Non-Rejection Aware Online Task Assignment in Spatial Crowdsourcing

Jiajun Yao, Lei Yang, Zhenyu Wang, and Xiaohua Xu

***Abstract*—Spatial crowdsourcing as a promising computing paradigm has received significant attention recently. A fundamental issue of spatial crowdsourcing is online task assignment, i.e., the platform must make decisions immediately (assign or reject) for newly arriving objects (tasks or workers). Previous studies mostly focus on the rejection-aware assignment, which rarely considers non-rejection assignment for new arrival objects. To solve this new allocation model, in this paper, we first formulate a novel problem, namely Online Non-rejection aware Task Assignment (ONRTA) in spatial crowdsourcing, where an object cannot be rejected by the platform as long as there is a neighbor that satisfies the matching constraint with it. Then, we develop a non-rejection threshold-based random algorithm ONRTA-RT under the adversarial order model while obtaining a theoretical bound on the competitive ratio. More importantly, we consider a more natural random order model and propose a two-stage-based non-rejection aware task assignment approach, ONRTA-Base, which achieves a competitive ratio of $\frac{1}{4}$. Based on this framework, we further devise two non-rejection assignment approaches, ONRTA-OP and ONRTA-Greedy, which are more effective and run faster with a competitive ratio of $\frac{1}{4}$ and $\frac{1}{8}$, respectively. Finally, experiments on synthetic and real datasets demonstrate that our proposed methods outperform the representative methods.**

***Index Terms*—Non-rejection aware, task assignment, online matching, spatial crowdsourcing.**

I. Introduction

RECENTLY, Spatial Crowdsourcing (SC) is an emerging and promising paradigm of traditional crowdsourcing [1], [2], [3]. In the SC system, requesters post some location-based tasks to the platform; the platform then conducts workers move to the task location before the task's deadline to serve it and get the payment. At present, various spatial crowdsourcing systems have shown excellent potential in many areas, such as car-hailing applications (e.g., DiDi [4] and Grab [5]), on-demand delivery services (e.g., Ele.me [6] and Doordash [7]), and crowd sensing systems (e.g., OpenStreetMap [8] and Waze [9]).

A critical issue of SC platforms is online task allocation [10], [11], [12], [13], [14], [15], [16], [17], which allocates spatial tasks to the appropriate workers in a timely manner. The main difficulty in online assignments is how to decide whether the current arriving task or worker should be assigned or not. Previous work usually focuses on rejection-aware assignments, allowing rejections to respond to newly arriving tasks or workers or making them wait temporarily. For example, [15], [18] attempts to set some thresholds based on the SC platform's historical data. A newly arriving object is assigned if and only if the utility of the matching pair containing the object is not less than the threshold. [11] considers the random arrival model and proposes a near-optimal two-sided online algorithm, which compares the matching pair formed by the new object with the current global optimal allocation to decide whether to assign the object. In addition, some recent works consider the future total rewards for newly arrived objects using machine learning methods [19], [20], i.e., whether assigning this new object is beneficial to the future total rewards. However, all of the above methods are based on rejection-aware assignment methods, without considering the assignment of newly arrived workers or tasks that cannot be rejected by the platform.

In this work, we try to develop a new assignment model, non-rejection allocation, where newly arriving tasks (workers) are not rejected by the SC platform as long as there are workers (tasks) that satisfy matching constraints with it. In fact, non-rejection aware allocation is an important model for many spatial crowdsourcing applications [4], [21]. We illustrate the significance of this model as follows.

- Non-rejection aware assignments can be applied to many scenarios in SC. Existing research mainly focuses on rejection-aware assignments to maximize platform or worker benefits or other set goals, and few attempts have been made for many SC applications that require non-rejection assignment tasks. The non-rejection model is quite common in real-world applications, such as emergency rescue [20], car-hailing services [22], semi-autonomous network [23], cloud systems [24], etc. For example, in a car-hailing service, a user who is far from the city center posts a request to the platform; if there are some idle drivers on the platform at this time, the platform should not reject the user's request even though it may only bring less actual income.
- Non-rejection aware assignments can provide a simple and efficient benchmark algorithm for online task assignments in

Manuscript received 25 March 2023; revised 23 October 2023; accepted 23 October 2023. Date of publication 26 October 2023; date of current version 13 December 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 61972161, in part by Guangdong Basic and Applied Basic Research Foundation under Grant 2022A1515010374, in part by Guangzhou Basic and Applied Basic Research Foundation under Grant 202201010715, in part by Key-Area Research and Development Program of Guangdong Province of China under Grant 2021B0101190002, and in part by the RGC Grant for Theme-based Research Scheme Project under Grant T43-513/23-N. *(Corresponding author: Lei Yang.)*

Jiajun Yao, Lei Yang, and Zhenyu Wang are with the School of Software Engineering, South China University of Technology, Guangzhou 510641, China (e-mail: 201810102795@mail.scut.edu.cn; sely@scut.edu.cn; wangzy@scut.edu.cn).

Xiaohua Xu is with the Department of Computer Science, University of Science and Technology of China, Hefei 230026, China (e-mail: xiaohuaxu@ustc.edu.cn).

Digital Object Identifier 10.1109/TSC.2023.3327858

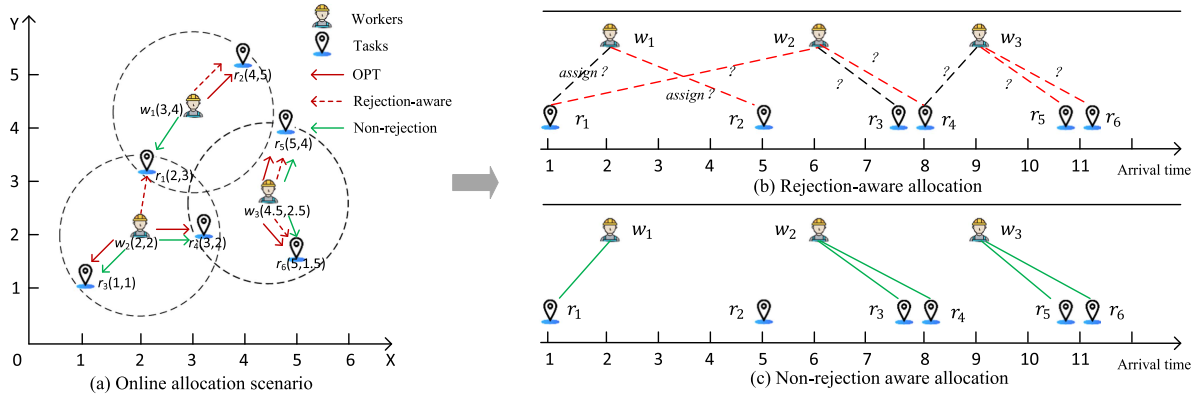


Fig. 1. Motivation example.

spatial crowdsourcing. Non-rejection methods can bring the following advantages: First, the non-rejection model is simple and easy to understand, and can be better used as a general benchmark algorithm for online matching problems. Second, non-rejection allocation does not perform worse than rejection-aware allocation; on the contrary, it outperforms rejection-aware allocation in many cases. This has been shown in the experimental results of the literature [11], e.g., the Greedy algorithm (also a classical non-rejection algorithm) tends to outperform rejection-aware assignments. Third, non-rejection assignments can respond to newly arriving tasks or workers more quickly, which is just suitable for large-scale dynamic scenarios with strict requirements on response time.

Inspired by the above issues, in this paper, we formally define a novel problem, called Online Non-rejection aware Task Assignment (ONRTA) in spatial crowdsourcing. We focus on two classical arrival order models, namely the adversarial order model (AO) [18], [25] and the random order model (RO) [26]. The AO model can provide an optimal theoretical guarantee for the algorithm under the worst scenario, and the RO model is the most natural sequential model in real life, so the algorithm under the RO model can provide better performance than the algorithm under the AO model. To the best of our knowledge, this is the first work that investigates the non-rejection allocation online model for spatial crowdsourcing. Our contributions are listed as follows:

- We identify a novel task assignment in SC, called Online Non-rejection aware Task Assignment (ONRTA) in spatial crowdsourcing problem.
- we develop a non-rejection threshold-based random algorithm ONRTA-RT in the AO model, and obtain a competitive ratio of $\frac{1}{2U_{\max}}$ for the ONRTA problem, where $U_{\max}$ is the estimated maximum utility score.
- We consider a more natural RO model and present three non-rejection aware task allocation approaches based on a two-stage framework, namely ONRTA-Base, ONRTA-OP and ONRTA-Greedy, which outperform ONRTA-RT while having competitive ratios of $\frac{1}{4}$, $\frac{1}{4}$ and $\frac{1}{8}$, respectively.
- As shown by the experimental results, our proposed approaches can achieve higher the total utility and the matching size than the representative methods.

TABLE I
ARRIVAL TIME OF TASKS AND WORKERS

Time	1	2	5	6	8	8	9	11	11
Order 1	r_1	w_1	r_2	w_2	r_3	r_4	w_3	r_5	r_6
Order 2	w_1	r_2	r_3	w_2	r_1	r_4	w_3	r_5	r_6

TABLE II
UTILITY BETWEEN TASKS AND WORKERS

	r_1	r_2	r_3	r_4	r_5	r_6
$w_1(1)$	2	5	6	3	4	1
$w_2(2)$	1	5	4	7	6	2
$w_3(2)$	7	2	4	2	5	3

The rest parts of the paper are organized as follows. Section II presents an example of motivation. Section III introduces the problem definition and formulation. Section IV develops a non-rejection randomized method and achieves a theoretical bound. Section V presents three two-stage-based non-rejection approaches with constant competitive ratios. Section VI presents the performance evaluation. Section VII reviews the related work. Section VIII concludes this paper.

II. MOTIVATION EXAMPLE

Next, we further illustrate the ONRTA problem by a motivation example of online task allocation in Fig. 1.

As shown in Fig. 1, there are three workers $(w_1, \ldots, w_3)$ and six tasks $(r_1, \ldots, r_6)$ in a two-dimensional (2D) space. Workers and tasks are labeled with a location (i.e., 2D coordinates) when they appear on the platform. Table I gives the time and order of arrival of workers and tasks. Table II shows the utility scores for matching between tasks and workers. Each worker is associated with a capability (in brackets) indicating the maximum number he/she can match. Assuming that the duration of the worker and task is set to 6 (e.g., r_1 will expire at time 7), we present the allocation results for three different methods.

First, if the information about workers and tasks can be known in advance, as illustrated in Fig. 1(a), the red solid lines indicate an optimal task allocation $\{(w_1, \{r_2\}), (w_2, \{r_3, r_4\}), (w_3, \{r_5, r_6\})\}$ with a total utility of 24 and the average response time is 1.8 (i.e., the response

time is the length of the interval between the time workers or tasks arrive and the time they are matched).

