Information Sciences 622 (2023) 512–535

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Satisfaction-aware Task Assignment in Spatial Crowdsourcing

Yuan Xie [a], Yongheng Wang [b,*], Kenli Li [a], Xu Zhou [a], Zhao Liu [a], Keqin Li [a,c]

[a] College of Computer Science and Electronic Engineering, Hunan University, China
[b] Big Data Intelligence Research Center of Zhejiang Lab, China
[c] Department of Computer Science, State University of New York, New Paltz, New York 12561, USA

ARTICLE INFO

Article history:
Received 4 December 2021
Received in revised form 8 October 2022
Accepted 16 November 2022
Available online 22 November 2022

Keywords:
Cooperation quality
Spatial crowdsourcing
Task assignment
User satisfaction

ABSTRACT

With the ubiquitous of GPS-equipped devices, spatial crowdsourcing (SC) technology has been widely utilized in our daily life. As a novel computing paradigm, it hires mobile users as workers who physically move to the location of the task and perform the task. Task assignment is a fundamental issue in SC. In real life, there are many complex tasks requiring different workers, among which the quality of worker cooperation and the price satisfaction of users should not be ignored. Hence, this paper examines a *satisfaction-aware task assignment* (SATA) problem with the goal of maximizing overall user satisfaction, where the user satisfaction integrates the satisfaction towards price and cooperation quality. The SATA problem has been proved to be NP-hard by reducing it from the *k*-set packing problem. In addition, two algorithms, namely, *conflict-aware greedy* (CAG) algorithm and *game theoretic* (GT) algorithm with an optimization strategy, are proposed for solving the SATA problem. The CAG algorithm can efficiently obtain a result with provable approximate bound, while the GT algorithm is proven to be convergent which can find a Nash equilibrium. Extensive experiments have demonstrated the effectiveness and efficiency of our proposed approaches on both real and synthetic datasets.

© 2022 Published by Elsevier Inc.

1. Introduction

With the popularity of smart mobile devices, spatial crowdsourcing (SC), as a novel concept, is proposed to employ sensor-equipped people as workers who would move to the designated location and contribute to the task. The SC concept has been extensively applied in many successful crowdsourcing platforms (e.g., TaskRabbit.[1], DiDi,[2] and Grab[3]) and aroused interest from both industry and academia.

As a fundamental problem in SC, task assignment has drawn considerable attention, most of which focuses on allocating simple tasks to workers, like taking photos, reporting traffic conditions and delivering passengers. In practice, however, many real-life tasks are complex requiring multiple workers, such as event organization, house decoration and wedding preparation. These complex tasks require multiple workers who provide diverse skills to play different roles in the task.

* Corresponding author.
E-mail addresses: yxie@hun.edu.cn (Y. Xie), wangyh@zhejianglab.com (Y. Wang), lkl@hnu.edu.cn (K. Li), zhxu@hnu.edu.cn (X. Zhou), lzhao@hnu.edu.cn (Z. Liu), lik@newpaltz.edu (K. Li).

[1] https://www.taskrabbit.com/.
[2] https://www.didiglobal.com/.
[3] https://www.grab.com/sg/.

https://doi.org/10.1016/j.ins.2022.11.081
0020-0255/© 2022 Published by Elsevier Inc.

An obscure assumption recognized by existing SC studies is that the workers are volunteered to complete the tasks assigned to them. However, different combinations of workers will exhibit different behaviors when completing the same task.

For example, two worker sets with different cooperation qualities for the same task may lead to different types of behaviors. One group of workers with plentiful cooperation experience and close proximity to the task as well usually complete the tasks quickly and fabulously, while another group may not achieve a high satisfaction for both workers and task requesters since the worker group requires a higher travel cost and lacks any experience of collaboration. Actually, when a worker is forced to cooperate with other unfamiliar workers, it is very likely that they will not be able to perform at their best, which will reduce the quality of task completion. Moreover, selecting nearing workers from the assigned tasks can significantly reduce the travel cost, which is related to the workers' profit. It is of great importance since profit can fundamentally drive workers to complete tasks in high quality. Therefore, the key to control quality for task accomplishment is how to combine workers who have been frequently cooperating with each other. Furthermore, incorporating price satisfaction of workers also can improve the effectiveness of spatial task assignments.

Inspired by our observation, it is the key to optimizing price satisfaction and worker cooperation quality satisfaction to improve task allocation quality. For example, repairing a house requires a group of workers with various challenging skills, including repairing floors, installing pipe systems and electronic components, painting walls, and finally cleaning rooms. If only considering the price satisfaction, the house owner would probably employ people who are willing to work for low prices but less effective in working as part of a team. In that case, the decoration quality of the new house will be impacted. On the other hand, merely asking workers to strive for optimized cooperation quality but ignoring the task budget will also influence the task completion quality of the assigned task since the owner cannot afford the cost.

Therefore, when the platform provides the optimal combination of workers by optimizing price satisfaction and cooperation satisfaction, it not only improves the quality of task assignments to increase the satisfaction of the requester but also increases the enthusiasm of workers to participate in the assigned task. In this way, workers and requesters will benefit at the same time. Moreover, the platform will also benefit since more and more task requesters and workers will be attracted to the platform for achieving long-term stability and high profit.

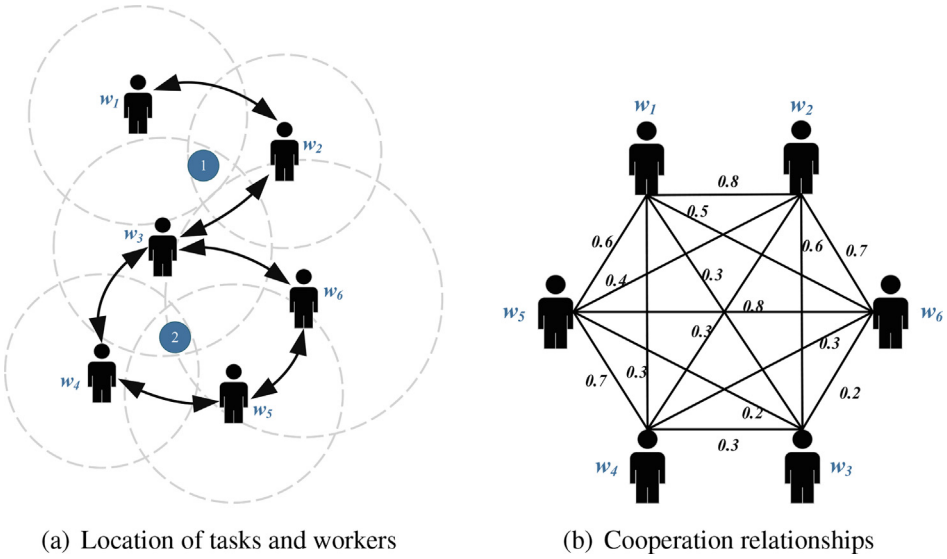
Based on the aforementioned analysis, we formulate user satisfaction by trading off the cooperation quality and price information. Next, we propose a novel task assignment problem in SC, namely the satisfaction-aware task assignment (SATA) problem, which aims to maximize overall user satisfaction. To the best of our knowledge, this is the first study to propose the problem to optimize the perspective of above factors in the Spatial Crowdsourcing.

In order to better understand the motivation of our SATA problem, we introduce a concrete example as follows.

Example 1. In Fig. 1(a), there are two tasks t_1, t_2, and six workers $w_1 \sim w_6$. The available working range of workers are represented by the dotted circle, and the double-headed arrow indicates the cooperation between any worker pairs. Fig. 1(b) presents the cooperation scores of all worker pairs, where the number be attached in the black line is the cooperation score of the linked worker pair (i.e., the cooperation score of w_2 and w_4 is 0.3). Moreover, the task's skills requirement, the worker's professional skills and the price satisfaction of each available worker pair are shown in Table 1.

After the workers and task requesters upload their information, the platform can know that t_1 requires skills k_1 and k_2 while t_2 requires the skills k_1 and k_3. According to the range limits and skill constraints, there are three workers, w_1, w_2 and w_3, who are eligible to be assigned to the task t_1. On the other hand, the available worker set of t_2 is $\{w_3, w_4, w_5, w_6\}$. After the skill matching process, the worker group $\{w_1, w_2\}$ and $\{w_2, w_3\}$ are eligible to complete t_1 while $\{w_3, w_4\}, \{w_4, w_5\}, \{w_5, w_6\}$ and $\{w_3, w_6\}$ have the qualification to accomplish t_2. The satisfaction score is $S(t) = \alpha * P(t, W) + (1 - \alpha) * C(t, W)$, where $P(\cdot)$ is the price satisfaction score, $C(\cdot)$ is the cooperation satisfaction score. For simplicity, assume the coefficient of a linear relationship between the two factors is 0.5. If the platform merely consider optimizing the cooperation quality, thus, $\{w_1, w_2\}$ is assigned to t_1 and $\{w_5, w_6\}$ is assigned to t_2. The overall satisfaction score is 1.1 (i.e., $S(t_1) = 0.5 \times 0.2 + 0.5 \times 0.8 = 0.5, S(t_2) = 0.5 \times 0.4 + 0.5 \times 0.8 = 0.6$). On the other hand, if only maximizing the price satisfaction in the assignment, $\{w_2, w_3\}$ is assigned to t_1 and $\{w_3, w_6\}$ is assigned to t_2. The total satisfaction score is 1.2 (i.e., $S(t_1) = 0.5 \times 0.8 + 0.5 \times 0.6 = 0.7, S(t_2) = 0.5 \times 0.8 + 0.5 \times 0.2 = 0.5$). However, this is not the optimal assignment result for achieving maximum user satisfaction regarding considering price satisfaction and cooperation satisfaction at the same time. The SATA task assignment problem generates the best assignment result, in which $\{w_2, w_3\}$ be assigned to t_1 and $\{w_4, w_5\}$ be assigned to t_2. The user satisfaction score of t_1 is $S(t_1) = 0.5 \times 0.8 + 0.5 \times 0.6 = 0.7$ while the score of t_2 is $S(t_2) = 0.5 \times 0.7 + 0.5 \times 0.7 = 0.7$. The overall satisfaction score is 1.4, which is higher than that of unilaterally considering cooperation quality or price satisfaction.

The SATA problem is a new extension of the task assignment problem in the SC since existing studies fail to consider optimizing the price satisfaction and cooperation quality simultaneously. Some existing studies have explored the complex task assignment problem in SC [1,2], however, they do not concentrate on the cooperation relations among workers. Besides, they also fail to effectively incorporate price information [3,4]. Cheng et al. [1] sought to optimize the profits of the platform in the multi-skill task assignment, but they overlooked the quality of cooperation between the workers. Additionally, Cheng et al. [3] focused on maximizing the overall cooperation scores in the SC task assignment problem, in which the task requires a fixed number of workers but ignores the fact that they are cooperating according to the diverse skills. Besides, it also overlooks the price information. Liang et al. [5] proposed a two-stage coopetition model by constructing novel three-way decision to availably solve the crowdsourcing task allocation problem, ignoring profits of crowdsourcing platforms. Gao et al. [6]



(a) Location of tasks and workers (b) Cooperation relationships

Fig. 1. Example of the SATA problem.

Table 1
Skill description and price satisfaction of tasks/workers.

Task/Worker	**Skill**
t_1	$\{k_2, k_3\}$
t_2	$\{k_1, k_3\}$
w_2	$\{k_2\}$
w_4, w_6	$\{k_1\}$
w_1, w_3, w_5	$\{k_3\}$
Worker Set	**Price Satisfaction**
$\{w_1, w_2\}$	0.2
$\{w_2, w_3\}$	0.8
$\{w_3, w_4\}$	0.3
$\{w_4, w_5\}$	0.7
$\{w_3, w_6\}$	0.8
$\{w_5, w_6\}$	0.4

assigned team-oriented workers with the largest satisfaction for the task, while the satisfaction refers to the worker's preference for the task, ignored the cooperation among workers. Liu et al. [7] focused on assigning a group of workers to a multi-stage task for the maximal profits, which is applying greedy strategy and game theory method to solve their MSCTA problem. However, they ignored the cooperation quality among workers in the greedy algorithm. Besides, the worker in MSCTA problem is allowed to be quit from the game iteration when the assigned worker satisfied the complete condition of the corresponding task. Prantare et al. [8,9] proposed an SCSGA problem, which has a similar objective optimization to our SATA problem but without considering specific spatial–temporal limitations, such that skill limit and working range constraint. Therefore, existing methods cannot solve our problem directly and effectively.

