.5 x_\lambda / \beta_\lambda$. Thus, the fact that $\beta_\lambda \ge 0.5 x_\lambda$ for every $\lambda \in S_t$ and every $t \in [T]$ is a critical (and also sufficient) condition that ensures ALG-LP function properly. Under that assumption, we see that each valid assignment λ is scheduled successfully with a probability equal to $0.5 x_\lambda$. The ratio follows from the linearity of expectation. It remains to prove the fact that every assignment in Λ will be made with a probability of at least $0.5 x_\lambda$. The proof is by induction over a given order $\pi = \{\pi_t\}$ imposed on $\Lambda = \cup_t \Lambda(t)$, where $\Lambda(t)$ is the collection of all valid assignments that are scheduled at $t \in [T]$. Consider the base case when $t = 1$. Let $\lambda = (j, t' = 1, i, t = 1)$ be the first assignment under order π_1 on $\mathcal{S}_1$. In this case, we see $\beta_\lambda = p_{j,t'} \ge 0.5 x_\lambda$, which follows from the fact that $x_\lambda \le p_{j,t'}$ due to Constraint (2). Consider a given valid assignment $\bar{\lambda} = (\bar{j}, \bar{t}', \bar{i}, \bar{t}) \in \Lambda$. Let $\Lambda(\bar{\lambda})$ be an *ordered* collection of all valid assignments before $\bar{\lambda}$ that are arranged following the given order imposed on Λ. Now assume that any $\lambda \in \Lambda(\bar{\lambda})$ is made with a probability equal to $0.5 x_\lambda$, and we show $\bar{\lambda}$ will be made with a probability equal to $0.5 x_{\bar{\lambda}}$. It suffices to prove that $\beta_{\bar{\lambda}} \ge 0.5 x_{\bar{\lambda}}$. Assume $\bar{j}$ arrives at time $\bar{t}'$, which occurs with probability $p_{\bar{j},\bar{t}'}$. Observe that $(\bar{j}, \bar{t}')$ remains in Q upon ALG-LP. Checking $\bar{\lambda}$ should happen with a probability equal to $p_{\bar{j},\bar{t}'} - \sum_{\lambda = (\bar{j}, \bar{t}', *, *) \in \Lambda(\bar{\lambda})} 0.5 x_\lambda$. Note that $\bar{i}$ is safe at $\bar{t}$ upon ALG-LP. Checking $\bar{\lambda}$ with a probability equal to $1 - \sum_{\lambda = (*, *, \bar{i}, t) \in \Lambda(\bar{\lambda})} 0.5 x_\lambda \Pr[C_\lambda > \bar{t} - t] \ge 1 - 0.5$ *unconditionally* due to Constraint (3). Observe that the event $\bar{j}$ that arrives at $\bar{t}'$ will not affect the arrival of an online agent during any other $t \ne \bar{t}'$ (since arrivals are independent over different rounds); meanwhile, the event that $(\bar{j}, \bar{t}')$ remains in Q by ALG-LP checking $\bar{\lambda}$ implies that $(\bar{j}, \bar{t}')$ has not ever been matched with $\bar{i}$. This suggests that the occurrence of an event where $(\bar{j}, \bar{t}')$ remains in Q could only positively contribute to the chance that $\bar{i}$ is safe at $\bar{t}$. Thus, we claim that $\beta_{\bar{\lambda}} \ge 0.5 \cdot \left(p_{\bar{j},\bar{t}'} - \sum_{\lambda = (\bar{j}, \bar{t}', *, *) \in \Lambda(\bar{\lambda})} 0.5 x_\lambda\right) \ge 0.5 x_{\bar{\lambda}}$, where the second inequality is due to the fact that $\sum_{\lambda = (\bar{j}, \bar{t}', *, *) \in \Lambda} x_\lambda \le p_{\bar{j},\bar{t}'}$ from Constraint (2). □

[2] We assume i is indexed as $1, 2, \ldots, n$ with $n = |I|$. Similarly for $j \in J$.

[3] The number of simulations needed is $\text{poly}(|I|, |J|, T, 1/\epsilon)$ if we aim for a multiplicative error of $\epsilon > 0$.