Second, we present a delay-based rejection-aware online algorithm as shown by the red dotted arrows in Fig. 1(a). For example, when w_1 arrives, we do not assign w_1 to r_1 due to low payoff. When r_2 arrives at time 5, allocate w_2 to r_2. When w_2 arrives, allocate w_2 to r_1 as r_1 is close to expiring. When r_3 and r_4 arrive, assign r_4 to w_2. Similarly, we can obtain an assignment $\{(w_1, \{r_2\}), (w_2, \{r_1, r_4\}), (w_3, \{r_5, r_6\})\}$ with a total utility of 21 and the average response time is 2.6. In fact, this allocation method is difficult and unstable. As shown in Fig. 1(b), it is difficult for us to decide whether w_1 is allocated to r_1 or continues to wait, and whether r_2 is allocated to w_1 or continues to wait. Assuming that r_2 is not assigned to w_1 and chooses to continue waiting, if no subsequent worker can be assigned to r_2, the performance of the algorithm will become worse.

Finally, we give a simple but efficient algorithm, the non-rejection assignment, as shown by the green solid arrows in Fig. 1(a) and (c). When w_1 arrives, we assign w_1 to r_1 despite the low payoff of r_1. When r_3 and r_4 arrive, we assign r_3 and r_4 to w_2 because both r_3 and r_4 satisfy the matching constraint with w_2 and w_2 has capability 2, so a non-rejection allocation $\{(w_1, \{r_1\}), (w_2, \{r_3, r_4\}), (w_3, \{r_5, r_6\})\}$ can be obtained with a total benefit of 21 and the average response time is 1.4.

Besides, different arrival orders will also affect the allocation results. When the arrival order of workers and tasks obey "Order 2" in Table I, the non-rejection method can obtain an optimal allocation, $\{(w_1, \{r_2\}), (w_2, \{r_3, r_4\}), (w_3, \{r_5, r_6\})\}$.

As a result, non-rejection online matching is efficient, simple and more responsive, and can provide an effective benchmark for most online scenarios in SC. In particular, the method also has the same theoretical guarantees as the state-of-the-art rejection-aware online algorithms.

III. Problem Statement

In this section, we first introduce necessary prerequisites and formally define the ONRTA problem. Then, we present a theoretical method to evaluate the performance of online algorithms under different order models.

A. Problem Definitions

Definition 1. (Crowd Worker): A crowd worker, denoted by $w = <w.l, w.b, w.d, w.c, w.r, w.\theta>$, has a current location l_w, comes on the platform at the time s_w, and will stop the service for the requester at the deadline $w.b + w.d$. Moreover, each worker w has a capability $w.c$ indicating that he/she can complete the maximum number of tasks, and has an active area with $w.l$ as the center and $w.r$ as the radius, $\theta_w \in (0, 1]$ is a quality score that completes the task, which can be evaluated by historical data.

Definition 2. (Spatial Task/Request): A spatial task, denoted by $r = <r.l, r.b, r.d, r.\varphi>$ is submitted by a customer at the time $r.b$, and it will expire at the deadline $r.b + r.d$. $r.l$ is the location of r at which the request is required to be performed. $r.\varphi$ is a reward to accomplish the task r.

Definition 3. (Online Matching): Online matching is usually modeled as a bipartite graph $G = (W, R, E, U)$, where W and R represent the crowd workers and spatial tasks, respectively. Each edge $I = (w, r)$ connects a worker w and task r satisfying the matching constraints, and is labeled with a utility score $U(I)$. All workers and tasks come and depart the SC platform in an online manner, which means that information about tasks and workers is not known ahead of time.

Definition 4. (Utility): The utility score of a matching pair (w, r) of worker w and task r is defined as follows:

$$U(w, r) = w.\theta \times r.\varphi$$

Note that, here we use utility function $U(,)$ to better reflect the generalized setting for many practical applications of SC, including the total reward of assigned tasks [15] ($\forall w, w.\theta = 1$), the total quality of the tasks [27] ($\forall r, r.\varphi = 1$), the total matching size [10], [28] ($\forall r, r.\varphi = 1$ and $\forall w, w.\theta = 1$).

Definition 5. (ONRTA problem): Given a set of tasks R, a set of workers W, and their utility score $U(,)$. The workers and tasks are free to appear and leave the SC platform. The ONRTA problem is to calculate a feasible assignment A whose overall utility $MaxSum(A) = \sum_{w \in W, r \in R} U(w, r)$ ($\forall w \in W, r \in R \mid U(w, r) \geq 0$) is maximized, satisfying the following constraints:

- *Capacity Constraint:* The number of tasks assigned by the SC platform cannot be more than the capacity of each worker, i.e., $\sum_{r \in R} I(w, r) \leq w.c$.
- *Deadline Constraint:* A worker w and a task r can form a valid matching (w, r), the worker $w(r)$ must arrive before the deadline of $r(w)$ and leave after the start time of $r(w)$, i.e., $r.b + r.d \leq w.b$ and $r.b \geq w.b + w.d$ ($w.b + w.d \leq r.b$ and $w.b \geq r.b + r.d$).
- *Range constraint:* A worker w can only conduct requests for locations within a circle with $w.l$ as the center and $w.r$ as the radius, i.e., $(w.l - r.l)^2 \leq w.r^2$.
- *Invariable Constraint:* If a pair (w, r) of task r and worker w is conducted, it cannot be changed.

B. Valuation Model

The competitive ratio is a metric used to measure the performance of an online algorithm, which represents the ratio between the output of the online algorithm and its offline optimal result [29]. The evaluation method of the competition ratio is greatly affected by different arrival order models. We focus on two main order models [25], [26]: the adversarial order model (AO) and the random order model (RO). The former is an adversarial input sequence that makes the algorithm perform poorly, while the latter is an input sequence chosen randomly from a distribution.

Definition 6. (Competitive Ratio): Given an arbitrary input order O of tasks, workers and utility score. Let A be an online algorithm and OPT be an optimal offline algorithm for the ONRTA problem. Then, the competitive ratio of A in the AO model is defined as follow:

$$\alpha = \min_{\forall O(W,R,U)} \frac{Maxsum(A)}{Maxsum(OPT)}$$

where $Maxsum(A)$ is the total utility obtained by A on O, and $Maxsum(OPT)$ is the optimal overall utility of the offline scenario. Similarly, the competitive ratio of A in the RO model

Algorithm 1: ONRTA-RT Algorithm.

Input: R, W, $U(.,.)$
Output: An assignment: A
1 $\lambda \leftarrow \lceil ln(U_{\max} + 1) \rceil$;
2 $\theta \leftarrow$ randomly choosing a value from $\{e^0, e^1, \ldots, e^{\lambda-1}\}$;
3 **for** each new object σ arrives **do**
4 $S \leftarrow \{\forall v | (v, \sigma)$ is a match pair with $U(v, \sigma) \geq \theta$ and it satisfies all the constraints$\}$;
5 **if** $Cand \neq \emptyset$ **then**
6 $I_\sigma \leftarrow$ randomly choosing a pair in $Cand$;
7 $A \leftarrow A \cup \{I_\sigma\}$;
8 **else** // e.g., *non-rejection*
9 Greedily pick a pair containing σ and it satisfies all the constraints;
10 $A \leftarrow A \cup \{I_\sigma\}$;
11 remove workers/tasks with expired deadlines;
12 **return** A

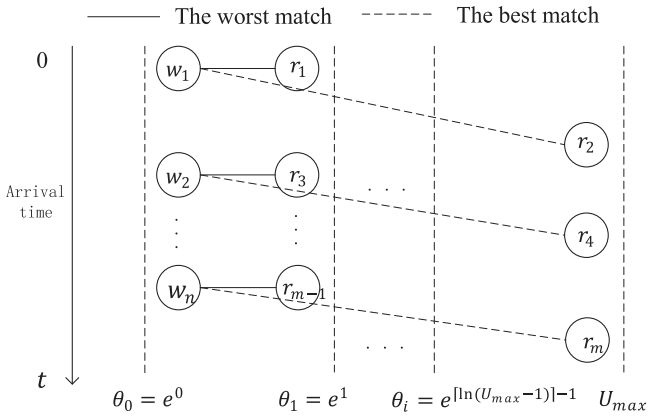


Fig. 2. A worst-case allocation example of ONRTA-RT with n workers and m tasks. The arrival time of workers and tasks increases gradually from top to bottom. For any threshold θ, we give a worst-case matching result $\{(w_1, r_1), (w_2, r_3), \ldots, (w_n, r_{m-1})\}$ by a solid line, and the optimal matching pair $\{(w_1, r_2), (w_2, r_4), \ldots, (w_n, r_m)\}$ is represented by a dotted line.

is

$$\alpha = \min_{\forall O(W,R,U)} \frac{E[Maxsum(A)]}{Maxsum(OPT)}$$

where $E[Maxsum(A)]$ is the expectation of the total utility obtained by A and $Maxsum(OPT)$ is the optimal overall utility.

IV. Randomized Algorithm

In this section, we design a non-rejection random method under the AO model by combining the state-of-the-art random threshold algorithm [18] with a greedy strategy for the ONRTA problem, and then prove that the extended algorithm, ONRTA-RT, has a theoretical lower bound of $\frac{1}{2U_{\max}}$, where $U_{\max}$ is the evaluated maximum utility.

Basic Idea: We first randomly select a threshold according to the maximum utility. When a new object arrives, a pair containing the object is added to assignment A if the worker-task pair whose utility is greater than the threshold and satisfies all constraints. In particular, if there is no pair larger than the threshold, we still choose a pair containing the object with the highest benefit score to A.

Algorithm Details: The procedure of ONRTA-RT is summarized in Algorithm 1. ONRTA-RT first evaluates the maximum utility of all pairs based on historical data, and then randomly selects an item from the threshold set obtained by $U_{\max}$ (lines 1–2). When a worker (task) comes to the platform, if a pair contains the worker (task) whose benefit score is larger than the threshold θ and satisfies all the constraints, we add the pair to assignment A (lines 3–7); otherwise, we pick a pair containing σ with the highest utility to add to A (lines 8–10).

Example 2: Back to our motivation example in Fig. 1. Note that, the arrival sequence of workers and requests obeys the "Order 1" in Table Iin all the following examples. ONRTA-RT first calculates $\lambda \leftarrow \lceil ln(U_{max} + 1) \rceil$ according to $U_{max} = 5$, so the threshold set $\theta = \{e^0, e^1, e^2\}$ is obtained. If $\theta = e^0$, when worker w_1 arrives, a matching pair (w_1, r_1) is obtained because $U(w_1, r_1) = 2 > e^0$. Then, when r_3 arrives, the worker candidate set is $\{w_2\}$, and r_3 is assigned to w_2. Similarly, an allocation $\{(w_1, \{r_1\}), (w_2, \{r_3, r_4\}), (w_3, \{r_5, r_6\})\}$ can be obtained with a total benefit of 21. If the threshold $\theta = e^1$ or $\theta = e^2$, the total utility is 24 and 7, respectively. Thus, for all possible θ, we have that the expected utility of ONRTA-RT is $\frac{21+24+7}{3} \approx 17.3$.