We generally handle the SATA problem in the batch-based framework, in which the SC platform periodically assigns the available workers to unassigned tasks. In this paper, we first show the NP-hardness of SATA by a reduction from the k-SP problem. To tackle the SATA problem, two algorithms, namely, conflict-aware greedy (CAG) algorithm and game theoretic (GT) algorithm with the threshold stop (TS) optimization strategy, are presented. The CAG algorithm introduces a greedy strategy iteratively choosing the task-and-worker pair with the highest satisfaction score. Besides, we prove its submodularity and give an approximate ratio. However, the workers tend to choose the task proactively for higher satisfaction in many real-life scenarios. Accordingly, we develop the GT algorithm by formulating the SATA problem to a SATA strategy game and providing theoretical proof of the Nash equilibrium. For improving the effectiveness of GT, we propose the TS strategy which can reduce the running time but only at the slightly cost of total satisfaction score.

Briefly, the contributions are illustrated as follows,

- We formally define the satisfaction-aware task assignment (SATA) problem in spatial crowdsourcing and prove its NP-hardness, where the satisfaction is comprised of the cooperation quality and price satisfaction in Section 3.
- We propose the conflict-aware greedy (CAG) algorithm iteratively assigns the task-and-worker pair with maximum satisfaction increment, and show the approximate bound in Section 5.
- We propose the game theory (GT) algorithm to allow workers choose tasks proactively, and prove the existence, convergence and quality of Nash equilibrium in Section 6. In addition, we propose a threshold stop (TS) optimization strategy for the GT.
- We conduct extensive experiments on real and synthetic datasets to demonstrate the efficiency and effectiveness of our proposed algorithms in Section 7.

The rest of the paper follows the following structure. In Section 2, related studies are summarized. The SATA problem is defined in Section 3, followed by a framework for solving it in Section 4. Section 8 discusses the article and Section 9 concludes the article.

2. Related work

Spatial crowdsourcing is a new computing paradigm requiring workers to move to a specific location and accomplish the assigned tasks. Tong et al. [10] identified four core issues in spatial crowdsourcing: (1) task assignment [11–13], (2) quality control [14,15], (3) incentive mechanism design [16,17], and (4) privacy protection [18,19]. Among them, the most fundamental challenge problem in spatial crowdsourcing is task assignment.

Offline Mode and Online Mode: Based on the input objective, task assignment can be classified into the offline [20,21] and online [22] categories. The offline mode has to know all the information about tasks and workers in advance. Comparatively speaking, the online mode is more complex, i.e., workers and tasks are arrived randomly. In addition, according to the algorithmic assignment model, the task assignment consists of the matching problem [23–25] and the planning problem [26,22]. To be specific, the matching problem refers to assign the worker to the tasks while the planning problem involves both assignment and route planning. Among them, our SATA problem is a batch-based assignment problem.

Task Publishing Mode: Kazemi et al. [27] defined two task publishing modes in SC problems, namely, server assigned tasks (SAT) mode [28–30,20,21], and worker selected tasks (WST) mode [31]. SAT is a standard mode, which assigns workers (tasks) to tasks (workers) under the control of the platform. Many studies on task assignment problem in SC have focused on the SAT mode. Zhao et al. [20,21] assigned range and time-constrained spatial tasks and planed the route for completing all tasks to workers, aiming at maximizing number of assigned tasks and minimizing total travel costs. To et al. [29] tried to maximize the number of assigned tasks under budget constraint in the SAT mode. Prantare et al. [8,9] focused on the SCSGA problem which has a similar objective optimization to our SATA problem, aiming at maximizing the total weight of the coalition of agents. However, our SATA problem considers the skill matching and other spatial–temporal constraints such that work available range and task budget, which is more complex. On the other hand, in WST mode, the platform allows workers to choose the tasks proactively or reject the assigned tasks, which is more in line with the worker's expectations. Cheng et al. [3] applied the game theoretic algorithm and enabled the workers to do so. Likewise, Shan et al. [31] recommended tasks to workers via supervised learning methods so that the workers could choose the tasks they preferred. Moreover, Zheng et al. [32] took into consideration workers' rejection and tried to maximize workers' acceptance. In our paper, the CAG algorithm for our SATA problem is associated with the SAT mode in Section 5 while the GT algorithm is applied to the WST mode in Section 6.

Optimization of Task Assignment: Existing studies have presented different ways of assigning tasks to workers, and the task assignment of SC seeks to achieve different optimization goals, such as maximizing the number of assigned tasks [27] or maximizing the total payoff of the SC platform in which the price fluctuates according to the supply and demand [33], minimizing the distance cost of workers [34] or minimizing the delay in task response time [35], or improving the quality of assignment decisions based on the preference of workers [36,37]. Excepting for single-objective optimization problems, many studies concerned two goals at the same time, including maximizing the number of tasks and minimizing workers' travel distance simultaneously [2]. In addition to this, task assignment is also a foundation problem in the crowdsensing market, such as recommending a trajectory to the worker allowing him to complete the assigned tasks [38], assigning suitable tasks to the location-sensitive workers according to their skills and preference [39], and suggesting a trajectory to the worker for maximizing the satisfaction of workers, requesters and the system [40]. This paper proposes the satisfaction-aware task assignment problem aiming at maximizing the overall user satisfaction for optimizing price satisfaction and cooperation quality, which is of importance to spatial crowdsourcing. Therefore, the satisfaction-aware task assignment problem is a new extension of Spatial Crowdsourcing.

3. Problem statement

Some basic definitions of SC are presented before we illustrate our problem. Furthermore, we verify the SATA problem is NP-hard at the end of this section.

Definition 1. (Spatial Worker) Let $W = \{w_1, w_2, \cdots, w_m\}$ denotes the spatial workers appearing on platform. The worker $w_j = (L_{w_j}, K_{w_j}, R_{w_j}, v)$ claims he has the skill set K_{w_j} and is allowed to move within working range R_{w_j} in his location L_{w_j}. Besides, each worker has an average unit cost $v$for traveling.

In addition to some basic information about workers, crowdsourcing platforms also know previous cooperation activities of all workers. For the given workers w_i and w_j, the cooperation score is $q_j(w_k)$, such that

$$q_j(w_k) = \beta \cdot \omega + (1-\beta) \cdot \frac{|G_{w_j} \cap G_{w_k}|}{|G_{w_j} \cup G_{w_k}|}, \tag{1}$$

where ω is a basic cooperation score set by the platform, which is set between $[0, 1]$. For any worker pair, whether they participate in the same tasks or not, they have a basic cooperation score for ensuring the cooperation score is not equal to zero. β is the linear coefficient to reconcile the basic cooperation score and the historical average cooperation quality, varying from 0to 1. G_{w_j} and G_{w_k} denote the task set they have participated in. Thus, the cooperation score is the absolute ratio value of intersection task set (i.e., $G_{w_j} \cap G_{w_k}$) and union task set (i.e., $G_{w_j} \cup G_{w_k}$). The more tasks the workers participate in, the higher the cooperation score. The cooperation quality [3] is measured by the historical cooperating times in our work, e.g., the higher cooperation score is, the better the service provided by the two workers is. Therefore, the collaboration times of workers in the system is the guarantee of worker's cooperation quality since the cooperation score is linear to collaboration times as shown in Definition 1.

The intuition of cooperation score in Eq. (1) reflects the balance of the historical performance and the priori assumption (i.e., the average cooperation quality between any two workers, such as ω, which is set to 0.5 by default in our experiment). Moreover, the unit cost v in our work is unit traveling cost, however, nothing would prevent to consider the v to be other costs such as energy, computation resource, and storage, etc. We only need to modify the budget constraints in this paper.

Definition 2. (Spatial Task) Let $T = \{t_1, t_2, \cdots, t_n\}$ be a set of spatial tasks. Task $t_i = (L_{t_i}, K_{t_i}, B_{t_i}, D_{t_i})$ is reported by the requester with a specific location L_{t_i} to complete. The corresponding requester also gives task budget B_{t_i}, skill requirement K_{t_i}, and expiration D_{t_i}.

Each task t_i needs workers to move from their current place to the designated location L_{t_i} and process it. Apart from this, the requester will not pay for assigned workers more than the task budget B_{t_i}. The requiring skills K_{t_i} also need to be shown in the platform for the tasks-workers matching. Further, task t_i will not be matched with any workers in the platform if the current matching time exceeds D_{t_i}.

Definition 3. (Candidate Worker Set) Candidate worker set (CWS) is the available workers for task t_i, which satisfy constraints such that

1) Each worker in CWS has at least a valid skill matching to the task t_i (i.e., $\forall w_j \in$ CWS, $K_{w_j} \cap K_{t_i} \neq \varnothing$).
2) The location of task t_i is located within reachable work range of workers in the CWS (i.e., $\forall w_j \in CWS, dist(t_i, w_j) \leqslant R_{w_j}$).

For example, in Fig. 1, workers w_1, w_2 and w_3 are the CWS of task t_1. The reason is that t_1 is located in the effective work range of w_1, w_2 and w_3. Besides, they have a valid skill-to-job match for task t_1, where the skill set of t_1 is $\{k_2, k_3\}$. The skill of workers is shown in the Table 1 (i.e., $K_{w_1} = k_3, K_{w_2} = k_2$ and $K_{w_3} = k_3$). Thus, w_1, w_2, and w_3 are available to task t_1 (i.e., $K_{t_1} \cap K_{w_1} \neq \varnothing, K_{t_1} \cap K_{w_2} \neq \varnothing$, and $K_{t_1} \cap K_{w_3} \neq \varnothing$).

Definition 4. (Tasks-and-Worker Set Pair) A successful matching pair $\langle t_i, W_i \rangle$ denotes the task t_i will be completed by the workers in W_i.

All workers in W_i are selected from the CWS of t_i. The skills of the worker set satisfy the complete condition of t_i. Moreover, the effective skills of workers in W_i must be different from each other for avoiding hiring duplicate workers. The users are not willing to hire two workers who have the same function for the task since the budget is limited. For instance, the house owner only needs one pipe repair worker for the house repairing and needs other workers for different jobs such as cleaning, but doesn't ask for another pipe repair worker under the limited budget.

Definition 5. (User Satisfaction) Satisfaction is comprised of price satisfaction and cooperation satisfaction:

$$S(t_i, W_i) = \frac{\alpha}{P_{max}} \cdot P(t_i, W_i) + \frac{1-\alpha}{C_{max}} \cdot C(W_i), \tag{2}$$

where $P(t_i, W_i)$ denotes the price satisfaction of assigned workers W_i to t_i, and $C(W_i)$ represents the cooperation quality of W_i. Moreover, P_{max} and C_{max} are the maximum price satisfaction score and cooperation satisfaction score of task t_i, respectively, given by the platform.

There is a negative correlation between price satisfaction and the total travel cost since higher travel cost of workers would incur lower price satisfaction. The mathematical expression is

$$P(t_i, W_i) = B_{t_i} - \sum_{w_j \in W_i} dist(t_i, w_j) \cdot v, \tag{3}$$

where B_{t_i} is the budget of the task t_i, $dist(t_i, w_j)$ is the distance from the location of t_i to the worker w_j, and v is the unit traveling cost. On the other hand, $C(W_i)$ is the average cooperation score of all worker pairs in W_i[3]. The mathematical expression is

$$C(W_i) = \frac{\sum_{w_j \in W_i} \sum_{w_k \in W_i, k \neq i} q_j(w_k)}{|W_i - 1|}, \tag{4}$$

where $q_j(w_k)$ is the cooperation score in Eq. (1). Based on the above description, we can formally define the satisfaction-aware task assignment problem as follows.

Definition 6. (Satisfaction-aware problem) Given a worker set $W$and a task set T, our problem is to find $|\mathscr{A}_p|$ task-and-worker set pairs in the form of $\langle t_1, W_1\rangle, \langle t_2, W_2\rangle \cdots \langle t_{|T|}, W_{|T|}\rangle$, the goal of which is maximizing the total satisfaction score. The assignment needs to satisfy several constraints as follows,

1) Range constraint: The task t_i must be located within the work range of all workers in W_i.
2) Skill constraint: The skill requirement of t_i has to be covered by the skill union of W_i.
3) Conflict constraint: Every worker is assigned to only one task at a timestamp until he/she completes the task. After that, the worker can be released.
4) Budget constraint: The total travel cost of W_i cannot exceed task budget of t_i.
5) Duplicate constraint: The skills of workers in W_i are not duplicate with each other.