Algorithm 1: An LP-based sampling policy with attenuations for OMDA-KD: ALG-LP(·).

```
1  Offline Phase:
2  Solve LP (1) and let $\{x_\lambda\}$ be an optimal solution.
3  Set $Q = \emptyset$, $\mathcal{A} = I$.
   /* $\mathcal{A}$ is a set storing all currently available offline agents; $Q$ is a set storing all unprocessed tasks
      arriving so far in the form of $(j, t)$ (the task type together with its arriving time).   */
4  Online Phase:
5  for $t = 1, \dots, T$ do
6      Let a task of type $j$ arrive at (the beginning of) time $t$. Update $Q = Q \cup \{(j, t)\}$.
7      Let $\mathcal{S}_t = \{\lambda = (j, t', i, t) \in \Lambda : i \in \mathcal{A}, (j, t') \in Q\}$, the collection of all available unprocessed valid assignments that are scheduled
        at $t$.
8      for each $\lambda \in \mathcal{S}_t$ (following a specific order $\pi_t$ over $\mathcal{S}_t$; see Remarks on ALG-LP) do
9          Let $\lambda = (j, t', i, t)$ and $\beta_\lambda = \Pr[\lambda \in \mathcal{S}_t]$, i.e., $i \in \mathcal{A}$ and $(j, t') \in Q$ at $t$, which can be obtained via simulations (see Remarks
            on ALG-LP).
10         With probability $0.5x_\lambda/\beta_\lambda$: Assign $(j, t')$ to $(i, t)$ with $\lambda = (j, t', i, t)$, update $\mathcal{A} \leftarrow \mathcal{A} \setminus \{i\}$, $Q \leftarrow Q \setminus \{(j, t')\}$, and remove any
            assignment in $\mathcal{S}_t$ in the form of either $(*, *, i, t)$ or $(j, t', *, t)$; with probability $0.5x_\lambda/\beta_\lambda$: Skip. (see Remarks on ALG-LP).
11     For any offline agent $i$ that finishes its task by (the end of) time $t$, add it to $\mathcal{A}$ by $\mathcal{A} \leftarrow \mathcal{A} \cup \{i\}$.
```

4.4 Impossibility

We consider two types of impossibility. First, we claim that no online algorithm can achieve a competitive ratio better than 0.5 with respect to the benchmark LP (1). This claim follows from the proof of Theorem 2 in the work of Dickerson et al. [7]. Second, Manshadi et al. [26] showed that for a known arrival distribution, no algorithm can have a better competitive ratio than 0.823 with respect to the optimal offline solution. In their setting, no delay is allowed and offline resources are disposable. However, their problem is a private case of ours and hence their results hold for our problem as well.

5 SECOND CASE: ADVERSARIAL ARRIVALS

In this section, we describe the case where the distribution is unknown. In this case, we must consider arbitrary bad arrival rates. We first define what a competitive ratio is and then show that it is impossible to find an algorithm with a constant competitive ratio for the problem with $|I| > 1$.

5.1 Competitive Ratio (CR)

In the context where the arrival distribution is unknown, ALG does not use information about arrival times. OPT can access all arrival times of the tasks at the very beginning. Both have access to the distribution of usage times. Here again, the ratio is defined as $\mathbb{E}[\text{ALG}]/\mathbb{E}[\text{OPT}]$.

5.2 Impossibility Results

Proposition 1. *In the adversarial arrival model, there is no online algorithm with a constant competitive ratio.*

Proof. We rely on Theorem 2 in the work of Gong et al. (2019). The theorem is as follows (reformulated according to our notations).

Theorem 2. *For online matching with a single type reusable agent and an arbitrary number of agents, if the random duration of use depends on the task, no online (randomized) algorithm can have a competitive ratio better than $O(\log T/T)$.*

In their proof, Gong et al. [12] construct arrival sequences and prove by contradiction that there is no algorithm with a competitive ratio better than $O(\log T/T)$ for these sequences. We construct similar sequences and adjust their proof such that it holds for our setting. We define a sequence $C(c, t)$ as a sequence of $c \cdot T^t$ tasks, each having identical usage duration distributions. In our construction, the workers are identical and for each pair with a task and a worker, the task is completed after a single unit of time with probability $p_t = 1 - \frac{1}{T^t}$ or after $T+1$ units of time, i.e., $\Pr(d_t = 0) = p_t = 1 - \frac{1}{T^t}$ and $\Pr(d_t = T + 1) = 1 - p_t = \frac{1}{T^t}$, where c denotes the number of agents. The set of T sequences considered in the proof is defined as $D(T) = \{C(c, 1), .., C(c, T)\}$.