Complexity Analysis: For each newly arrived worker or task, it takes $O(\max\{|R|, |W|\})$ time and space to select a match with a benefit score greater than the threshold. If there is no matching pair that satisfies the condition, it still requires the time and space of $O(\max\{|R|, |W|\})$ to find a pair with the highest benefit. Thus, the time and space complexity of ONRTA-RT are $O(\max\{|R|, |W|\})$.

Competitive Analysis: We analyze the competitive ratio of ONRTA-RT inspired by the theoretical analysis in [18], [30]. The competitive ratio of ONRTA-RT is obtained by two parts. The first part is that the algorithm executes lines 5–7 if there is no pair containing σ with a benefit greater than θ; the second part is that the algorithm executes lines 8–10 if the candidate set is empty. We let A_{RT} and A_{Greedy} denote an assignment output by lines 5–7 and lines 8–10 of ONRTA-RT respectively, and OPT be the offline optimal assignment. Hence, we have the following theorem.

Theorem 1: The competitive ratio of the ONRTA-RT algorithm is $\frac{1}{2U_{\max}}$ under the AO model.

Proof: We consider the worst-case allocation scenario as shown in Fig. 2. There are n workers and m tasks with arrival times. Assume that the matching pair $I(w_1, r_1), I(w_2, r_3), \ldots, I(w_n, r_{m-1})$ produces a utility score in the interval $[e^0, e^1]$, and the matching pair $I(w_1, r_2), I(w_2, r_4), \ldots, I(w_n, r_m)$ in the interval $[e^{\lceil \ln(U_{max}+1) \rceil - 1}, U_{max}]$. When θ is chosen to be e^0, for any newly arrived σ, the ONRTA-RT algorithm always executes lines 5–7. When the selected threshold θ is greater than e^0, although there is no pair containing σ greater than θ (i.e., for any σ, we have $U(\sigma) < e^i$ $(i \geq 1)$), ONRTA-RT still greedily select

pairs containing σ to add to A (lines 8–10). According to Lemma analysis of[12], we know that the number of vertexs matched by A_{RT} and A_{Greedy} is at least $1/2$ of those matched by OPT. Thus, we have $|A_{RT_{\geq e^i}}| \geq \frac{|OPT|}{2}$ and $|A_{Greedy}| \geq \frac{|OPT|}{2}$. Due to θ chooses an item from $e^0, e^1 \ldots, e^{\lambda-1}$ uniformly, for $\forall i \in \{0, 1, \ldots, \lambda-1\}$, we know

$$
\begin{aligned}
a &= \min_{\forall G(W,R,U)} \frac{E\left[U(A)\right]}{U(OPT)} = \frac{\frac{1}{\lambda}\sum_{0\leq i<\lambda} U\left(A_{\geq e^i}\right)}{U(OPT)} \\
&\geq \frac{\frac{1}{\lambda} U\left(A_{RT_{\geq e^0}}\right) + \sum_{1\leq i<\lambda} U\left(A_{Greedy}\right)}{U(OPT)} \\
&\geq \frac{e^0 \cdot |A_{RT\geq e^0}|}{\lambda \left|OPT_{[e^0,e^1]}\right| \cdot U_{\max}} + \frac{\lambda-1}{\lambda} \frac{e^0 \cdot |A_{Greedy}|}{|OPT| \cdot U_{\max}} \\
&= \frac{1}{2\lambda U_{\max}} + \frac{\lambda-1}{2\lambda U_{\max}} = \frac{1}{2U_{\max}}
\end{aligned}
$$

Therefore,

$$
E[U(A)] \geq \frac{1}{2U_{\max}} U(OPT)
$$

□

V. Two-Stage-Based Approach

Although we have shown that the ONRTA-RT algorithm can obtain a theoretical performance guarantee under the AO model, the algorithm is designed for worst-case arrival order [18]. Therefore, it is difficult for us to obtain a good performance through the ONRTA-RT algorithm. In practical scenarios, the arrival of requests and workers is more natural appear on the SC platform in a random manner. Therefore, in this section, we first develop a two-stage-based online non-rejection aware algorithm under the RO model, ONRTA-Base, with a competition ratio of $\frac{1}{4}$. Based on this framework, we further propose the ONRTA-OP and ONRTA-Greedy algorithm with a competitive ratio $\frac{1}{4}$ and $\frac{1}{8}$, respectively.

. *ONRTA-Base Algorithm*

Basic Idea: We first divide the ONRTA-Base algorithm into two stages, which are inspired by the approaches in [11], [31], [32]. In the first stage, for each newly arrived object σ, we find a match containing σ that has maximum utility and satisfies all constraints via a greedy approach. For the second stage, we compute an optimal allocation over all unmatched objects that have arrived and not expired. If the valid pair containing σ is not included in the global optimal assignment, we continue to run the greedy strategy to select a pair that contains σ with the highest benefit.

Algorithm Details: Algorithm 2 shows the pseudo-code of our ONRTA-Base algorithm. Initially, we first computes the total number of tasks and worker capabilities, respectively, which can be evaluated through historical data. Therefore, we can divide the algorithm into two phases by $\lfloor \frac{a+b}{2} \rfloor$. Then, we initialize a worker set W^{Δ}, task set R^{Δ} and assignment A, and use R' and W' to record the number of removed requests and workers (lines 1–2). In the first stage (lines 8–11), for newly arrived object σ, a pair containing σ with the highest utility is added to A. For the second stage (lines 12–18), we run the optimal algorithm (i.e., the Hungarian algorithm) over all unmatched workers and tasks that have arrived and not expired (line 13). If I_σ is included in A_σ, we add I_σ to A (lines 15–16); otherwise, we continue to pick a pair that contains σ with the highest benefit to add to A (lines 17–18). Finally, we remove all workers and tasks that exceed the deadline, and record the number of removed workers and requests in line 19.

Algorithm 2: ONRTA-Base Algorithm.

Input: $R, W, U(.,.)$
Output: An assignment: A
1 $R' = 0, W' = 0, a \leftarrow |R|, b \leftarrow \sum_{w\in W} w.c$;
2 $R^{\Delta} \leftarrow \phi, W^{\Delta} \leftarrow \phi, A \leftarrow \phi$;
3 **for** each new object σ arrives **do**
4 **if** σ is a task **then**
5 $R^{\Delta} \leftarrow R^{\Delta} \cup \{\sigma\}$;
6 **else**
7 $W^{\Delta} \leftarrow W^{\Delta} \cup \{\sigma\}$;
8 **if** $|R^{\Delta}| + R' + |W^{\Delta}| + W' \leq \lfloor \frac{a+b}{2} \rfloor$ **then** *// e.g., the first stage*
9 $I_\sigma \leftarrow \{\forall i \mid i$ is a pair containing σ with the highest utility and satisfies all the constraints$\}$;
10 **if** $I_\sigma \neq \phi$ **then**
11 $A \leftarrow I_\sigma$;
12 **else** *// e.g., the second stage*
13 $A_\sigma \leftarrow$ optimal allocation on $(R^{\Delta} \cup W^{\Delta} \cup \sigma)$;
14 $I_\sigma \leftarrow$ a pair containing σ in A_σ;
15 **if** $A \cup \{I_\sigma\}$ is a matching **then**
16 $A \leftarrow I_\sigma$;
17 **else** *// e.g., non-rejection*
18 Greedily pick a pair containing σ and it satisfies all the constraints;
19 Remove all workers/requests whose deadlines have passed and use R' and W' to record the umber of removed requests and workers;
20 **return** A;

Example 3: Considering Example 1. ONRTA-Base first divides the algorithm into two phases by $\lfloor \frac{5+6}{2} \rfloor$. In the first stage, there are two tasks $\{r_1, r_2\}$ and three workers $\{w_1, w_2, w_2\}$. Note that, we consider a worker as duplicate $w.c$ of that worker arriving at the same time when he/she has capability $w.c$. A greedy task assignment $\{(w_1, r_1)\}$ can be obtained. In the second stage (i.e., $|R^{\Delta}| + R' + |W^{\Delta}| + W' > 5$), when r_3 and r_4 arrive at time 8, $R^{\Delta} = \{r_2, r_3, r_4\}$ and $W^{\Delta} = \{w_1, w_2, w_2\}$, the current global optimal assignment is $\{(w_1, \{r_2\}), (w_2, \{r_3, r_4\})\}$. Because the pairs (w_2, r_3) and (w_2, r_4) are included in the optimal allocation and w_2 is unmatched, we assign r_3 and r_4 to w_2. When r_5 and r_6 arrive, $R^{\Delta} = \{r_2, r_3, r_4, r_5, r_6\}$ and $W^{\Delta} = \{w_2, w_2, w_3, w_3\}$, the global optimal assignment is $\{(w_2, \{r_3, r_4\}), (w_3, \{r_5, r_6\})\}$, the pair (w_3, r_5) and (w_3, r_6) can be obtained. Thus, the total utility is 21, which is better than that of ONRTA-RT.

Complexity Analysis: When a new object arrives in the first stage (lines 8–11), it takes $O(\max\{|R|,|W|\})$ both time and space to obtain the highest utility pair containing the object. In the second stage, it takes $O(\max\{|R^{\Delta}|^3,|W^{\Delta}|^3\})$ time and $O(\max\{|R^{\Delta}|^2,|W^{\Delta}|^2\})$ space to run the Hungarian algorithm over unmatched workers and requests that have arrived and not expired. If $A\cup\{I_\sigma\}$ is not a matching (line 15), ONRTA-Base requires the same time and space as the first stage to run the greedy algorithm. Thus, ONRTA-Base has a time and space complexity of $O(\max\{|R^{\Delta}|^3,|W^{\Delta}|^3\})$ and $O(\max\{|R^{\Delta}|^2,|W^{\Delta}|^2\})$, respectively.

We next study the competitive ratio of ONRTA-Base under the RO model. Let I_σ be a match pair containing σ and U_σ be the benefit score of I_σ. Let $E[U_\sigma]$ and $E[MaxSum(A)]$ be the expected utility of I_σ and the expected overall utility, respectively. Let $OPT_{(X,Y)}=\{(x,y)\in OPT \mid x\in X, y\in Y\}$ be the offline optimal assignment containing X tasks and Y workers. For simplicity, We still use R' and W' to represent a set of removed tasks and workers, respectively.