For simplicity, the unit cost for completing the assigned tasks of different workers is set to the same value v. We formulate our problem with constraints in the form of mathematical expressions as follows,

$$\max \sum_{t_i \in T} S(t_i, W_i)$$
$$s.t. \begin{cases} dist(t_i, w_j) \leqslant R_{w_j}, & 1 \leqslant i \leqslant n, 1 \leqslant j \leqslant m, \\ K_{t_i} \cap K_{w_j} \neq \varnothing, & t_i \in T, w_j \in W, \\ \sum_{i=1}^{|T|} w_{ji} \leqslant 1, & 1 \leqslant j \leqslant m, \\ \sum_{j=1}^{|W_i|} dist(t_i, w_j) \cdot v \leqslant B_{t_i}, & 1 \leqslant i \leqslant n, \\ K_{w_i} \cap K_{w_j} = \varnothing, & \forall w_i, w_j \in W_i, \end{cases} \tag{5}$$

where $dist(t_i, w_j)$ denotes the distance between location of t_i and location of w_j, R_{w_j} is the work range of w_j, K_{t_i} represents the skill requirement of t_i, K_{w_i} and K_{w_j} are the skills of worker w_i and w_j respectively and B_{t_i} denotes the task budget of t_i.

3.1. The hardness of the SATA problem

Theorem 1. The problem of satisfaction-aware task assignment is NP-hard.

Proof. We proved the SATA problem to be NP-hard by reducing it from the k-set packing problem (k-SP).

The k-set packing problem: Given a universe of elements $E = \{e_1, e_2, \ldots, e_{|E|}\}$, a collection $C = \{C_1, C_2, C_3, \ldots, C_{|C|}\}$ of element subsets with size at most k. For each subset $C\prime \in C$, it has $C' \subseteq E$ and C' is associated with a weight $W(C') > 0$, the k-set packing problem is to compute subsets $C^* \subseteq C$ to maximize $\sum_{C' \in C^*} W(C')$ satisfying $C_i \cap C_j = \varnothing$ for $C_i, C_j \in C^*$.

For a given k-set packing problem, we can transform it to an instance of the SATA problem in the following way. .

Let each element in Erepresent a spatial task, and each subset C_i corresponds to the assigned worker set of task t_i with the maximum number of workers $t_i.c = k$. We assume each worker has only one skill in the instance. Besides, $\widehat{W_i}$ and $\widehat{W_j}$ denote the worker sets assigned to tasks t_i and t_j, where $|\widehat{W_i}|$ and $|\widehat{W_j}|$ are both at most $t.c$, respectively. It holds that $\widehat{W_i} \cap \widehat{W_j} = \varnothing$ since one worker can only be assigned to one task. This is consistent with the constraint $C_i \cap C_j = \varnothing$ of the k-set parking problem where $\widehat{W_i} = C_i$ and $\widehat{W_j} = C_j$.

Regarding the SATA problem, its goal is to maximize the overall satisfaction score, which can be rewritten as $S_M = \sum_{t_i \in T} S'(t_i)$. For our SATA problem, we assign workers w_i to the task t_i for maximizing the overall satisfaction S_M. This is the same as the goal in k-set packing problem, which seeks to maximize the sum of weights for subsets.

From the above way, we can reduce the k-set packing problem to the SATA problem. Since the k-set packing problem is known to be NP-hard [37], the SATA problem is also NP-hard.

Table 2 lists most essential parameters in this paper.

4. Framework

In this part, we illustrate the framework for solving our satisfaction-aware task assignment problem. Two methods are proposed for obtaining the maximal total satisfaction score of all assigned tasks, named conflict-aware greedy (CAG) algorithm and game theory (GT) algorithm, respectively. The CAG method primarily adopts the greedy strategy, and the GT is designed based on the game theory.

The conflict-aware greedy (CAG) algorithm is comprised of two stages. In the first stage, it assigns a set of workers with the highest satisfaction score allowing workers can be assigned multiple tasks, which is denoted to the conflict workers. And then, delete all conflict workers in the task assignment result. The second stage is proposed to iteratively select the task-and-worker pair with the maximal satisfaction increment and force fully assigned tasks and workers to leave from the assignment iterations in advance, which is detailed in Section 5. However, it is unfair for workers since they cannot choose tasks by themselves though the CAG achieves a good satisfaction score. Accordingly, the game theory (GT) method has been proposed to allow workers to proactively choose their best task until the Nash equilibrium is reached. The final Nash equilibrium achieves an assignment result with a high overall satisfaction score where all workers can choose their optimal strategies, which is detailed in Section 6.

Algorithm 1: Batch-based Framework

Input: A time interval T_p
Output: A set of task-and-worker assignments $\mathcal{A}_p$ in the time interval T_p

```
1 initialize A_p ← ∅;
2 while current timestamp p in T_p do
3     collect all available tasks for T;
4     collect all available workers for W;
5     foreach t_i ∈ T do
6         select the candidate worker set (CWS) for t_i under the skill and range constraints;
7     utilize our greedy method or game theoretic approach to obtain a good assignment A_p;
8     foreach ⟨t_i, W_i⟩ ∈ A_p do
9         inform all worker in W_i to execute the task t_i;
```

Algorithm 1 shows the batch-based framework, which assigns suitable workers to proper tasks iteratively in one batch of the time interval. In each batch, the information of all available workers and valid tasks are known in advance. It is worth noting that the platform needs to check the deadline of all tasks in each batch. The available tasks which do not exceed the deadline are successfully recognized as the available tasks; otherwise, the invalid tasks are deleted from the platform, workers as well. We first initialize $\mathcal{A}_p$ as the final assignment result (Line 1). Then, collecting available workers and tasks in the current batch (Lines 3–4). The valid tasks include those that are not yet assigned with enough workers in the previous batch and new tasks that appear in the current batch. Additionally, valid workers contain those that are not assigned to any task in the previous batches and newly appeared in the current batch. To reduce the time-consuming for some invalid task-and-worker pairs, we select candidate workers, which are denoted as CWS, according to the skill limit and range constraint (Lines

Table 2
Symbols and Descriptions.

Symbols	Descriptions
t_i	The task t_i in T
w_j	The worker w_j in W
T_p	A batch timestamp
T	The number of time-constrained spatial tasks at timestamp T_p
W	The number of moving workers at timestamp T_p
L_{t_i}	The location of task t_i
L_{w_j}	Location of worker w_j
K_{t_i}	Skill requirement of task t_i
K_{w_j}	Skills equipment of worker w_j
B_{t_i}	Budget of task t_i
R_{w_j}	Range area of worker w_j
v	Unit travel cost of workers
$dist(t_i, w_j)$	Distance between the location of task t_i and worker w_j
W_c	Conflict workers competed by more than one task
W_i	The worker set is assigned to the task t_i

5–6). Among the available tasks T, available workers Win the timestamp T_p, we utilize our proposed algorithms, including the CAG algorithm and the GT algorithm, for achieving two good assignment results with a high overall satisfaction score (Line 7). At the end of each assignment batch, we inform all workers in W_i to conduct task t_i according to the task assignment result (Lines 8–9).

It is noted that, for the unassigned tasks and workers in the current batch, they will be left in the next batch. Further, new workers and tasks are emerged with unassigned workers and tasks for a new assignment result in the next batch. On the other hand, if the task cannot be assigned before its expiration, the task will be rejected form the SC platform.

5. The greedy approach

We first show the definition of satisfaction increment which is comprised of cooperation increment score and price increment score in Section 5.1. Then, we propose the *conflict-aware greedy* (CAG) algorithm to solve the SATA problem, detailed in the form of pseudo-code in Section 5.2. Finally, we illustrate the proof that the CAG algorithm is submodular and give the corresponding approximation bound in Section 5.3.

5.1. The satisfaction increment

Before we present the CAG algorithm, it is necessary to measure the satisfaction increment when a single worker is assigned to a task. The satisfaction increment is indicated by ΔS, consisting of two subsections, cooperation increment score ΔC and price increment score ΔP, as follows,

$$\Delta S(w_j, t_i) = \frac{\alpha}{P_{max}} \cdot \Delta P + \frac{1-\alpha}{C_{max}} \cdot \Delta C. \tag{6}$$

The price increment score is the increased score when w_i is added into task t_i as

$$\Delta P = P(t_i, W_i \cup \{w_j\}) - P(t_i, W_i), \tag{7}$$

where $P(t_i, W_i \cup \{w_j\})$ is the price satisfaction when w_j is added into the assigned worker set W_i of task t_i, and $P(t_i, W_i)$ is the price satisfaction of worker set W_i. The calculation of price satisfaction score is presented in Eq. (3).

The cooperation increment score is the increased score when w_j is added to W_i, as

$$\Delta C = C(W_i \cup \{w_j\}) - C(W_i). \tag{8}$$

where the calculation of cooperation score is shown in Eq. (2).

5.2. The greedy algorithm

Algorithm 2: Conflict-aware Greedy Algorithm

Input: A task set T and a worker set W
Output: Assignment $\mathcal{A}_p$

1 initialize $\mathcal{A}_p \leftarrow \emptyset$;
2 **foreach** $t_i \in T$ **do**
3 select candidate workers under the constraints of skill matching and working range among all workers as the CWS;
4 find a set of workers W_i^* with a higher satisfaction score for task t_i;
5 **while** *there is a task pair compete for the same worker set* **do**
6 let W_c be the conflict workers among W_i^* and W_j^* of task t_i and t_j, repectively;
7 **foreach** w_c *in* W_c **do**
8 $\Delta S_i \leftarrow S(t_i, W_i^*) - S(t_i, W_i^* \setminus \{w_c\})$;
9 $\Delta S_j \leftarrow S(t_j, W_j^*) - S(t_j, W_j^* \setminus \{w_c\})$;
10 **if** $\Delta S_i > \Delta S_j$ **then**
11 $W_j^* \leftarrow W_j^* \setminus \{w_c\}$;
12 **else**
13 $W_i^* \leftarrow W_i^* \setminus \{w_c\}$;
14 $W_c \leftarrow W_c \setminus \{w_c\}$;
15 **if** W_i^* *is able to complete* t_i **then**
16 $T \leftarrow T \setminus \{t_i\}$;
17 $\mathcal{A}_p \leftarrow \mathcal{A}_p \cup \bigcup_{w_i \in W_i^*} \{(w_i, t_i)\}$;
18 $W \leftarrow W \setminus \{W_i^* \cup W_j^*\}$;
19 **while** $W \neq \emptyset$ *or* $T \neq \emptyset$ **do**
20 select a task-and-worker pair (w_j, t_i) with the highest satisfaction increment;
21 $\mathcal{A}_p \leftarrow \mathcal{A}_p \cup \{(w_j, t_i)\}$;
22 $W \leftarrow W \setminus \{w_j\}$;
23 **if** W_i *satisfy the complete condition of* t_i **then**
24 $T \leftarrow T \setminus \{t_i\}$;
25 **return** $\mathcal{A}_p$

In this part, we present a two-stage conflict-aware greedy approach shown in Algorithm 2. In the first stage, we assign a set of workers which is a combination worker set to each task with a high satisfaction score and delete the conflict worker set of task pairs (Lines 2–18); in the second stage, we iteratively select a new task-and-worker pair with the highest satisfaction increment score into the final assignment (Lines 19–24).

Specifically, we first initialize $\mathcal{A}_p$ as the optimal task assignment result (Line 1). Next, we move out invalid candidate workers for all tasks in advance who is unsatisfied the skill requirement and work range limitation (Line 3). Then, we calculate the *best* set of workers to each task with the highest satisfaction without taking into consideration the conflict constraint, allowing one worker can be assigned to multiple tasks (Line 4). However, according to the conflict constraint shown in Definition 6, one worker can only be assigned to one task before he is released again. By comparing the assigned worker set of any two tasks, the workers assigned to multiple tasks simultaneously are named conflict worker set in our paper, which is denoted as W_c. Next, we delete the conflict worker among all task pairs (i.e., any two tasks in the task set) (Line 7–14). Particularly, we obtain the satisfaction increment score of the task-and-worker pair of task t_i and task t_j(e.g., $\Delta S_i = \Delta S(w_c, t_i)$ and $\Delta S_j = \Delta S(w_c, t_j)$) (Lines 8–9). For achieving a high overall satisfaction score, we allow the task-and-worker pair with a higher satisfaction increment is eligible to keep the conflict worker w_c(e.g., $w_c \in W_c$) and delete w_c from the worker set of another task. That is, in the case that $\Delta S_i > \Delta S_j$, we assign the conflict worker to t_i and delete w_c from the assigned worker set of t_j; otherwise, we assign the conflict worker to t_j and delete w_c from the assigned worker set of t_i(Lines

10–13). Moreover, if $\Delta S_i = \Delta S_j$, which means w_c achieves the same satisfaction increment for t_i and t_j, respectively, this conflict worker is assigned randomly to one of the task pair. After that, if the assigned worker set can complete corresponding task, we delete the task from the unassigned task set T, remove the assigned workers from $W$and obtain the assignment result of the first stage to $\mathscr{A}_p$ (Lines 15–18).