□

6 EXPERIMENTS

Teleoperation of autonomous vehicles has been gaining a lot of attention recently (e.g., [3, 11, 33, 40]) and is expected to play an important role in helping autonomous cars handle challenging situations which they cannot handle on their own. Efficient assignment of online requests for interventions arriving from the vehicles to the appropriate operators is essential for making teleoperation centers feasible by enabling the reduction of the number of human operators employed at a given time. In the next sections, we will describe, in detail, the extensive experiments we ran in a simulated teleoperation environment in order to evaluate the proposed algorithm. We will compare it with a greedy algorithm and with a heuristic that is based on the LP approach.

In order to evaluate the LP approach in the teleoperation of the autonomous cars simulation environment, we need data on the expected duration time of a task of type j performed by a given operator (C_λ) and the arrival distribution of the task types over time ($p_{j,t}$). Due to the early stage of the technology and regulatory and economic barriers, data has not yet been collected from operational teleoperation centers. Therefore, we used two sources for data. First, we ran experiments with human subjects driving in a simulator to generate the duration time. Second, we estimated the arrival times based on estimates presented in the literature on autonomous cars and their requests for interventions. Finally, we had to define a benefits function that will be used for the evaluation of the teleoperation centers. We did this in consultation with members of the industry.

Duration time dataset. We collected data on expected duration times using the CARLA platform [8]. The CARLA platform is an open-source driving simulation, which is widely in use for research in autonomous vehicles(e.g., Caesar et al. [4], Codevilla et al. [5]). We simulated some challenging driving scenarios and asked human participants to drive in these situations. We defined four driving task types and asked ten subjects to drive vehicles to handle tasks of these types. The subjects were students of computer science - 5 women and 5 men aged 22 to 31 years (the average age was 25.2 years). The tasks included driving in extreme weather conditions, turning left at a signalized intersection when cars were approaching from the opposite direction, and passing static and dynamic obstacles. The subjects (operators) showed different levels of skill in their performance both in general and specifically for each type of task. That is, the average time to complete a task depends on both the operator and the type of the specific task. Among relevant data, we collected the usage duration times and used them to predict the duration, $C_\lambda \in [T]$, for each pair of a human operator and an intervention request.

Arrival Distribution. Arrival rates were determined following the data presented in [16], which describes the expected request rate for remote operators to assist autonomous cars in "edge" driving scenarios. The expected number of requests for intervention at a given hour of a day was calculated based on the estimated mileage that an autonomous car will drive in a given city (Table 1 in [16], the distribution over the day (Figure 2 in [16]) and the estimation of the number of requests for intervention per mileage specified in [16]. We considered a setting where the operators were responsible for autonomous cars in the entire area of New York, NY, Washington DC, Philadelphia, PA, and Atlanta, GA. We picked a time window of 5 hours, from 3 pm to 8 pm, with different traffic loads.

Experiment scenarios. In order to evaluate ALG-LP, we generated three different scenarios where tasks arrived according to the defined arrival time from Section 6 and the expected duration as defined in Section 6. In particular, in each scenario, we sampled the type of the arriving tasks, their parameters, and the usage duration according to the described distributions. Assuming that for a certain task type an operator performs tasks of this type differently each time (due to the effects of the environment and the mood of the operator, as well as basic variation in performance arising from reality), we sampled the usage duration 5 times for each scenario. So, in total, we created 15 different simulations for the 3 scenarios. Since ALG-LP involves random sampling, we ran it 10 times for each simulation and calculated the average among them. Each task type j was associated with a quality q_j. For an assignment (j, t, i, t'), we refer to $t' - t$ as the waiting time of task of type j that arrived at time t. In the experiments, we bounded the waiting time to make the scenario more realistic and to reduce the computation time of solving the LP.

The algorithm performance is measured by a score function that considers both the quality of the performed tasks which is determined by the quality(importance) of their types as well as the time it took to complete a task (including the waiting times). We use two normalization functions. For time normalization, we use $\phi_t(y) = \ln(y)/\ln(\text{max-time})$ where max-time is the maximal overall time including the actual performance and the waiting time. For quality $\phi_q(q_j) = q_j/\text{max-quality}$ where max-quality is the maximum quality associated with any task type. Using these functions, any request of type $j \in J$ that arrived at time $t \in [T]$, was assigned at time $t' \in [T]$ and its duration time was $0 < c < T$, we associated a value using the following score function:

$$v(j, c, t, t') = -\gamma\phi_t(c + t - t') + (1 - \gamma)\phi_q(q_j) + 1$$

where $0 < \gamma < 1$. We aim at maximizing the sum of $v(j, c, t, t')$ for all performed tasks.