Lemma 1: For $|R^{\Delta}|+|R'|+|W^{\Delta}|+|W'|>\left\lfloor\frac{a+b}{2}\right\rfloor$ and $\sigma\in R\bigcup W$:

$$\begin{cases} E[U_\sigma]\geq \frac{|W^{\Delta}+W'|}{ab}OPT_{(a,b)} & \sigma\in R\\ E[U_\sigma]\geq \frac{|R^{\Delta}+R'|}{ab}OPT_{(a,b)} & \sigma\in W\end{cases}$$

Proof: If a new object $\sigma\in R$, we have $E[U_\sigma]=\frac{1}{|R^{\Delta}+R'|}E[MaxSum(A_{(|R^{\Delta}+R'|,|W^{\Delta}+W'|)})]$, which the object can be considered as a uniformly random selection from the task set $R^{\Delta}\cup R'$. Similarly, the set $R^{\Delta}\cup R'$ can also be considered as a uniformly random selection from R with a size of $|R^{\Delta}+R'|$. Then, we know

$$\begin{aligned} &E[\text{MaxSum}\ \left(A_{(|R^{\Delta}+R'|,|W^{\Delta}+W'|)}\ z\right)]\\ &\quad\geq \frac{|R^{\Delta}+R'|}{a}OPT_{(a,|W^{\Delta}+W'|)}\\ &\quad\geq \frac{|R^{\Delta}+R'|}{a}\frac{|W^{\Delta}+W'|}{b}OPT_{(a,b)}\end{aligned}$$

Together, we have,

$$\begin{aligned} E[U_\sigma]&\geq \frac{1}{|R^{\Delta}+R'|}\frac{|R^{\Delta}+R'|}{a}\frac{|W^{\Delta}+W'|}{b}OPT_{(a,b)}\\ &\geq \frac{|W^{\Delta}+W'|}{ab}OPT_{(a,b)}\end{aligned}$$

Similarly, we can also obtain the expected utility of I_σ if $\sigma\in W$,

$$E[U_\sigma]\geq \frac{|R^{\Delta}+R'|}{ab}OPT_{(a,b)}$$

Lemma 1 gives the expected utility obtained when an object σ arrives. We next analyze the probability of adding the pair containing σ to A.

Lemma 2: For $|R^{\Delta}|+|R'|+|W^{\Delta}|+|W'|>\left\lfloor\frac{a+b}{2}\right\rfloor$ and $\sigma\in R\bigcup W$:

$$\begin{cases}\Pr\{A\cup\{I_\sigma\}\text{ is a matching}\}\geq\frac{\left\lfloor\frac{a+b}{4}\right\rfloor}{|R^{\Delta}+R'|-1} & \sigma\in R\\ \Pr\{A\cup\{I_\sigma\}\text{ is a matching}\}\geq\frac{\left\lfloor\frac{a+b}{4}\right\rfloor}{|W^{\Delta}+W'|-1} & \sigma\in W\end{cases}$$

Proof: When a new object σ arrives, σ can only be added to allocation A if its neighbor σ' has not already been matched before σ arrived. If $\sigma\in R$, we construct $R=\{r_{\left\lfloor\frac{a+b}{4}\right\rfloor+1+|R'|},\ldots,r_{|R^{\Delta}+R'|-1}\}$, which is a set of available requests that arrive after the second stage but before σ. For i-th request arrives, the probability that its neighbor σ' can be assigned to A is at most $\frac{1}{i}$, which can be considered as uniformly selected from the i request. Thus, σ' is unmatched with a probability of at least $\frac{i-1}{i}$. Then,

$$\begin{aligned}&\mathbb{P}\left\{\sigma'\text{ unmatched in }\left[\left\lfloor\frac{a+b}{4}\right\rfloor+1+|R'|,\left|R^{\Delta}+R'\right|-1\right]\right\}\\ &=\mathbb{P}\left[\bigwedge_{i=\left\lfloor\frac{a+b}{4}\right\rfloor+1+|R'|}^{|R^{\Delta}+R'|-1}\sigma'\notin I_{\sigma'}\right]\geq\prod_{i=\left\lfloor\frac{a+b}{4}\right\rfloor+1+|R'|}^{|R^{\Delta}+R'|-1}\frac{i-1}{i}\\ &=\frac{\left\lfloor\frac{a+b}{4}\right\rfloor+|R'|}{|R^{\Delta}+R'|-1}\geq\frac{\left\lfloor\frac{a+b}{4}\right\rfloor}{|R^{\Delta}+R'|-1}\end{aligned}$$

Lemma 3: For $|R^{\Delta}|+|R'|+|W^{\Delta}|+|W'|>\left\lfloor\frac{a+b}{2}\right\rfloor$ and $\sigma\in R\bigcup W$:

$$\begin{cases}E[U_\sigma\in A]\geq\frac{\left\lfloor\frac{a+b}{4}\right\rfloor\left|R^{\Delta}+R'\right|}{ab(|W^{\Delta}+W'|-1)}OPT_{(a,b)} & \sigma\in R\\ E[U_\sigma\in A]\geq\frac{\left\lfloor\frac{a+b}{4}\right\rfloor\left|W^{\Delta}+W'\right|}{ab(|R^{\Delta}+R'|-1)}OPT_{(a,b)} & \sigma\in W\end{cases}$$

Proof: According to the Lemmas 1 and 2, we can get the expected utility $U_\sigma\in A$, if $\sigma\in R$,

$$E[U_\sigma\in A]\geq\frac{\left\lfloor\frac{a+b}{4}\right\rfloor\left|R^{\Delta}+R'\right|}{ab\left(|W^{\Delta}+W'|-1\right)}OPT_{(a,b)}$$

Then, if $\sigma\in W$,

$$E[U_\sigma\in A]\geq\frac{\left\lfloor\frac{a+b}{4}\right\rfloor\left|W^{\Delta}+W'\right|}{ab\left(|R^{\Delta}+R'|-1\right)}OPT_{(a,b)}$$

Theorem 2: The competitive ratio of the ONRTA algorithm is $\frac{1}{4}$ under the RO model.

Proof: The total expected utility $E[\text{MaxSum}(A)]$ is calculated by summing $E[U_\sigma]$. According to Lemma 3, we get, for $\sigma\in R\bigcup W$,

$$\begin{aligned}E[\text{MaxSum}(A)]&=\frac{1}{2}\left(E[U_\sigma(\sigma\in R)]+E[U_\sigma(\sigma\in W)]\right)\\ &\geq\frac{1}{4}\sum_{\sigma=\left\lfloor\frac{a+b}{2}\right\rfloor+1}^{a+b}\left(E[U_\sigma(\sigma\in R)]+E[U_\sigma(\sigma\in W)]\right)\end{aligned}$$

Algorithm 3: ONRTA-OP Algorithm.

Input: $R, W, U(.,.)$
Output: An assignment: A
1 **for** each new object σ arrives **do**
2 **if** $|R^{\Delta}| + R' + |W^{\Delta}| + W' > \lfloor \frac{a+b}{2} \rfloor$ **then**
  // e.g., the second stage
3 $A_\sigma \leftarrow$ optimal matching on $(R^{\Delta} \cup W^{\Delta} \cup \sigma)$;
4 $I_\sigma \leftarrow$ a pair containing σ in A_σ;
5 **if** $A \cup \{I_\sigma\}$ is a matching **then**
6 $A \leftarrow I_\sigma$;
7 $R^{\Delta} \leftarrow R^{\Delta}/(r \in I_\sigma), R' + +$;
8 $W^{\Delta} \leftarrow W^{\Delta}/(w \in I_\sigma), W' + +$;
9 **return** A;

$$\geq \frac{1}{4}\left(\sum_{\sigma=\lfloor \frac{a+b}{2} \rfloor+1}^{a+b} \frac{\lfloor \frac{a+b}{4} \rfloor \left|R^{\Delta}+R'\right|}{ab\left(|W^{\Delta}+W'|-1\right)} OPT_{(a,b)} + \frac{\lfloor \frac{a+b}{4} \rfloor \left|W^{\Delta}+W'\right|}{ab\left(|R^{\Delta}+R'|-1\right)} OPT_{(a,b)}\right)$$

$$\geq \frac{1}{4}\frac{\lfloor \frac{a+b}{4} \rfloor}{ab} \sum_{\sigma=\lfloor \frac{a+b}{2} \rfloor+1}^{a+b} \left(\frac{\left|R^{\Delta}+R'\right|}{|W^{\Delta}+W'|-1} + \frac{\left|W^{\Delta}+W'\right|}{|R^{\Delta}+R'|-1}\right)$$

$$\geq \frac{1}{2}\frac{\lfloor \frac{a+b}{4} \rfloor}{ab}\left(a+b-\left\lfloor \frac{a+b}{2} \right\rfloor+1\right) OPT \geq \frac{1}{4} OPT.$$

B. ONRTA-OP Algorithm

During the allocation process, we found that the second stage of the ONRTA-Base algorithm will no longer run greedy allocation (i.e., lines 17–18) when the already matched workers and tasks are consistently removed. Moreover, this approach can obtain better performance than the original algorithm (i.e., ONRTA-Base) while having the same competitive ratio. Based on this, we further propose the following algorithm.

Basic Idea: We propose the ONRTA-OP algorithm using the framework of ONRTA-Base by continuously removing allocated workers and tasks in the second phase.

Algorithm Details: Algorithm 3 shows the procedure of our ONRTA-OP method. All objects that have completed matching are removed and the number of workers and requests removed is recorded (lines 7–8).

Example 4: Considering Example 1. The allocation result of ONRTA-OP in the first phase is the same as that ONRTA-Base. For the second stage, when r_3 and r_4 arrive, $R^{\Delta} = \{r_2, r_3, r_4\}$ and $W^{\Delta} = \{w_2, w_2\}$, the matching pair $\{(w_2, \{r_3, r_4\})\}$ can be obtained. When r_5 and r_6 arrive, $R^{\Delta} = \{r_2, r_5, r_6\}$ and $W^{\Delta} = \{w_3, w_3\}$. Thus, we can obtain a final assignment $\{(w_1, \{r_1\}), (w_2, \{r_3, r_4\}), (w_3, \{r_5, r_6\})\}$ with a total utility of 21, which is still more effective than ONRTA-RT.

Complexity Analysis: For the first phase, we have the same time and space complexity as ONRTA-Base. For the second phase, the ONRTA-OP algorithm has a smaller set of available tasks and workers by removing matched workers and tasks. Thus, we let $\mathcal{R}$ and $\mathcal{W}$ be the maximum number of workers and tasks after removing the matched workers and tasks from R^{Δ} and W^{Δ}, respectively. Thus, the time and space complexity of ONRTA-OP is $O(\max\{|\mathcal{R}|^3, |\mathcal{W}|^3\})$ and $O(\max\{|\mathcal{R}|^2, |\mathcal{W}|^2\})$ $(\mathcal{R} < R^{\Delta}, \mathcal{W} < W^{\Delta})$.

Next, we analyze that the ONRTA-OP algorithm has the same competitive ratio as ONRTA-Base. First, tasks and workers that have been matched are removed continuously not affect the calculation of the expected utility $E[U_\sigma]$, so Lemma 1 still holds. We then try to demonstrate that Lemma 2 also holds. Let $\bar{R}$ and $\bar{W}$ denote a set of tasks and workers that have been matched, respectively.

Theorem 3: The competitive ratio of the ONRTA-OP algorithm is $\frac{1}{4}$ under the RO model.