In the second stage, we select the task-and-worker pair with the highest satisfaction increment score from the remaining workers and tasks (Line 20). The pair with the highest satisfaction increment is added into $\mathscr{A}_p$(Line 21). In addition, if enough workers are assigned to t_i, it will be removed from the unassigned task set (Lines 23–24). Until all tasks have been assigned enough workers or all valid workers have been assigned, we return the assignment result and inform the workers to complete tasks (Line 25). Noted that, in the second stage, we always assumed that the task-and-worker pair could be added to the final assignment result. If there no pair is valid for the assignment, the current batch assignment will end up as well.

We give the following example to better understand the CAG method.

Example 2. There are two tasks t_1 and t_2 and six workers $w_1, \ldots, w_6$. If we don't consider the duplicate constraint, we assume the worker set $\{w_1, w_2, w_3\}$ is assigned to t_1 and $\{w_1, w_4, w_5\}$ is assigned to t_2 can achieve the highest satisfaction score. Comparing these two tasks, worker w_1 is the conflict worker. Next, we compare the satisfaction increment score of task-and-worker pairs. If $\Delta S(w_c, t_1) > \Delta S(w_c, t_2)$, we allow w_1 is assigned to t_1 and delete w_1 from the assigned worker set of t_2. Finally, after deleting all conflict workers among all tasks (i.e., there are multiple conflict workers in W_c), the assigned result is $\{w_1, w_2, w_3\}$ to t_1 and $\{w_4, w_5\}$ to t_2. Next, we choose the task-and-worker pair with the highest increment satisfaction score. Thus, we assign w_6 to t_2. Therefore, the final assignment result is the worker set $\{w_1, w_2, w_3\}$ assigned to t_1 and $\{w_4, w_5, w_6\}$ is assigned to t_2.

5.3. Analysis of the greedy algorithm

In this part, we give the time complexity and approximate bound of our CAG algorithm.

5.3.1. Time complexity

We assume that there are n available tasks, $m$available workers, and every task is valid for $\hat{m}$ workers. Further, each task requires k workers to complete. Specifically, for each task in T, selecting all candidates each task requires $O(nm)$(Line 2–3). Then, greedily finding the set of workers with the highest satisfaction score, it needs $O(\hat{m})$(Lines 5). For the conflict part, it needs at most $O(n^2\hat{m})$ since there are at most $n(n-1)/2$ task pairs need to be compared and at most $\hat{m}$ conflict workers for each conflict task pair which requires $O(1)$ for deleting conflict worker. Checking the assigned worker set satisfies the completion condition also requires $O(1)$(Lines 15–18). Overall, it requires at most $O(nm + n^2\hat{m})$(Lines 3–18). Next, there are at most $n\hat{m}$ task-and-worker pairs of the remaining workers and tasks. Moreover, including the best pair into the assignment result and checking the valid worker set requires $O(1)$. Thus, it needs $O(n\hat{m})$ in the second stage (Lines 19–24). Therefore, the total time complexity of CAG algorithm is $O(nm + n^2\hat{m}) + O(n\hat{m})$.

5.3.2. Quality

Let $\Delta S(w, t)$ be the satisfaction increment of task-and-worker pair in assignment $\mathscr{A}_p$, then the objective function could be rewritten as $\mathrm{Sum}(A) = \sum_{t_i \in T, W_i \in W} S(t_i, W_i) = \sum_{(w,t)\in A} \Delta S(w, t)$. The satisfaction of each assigned task is a number greater than zero, thus, $\Delta S(w, t)$ is always positive, and $\mathrm{Sum}(A)$ is monotone.

Lemma 1. If $\forall \langle w_s, t_s\rangle \notin A, \forall \langle w_j, t_j\rangle \notin A$, and $\mathrm{Sum}(\widehat{A}) - \mathrm{Sum}(\widetilde{A}) \leqslant \mathrm{Sum}(\overline{A}) - \mathrm{Sum}(A)$, $\mathrm{Sum}(A)$ is submodular.

Proof. Let $\widehat{A} = A \cup \{\langle w_s, t_s\rangle, \langle w_j, t_j\rangle\}, \widetilde{A} = A \cup \{\langle w_s, t_s\rangle\}$ and $\overline{A} = A \cup \{\langle w_j, t_j\rangle\}$. Although $\langle w_j, t_j\rangle$ is assigned both in $\widehat{A}$ and $\overline{A}$, we assume the satisfaction score increment (i.e., $\Delta S(w_j, t_j)$) of worker w_j in $\widehat{A}$ is larger than the score in $\overline{A}$. And then, when the worker w_s is assigned to the task t_s, we move out w_s and t_s from $W$and T, respectively. The satisfaction score of w_j(i.e., $\Delta S(w_j, t_j)$) will not be affected. Thus, the $\Delta S(w_j, t_j)$ of $\langle w_j, \Delta S(w_j, t_j)\rangle$ in $\overline{A}$ is available and $\mathrm{Sum}(\overline{A}) - \mathrm{Sum}(A) = \Delta S(w_j, t_j)$. The workers $\overline{W_p}$ is the subset workers of W_p, where $\overline{W_p}(W_p)$ is the available workers of the assignment after (before) completing the task t_j(i.e., $\overline{W_p} \subseteq W_p$). If t_j can be finished by the workers in W_p but cannot be completed by the workers in $\overline{W_p}$, t_j will not be assigned in $\widehat{A}$. Then, $\mathrm{Sum}(\widehat{A}) - \mathrm{Sum}(\widetilde{A}) = 0$, but $\mathrm{Sum}(\overline{A}) - \mathrm{Sum}(A) = \Delta S(w_j, t_j)$. In conclusion, $\mathrm{Sum}(A)$ is monotone and submodular. Thus, Lemma 1 has been proven.

Based on the above analysis, we have proved that the CAG algorithm is monotone and submodular. According to [41], our greedy-based algorithm, CAG, can obtain an approximate bound, that is, $(1 - \frac{1}{e}) \cdot S(T^*)$, where the $S(T^*)$ is the optimal satisfaction in the SATA problem.

6. The game theory approach

For solving the SATA problem, we propose the CAG algorithm for a good assignment result. However, the workers prefer to choose their own optimal task proactively for a higher utility by themselves in real applications. Based on this, we develop a game theory (GT) algorithm allowing workers to choose their task proactively for the best single utility to achieve the goal of the SATA problem.

In this section, we first give the general definition of the game theory. The *game theory* (GT) approach of our SATA problem is illustrated in Algorithm 3. In the end, we analyze the result of Nash equilibrium from three aspects, existence, quality, and convergence.

6.1. The game theory

The game theory model has been widely applied in economics, politics, computer science, even in law, biology, and sport. In a strategy game, there are three essential parts—players, strategy space, and the payoff. For each player *i*of the players set *N*, the player chooses his dominant strategies in the strategy space *S*, which corresponds to utility u_i of the payoff set *U*. In our paper, each player is allowed to choose one strategy in a time. In other words, when the player has his best strategy s_i^* and the strategy set of other players is s_{-i}, and the utility of player *i* satisfies the following condition:

$$U(s_i^*, s_{-i}) \geqslant U(s_i, s_{-i}), \tag{9}$$

where s_i^* denotes the optimal strategy of player *i*, s_i denotes one of other strategies except for the optimal strategy of player *i*, and s_{-i} is the union strategy set of other players excluding player *i*, which is denoted as $s_{-i} = \{s_1 \times s_2 \times \cdots \times s_{|n|}\}$, the Cartesian product of the actions of all other players. When all players choose their best strategy or best response, the Nash equilibrium can be reached in the whole picture, which is also a pure strategy game. According to [42], there is a pure Nash equilibrium in the pure strategy game if and only if all players choose the best response in their dominant strategy space.

6.2. The game theoretic algorithm

Algorithm 3: Game Theoretic Algorithm

Input: A set of tasks T and a set of workers W;
Output: Assignment $\mathcal{A}_p$

```
apply the CAG algorithm to obtain the initial assignment;
initialize the strategy of workers in W;
while Nash Equilibrium is not reached do
    foreach w_j ∈ W do
        \* select the best response task t_i to w_j;
        U_max ← −∞;
        for t_i in T_cj do
            compute the Utility(w_j, w_−j) of w_j;
            if U(w_j, w_−j) > U_max then
                U_max ← U(w_j, w_−j);
        \* assign the worker w_j to task t_i;
        s_wj ← t_i;
return A_p
```

In this section, we model a SATA problem instance as a strategic game and propose a game theoretic (GT) approach based on the best-response framework to find a Nash equilibrium, where every worker is assigned to his "best" task such that a high total satisfaction score is achieved. Specifically, we model each worker w_i as a player *i*, whose target is to select a "best" task with the highest satisfaction score (utility) for him. For each player w_i, his strategy set S_i indicates all possible strategies that he can choose (e.g., all the valid tasks he can conduct). Then, a joint strategy *S* for the strategic game corresponding to an assignment $\mathcal{A}_p$ of the SATA problem instance.

The SATA strategy game G, which is consists of three parts, W, T_{cj} and ΔS. Specifically, each available worker in $W$is modeled as player i, whose strategy in each iteration is the candidate tasks T_{cj} filtered by skill constraints and range constraints, and they compete with other workers for the highest utility ΔS. For every player in our SATA game, his goal is to maximize his utility, which is related to the objective function of our SATA problem. Thus, we obtain the utility of each player which is expressed as follows,

$$U(s_i, s_{-i}) = \Delta S(w_j, t_i) = S(W_i \cup w_j) - S(W_i), \tag{10}$$

where $\Delta S(w_j, t_i)$ is shown in Eq. (6). The utility of the worker is the increased satisfaction score when he participate in the assigned task.

We propose a basic game theoretic approach, shown in Algorithm 3, to achieve a Nash equilibrium for a set of workers W. Specifically, we first apply the CAG approach to achieve an initial assignment and initialize the strategy of all workers (Lines 1–2). Secondly, in each round, we iteratively adjust each worker's strategy to his best-response strategy that maximizes his utility function U_i, which is defined in Eq. (10). The Nash equilibrium is reached when all workers select their best-reponse strategy (Lines 5–11). Particularly, set the maximal utility of each worker w_j to be the infinitesimal (Line 5). Next, traverse all dominant strategies in w_j's strategy space and give the corresponding utility (Line 7). By comparing the utility of all strategies, determining the best response of w_j (Lines 7–10). After that, set the best response task of w_j as the t_i(Lines 11–12). It is noted that although all workers choose their best response strategy, the available task-and-worker sets are eligible to achieve a satisfaction score. Accordingly, the strategy choices of some workers are meaningless since it does not contribute to the overall satisfaction score. For achieving a higher satisfaction score, these workers need to choose other tasks in the next rounds. Furthermore, the best response task of players also changed in the next round because the assigned worker set of the task is not fixed in each round. Therefore, it requires multiply rounds for reaching Nash equilibrium. Finally, we obtain the best assignment result which is the Nash equilibrium where all workers choose their best response (Line 13). Noted that, if two players compete for the same best strategy and achieve the same utility in our work, randomly choose one of them to obtain the strategy as the best response. Here, each iteration of the WHILE loop (Lines 3–14) is called a round.

We give the following example to better illustrate the GT algorithm.

Example 3. Workers w_1, w_2, and w_3 are assigned to t_1, thus, the strategy of w_1, w_2, and w_3 is t_1. w_4 is an unassigned worker as a player in the SATA strategy game model. We assume task t_1 is the best strategy to w_4 in the current round and w_4 and w_2 have the same effective skill for the task t_1. Next, calculate the utility of w_4(e.g., $U(s_{w_4}, s_{-i}) = S(\{w_1, w_4, w_3\}, t_1) - S(\{w_1, w_2, w_3\}, t_1)$). If $U(s_{w_4}, s_{-i}) > 0, w_4$ is assigned to t_1 and turns to the assigned state. Finally, w_2 turns to the unassigned state and joins the next game round as the new player to choose other tasks.