For the LP (1) we set the reward to be

$$w_{j,t',i,t} = -\gamma\phi_t(C_{j,t',i,t}) + (1 - \gamma)\phi_q(q_j) + 1.$$

The datasets used in the experiments and the source code are available in a public repository[4].

6.1 Heuristics

When the number of agents is much higher than the number of arriving tasks, there are many solutions to LP (1). However, all of the solvers that we considered generate solutions in which the number of variables that are set to zero is maximized. As a result, and since constraint (3) of LP (1) is relatively weak, it occurs that, for several agents, all of their relevant variables are set to zero, and therefore these agents are not used in the matching. This leads to tasks being rejected even though there are agents that are available. To mitigate this problem, we modify the LP (1) as follows: Let Ω be the total expected reward achieved by the original LP (1), $0 < \epsilon < 1$, $0 < \delta < 1$ and $\kappa : I \times J \times T \times T \times [0, 1] \to [0, 1]$. $\kappa(i, j, t, t', \gamma)$ was determined using trial and error.

We add two constraints to LP (1) and refer to this heuristic as ALG-LP-Non-Zero (ALG-LP-NZ).

$$\sum_{\lambda \in \Lambda} x_\lambda \cdot w_\lambda \geq \Omega - \epsilon, \tag{5}$$

$$x_{j,t,i,t'} \geq p_{j,t} \cdot \delta \cdot \kappa(i, j, t, t', r), \forall i \in I, \forall j \in J, \forall t', t \in [T] \tag{6}$$

We also slightly change the sampling method. For any $i \in \mathcal{A}$ following an order π_{ti}, let $\mathcal{S}_{ti} = \{\lambda = (j, t', i, t) \in \Lambda : (j, t') \in \mathcal{Q}\}$. Let $\psi = \sum_{\lambda \in \mathcal{S}_{ti}} x_\lambda$. We take a sample of λ with $0.5x_\lambda/\psi\beta_\lambda$. That is, nothing is skipped. When analyzing the competitive ratio of ALG-LP-NZ, we consider the differences between the original ALG-LP and ALG-LP-NZ. The first difference is the addition of constraints (5) and (6). These constraints could reduce the ratio by at most $\epsilon/2$. The other is the modified sampling (a boost-like). We hypothesize that the change in sampling does not reduce the competitive ratio. This hypothesis is supported by the experimental evaluation, where the actual ratio was much larger than 0.5.

The other heuristic that we considered is a greedy heuristic that does not use the arrival distribution. It uses w as its evaluation function. The suggested greedy heuristic, UKD-G, works as follows: for each task (j, t) that arrives or has not yet been assigned, and

[4]https://github.com/yohayt/Allocation_Reusable_Resources_Delayed_Assignments

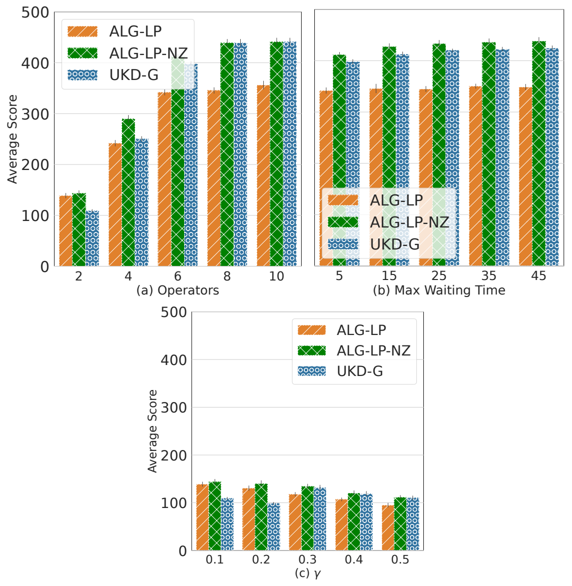

Figure 1: Average sum of scores as a function of the number of operators when the maximum waiting time is 5-time units in (a), and as a function of the maximum waiting time (6 operators) in (b) and as a function of γ (with 2 operators) (c).