Proof: If $\sigma \subset R$, we construct task set $R = \{r_{\lfloor \frac{a+b}{4} \rfloor+1+|R'|}, \ldots, r_{|R^{\Delta}|+R'|+|\bar{R}|-1}\}$, which is a set of available tasks that arrive after the second stage but before current object σ (i.e., $|\sigma| = |R^{\Delta} + R'| + |\bar{R}| - 1$). As we continuously remove matched tasks, this reconstructed task set can be denoted as $R' = \{r_{\lfloor \frac{a+b}{4} \rfloor+1+|R'|+|\bar{R}|}, \ldots, r_{|R^{\Delta}|+R'|+|\bar{R}|-1}\}$. According to the analysis in Lemma 2. Then,

$$\mathbb{P}\left\{\sigma' \text{ unmatched in } \left[\left\lfloor \frac{a+b}{4} \right\rfloor + 1 + |R'| + |\bar{R}|, |\sigma| - 1\right]\right\}$$
$$= \mathbb{P}\left[\bigwedge_{i=\lfloor \frac{a+b}{4} \rfloor+1+|R'|+|\bar{R}|}^{|R^{\Delta}|+R'|+|\bar{R}|-1} \sigma' \notin I_{\sigma'}\right]$$
$$\geq \prod_{i=\lfloor \frac{a+b}{4} \rfloor+1+|R'|+|\bar{R}|}^{|R^{\Delta}|+R'|+|\bar{R}|-1} \frac{i-1}{i}$$
$$= \frac{\left(\lfloor \frac{a+b}{4} \rfloor + |R'| + |\bar{R}|\right)}{(|R^{\Delta} + R'| + |\bar{R}| - 1)} \geq \frac{\lfloor \frac{a+b}{4} \rfloor}{(|R^{\Delta} + R'| - 1)}$$

Similarly, if $\sigma \subset W$, we also have the same conclusion. Therefore, Lemma 2 also holds. According to Lemmas 1, 2, and 3, we get

$$E[\text{MaxSum}(A)] \geq \frac{1}{4} OPT$$

□

C. ONRTA-Greedy Algorithm

In this subsection, we develop an ONRTA-Greedy algorithm with a $\frac{1}{8}$-competitive ratio, which runs faster but is slightly less effective than ONRTA-base and ONRTA-OP.

Basic Idea: ONRTA-Greedy is developed based on the framework of ONRTA-OP. The main difference is that ONRTA-Greedy adopts the relatively global greedy strategy in line 3 of Algorithm 4 instead of the optimal algorithm.

Algorithm Details: The procedure of ONRTA-Greedy is shown in Algorithm 4. We ignore the present of the first stage of ONRTA-Greedy, which is the same as ONRTA-OP. When $|R^{\Delta}| + R' + |W^{\Delta}| + W' > \lfloor \frac{a+b}{2} \rfloor$, we calculate a global greedy matching on $(R^{\Delta} \cup W^{\Delta} \cup \sigma)$ for each newly arrived object σ (line 3).

Algorithm 4: ONRTA-Greedy Algorithm.

Input: $R, W, U(.,.)$
Output: An assignment: A
1 **for** each new object σ arrives **do**
2 **if** $|R^{\Delta}| + R' + |W^{\Delta}| + W' > \lfloor \frac{a+b}{2} \rfloor$ **then**
// e.g., the second stage
3 $A_{\sigma} \leftarrow$ greedy matching on $(R^{\Delta} \cup W^{\Delta} \cup \sigma)$;
4 **return** A

Example 5: Considering Example 1. We ignore the analysis of the first stage which is the same as the above example 4. For the second stage, when r_3 and r_4 arrive, the current global greedy assignment is $\{(w_1, \{r_2\}), (w_2, \{r_3, r_4\})\}$. Because tasks r_3 and r_4 are included in the allocation, we assign r_3 and r_4 to w_2. Thus, the final assignment $\{(w_1, \{r_1\}), (w_2, \{r_3, r_4\}), (w_3, \{r_5, r_6\})\}$ can be obtained and the total utility is also 21.

Complexity Analysis: In line 3 of Algorithm 4, the global greedy strategy is implemented using a heap over available workers and requests. Thus, ONRTA-Greedy has a time and space complexity of $O(|\mathcal{R}||\mathcal{W}| \log(|\mathcal{R}||\mathcal{W}|))$ and $O(\max\{|\mathcal{R}|^2, |\mathcal{W}|^2\})$, respectively.

Since ONRTA-Greedy replaces the global optimal strategy with a global greedy algorithm, the expectation of the benefit generated by each new object will change. However, the probability of adding each newly object to allocation A remains the same and obeys Lemma 2. Thus, we next only analyze the expected utility obtained by the pair I_{σ}.

Theorem 4: The competitive ratio of the ONRTA-Greedy algorithm is $\frac{1}{8}$ under the RO model.

Proof: When a new σ arrives, ONRTA-Greedy will calculate a global greedy solution A_{σ} in line 3 of the algorithm. Since the offline greedy algorithm provides at least an approximation ratio of 2 [14], [33], we have $E[MaxSum(A)] \geq \frac{1}{2} OPT$. Combined Lemmas 4 and 6, if $\sigma \in R$, the expected utility $E[U_{\sigma}]$ is

$$E\left[U_{\sigma} \in A\right] \geq \frac{\lfloor \frac{a+b}{4} \rfloor}{(|R^{\Delta} + R'| - 1)} \frac{|W^{\Delta} + W'|}{2ab} OPT_{(a,b)}$$

Similarly, if $\sigma \in W$

$$E\left[U_{\sigma} \in A\right] \geq \frac{\lfloor \frac{a+b}{4} \rfloor}{(|R^{\Delta} + R'| - 1)} \frac{|W^{\Delta} + W'|}{2ab} OPT_{(a,b)}$$

Thus, we have

$$E[MaxSum(A)] = \frac{1}{2} E\left[\sum_{\sigma=1}^{a+b} U_{\sigma}\right] \geq \frac{1}{8} OPT$$

VI. Experimental Evaluation

In this section, we evaluate the performance of the proposed approaches based on two real-world datasets and a synthetic dataset.

TABLE III
Real Dataset

Platform	$\|R\|$	$\|W\|$	$w.d, r.d$	$w.\theta$	w.r	w.c
EverySender	4036	400,500,600,700,800	600	0.6	10	1
DiDi Chuxing	5000	1K,2K,3K,4K,5K	300	0.5	300	1

TABLE IV
Synthetic Dataset

Factor	Setting
$\|R\|$	1000,2000,**3000**,4000,5000
$\|W\|$	100,200,**500**,1000,2000
$w.c$	**1**,3,5,7,9
$w.r$	2,4,**6**,8,10
$w.\theta$	0.1,0.3,**0.5**,0.7,0.9
$w.e, r.e$	2,5,**10**,15,20
μ, λ and U	2,5,**10**,15,20
scalability ($\|$W$\|$ × $\|$T$\|$)	2K×10K,4K×20K,6K×30K,8K×40K,10K×50K

A. Experimental Setup

Datasets: We evaluate the performance of our proposed approaches on two extensively used real datasets: EverySender (a dataset on the general Spatial Crowdsourcing Platform) [11], [12] and DiDi (an open dataset on the DiDi Chuxing application) [34]. In the EverySender dataset, each worker and task are associated with some basic information such as location, start time, due time, the reachable distance of the worker, payment for completing the task, etc. Similarly, the DiDi dataset has a similar information description as the EverySender dataset. For example, the pick-up time and location of ride-hailing orders are used as the start time and location of tasks, and the drop-off time and location of orders are taken as the arrival time and location of workers. Since the DiDi dataset has no associated reward for tasks, we will uniformly generate the reward $r.\varphi$ from the range of (0, 10]. In particular, we select different numbers of workers $|W| = 400, 500, 600, 700, 800$ and $|W| = 1K, 2K, 3K, 4K, 5K$ from the EverySender and DiDi datasets as parameters for varying the number of workers, respectively. Table III shows the statistics for both datasets.

For the synthetic dataset (Syn), we use the same generation method as the synthetic dataset setup in the previous work [11] for consistency and fair comparison. We generate the positions of workers and tasks following a random manner in a two-dimensional space $[0, 100]^2$. In particular, we change the number of workers and tasks, the ability of workers, the quality of completed tasks, and the end time of tasks and workers, etc. The specific attribute settings for our experiments are presented in Table IV, where default settings are shown in bold.

Comparison Approaches: Our proposed algorithm is compared with the following algorithms.

Basic-RT: This is a state-of-the-art threshold-based randomized method that achieves a near-optimal theoretical bound under the AO model [13], [15], [35].

Greedy: This is a greedy-based task allocation approach, which greedily chooses a worker-task pair with the maximum benefit for each new object that arrives at the platform [12], [14], [16].

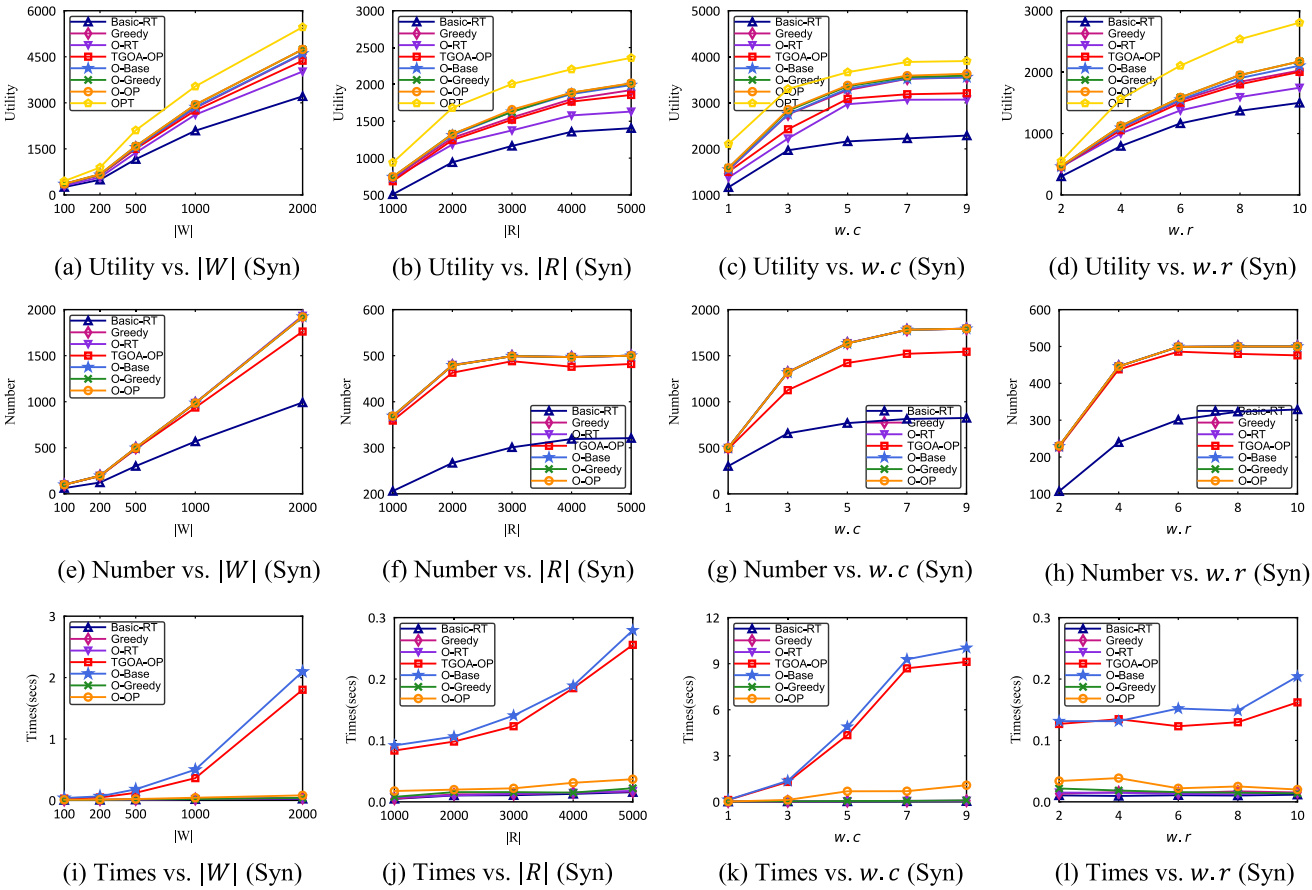


Fig. 3. Results on varying workers $|W|$ and tasks $|T|$, capacity $w.c$ and radius $w.r$.