6.3. Analysis of the game theoretic algorithm

The Nash equilibrium analysis of the GT algorithm is divided into three parts: existence, quality, and convergence. The Nash equilibrium that exists in the SATA problem is proven by formulating it into a potential game. Next, we show a provable quality bound of our proposed GT algorithm. Finally, we prove the GT approach will converge in the limited rounds and the time complexity is presented.

6.3.1. Existence

In this section, we give the proof that our SATA strategy game is an exact potential game if and only if we could find a potential function to combine the utility of the player with the global optimization in SATA problem, as

$$U(s_i^*, s_{-i}) - U(s_i, s_{-i}) = Q(s_i^*, s_{-i}) - Q(s_i, s_{-i}), \tag{11}$$

where s_i^* represents the best response in his dominant strategy space of w_i and s_i is another strategy. s_{-i} is the joint strategy set of other players except for w_i. When the worker changes his best strategy s_i^* to another strategy s_i in the potential game, the utility difference in the shift movement will change in the same way as the potential function value does. Accordingly, we give the proof that our SATA strategy game is a potential strategy game in the following theorem.

We define the potential function of the SATA game as

$$Q = \sum_{i=1}^{|T|} S(t_i, W_i), \tag{12}$$

where $S(t_i, W_i)$ is the user satisfaction score of t_i. The potential function is defined as equal to the objective function of our problem as Eq. (2).

Theorem 2. The SATA strategy game is a potential game.

Proof. We note strategies s_i^* and s_i as w_i chooses tasks t_j and t_k, respectively. s_{-i} indicates a given joint strategy for excepting t_j and t_k. Then we have:

$$
\begin{aligned}
& Q(s_i^*, s_{-i}) - Q(s_i, s_{-i}) \\
= & \left(S(W_j \cup \{w_i\}) + S(W_k) + \sum_{t_s \in T - \{t_j, t_k\}} S(W_s) \right) \\
& - \left(S(W_j) + S(W_k \cup \{w_i\}) + \sum_{t_s \in T - \{t_j, t_k\}} S(W_s) \right) \\
= & \frac{\alpha}{P_{max}} \cdot \left(P(t_j, W_j \cup \{w_i\}) + P(t_k, W_k) + \sum_{t_s \in T - \{t_i, t_k\}} P(t_s, W_s) \right) \\
& + \frac{(1-\alpha)}{C_{max}} \cdot C(W_j \cup \{w_i\}) + C(W_k) + \sum_{t_s \in T - \{t_i, t_k\}} C(W_s) \\
& - \frac{\alpha}{P_{max}} \cdot \left(P(t_j, W_j) + P(t_k, W_k \cup \{w_i\}) + \sum_{t_s \in T - \{t_i, t_k\}} P(t_s, W_s) \right) \\
& - \frac{(1-\alpha)}{C_{max}} \cdot \left(C(W_j) + C(W_k \cup \{w_i\}) + \sum_{t_s \in T - \{t_i, t_k\}} C(W_s) \right) \\
= & \frac{\alpha}{P_{max}} \cdot \left(P(t_j, W_j \cup \{w_i\}) - P(t_j, W_j) \right) + \frac{(1-\alpha)}{C_{max}} \cdot \left(C(W_j \cup \{w_i\}) - C(W_j) \right) \\
& - \left(\frac{\alpha}{P_{max}} \cdot \left(P(t_k, W_k \cup \{w_i\}) - P(t_k, W_k) \right) + \frac{(1-\alpha)}{C_{max}} \cdot \left(C(W_k \cup \{w_i\}) - C(W_k) \right) \right) \\
= & S(W_j \cup \{w_i\}) - S(W_j) - \left(S(W_k \cup \{w_i\}) - S(W_k) \right) \\
= & U(s_i^*, s_{-i}) - U(s_i, s_{-i}),
\end{aligned}
$$

where the calculation of $S(W_j), S(W_j \cup \{w_i\})$, and $\sum_{t_s \in \{T - \{t_j, t_k\}\}} S(W_s)$ are shown in Eq. (2).

Based on the theory of potential games [43], we achieve the conclusion that when all players choose their best-response based on the GT algorithm can finally reach Nash equilibrium since our SATA strategy game is also a potential game model. Therefore, the strategic game of the SATA problem can find a Nash equilibrium after all players choose their best strategy after multiple game rounds.

6.3.2. Quality

In the strategy game, there may be more than one Nash equilibrium among the equilibriums. Thus, we propose three measures [3] to assess the quality of the Nash equilibrium of SATA strategy game: 1) social optimum (OPT); 2) price of stability (PoS); 3) price of anarchy (PoA). OPT refers to the result that achieves the global optimal strategy joint quality, in which the given objective optimization (e.g., the total utility or cost) of the proposed problem is maximized or minimized. It is also the goal of the related optimization problem. PoS represents the ratio of the best equilibrium utility among all Nash equilibriums to the OPT (i.e., the best utility of equilibrium/OPT). PoA indicates the ratio of the worst equilibrium among all Nash equilibriums to OPT (i.e., the worst utility of equilibrium/OPT). Moreover, PoS reflects the upper bound of the equilibrium while PoA reflects the lower bound, which is the ratio of the achieved equilibrium utility to the OPT. We show the upper bound of PoS and the lower bound of PoA as follows.

Theorem 3. In the pure strategy game of the SATA problem, the upper bound of PoS is 1 while the lower bound of PoA is $|T| \cdot \left(\alpha \cdot \frac{P(t_i, W_i)_{min}}{P_{max}} + (1 - \alpha) \cdot \frac{C(W_i)_{min}}{C_{max}} \right) / OPT$.

Proof. Let T denotes the strategy joint of the potential strategy game, and $S(T)$ denotes the overall satisfaction of the objective function of SA, which equals the total utility of SATA strategy game as Eq. (12) (*i.e.*, $S(T) = Q(T)$). We set T^* as the optimal strategy joint, $\widehat{T}$ as the strategy joint assignment of the best equilibrium, and $\widetilde{T}$ as the strategy joint assignment of the worst equilibrium. Thus, we have $S(T^*) = Q(T^*), S(\widehat{T}) = Q(\widehat{T})$, and $S(\widetilde{T}) = Q(\widetilde{T})$. For estimating the lower bound of PoA, we must first analyze the lower bound of the global satisfaction score achieved by an equilibrium. In each game round, the total satisfaction score will increase when players change their strategy. Thus, if the strategy game reaches Nash equilibrium, the assignment result must be better than the initial assignment satisfaction score. We have

$$\begin{aligned} S(\widetilde{T}) &\geqslant \sum_{t_i \in T} \widetilde{S}(t_i, W_i) \\ &\geqslant \sum_{t_i \in T} \left(\alpha \cdot \frac{P(t_i, W_i)_{min}}{P_{max}} + (1-\alpha) \cdot \frac{C(W_i)_{min}}{C_{max}} \right) \\ &= |T| \cdot \left(\alpha \cdot \frac{P(t_i, W_i)_{min}}{P_{max}} + (1-\alpha) \cdot \frac{C(W_i)_{min}}{C_{max}} \right), \end{aligned}$$

where $\widetilde{S}(t_i, W_i)$ is the lowest satisfaction among all assigned tasks, and $|T|$ means the number of assigned tasks in the game. C_{min} is the lower bound of cooperation satisfaction score among all tasks. P_{min} is the lower bound of price satisfaction when the workers who are assigned to the task have the maximal total travel cost. Thus,

$$PoA = \frac{S(\widetilde{T})}{S(T^*)} \geqslant \frac{|T|}{OPT} \cdot \left(\alpha \cdot \frac{P(t_i, W_i)_{min}}{P_{max}} + (1-\alpha) \cdot \frac{C(W_i)_{min}}{C_{max}} \right).$$

For the upper bound of *PoS*, we have $Q(T^*) = S(T^*)$ and $Q(\widehat{T}) = S(\widehat{T})$. We also know that $Q(T^*) \geqslant Q(\widehat{T})$ since the optimal assignment quality is always superior to the best equilibrium result, as seen in $Q(T^*) = S(T^*) \geqslant S(\widehat{T}) = Q(\widehat{T})$. Consequently, we have

$$PoS = \frac{S(\widehat{T})}{S(T^*)} \leqslant 1.$$

The theorem holds.

6.3.3. Convergence

To obtain the convergence speed of the GT approach, it is necessary to know the time complexity of each round and how many rounds it needs for reaching the Nash equilibrium. For answering these questions, we give some lemmas as follows and the proofs are presented in what follows.

Lemma 2. The GT approach reaching the pure Nash equilibrium requires at most $Q_z(T^*)$ rounds, which is a scaled potential function (i.e., $= d \cdot Q(T^*)$, where $Q(T^*)$ is the optimal satisfaction of SATA strategy game, and T^* is the optimal strategy joint).

Proof. The GT algorithm reaches the Nash equilibrium when all workers select their best strategy, and no worker tends to deviate from his current strategy in the last round. For the potential game, when the player changes his strategy, the improvement of a single player's utility is equal to the increasement of the potential function. In each round of our potential game, there is at least one worker switching from his current strategy s_i to a better strategy s_i^* in order to improve utility by at least 1 unit utility in the scaled potential function $(i.e., Q_z(s_i', s_{-i}) - Q_z(s_i, s_{-i}) \geqslant 1)$. $Q_z(T^*)$ is an integer utility to scaled potential function. Thus, the improvement of the potential function is always positive and the upper bound of the rounds is $Q_z(T^*)$.

Lemma 3. The time complexity in each round is $O(m\hat{n})$ and the time complexity of the GT algorithm is $O(m\hat{n}Q_z(T^*))$.

Proof. Assuming there are *m*workers and each worker has $\hat{n}$ tasks in his dominant strategy space. In each round, every player needs to choose their best response. For each player, finding his best response strategy requires at most $\hat{n}$ times for computing the individual utility in the corresponding strategy space. Thus, the total time cost of computing individual utility is $O(m\hat{n})$. We have already proved the number of the game round is $O(Q_z(T^*))$ as Lemma 2. Consequently, the total time complexity of GT algorithm is $O(m\hat{n}Q_z(T^*))$.

6.4. Optimization strategy

Workers in the GT algorithm cannot improve their utility by unilaterally changing their strategy since each worker is allowed to keep adjusting the strategy until a Nash equilibrium is reached. Therefore, the GT algorithm may be slow when solving SATA problem instances on large scale. Then, a threshold stop (TS) optimization strategy is proposed to improve the efficiency of the GT algorithm.

Threshold Stop (TS) Optimization Strategy: The GT algorithm is an iteration algorithm. All workers will go through multiple rounds of competition until their strategy stabilizes before reaching the Nash equilibrium, which means the GT algorithm can interrupt at any round and a valid score still be returned. GT is expected to achieve a better total satisfaction score than the last round at the end of each iteration. In our experiment, A truth is revealed that the increment of total satisfaction tends to get smaller and smaller until it reaches zero in the final round. For real applications, a value closer to the optimal

result but quickly obtained is more acceptable. Thus, we propose an optimization strategy for stopping the iteration when total satisfaction reaches an acceptable value. Specifically, if the satisfaction score increment is smaller than $\gamma \cdot S_c$, the iterations are stopped, where γ is a given parameter and S_c denotes the total satisfaction scores of the last round. Demonstrated by our experiments, the TS optimization strategy can reduce the running time of GT approach but only at the slightly cost of total satisfaction score.

7. Experiments

This section illustrates the experimental results of our approaches to show effective performance through synthetic data sets and real data sets. Table 3 shows the setting value of all parameters, and we bold the default values.

7.1. Experimental setup

We test the proposed approaches of the satisfaction-aware task assignment problem on both real datasets and synthetic datasets, particularly in three aspects, i.e. the running time of the CPU, the number of assigned tasks and the total satisfaction score of the assignment. The running time reflects the efficiency of the proposed algorithms. The satisfaction score and number of assigned tasks show the effectiveness of the assignment. Additionally, we change only one parameter and others are kept in the default value on each experiment. All algorithms are implemented on an Intel Core i7-8565U CPU @ 1.99GHZ with 16 GB RAM in Python 3.6.

The real data set we applied is published on the Meetup website.[4] There are millions of users and location-sensitive events, and ten thousands of groups in Meetup, where each user reports a specific location, each group tends to attract a group of users to participate, and each event is associated with a location to be released. We only allow workers to move to the location of tasks within the same city since workers are unwilling to move to another city to perform the task, preventing the higher traveling costs. Specifically, we select one popular meetup city, California, and extract Meetup records from the area of California, and we randomly choose 500 tasks and 1000 workers from the data. Each event has its topic with several keywords, which are associated with the skill requirement of the task. Each member of the meetup has an interesting topic for the event, which is regarded as the skill of the worker. Further, all events and workers release their specific location. The system assigns the proper workers to the suitable tasks based on spatial–temporal information.