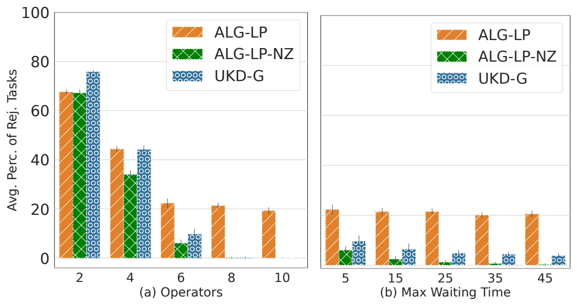

Figure 2: The average percentage of rejected tasks as a function of the number of operators (when the maximum waiting time is 5-time units in (a)) and as a function of the maximum waiting time (with 6 operators) in (b).

for all available agents i, it computes $w_{j,t',i,t}$. Out of all of the options, the match with one of the highest values is selected. We ran a preliminary experiment where we compare the UKD-G with another heuristic that does not use the arrival distribution and is based on the Hungarian matching method H.W. and B. [19], which is an efficient algorithm for solving the offline weighted matching problem in bipartite graphs. In this heuristic, for each time unit, we use the Hungarian algorithm for assigning tasks to the appropriate agents. Our results indicate that the UKD-G yields a higher average sum of scores of the performed tasks and rejects fewer tasks than the Hungarian-based heuristic. Therefore, we focused on the UKD-G in our experiments.

Note that UKD-G does not use the arrival distributions (i.e., $p_{j,t}$), therefore it could be used in both arrival types that we consider,

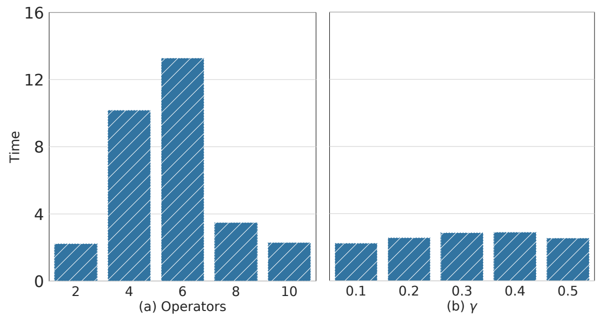

Figure 3: Runtime of the ALG-LP as a function of the number of operators when the maximum waiting time is 5-time units and $T = 360$ rounds in (a), and as a function of γ(2 operators,$T = 360$ rounds) in (b).

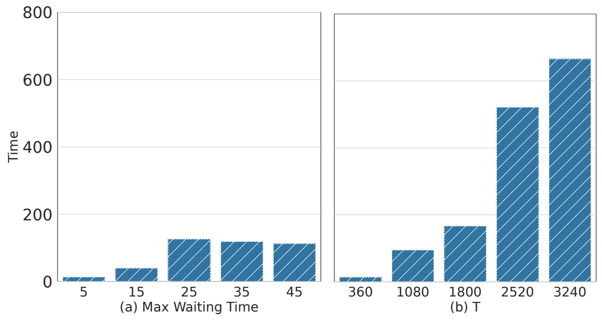

Figure 4: Runtime of the ALG-LP as a function of the maximum waiting time (6 operators, $T = 360$ rounds) in (a) and as a function of T (6 operators, maximum waiting time of 5 units) in (b).

known and unknown distributions. However, It does use the usage duration $C_\lambda \in [T]$.

6.2 Results

We used $\epsilon = 0.1$ in all of the experiments. In the experiments reported in Figure 1, in graphs (a-b) we used $\gamma = 0.1$ as recommended by our industrial consultants. In graph (a) the maximum waiting time was set to 5 and we varied the number of operators. In graph (b) the number of operators was set to 6 and we varied the maximum waiting time. The average score refers to the average sum of $v(j, c, t, t')$ for all performed tasks in all the runs of a given setting. As can be observed in these graphs, ALG-LP-NZ always yields the highest average scores. When the number of operators is large (i.e., 8 and 10 operators), UKD-G yields the same results as the ALG-LP-NZ. However, when the number of operators is relatively small (2-6), ALG-LP-NZ does significantly better than the UKD-G. Interestingly, when the number of operators is very small ALG-LP yields a higher score than the UKD-G, but the ALG-LP-NZ still yields statistically significantly higher scores than the LP. However, when the number of operators increases, the ALG-LP yields much lower scores than both the UKD-G and the ALG-LP-NZ. These results can be explained by observing the number of rejected tasks presented in Figure 2. Graphs (a-b) of this figure correspond to the same settings as those of (a-b) of Figure 1. It can be observed that the number of rejected tasks by ALG-LP-NZ is always the lowest, and when there are enough operators (8-10), neither ALG-LP-NZ nor UKD-G rejects any tasks. However, ALG-LP rejects many tasks even with 10 operators. Finally, we tested the effect of the change in the balance between the time execution and the quality of tasks in the score function, i.e., γ. The number of operators was set to 2 and the waiting time to 5, studying situations where there are not enough operators. As can be observed from graph (c) in Figure 1, for all values of γ, ALG-LP-NZ does significantly better than the UKD-G. However, for $\gamma \geq 0.4$ the differences between the results yielded by UKD-G and those by ALG-LP-NZ are relatively small. In addition, in many cases the gap between ALG-LP and ALG-LP-NZ is quite small. We argue that for a significant distance between the two heuristics, two conditions must be met. The first is that there is a significant difference between the available operators. The second is that there are enough scenarios in which waiting for an operator to become free before making an assignment is beneficial. In some problems, the combination of these conditions does not occur, and therefore, the gap is small.