ONRTA-RT(O-RT): This approach is a non-rejection aware random online algorithm, which is extended from the Basic-RT and is shown in Section IV.

TGOA-OP: This is a state-of-the-art two-sided online approach with a $\frac{1}{4}$-competitive ratio in the RO model, which utilizes a local optimal solution to approximate the global optimal solution in each round for each newly arrived object [11].

ONRTA-Base (O-Base): Our non-rejection online matching method based on a two-stage framework is presented in Section V-A.

ONRTA-OP(O-OP): Our improved ONRTA-Base algorithm by continually removing matched workers and tasks and is presented in Section V-B.

ONRTA-Greedy(O-Greedy): Our improved ONRTA-OP method enhances the efficiency and scalability in large-scale dataset scenarios by replacing the global optimal algorithm with a global greedy algorithm.

OPT: This is an optimal algorithm obtained by running the Hungarian Algorithm (ie, the optimal approach for solving the maximum bipartite matching problem) over all tasks and workers in offline scenarios [10], [11].

Metrics and Implementation: In the experiments, we utilize the following three metrics to investigate the performance of our proposed approaches. 1) Overall utility (utility for short): The overall utility represents the total benefit from assigning worker-task pairs, which is the objective of the ONRTA problem [13]. 2) Matching Size (number for short): The matching size represents the total number of matching pairs formed by workers and tasks [28], [36]. 3) Running time (time for short): The running time represents the time it takes to execute an algorithm to solve the ONRTA problem [12], [35].

For each experiment, we repeatedly test the algorithm 100 times and take the average as a result. Our experiments are performed on a GUN C++ machine equipped with an Intel i7-8750H 2.20 GHz and 16 GB RAM.

B. Experiment Results

Effect of $|W|$: We first study the effect of the number of workers $|W|$ by varying it from 100 to 2000. From Fig. 3(a), we can see that the total utility for all approaches naturally increases as $|W|$ gets larger. This is because there are more requests that can be completed by the newly added workers. Since OPT is an offline optimal algorithm, we ignore its comparison with other algorithms. Our proposed O-Base, O-Greedy and O-OP approach obtains more utilities than the baseline Basic-RT and TGOA-OP. In terms of matching size, as shown in Fig. 3(e), the TGOA-OP and Basic-RT algorithms are also less efficient compared to other algorithms. This is because, for each newly arrived task, the other algorithms always respond

to this request/task as long as there are workers that satisfy the matching constraints with that task. Fig. 3(i) shows that our proposed O-OP and O-Greedy have a higher running time than the baseline Basic-RT and Greedy, but have a shorter running time than the state-of-the-art online algorithm TGOA-OP under the RO model. The former is because, O-OP and O-Greedy always repeatedly compute a relative global allocation over all available workers and requests, while Basic-RT just searches for a neighbor that satisfies the policy for each newly arriving worker or task in each iteration. The latter is because, O-OP and O-Greedy continuously remove matched workers and tasks, resulting in a smaller maximum cardinality in each iteration round, so the time complexity is smaller than TGOA-OP.

Effect of $|R|$: Next, we study the effect of the number of tasks $|R|$ by varying it from 1000 to 5000. As can be seen in Fig. 3(b), we can observe that when $|R|$ grows, the total utility scores of all methods show a similar growth trend. This is because each worker has the opportunity to perform more tasks. Besides, O-OP obtains the highest total benefit, followed by O-Greedy, O-Base, Greedy, TGOA-OP, O-RT and Basic-RT. In Fig. 3(f), our proposed O-OP and O-Greedy still complete more tasks than the baseline Basic-RT and TGOA-OP. The reason is the above-mentioned as explained in the effect of $|W|$. In terms of CPU time, as shown in Fig. 3(j), although our proposed O-OP algorithm has a higher running time than the baseline algorithm, the difference in execution time is less than 15 ms on average.

Effect of $w.c$: The third column in Fig. 3 presents the effect of varying $w.c$. Fig. 3(c) and (g) show that the total utility and matching size of all the algorithms first increase and then stabilize. This is because when $w.c$ increases from 1 to 5, each worker gets more capacity, i.e., the worker can conduct more requests during his/her service time window. Since $|R| = 3000$ and $|W| = 500$, we can know that each worker can perform six tasks on average, so the total utility and number of matches only slightly increase when $w.c$ increases from 7 to 9. Our proposed O-OP is the most effective, and the Basic-RT algorithm is the least effective. In terms of running time, O-Greedy has a lower running time than O-OP as illustrated in Fig. 3(k). The reason is that O-Greedy adopts a greedy strategy to replace the Hungarian algorithm with a cubic time complexity in the second phase of the method.

Effect of $w.r$: The fourth column in Fig. 3 shows the effect of varying $w.r$. Fig. 3(d) shows that the overall utility of the seven methods increases gradually as $w.r$ grows. This is due to the fact that with a larger $w.r$, each worker can perform longer distance requests, which means she/he can have more requests with high pay to choose from. From Fig. 3(d), (h) and (l), we also know that TGOA-OP achieves a better overall utility than the threshold-based random algorithm O-RT, but is ineffective in terms of both the number of completed requests and running time. The reason is that, Even if there are no candidate pairs larger than the set threshold in the newly arrived request, the O-RT algorithm will still greedily select a pair from the candidate pairs for allocation. In terms of CPU time, the running time of the seven methods varies only slightly due to the constant number of workers and tasks.

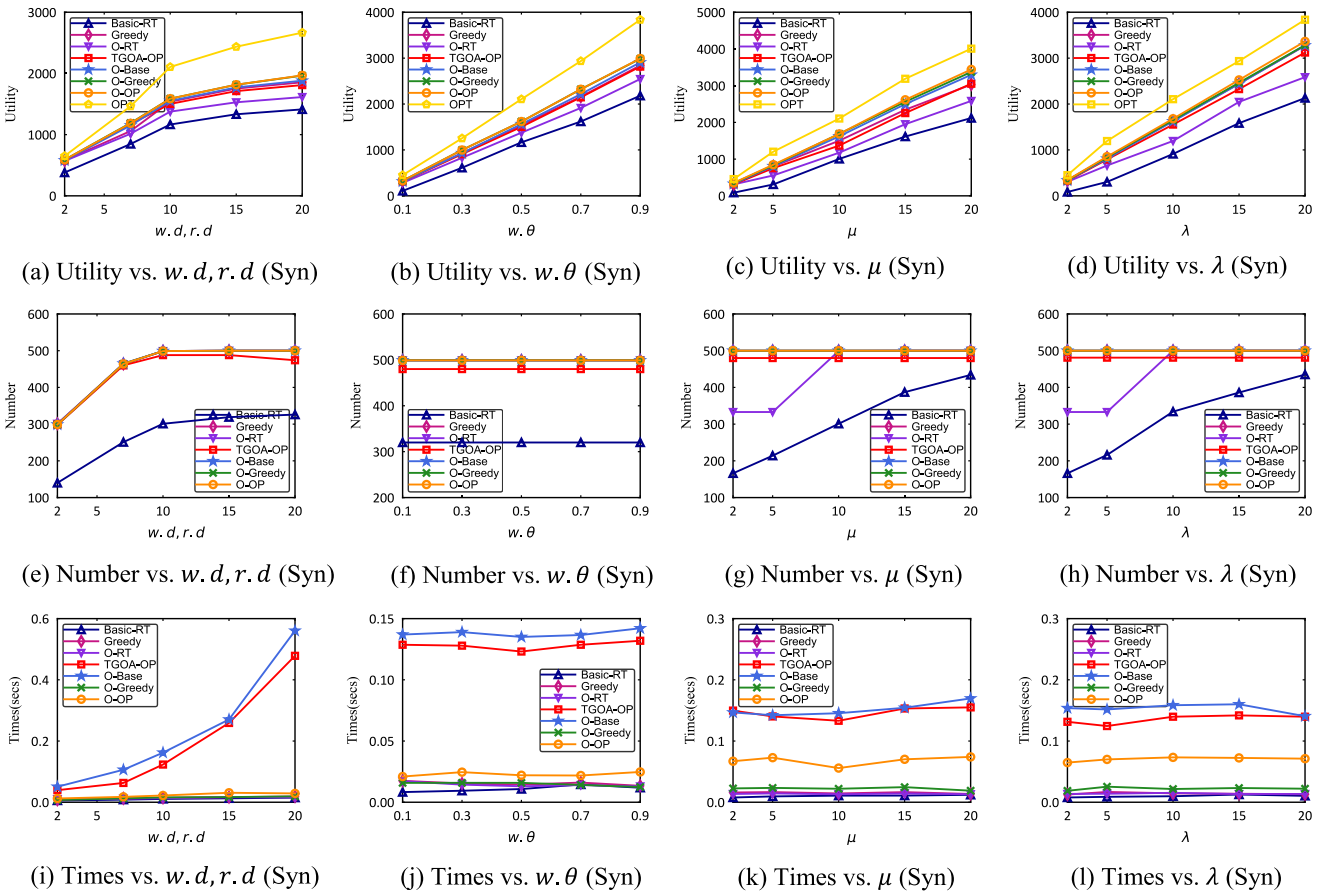
Effect of $w.d, t.d$: Fig. 4(a) and (e) show the variation of overall utility and number of matches with $w.d, t.d$. When the duration of workers and tasks increase from 2 to 10 and other attributes are fixed. We can notice that the trend of total utility and matching size is similar to the trend of varying $w.r$. As for CPU time, the execution time of the algorithm increases continuously when $w.d$ and $t.d$ get larger, as illustrated in Fig. 4(i). This is due to the fact that a larger deadline window means more time is needed to traverse the tasks or workers in each iteration. Our O-OP and O-Greedy are still more efficient than TGOA-OP in terms of time consumption.