In synthetic data, we randomly generate tasks and workers in two-dimensional space $[0, 1]^2$ and set activity ranges of workers into circles. For simplicity, we also normalize the location information to $[0, 1]^2$ space in Max–Min normalization for both workers and tasks. To estimate the cooperation score, we use information of the number workers in the same groups where they participated in. The cooperation score of any two workers could be calculated by Eq. (1) where Gdenotes the total number of groups that workers had participated in, the basic cooperation score (i.e., ω) is 0.5, and the linear coefficient (i.e., β) is 0.5. We propose the Random algorithm, Greedy algorithm, Simulated Annealing (SA) algorithm and MIP method as baseline methods in our paper. The main idea of the greedy approach is greedily choosing the worker set that achieves the most satisfaction score for each task. In the random method, workers are chosen at random for each task according to their skill constraints. The simulated annealing approach is a heuristic algorithm that performs well in [8,9] as the benchmark method. The MIP algorithm is the standard algorithm for solving the mixed integer problem.

Next list a comparison between our approaches and the baseline algorithms, including the MIP, Random, Greedy, SA, CAG, GT, and GT + TS in the following:

- **MIP**: The standard method for solving the mixed integer problem [44].
- **Random**: The baseline algorithm that randomly selects the workers in the candidate worker set.
- **Greedy**: The baseline algorithm that greedily chooses the worker set with the maximal satisfaction iteratively.
- **SA**: The simulated annealing heuristic algorithm is introduced from [8,9].
- **CAG**: The conflict-aware greedy method introduced in Section 5.
- **GT**: The game theoretic approach shown in Section 6.
- **GT + TS**: The game theory approach with the threshold stop (TS) optimization strategy for GT.

7.2. Experiment result

In this section, we show the experimental results on synthetic datasets and real datasets to demonstrate the efficiency and effectiveness of our proposed approaches.

7.2.1. Experiments on synthetic datasets

Here we evaluate our approaches through the satisfaction score, number of tasks assigned and the running time on synthetic data sets that include a variety of parameters, such as the number of tasks, the number of workers, and the threshold parameter of TS strategy.

[4] https://www.meetup.com/.

Table 3
Experiment setting.

Parameter	Values
Number of task	300, **500**, 800, 1000, 1200
Number of worker	500, 800, **1 k**, 1500, 2000
Unit cost of worker	10, **20**, 30, 40
Task budget	[5,10], [10,15,20,25]
Range of worker	0.05, **0.1**, 0.15, 0.2
Skill of task	**3**, 4, 5, 6
Parameter of γ	0.01, 0.03, **0.05**, 0.08
Parameter of α	**0.5**

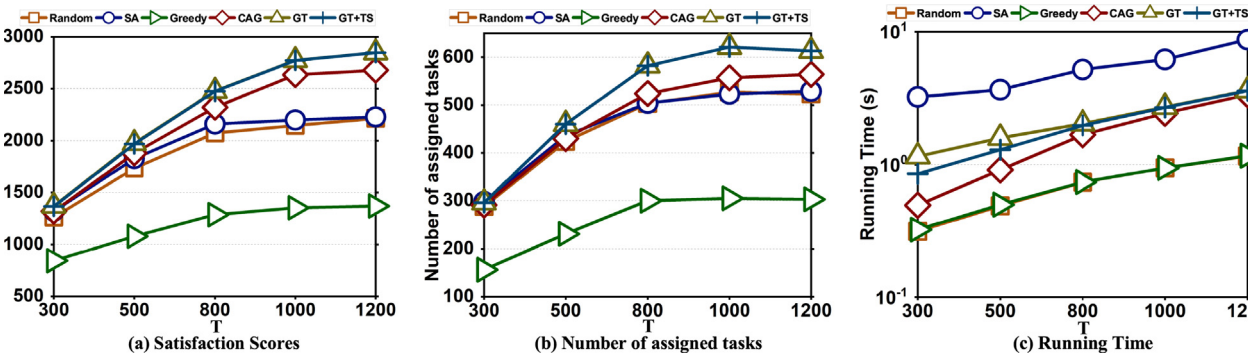


Fig. 2. Effect of the number of task.

Effect of the number of tasks. As the number of tasks increases from 300 to 1200, Fig. 2 shows the running time and overall satisfaction score for all tested approaches. All other parameters are set to their default values. In Fig. 2(a), it is noticeable that the satisfaction score of all methods achieves higher satisfaction score when the number of tasks varies from 300 to 1000; then, the total satisfaction score does not change much as the number of tasks in each batch grows from 1000 to 1200. It occurs because at first, with the increase number of tasks, more workers devoted to enough tasks, resulting in more assignments with higher satisfaction scores. However, the maximum number of tasks is limited. When the number of tasks reaches 1000, enough workers have been assigned to all tasks. Therefore, more tasks do not result in significantly higher satisfaction scores once the number of tasks reaches 1000. The satisfaction score of baseline methods, Greedy, Random, and SA are lower than that of CAG, GT, and GT + TS. SA achieves the best satisfaction score among all baseline methods while GT has the highest satisfaction score among all proposed approaches. Fig. 2(b) presents the number of tasks assigned, which is increasing first and keep a constant later. That is, more tasks are available and the number of workers is fixed. GT and GT + TS achieve the largest number of tasks assigned of all proposed approaches. In Fig. 2(c), the running time of all approaches increases as the number of tasks increases. Because it takes time to compare more task-and-worker pairs and choose an appropriate task for each worker as the number of tasks increases.

Effect of the number of workers. As shown in Fig. 3, the varying number of workers is between 500 and 2000 for synthetic data sets while all other parameters are set as defaults. When increasing the number of workers, satisfaction scores of all proposed methods increases at the same time. As demonstrated in Fig. 3(a), the total satisfaction increases quickly if we enlarge the size of workers from 500 to 1500 and gradually be slow when exceeds 1500. This is because more workers are able to be assigned to more tasks for achieving noticeable increment of satisfaction scores at the begin. However, when the number of workers exceeds 1500, workers are already enough to the fixed number of tasks. Similar satisfaction scores are achieved of GT and GT + TS compared to other approaches but higher than Random, Greedy, and SA. In Fig. 3(b), GT and GT + TS achieve the highest number of tasks assigned than other baseline methods. Besides, CAG, SA, Random have s similar number of tasks assigned and higher than Greedy. Fig. 3(c) shows that as the number of workers increases, all approaches require more time, which is because more task-and-worker pairs are to be tested by all methods. SA requires the most time cost among all proposed methods due to the multiple iteration times. GT and GT + TS require more time as the number of workers exceeds 1000. The reason is that more workers as the players in GT and GT + TS, it costs more time to find the best tasks (strategies). GT + TS is quicker than GT, which demonstrates the optimization strategy is effective.

Effect of the threshold parameter γ. As shown in Fig. 4, to demonstrate the effect of the threshold parameter γ of the Threshold Stop optimization method for GT, we present the results of GT + TS with different threshold parameters ranging from 0 to 0.08. As shown in Fig. 4(a), different threshold parameters for GT + TS achieve similar satisfaction scores. In particular, satisfaction scores decrease significantly when $\gamma = 0.08$. In Fig. 4(b), the running time of GT + TS decreased when increasing the value of parameter γ, which is because the iteration is lesser when expanding the value of γ.

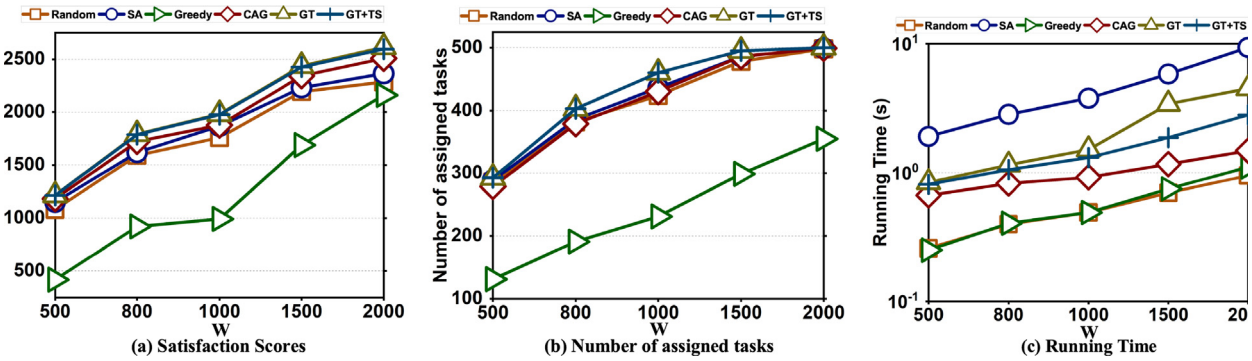


Fig. 3. Effect of the number of workers.

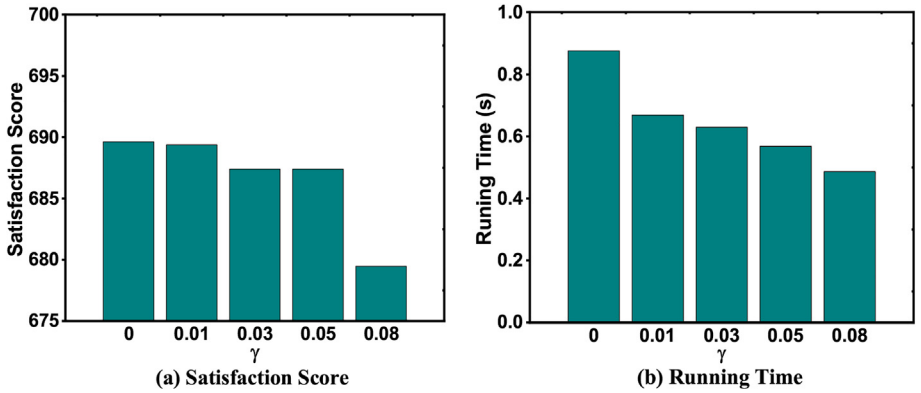


Fig. 4. Effect of the threshold parameter.

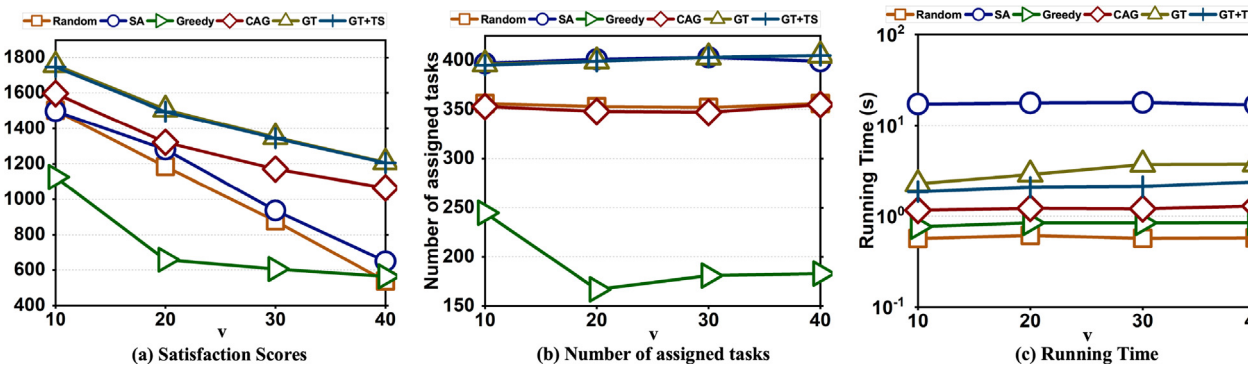


Fig. 5. Effect of the unit cost of workers.

7.2.2. Experiments on real datasets

In this section, we present the efficiency of our algorithms on real datasets in task budget, the range of workers, and the complexity of the task.