The runtimes of the application of the solver in ALG-LP were shown in Figures 3 and 4. In graph (a) of figure 3, the time is larger when there are 4-6 operators. The reason is that the problem is simple when there are too few operators, since there are few alternatives. On the other hand, the problem is also easy when there are many operators, since it is easier to find an optimal solution. In (b), we see that γ has no significant effect on the running time. In graph (a) of Figure 4, as the waiting time increases, the running time also increases. That is because as the waiting time increases, the number of valid assignments increases and, therefore, the number of variables X_Λ in the linear program increases. Thus, the running time of solving the linear program increases. However, there is a threshold effect when the waiting time is at least 25. We assume that in this case, the solver knows that it needs to perform optimization and ignores the unnecessary waiting times when necessary. Finally, we can see in (b) that as the number of rounds (T) increases, so does the running time.

6.3 Discussion

In the case where the arrival probabilities $\{p_{j,t}\}$ of the tasks are known, we presented an algorithm with a guaranteed competitive ratio of 0.5 while the known upper bound of an achievable competitive ratio is 0.823. Currently, it is still an open question whether there is a better upper bound, and it is also unknown whether it is possible to develop an algorithm with a better guarantee.

Regardless, the guarantee of the algorithm does not necessarily lead to good practical performance. Indeed, it has been shown to be practical only for a relatively small number of operators.

Consider situations where there are 10 operators that are enough to perform all tasks in our settings as indicated by the performance of the heuristics. In this case, indeed, the bound is respected, and there is no situation where more than 50% of the tasks are rejected. However, even rejecting 20% of the requests (as presented in Figure 2) when there are enough operators to perform all of them is not acceptable. To understand the intuition behind these rejections we present the following example.

Example 1. *Suppose there are two agents, $I = \{1, 2\}$, and one type of task $J = \{1\}$. Both agents can perform both types of tasks and $p_{1,1} = p_{1,2} = 0.25$. Furthermore, suppose that $\Pr[C_{(1,t,i,1)} > 1] = 1$, i.e., the duration of the task, is always greater than one time period. There are many solutions to the associated* LP*, but all of the solvers that we checked will return a solution similar to the following: $x_{1,1,1,1} = 0.25$ and $x_{1,2,1,2} = 0.25$ and the rest of the xs will be set to zero. Suppose tasks of type 1 will arrive both at time 1 and at time 2. Only the first one will be performed, but it could be performed by the second agent. In* ALG-LP *the latter allocation has a probability of zero, but any greedy algorithm will choose an assignment that will lead to the completion of both tasks.*

To handle this practical problem of rejecting too many tasks, we proposed the ALG-LP-NZ heuristic which forces the solvers to increase the number of non-zero variables. In some cases, this leads to a very small decrease in the optimal value of the LP.

In this paper, we discussed two types of arrivals: known arrival distribution and unknown distribution. The UKD-G can be used in both cases and the results presented above demonstrate the performance of teleoperation systems when the distribution is unknown. Determining the arrival distribution, i.e., determining the $p_{j,t}$, usually requires a lot of effort. Our results can help a decision maker decide when it is beneficial to do so. In particular, when the number of agents is large in comparison to the number of tasks, and all tasks could be completed, the UKD-G yields only slightly lower scores than ALG-LP-NZ in almost all cases. However, when the number of agents is small, collecting data to determine the arrival time distribution is beneficial and significantly increases the number of tasks that are performed and the overall score.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we have considered online stochastic matching, which allows for delayed assignments. We first considered a situation where the arrival distribution of dynamic tasks is known in advance. We have shown an algorithm with a guaranteed competitive ratio for the problem and presented the gap between its ratio and the best-known possible ratio. For a setting when arrival distribution is not known in advance we show that there is no algorithm with a reasonable competitive ratio. For the case where the distribution is known, we then showed that there are situations where the algorithm with the competitive ratio rejects too many tasks and therefore suggested a heuristic variation of the algorithm. We then presented a greedy algorithm that does not use the arrival distribution and compared the three algorithms. We ran experiments on teleportation datasets and showed that in most situations the heuristic that is based on the competitive ratio algorithm does significantly better than the greedy heuristic.