Effect of $w.\theta$: Fig. 4(b) reports the results of the total utility for varying $w.\theta$ on the synthetic dataset. When $w.\theta$ increases from 0.1 to 0.9, the overall utility calculated by all the approaches increases gradually. This is because a larger $w.\theta$ indicates that a worker gets a higher utility score for each request completed. Moreover, our proposed O-Base, O-OP, O-Greedy and O-RT can achieve more overall utility than the baseline Basic-RT but sacrifice some efficiency. Fig. 4(f) and (j) show that the seven algorithms are almost unaffected by differences in $w.\theta$. The reason is that changes in the quality score $w.\theta$ only affect how much each worker gets payoff for completing each request.

Effect of μ, λ and U: Figs. 4(c), (d) and 5(a) report the results of the total utility with μ, λ and U. As we can see, the overall utility of all methods gradually increases under three different distributions. Our proposed O-OP is still the most effective, and Basic-RT is the least effective. In terms of matching size, as shown in Figs. 4(g), (h) and 5(e), all algorithms remain basically the same except the random threshold algorithm O-RT and Basic-RT. This is because different maximum benefit values affect the threshold setting of the algorithm. In Figs. 4(k), (l) and 5(i), all methods follow a similar trend to $w.\theta$. In particular, we observe similar results for the variation of total utility, matching size and CPU time for three different distributions.

Scalability: To better evaluate the performance of our proposed methods, we expand the size of the synthetic dataset Syn. In Fig. 5(b), the total utility of all the algorithms increases with the increment of $|W|$ and $|R|$, which is natural as more worker-task pairs can be assigned. O-OP obtains the highest total utility, followed by O-Greedy, O-Base, Greedy, TGOA-OP, O-RT and Basic-RT. Fig. 5(f) shows that our proposed four non-rejection methods still achieve more matching sizes than TGOA-OP. Considering the running time in Fig. 5(j), O-Greedy is as efficient as the baseline Basic-RT; the running time of O-Greedy is only $4.2\%-28.5\%$ longer than that of Basic-RT and $2.5\%-24.6\%$ longer than that of Greedy.

Performance on Real Datasets: Fig. 5(c) and (d) illustrate the effect of the total utility of the varying $|W|$ on datasets DiDi and EverySender. We can observe that the increment in $|W|$ leads to an increase in the overall utility of all seven methods. Moreover, the total utility of O-OP increases faster than those of the baseline Basic-RT as $|W|$ increases. In terms of the number of completed requests, as illustrated in Fig. 5(g) and (h), TGOA-OP and Basic-RT are still the least effective. Fig. 5(k) and (l) show that the running time of all methods also exhibits a similar increasing trend with $|R|$. The time overhead of O-Greedy is close to that of Basic-RT and Greedy. And the running time of



(a) Utility vs. $w.d, r.d$ (Syn) (b) Utility vs. $w.\theta$ (Syn) (c) Utility vs. μ (Syn) (d) Utility vs. λ (Syn)

(e) Number vs. $w.d, r.d$ (Syn) (f) Number vs. $w.\theta$ (Syn) (g) Number vs. μ (Syn) (h) Number vs. λ (Syn)

(i) Times vs. $w.d, r.d$ (Syn) (j) Times vs. $w.\theta$ (Syn) (k) Times vs. μ (Syn) (l) Times vs. λ (Syn)

Fig. 4. Results on varying duration $w.e, t.e$, quality score $w.\theta$, normal distribution μ and index distribution λ.

O-OP is higher than that of Basic-RT, but lower than that of O-Base and TGOA-OP. In particular, our O-Greedy and O-OP outperform both the baseline Basic-RT and TGOA-OP in terms of overall utility and matching size.

Summary of Results: The experimental study can be summarized as follows.

- O-OP significantly outperforms other algorithms in total utility and number of assigned tasks while sacrificing some efficiency. In particular, O-OP outperforms the state-of-the-art two-sided online algorithm TGOA-OP while having the same theoretical guarantees.
- O-Greedy provides a good balance between efficiency and effectiveness.
- The O-RT algorithm achieves a higher utility and matching size than the baseline Basic-RT but loses some theoretical performance.

VII. Related Work

In recent years, SC as an emerging topic has attracted extensive attention from industry and academia [37], [38]. The important issue in SC is online assignment, assigning tasks to the appropriate workers in a timely manner. The task allocation approaches can be categorized into two types: offline and online matching [12].

1) *Offline matching:* In offline scenarios, all algorithms are developed under the premise of known worker and request information. One of the earliest works on spatial crowdsourcing research is the paper by Kazemi et al. [1], who consider spatial and temporal constraints and propose a spatial task assignment algorithm for mobile crowdsourcing. [36] designed an optimal allocation method that reduces task allocation in offline scenarios to a maximum flow problem to maximize the number of allocated tasks. Wang et al. [39] proposed a multi-objective optimization approach for task assignment in spatial crowdsourcing, including task completion time, worker travel distance, and worker payment. Some recent work tries to consider various variants of the problem in SC. A variety of perspectives on fairness among workers are studied to minimize the difference in earnings between workers, which can be measured by travel time, distance and compensation, etc [40]. Since tasks may be complex and require multiple workers to collaborate to complete them, [41] proposes a coalition-based approach to assign tasks by forming a coalition of multiple workers. Moreover, there may be dependency problems among tasks, and [28] proposes a dependency-aware assignment method to maximize the number of assignments where tasks have dependencies.

2) *Online matching:* Online matching has been extensively studied and is often modeled as the problem of Online Maximum Weighted Bipartite Matching problem [10], [30]. In [26],

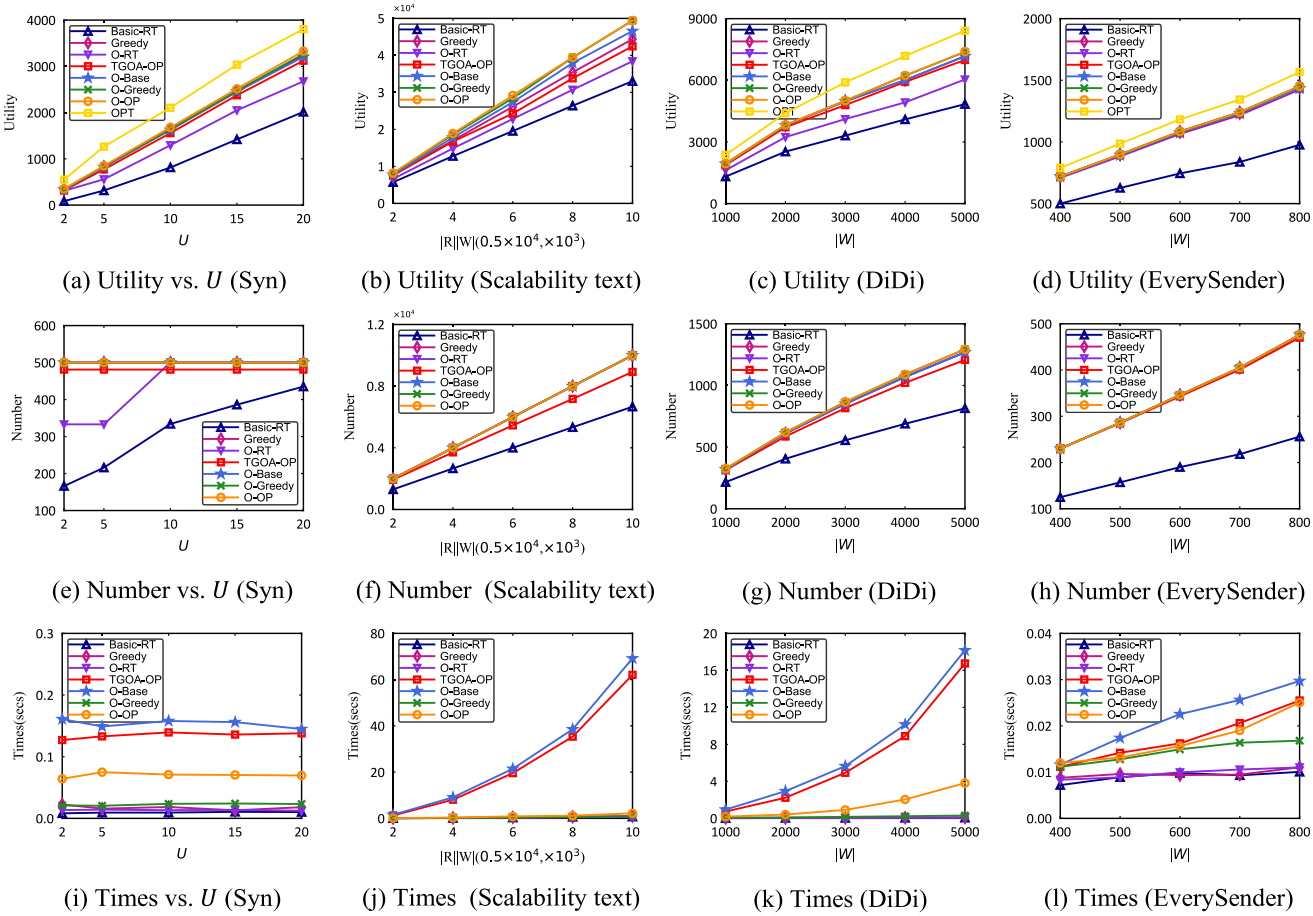


Fig. 5. Results on uniform distribution U, scalability text and real datasets.

Kesselheim et al. studied a one-sided online secretary matching problem, and it obtained a known near-optimal competitive ratio of $\frac{1}{e}$. Tong et al. extended the one-sided online to the two-sided online matching problem [11], i.e., both workers and requests dynamically appear on the platform, obtaining a $\frac{1}{4}$-competitive ratio. [15] considered a specific online matching scenario in SC, namely the real-time car-hailing service, where drivers of different platforms can service requests across platforms, and proposed a random threshold algorithm with a competitive ratio of $\frac{1}{8e}$. Considering the characteristics of car-hailing service, [42] further studied the problem of online task assignment with reusable resources, where drivers are reusable (i.e., drivers can rejoin the platform after completing a user request), establishing a one-to-many matching approach with a $\frac{1}{2}$-competitive ratio. Some studies considered a new variant in SC, 3D online matching, where platforms need to conduct three stakeholders (requests, workers, and workplaces) to match [12]. Online food delivery is a typical 3D matching application. [16] considered a 3D online food delivery problem and proposed a skyline kinetic tree-based solution to maximize the comprehensive matching benefits. Recently, [17] considered an online matching problem with preference awareness in opportunistic mobile crowdsourcing, and a stable task online assignment method was proposed to maximize worker benefits. [35] proposed a two-sided online priority assignment problem, in which tasks have dynamic priorities in the assignment process. However, all the above approaches decide whether new requests are matched or not based on the benefits of the platform or worker, and do not consider the scenario where the user's request cannot be rejected by the platform or worker.