Effect of the unit cost of workers. Fig. 5 presents the trend of all tested algorithms when the unit cost varies from 10 to 40. In Fig. 5(a), the satisfaction scores decrease when the unit cost of traveling distance increase. The reason is that the higher the travel cost of workers, the lower the price satisfaction scores. GT, GT + TS, and CAG achieve a higher satisfaction score than Greedy, Random, and SA, which illustrates the efficiency of our proposed methods. Fig. 5(b) shows the number of assigned tasks of GT, GT + TS and SA achieve the greatest number of assigned tasks than that of Random, CAG and Random. In Fig. 5(c), the running time remains stable when the unit cost of traveling distance increase. On the other hand, SA requires the most running time of all proposed methods. GT and GT + TS require more running time than other approaches when increasing the unit cost. The reason is that more workers as players in the game period until reaching the Nash equilibrium

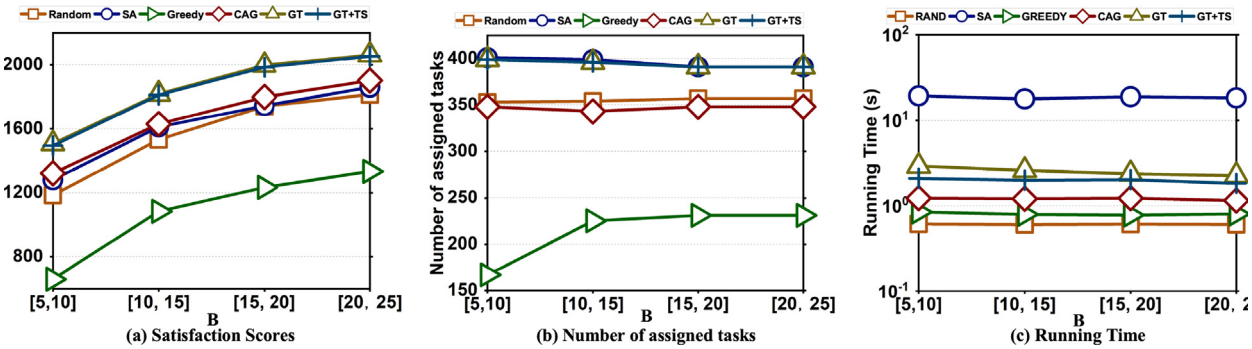


Fig. 6. Effect of the budget of tasks.

when the unit cost increases. GT + TS is slightly faster than GT when they have been achieved a similar satisfaction score, which shows the effectiveness of TS strategy.

Effect of the task budget. Fig. 6 illustrates the result of varying task budget from [5,10] to [20,25]. In Fig. 6(a), the satisfaction score increases when enlarging the task budget. As the average budget increases, the flexible budget of each task will also increase. Greedy, Random and SA achieve lower satisfaction scores than GT, GT + TS, and CAG. When the task budget varies from [5,10] to [15,20], price satisfaction increases. In addition, more task-and-worker pairs satisfy the budget constraints, then the satisfaction score increases. However, the scores almost are not increasing when the task budget goes from [15,20] to [20,25] due to all fixed numbers of workers being assigned to their best tasks for the optimal satisfaction scores. Fig. 6(b) shows the number of tasks increasing slightly when expanding the task budget. That is, the number of tasks and workers are fixed. GT, GT + TS and SA achieve the largest number of tasks assigned, which is higher than Random and Greedy. CAG also obtains a higher number of assigned tasks than other baseline algorithms. In Fig. 6(c), the time cost of CAG, RANG, Greedy, and SA remains stable because the time cost of satisfaction calculation is unchanged no matter what the task budget is. The running time of GT slightly decreases due to more workers assigned in the initial part of GT when increasing the task budget. Thus, few workers are participating in the game period, which results in the running time decreasing. Moreover, CAG needs more time cost than Random and Greedy while the CAG has the better performance on the satisfaction score. Besides, SA has the higher time consumption than CAG while SA has lower satisfaction than CAG.

Effect of the available range of workers. Fig. 7 shows the experimental results comparing different sizes of workers' working areas, ranging from 0.05 to 0.2. In Fig. 7(a), all approaches except for Random and SA achieve increasing satisfaction scores when expanding the working range of the workers from 0.05 to 0.1; then they almost stop growing when the working range of workers increases from 0.1 to 0.2. That is, with expanding the working range, more workers are available assigned to tasks. After that, the number of tasks and workers is fixed, and all workers assign to the tasks for higher satisfaction scores when the working range reaches 0.1. On the other hand, the satisfaction score of Random and SA is decreasing from 0.1. The reason is that at the beginning, the available workers are close to tasks that achieve higher price satisfaction even if selecting workers randomly. When expanding the working range, it is likely to choose workers farther away for lower price satisfaction due to the random strategy. Thus, satisfaction scores decrease when the working range is varying from 0.1 to 0.2 on the Random method. The SA is affected by the initialization, therefore, SA has a similar tendency to Random but higher than Random. In Fig. 7(b), GT, GT + TS and SA achieve the most and similar number of assigned tasks, and higher than that of CAG, Random, and Greedy. In Fig. 7(c), all approaches require more time when expanding the range of workers since more workers are available to tasks and more task-and-worker pairs need to be tested. The time cost of SA increased significantly. That is, more workers mean more time for forming the random assignment in iteration rounds.

Effect of the complexity of tasks. In Fig. 8, we vary the number of task skills from 3 to 6 to test the effectiveness of our methods. As Fig. 8(a), CAG, GT, and GT + TS perform better than Random, Greedy and SA on satisfaction score when the skill complexity of tasks increases. Specifically, the satisfaction scores of Random, Greedy decreases while the satisfaction scores of other approaches are almost unchanged. The reason is that when we increase the complexity of task skill requirements, it needs to hire more workers. More workers mean higher cooperation scores and lower price satisfaction score, thus, the satisfaction scores of our proposed methods are increasing slightly. However, Random and Greedy cannot guarantee the satisfaction score increases at the same time, thus, the satisfaction scores decrease. Fig. 8(b) presents the GT, GT + TS, CAG and SA achieves a higher number of tasks assigned than Greedy and Random. In Fig. 8(c), all tested approaches need more time because the matching difficulty is increasing. GT, GT + TS, and CAG require more time than Greedy and Random but lower than the SA.

Overall, our CAG, GT, and GT + TS approaches yield higher satisfaction scores on both real and synthetic data sets than Random, Greedy and SA. When compared to baseline methods, our proposed approaches can yield a total satisfaction score of about 30% to 60% higher than the baseline methods. Most importantly, threshold stop optimization is effective at reducing the running time of GT and only costs a very small portion of its total satisfaction scores. This emphasizes the efficiency of GT and its optimization strategy.

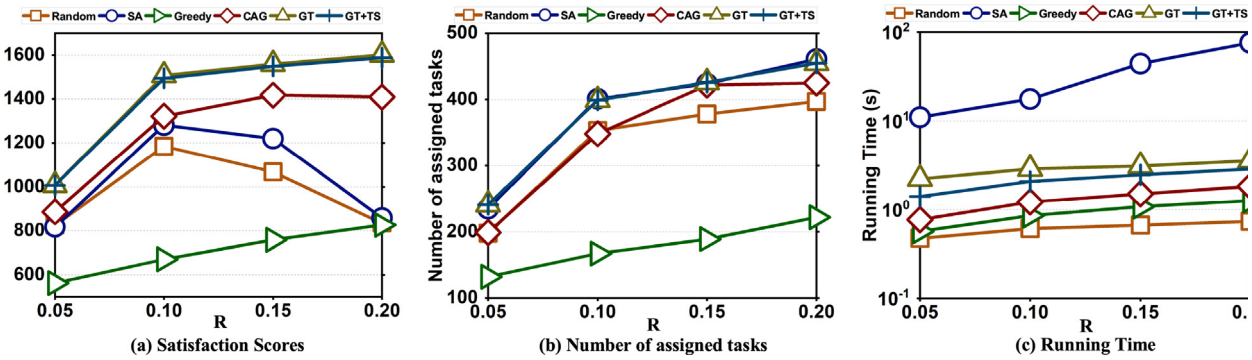


Fig. 7. Effect of the available range of workers.

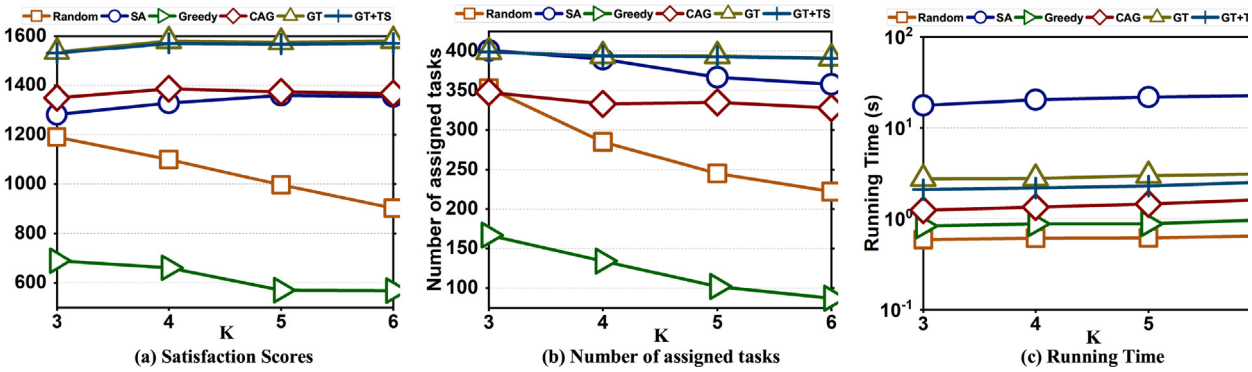


Fig. 8. Effect of the complexity of tasks.

7.2.3. Experiments on small datasets

In this section, we compare the proposed methods and the MIP method on a small dataset (e.g., 100 workers and 20 tasks) since the running time of MIP exceeds 10^4 seconds over large datasets (e.g., 500 tasks and 1000 workers). Other parameters are set as the default values.

Effect of the unit cost of workers. In Fig. 9(a), the satisfaction score decreases when the unit cost of workers rises. That is, as the traveling unit cost of workers rises, the number of available workers is limited since the fixed task budget. The best satisfaction score is achieved by MIP but it requires 3 orders of magnitude higher running time than our methods, which is shown in Fig. 9(b). In comparison to other baseline algorithms, the GT, GT + TS, and CAG achieve higher satisfaction scores and require less running time cost than SA and MIP. This fact shows the effectiveness and efficiency of our proposed methods.

Effect of the task budget. The total satisfaction score in Fig. 10(a) increases as the task budget is increased. That is as a result of the rising price satisfaction score. Furthermore, among all proposed methods, MIP achieves the highest satisfaction score. Nevertheless, GT achieves a satisfaction score that is only 7.59% lower than MIP in the best case. Additionally, MIP requires a running time that is more than 4 orders of magnitude in Fig. 10(b). This result also illustrates the effectiveness and efficiency of our proposed methods.

Effect of the available range of workers. In Fig. 11(a), the satisfaction score increases when enlarging the worker's range from 0.05 to 0.1. That is, more workers are available since increasing the working range. When the range is more than 0.1, the score remains constant after that. The available workers are fixed due to the limited task budgets. Although the MIP method achieves the highest satisfaction score among all algorithms, it requires the most running time, as shown in Fig. 11(b). Therefore, the proposed GT, GT + TS, and CAG methods obtain a good satisfaction score than other baseline algorithms with less time cost.

Effect of the complexity of tasks. When the complexity of the tasks is increased by raising the number of skills required for each task from 3 to 6, as shown in Fig. 12(a), the overall satisfaction score improves. Since the tasks require more workers to join in the task, the cooperation score increases, and the satisfaction score increases as well. As presented by the experiment, MIP requires longer than 10^4 seconds to obtain the assignment result when each task involves more than 4 different skills. In Fig, 12(b), we label the 10^4 seconds running time as INF since it is an unacceptable waiting time in our work. For the

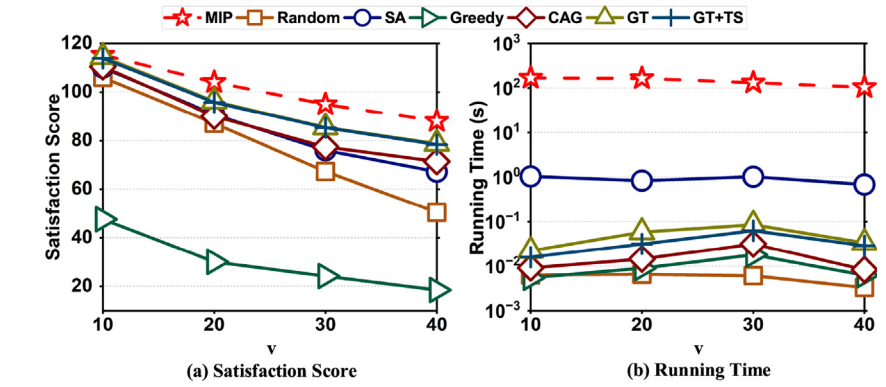


Fig. 9. Effect of the unit cost of workers.