In future work, we propose to improve our value and score functions by adding a notion of fairness. The aim is to improve human satisfaction while maintaining the system's performance. Finally, it is also interesting to handle scenarios in which both sides of the bipartite graph are dynamic. In our domains, operators may take some unexpected breaks and then return to the pool.

ACKNOWLEDGEMENTS

The research was supported in part by the Israeli Innovation Authority through the Andromeda consortium and by the Israel Science Foundation under grant 1958/20 and the EU Project TAILOR under grant 952215. The research of Pan Xu was partially supported by NSF CRII Award IIS-1948157.

REFERENCES

[1] Ali Aouad and Ömer Sarıtaç. 2022. Dynamic stochastic matching under limited time. *Operations Research* (2022).

[2] Saif Benjaafar, Zicheng Wang, and Xiaotang Yang. 2022. Human in the Loop Automation: Ride-Hailing With Remote (Tele-) Drivers. *Available at SSRN 4130757* (2022).

[3] Daniel Bogdoll, Stefan Orf, Lars Töttel, and J Marius Zöllner. 2022. Taxonomy and Survey on Remote Human Input Systems for Driving Automation Systems. In *Future of Information and Communication Conference*. Springer, 94–108.

[4] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. 2020. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11621–11631.

[5] Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. 2018. End-to-end driving via conditional imitation learning. In *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 4693–4700.

[6] Danial Dervovic, Parisa Hassanzadeh, Samuel Assefa, and Prashant Reddy. 2021. Non-Parametric Stochastic Sequential Assignment With Random Arrival Times. In *IJCAI '2021*. 4214–4220.

[7] John Dickerson, Karthik Sankararaman, Aravind Srinivasan, and Pan Xu. 2020. Allocation problems in ride-sharing platforms: Online matching with offline reusable resources. *ACM Transactions on Economics and Computation* (2020).

[8] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. 2017. CARLA: An open urban driving simulator. In *Conference on robot learning*. PMLR, Mountain View, United States, 1–16.

[9] Ezekiel J Emanuel, Govind Persad, Ross Upshur, Beatriz Thome, Michael Parker, Aaron Glickman, Cathy Zhang, Connor Boyle, Maxwell Smith, and James P Phillips. 2020. Fair allocation of scarce medical resources in the time of Covid-19. , 2049–2055 pages.

[10] Yiding Feng, Rad Niazadeh, and Amin Saberi. 2022. Near-optimal bayesian online assortment of reusable resources. In *Proceedings of the 23rd ACM Conference on Economics and Computation*. 964–965.

[11] Jean-Michael Georg and Frank Diermeyer. 2019. An adaptable and immersive real time interface for resolving system limitations of automated vehicles with teleoperation. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*. IEEE, 2659–2664.

[12] Xiao-Yue Gong, Vineet Goyal, Garud Iyengar, David Simchi-Levi, Rajan Udwani, , and Shuangyu Wang. 2019. Online assortment optimization with reusable resources. *Minor revision in Management Science* (2019).

[13] Xiao-Yue Gong, Vineet Goyal, Garud N Iyengar, David Simchi-Levi, Rajan Udwani, and Shuangyu Wang. 2021. Online assortment optimization with reusable resources. *Management Science* (2021).

[14] Vineet Goyal, Garud Iyengar, and Rajan Udwani. 2020. Online Allocation of Reusable Resources: Achieving Optimal Competitive Ratio. *CoRR* abs/2002.02430 (2020). arXiv:2002.02430 https://arxiv.org/abs/2002.02430

[15] Gaetano Graf and Heinrich Hussmann. 2020. User Requirements for Remote Teleoperation-based Interfaces. In *12th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*. 85–88.

[16] Robert C Hampshire, Shan Bao, Walter S Lasecki, Andrew Daw, and Jamol Pender. 2020. Beyond safety drivers: Applying air traffic control principles to support the deployment of driverless vehicles. *PLoS one* 15, 5 (2020), e0232837.

[17] C.-J. Ho and J. Vaughan. 2012. Online Task Assignment in Crowdsourcing Markets. *Proceedings of the AAAI Conference on Artificial Intelligence, 26(1)* (2012).

[18] Chien-Ju Ho and Jennifer Wortman Vaughan. 2012. Online task assignment in crowdsourcing markets. In *Twenty-sixth AAAI conference on artificial intelligence*.