VIII. Conclusion

In this article, we study a novel problem, called Online Non-rejection aware Task Assignment (ONRTA) in spatial crowdsourcing, where a newly arrived task (worker) cannot be rejected by the SC platform as long as there are workers (tasks) that satisfy matching constraints with it. To address the new problem, we focus on two main online arrival models, the AO model and the RO model. In the AO model, we design a non-rejection random threshold algorithm and prove that it has a theoretical guarantee on competitive ratio. In the RO model, we first propose an ONRTA-Base algorithm with a competition ratio of $\frac{1}{4}$. Based on this framework, we further propose the ONRTA-OP and ONRTA-Greedy, which are faster and more efficient algorithms with a competition ratio of $\frac{1}{4}$ and $\frac{1}{8}$, respectively. Finally, we validate the efficacy of our proposed solutions through extensive experiments on real and synthetic datasets. In particular, experimental results show that our proposed O-OP algorithm has higher effectiveness than online rejection-aware algorithms

with theoretical guarantees, and has the same competitive ratio as state-of-the-art rejection-aware online methods under the RO model. Therefore, it can provide an effective benchmark for online algorithms in spatial crowdsourcing.

REFERENCES

[1] L. Kazemi and C. Shahabi, "GeoCrowd: Enabling query answering with spatial crowdsourcing," in *Proc. 20th Int. Conf. Adv. Geographic Inf. Syst.*, 2012, pp. 189–198.

[2] S. R. B. Gummidi, X. Xie, and T. B. Pedersen, "A survey of spatial crowdsourcing," *ACM Trans. Database Syst.*, vol. 44, no. 2, pp. 1–46, 2019.

[3] H. To, C. Shahabi, and L. Kazemi, "A server-assigned spatial crowdsourcing framework," *ACM Trans. Spatial Algorithms Syst.*, vol. 1, no. 1, pp. 1–28, 2015.

[4] Didi Chuxing, Accessed: Jan. 16, 2023. [Online]. Available: https://www.didiglobal.com/

[5] Grab, Accessed: Jan. 17, 2023. [Online]. Available: https://www.grab.com

[6] Doordash, Accessed: Jan. 17, 2023. [Online]. Available: https://www.doordash.com

[7] Ele.me, Accessed: Feb. 2, 2023. [Online]. Available: http://www.openstreetmap.org

[8] Waze, Accessed: Feb. 2, 2023. [Online]. Available: https://www.waze.com

[9] Openstreetmap, Accessed: Feb. 2, 2023. [Online]. Available: http://openstreetmap.org

[10] Y. Tong et al., "Flexible online task assignment in real-time spatial data," in *Proc. VLDB Endowment*, vol. 10, no. 11, pp. 1334–1345, 2017.

[11] Y. Tong, Y. Zeng, B. Ding, L. Wang, and L. Chen, "Two-sided online micro-task assignment in spatial crowdsourcing," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 5, pp. 2295–2309, May 2021.

[12] T. Song et al., "Trichromatic online matching in real-time spatial crowdsourcing," in *Proc. IEEE Int. Conf. Data Eng.*, 2017, pp. 1009–1020.

[13] J.-X. Liu and K. Xu, "Budget-aware online task assignment in spatial crowdsourcing," *World Wide Web*, vol. 23, no. 1, pp. 289–311, 2020.

[14] Y. Wang and S. C.-W. Wong, "Two-sided online bipartite matching and vertex cover: Beating the greedy algorithm," in *Proc. Int. Colloq. Automata Lang. Program.*, Springer, 2015, pp. 1070–1081.

[15] Y. Cheng, B. Li, X. Zhou, Y. Yuan, G. Wang, and L. Chen, "Real-time cross online matching in spatial crowdsourcing," in *Proc. IEEE 36th Int. Conf. Data Eng.*, 2020, pp. 1–12.

[16] B. Zheng et al., "Online trichromatic pickup and delivery scheduling in spatial crowdsourcing," in *Proc. IEEE 36th Int. Conf. Data Eng.*, 2020, pp. 973–984.

[17] F. Yucel and E. Bulut, "Online stable task assignment in opportunistic mobile crowdsensing with uncertain trajectories," *IEEE Internet Things J.*, vol. 9, no. 11, pp. 9086–9101, Jun. 2022.

[18] H.-F. Ting and X. Xiang, "Near optimal algorithms for online maximum edge-weighted b-matching and two-sided vertex-weighted b-matching," *Theor. Comput. Sci.*, vol. 607, pp. 247–256, 2015.

[19] L. Yang, X. Yu, J. Cao, X. Liu, and P. Zhou, "Exploring deep reinforcement learning for task dispatching in autonomous on-demand services," *ACM Trans. Knowl. Discov. Data*, vol. 15, no. 3, pp. 1–23, 2021.

[20] Y. Li, Q. Wu, X. Huang, J. Xu, W. Gao, and M. Xu, "Efficient adaptive matching for real-time city express delivery," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 6, pp. 5767–5779, Jun. 2023.

[21] Z. Chen, P. Cheng, Y. Zeng, and L. Chen, "Minimizing maximum delay of task assignment in spatial crowdsourcing," in *Proc. IEEE 35th Int. Conf. Data Eng.*, 2019, pp. 1454–1465.

[22] J. P. Dickerson, K. A. Sankararaman, A. Srinivasan, and P. Xu, "Allocation problems in ride-sharing platforms: Online matching with offline reusable resources," *ACM Trans. Econ. Comput.*, vol. 9, no. 3, pp. 1–17, 2021.

[23] A. Chapman and M. Mesbahi, "Semi-autonomous consensus: Network measures and adaptive trees," *IEEE Trans. Autom. Control*, vol. 58, no. 1, pp. 19–31, Jan. 2013.

[24] M. O. Ogbole, E. Ogbole, and A. Olagesin, "Cloud systems and applications: A review," *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.*, vol. 7, pp. 142–149, 2021.

[25] Z. Huang, Z. G. Tang, X. Wu, and Y. Zhang, "Online vertex-weighted bipartite matching: Beating 1–1/e with random arrivals," *ACM Trans. Algorithms*, vol. 15, no. 3, pp. 1–15, 2019.

[26] T. Kesselheim, K. Radke, A. Tönnis, and B. Vöcking, "An optimal online algorithm for weighted bipartite matching and extensions to combinatorial auctions," in *Proc. Eur. Symp. Algorithms*, Springer, 2013, pp. 589–600.

[27] X. Miao, Y. Kang, Q. Ma, K. Liu, and L. Chen, "Quality-aware online task assignment in mobile crowdsourcing," *ACM Trans. Sensor Netw.*, vol. 16, no. 3, pp. 1–21, 2020.

[28] W. Ni, P. Cheng, L. Chen, and X. Lin, "Task allocation in dependency-aware spatial crowdsourcing," in *Proc. IEEE 36th Int. Conf. Data Eng.*, 2020, pp. 985–996.

[29] V. V. Vazirani, *Approximation Algorithms*. Berlin, Germany: Springer, 2013, vol. 1.

[30] C. Chen, L. Zheng, V. Srinivasan, A. Thomo, K. Wu, and A. Sukow, "Conflict-aware weighted bipartite B-matching and its application to E-commerce," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 6, pp. 1475–1488, Jun. 2016.

[31] R. Reiffenhauser, "An optimal truthful mechanism for the online weighted bipartite matching problem," in *Proc. 13th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2019, pp. 1982–1993.

[32] M. Goyal, "Secretary matching with vertex arrivals and no rejections," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 5051–5058.

[33] M. Hoefer and B. Kodric, "Combinatorial secretary problems with ordinal information," 2017, *arXiv: 1702.01290*.

[34] Y. Li, J. Fang, Y. Zeng, B. Maag, Y. Tong, and L. Zhang, "Two-sided online bipartite matching in spatial data: Experiments and analysis," *GeoInformatica*, vol. 24, pp. 175–198, 2020.

[35] Q. Zhang, Y. Wang, G. Yin, X. Tong, A. M. V. V. Sai, and Z. Cai, "Two-stage bilateral online priority assignment in spatio-temporal crowdsourcing," *IEEE Trans. Services Comput.*, vol. 16, no. 3, pp. 2267–2282, May/Jun. 2023.

[36] D. Deng, C. Shahabi, and U. Demiryurek, "Maximizing the number of worker's self-selected tasks in spatial crowdsourcing," in *Proc. ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, 2013, pp. 314–323.

[37] L. Yang, X. Yu, J. Cao, W. Li, Y. Wang, and M. Szczecinski, "A novel demand dispatching model for autonomous on-demand services," *IEEE Trans. Services Comput.*, vol. 15, no. 1, pp. 322–333, Jan./Feb. 2022.

[38] H. Wang, E. Wang, Y. Yang, J. Wu, and F. Dressler, "Privacy-preserving online task assignment in spatial crowdsourcing: A graph-based approach," in *Proc. IEEE Conf. Comput. Commun.*, 2022, pp. 570–579.

[39] L. Wang, Z. Yu, Q. Han, B. Guo, and H. Xiong, "Multi-objective optimization based allocation of heterogeneous spatial crowdsourcing tasks," *IEEE Trans. Mobile Comput.*, vol. 17, no. 7, pp. 1637–1650, Jul. 2018.

[40] F. Basık, B. Gedik, H. Ferhatosmanoğlu, and K.-L. Wu, "Fair task allocation in crowdsourced delivery," *IEEE Trans. Services Comput.*, vol. 14, no. 4, pp. 1040–1053, Jul./Aug. 2021.

[41] Y. Zhao, J. Guo, X. Chen, J. Hao, X. Zhou, and K. Zheng, "Coalition-based task assignment in spatial crowdsourcing," in *Proc. IEEE 37th Int. Conf. Data Eng.*, 2021, pp. 241–252.

[42] H. Sumita et al., "Online task assignment problems with reusable resources," 2022, *arXiv:2203.07605*.

Jiajun Yao is currently working toward the PhD degree with the School of Software Engineering, South China University of Technology, China. His research interests include spatial crowdsourcing and edge computing.

Lei Yang received the BSc degree from Wuhan University, in 2007, the MSc degree from the Institute of Computing Technology, Chinese Academy of Sciences, in 2010, and the PhD degree from the Department of Computing, Hong Kong Polytechnic University, in 2014. He is currently an associate professor with the School of Software Engineering, South China University of Technology, China. He has been a visiting scholar with Technique University Darmstadt, Germany from 2012 to 2013. His research interests include edge and cloud computing, distributed machine learning, and scheduling and optimization theories and techniques.

Zhenyu Wang received the BSc degree in computer science from Xiamen University, China, in 1987, and the MSc and PhD degrees in computer science from the Harbin Institute of Technology, China, in 1990 and 1993, respectively. He is currently a professor with the School of Software Engineering, South China University of Technology, China. His research interests include cloud and edge computing, social computing, and blockchain.

Xiaohua Xu received the bachelor's degree from the Zhu Kezhen College, Zhejiang University, and was awarded the Zhu Kezhen Honorary Certificate, and the PhD degree from the Illinois Institute of Technology, USA. He is a professor with the School of Computer Science and Technology, University of Science and Technology of China. His research interests include edge computing, algorithm design, and intelligent IoT.