[19] Kuhn H.W. and Yaw B. 1955. The Hungarian method for the assignment problem. *Naval Res. Logist.* (1955), 83–97.

[20] KE Jintao, Hai Yang, Jieping Ye, et al. 2020. Learning to delay in ride-sourcing systems: a multi-agent deep reinforcement learning framework. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[21] Richard M. Karp, Umesh V. Vazirani, and Vijay V. Vazirani. 1990. An Optimal Algorithm for On-line Bipartite Matching. *STOC-90* (1990).

[22] Zihao Li, Hao Wang, and Zhenzhen Yan. 2023. Fully Online Matching with Stochastic Arrivals and Departures. In *AAAI 2023(to be published)*.

[23] Philip Almestrand Linné and Jeanette Andersson. 2021. Regulating Road Vehicle Teleoperation: Back to the Near Future. In *2021 IEEE Intelligent Vehicles Symposium Workshops (IV Workshops)*. IEEE, 135–140.

[24] Cheng Long, Raymond Chi-Wing Wong, Yu Peng, and Liangliang Ye. 2013. On Good and Fair Paper-Reviewer Assignment. In *2013 IEEE 13th International Conference on Data Mining*. 1145–1150. https://doi.org/10.1109/ICDM.2013.13

[25] Vahideh Manshadi and Scott Rodilitz. 2022. Online policies for efficient volunteer crowdsourcing. *Management Science* (2022).

[26] Vahideh H Manshadi, Shayan Oveis Gharan, and Amin Saberi. 2012. Online stochastic matching: Online actions based on offline statistics. *Mathematics of Operations Research* 37, 4 (2012), 559–573.

[27] 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.

[28] A. E. Phillips, H. Waterer, M. Ehrgott, and D. M. Ryan. 2015. Integer programming methods for large-scale practical classroom assignment problems. *Computers & Operations Research* 53 (2015), 42–53.

[29] Guoyang Qin, Qi Luo, Yafeng Yin, Jian Sun, and Jieping Ye. 2021. Optimizing matching time intervals for ride-hailing services using reinforcement learning. *Transportation Research Part C: Emerging Technologies* 129 (2021), 103239.

[30] Rhonda Righter. 1987. The stochastic sequential assignment problem with random deadlines. *Probability in the Engineering and Informational Sciences* 1, 2 (1987), 189–202.

[31] Marc Rigter, Danial Dervovic, Parisa Hassanzadeh, Jason Long, Parisa Zehtabi, and Daniele Magazzeni. 2022. Optimal Admission Control for Multiclass Queues with Time-Varying Arrival Rates via State Abstraction. *arXiv preprint arXiv:2203.08019* (2022).

[32] Hanan Rosemarin, Ariel Rosenfeld, Steven Lapp, and Sarit Kraus. 2021. LBA: Online Learning-Based Assignment of Patients to Medical Professionals. *Sensors* 21, 9 (2021), 3021.

[33] Andreas Schrank and Carmen Kettwich. 2021. Roles in the Teleoperation of Highly Automated Vehicles in Public Transport. (2021).

[34] Autonomous Solutions. 2022. Teleoperated Mining Vehicles. https://asirobots.com/teleoperated-mining-vehicles/. Accessed: 2022-08-11.

[35] Felix Tener and Joel Lanir. 2022. Driving from a Distance: Challenges and Guidelines for Autonomous Vehicle Teleoperation Interfaces. In *CHI Conference on Human Factors in Computing Systems*. 1–13.

[36] Y. Tong, J. She, B. Ding, L. Wang, and L. Chen. 2016. Online mobile Micro-Task Allocation in spatial crowdsourcing. *IEEE 32nd International Conference on Data Engineering (ICDE), 2016* (2016), 49–60.

[37] Yansheng Wang, Yongxin Tong, Cheng Long, Pan Xu, Ke Xu, and Weifeng Lv. 2019. Adaptive dynamic bipartite graph matching: A reinforcement learning approach. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 1478–1489.

[38] Yifan Xu and Pan Xu. 2020. Trade the System Efficiency for the Income Equality of Drivers in Rideshare. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*. 4199–4205.

[39] Seunghwan Yoon and Mark E Lewis. 2004. Optimal pricing and admission control in a queueing system with periodically varying parameters. *Queueing Systems* 47, 3 (2004), 177–199.

[40] Tao Zhang. 2020. Toward Automated Vehicle Teleoperation: Vision, Opportunities, and Challenges. *IEEE Internet of Things Journal* 7, 12 (2020), 11347–11354.