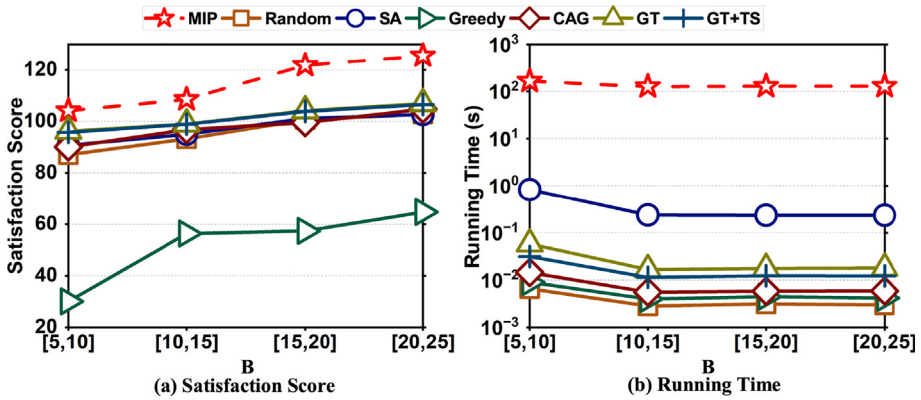


Fig. 10. Effect of the budget of tasks.

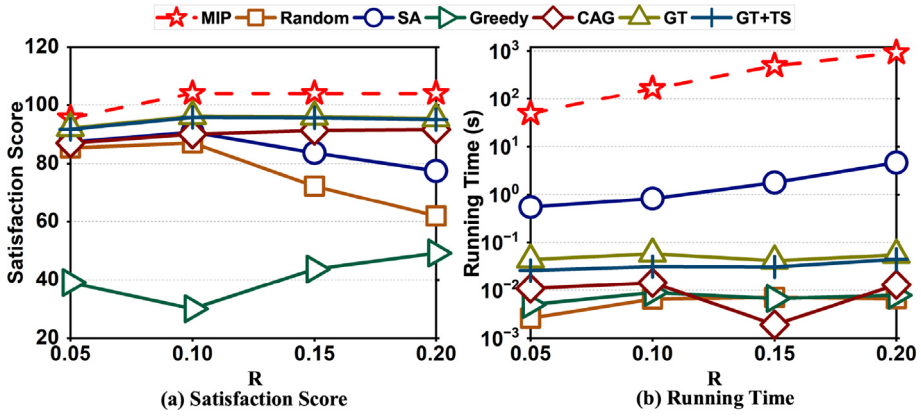


Fig. 11. Effect of the available range of workers.

proposed methods, they only require an acceptable running time that is within 0.1 seconds. Therefore, compared to the MIP approach, our methods have effectiveness and efficiency regarding to the total satisfaction score and the running time.

Overall, as shown in the experiment, the total satisfaction score of our proposed methods achieves 0.69% lower than that of MIP in the best case and 19.62% in the worst case. However, MIP demands a running time that is more than 3 orders of magnitude longer than our methods. Therefore, it is not suitable to apply the MIP method to handle large datasets. Additionally, while dealing with thousands of tasks and hundreds of workers, our proposed methods produce good assignment results in an acceptable amount of time.

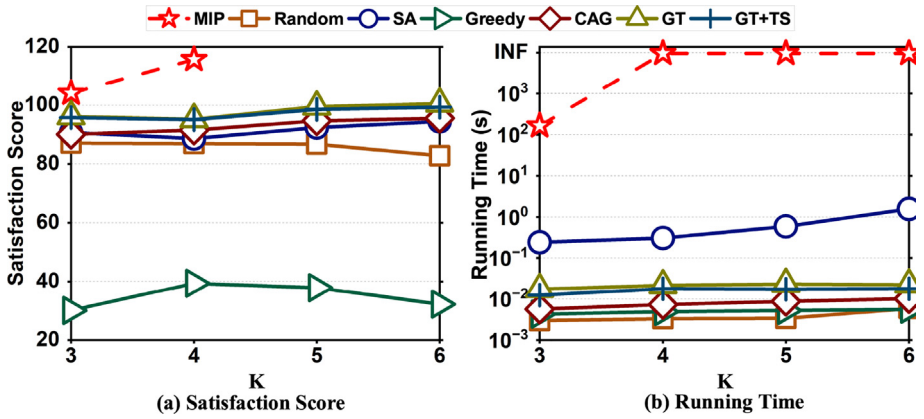


Fig. 12. Effect of the complexity of tasks.

8. Discussion

The experiment presents the efficiency and effectiveness of our proposed methods. The average time cost for dealing with hundreds of tasks and thousands of workers is 2–3 seconds, which meets the needs of real applications since users are not willing to wait for a long time. If the data size increases further, the users' requests probably wait for multiple seconds. Furthermore, we can extend our proposed methods to solve more complex multi-skill task assignment, in which workers have multiple tasks. In this way, the satisfaction increment of the task-and-worker pair depends on the effective skill of workers, which is the set of skills of workers that have been intersected by tasks. However, expanding the workers' skills and matching them to tasks according to their effective skills also can be solved in our proposed CAG and GT algorithm at the cost of more running time. Therefore, extending our algorithms to deal with the skill-oriented task assignment problem when workers have multiple complex skills is one of our future works. In many applications, the task assignment problem prefers to be dynamic rather than static. Due to the unevenness of arrived tasks and workers and the arrival time being random to the system, it is difficult to deal with the real-time task assignment problem in SC. Our proposed approaches in this paper are applied in the static state of each batch, where the spatial–temporal information of tasks and workers is obtained in advance. How to extend our methods in real-time dynamic task assignment problems in SC is also our future work. On the other hand, achieve the maximal number of tasks assigned is an important objective optimization in the task assignment in SC. Extending our approaches to optimize the overall satisfaction score and the number of tasks assigned simultaneously is one of our future works.

9. Conclusion

In this paper, we aim to define a satisfaction-aware task assignment (SATA) problem in spatial crowdsourcing for maximizing the user's overall satisfaction. Our SATA problem can be reduced to the *k*-SP problem to prove that its NP-hardness. To tackle this problem, we introduced the CAG method and GT approach with the TS strategy. Extensive experiments have demonstrated that our approaches are efficient and effective for the SATA problem on both real and synthetic data sets. In the future, we will extend our proposed methods on online scenarios to process the real-time tasks and apply them on the road network being close to real applications.

Data availability

Data will be made available on request.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

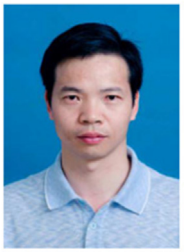
The research is supported by the National Key R\&D Program of China (Grant No. 2020YFB2104000), the NSFC under Grants 62172146, 62172157, and 62102143. The research is also supported by the Natural Science Foundation of Hunan Province under Grant 2022JJ30009 and the Open Research Projects of Zhejiang Lab under Grant 2021KD0AB02.

References

[1] P. Cheng, X. Lian, L. Chen, J. Han, J. Zhao, Task assignment on multi-skill oriented spatial crowdsourcing, IEEE TKDE (2016) 2201–2215.
[2] L. Zheng, L. Chen, Multi-campaign oriented spatial crowdsourcing, IEEE TKDE (2020) 700–713.
[3] P. Cheng, L. Chen, J. Ye, Cooperation-aware task assignment in spatial crowdsourcing, Proc. ICDE (2019) 1442–1453.
[4] S. Ma, Y. Zheng, O. Wolfson, T-share: A large-scale dynamic taxi ridesharing service, Proc. ICDE (2013) 410–421.
[5] D. Liang, W. Cao, Z. Xu, M. Wang, A novel approach of two-stage three-way co-opetition decision for crowdsourcing task allocation scheme, Inf. Sci. 559 (2021) 191–211.
[6] D. Gao, Y. Tong, Y. Ji, K. Xu, Team-oriented task planning in spatial crowdsourcing, in: Proceedings of the Web and Big Data, 2017, pp. 41–56.
[7] Z. Liu, K. Li, X. Zhou, N. Zhu, Y. Gao, K. Li, Multi-stage complex task assignment in spatial crowdsourcing, Inf. Sci. (2022) 119–139.
[8] F. Präntare, F. Heintz, An anytime algorithm for optimal simultaneous coalition structure generation and assignment, in: Proceedings of Autonomous Agents and Multi-Agent Systems, vol. 34, Springer, 2020, pp. 1–31.
[9] F. Präntare, H. Appelgren, F. Heintz, Anytime heuristic and monte carlo methods for large-scale simultaneous coalition structure generation and assignment, in: Proceedings of Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI Press, 2021, pp. 11317–11324.
[10] Y. Tong, Z. Zhou, Y. Zeng, L. Chen, C. Shahabi, Spatial crowdsourcing: a survey, VLDB J. (2020) 217–250.
[11] L. Chen, C. Shahabi, Spatial crowdsourcing: Challenges and opportunities, IEEE Data Eng. Bull. (2016) 14–25.
[12] Y. Tong, L. Chen, C. Shahabi, Spatial crowdsourcing: Challenges, techniques, and applications, Proc. VLDB Endow. (2017) 1988–1991.
[13] M. Xiao, K. Ma, A. Liu, H. Zhao, Z. Li, K. Zheng, X. Zhou, Sra: Secure reverse auction for task assignment in spatial crowdsourcing, IEEE TKDE, 2019, pp. 782–796.
[14] H. Hu, Y. Zheng, Z. Bao, G. Li, J. Feng, R. Cheng, Crowdsourced POI labelling: Location-aware result inference and task assignment, in: Proceedings of the ICDE, 2016, pp. 61–72.
[15] Y. Fang, H. Sun, G. Li, R. Zhang, J. Huai, Context-aware result inference in crowdsourcing, Inf. Sci. 460–461 (2018) 346–363.
[16] M. Asghari, D. Deng, C. Shahabi, U. Demiryurek, Y. Li, Price-aware real-time ride-sharing at scale: an auction-based approach, SIGSPATIAL (2016) 3:1–3:10.
[17] Y. Xu, M. Xiao, J. Wu, S. Zhang, G. Gao, Incentive mechanism for spatial crowdsourcing with unknown social-aware workers: A three-stage stackelberg game approach, IEEE TMC, 2022.
[18] K. Vu, R. Zheng, J. Gao, Efficient algorithms for k-anonymous location privacy in participatory sensing, Proceedings of INFOCOM (2012) 2399–2407.
[19] W. Tang, K. Zhang, J. Ren, Y. Zhang, X.S. Shen, Privacy-preserving task recommendation with win-win incentives for mobile crowdsourcing, Inf. Sci. (2020) 477–492.
[20] Y. Zhao, Y. Li, Y. Wang, H. Su, K. Zheng, Destination-aware task assignment in spatial crowdsourcing, Proc. CIKM (2017) 297–306.
[21] Y. Zhao, K. Zheng, Y. Li, H. Su, J. Liu, X. Zhou, Destination-aware task assignment in spatial crowdsourcing: A worker decomposition approach, IEEE TKDE (2020) 2336–2350.
[22] Y. Xu, Y. Tong, Y. Shi, Q. Tao, K. Xu, W. Li, An efficient insertion operator in dynamic ridesharing services, Proc. ICDE (2019) 1022–1033.
[23] Y. Tong, J. She, B. Ding, L. Wang, L. Chen, Online mobile micro-task allocation in spatial crowdsourcing, Proc. ICDE (2016) 49–60.
[24] Y. Tong, Y. Zeng, B. Ding, L. Wang, L. Chen, Two-sided online micro-task assignment in spatial crowdsourcing, IEEE TKDE (2021) 2295–2309.
[25] Y. Wang, Y. Tong, C. Long, P. Xu, K. Xu, W. Lv, Adaptive dynamic bipartite graph matching: A reinforcement learning approach, Proc. ICDE (2019) 1478–1489.
[26] D. Deng, C. Shahabi, L. Zhu, Task matching and scheduling for multiple workers in spatial crowdsourcing, SIGSPATIAL (2015) 21:1–21:10.
[27] L. Kazemi, C. Shahabi, in: Geocrowd: enabling query answering with spatial crowdsourcing, international conference on advances in geographic information systems, 2012, pp. 189–198.
[28] J. She, Y. Tong, L. Chen, C.C. Cao, Conflict-aware event-participant arrangement, Proc. ICDE (2015) 735–746.
[29] H. To, L. Fan, L. Tran, C. Shahabi, Real-time task assignment in hyperlocal spatial crowdsourcing under budget constraints, in: IEEE International Conference on Pervasive Computing and Communications (PerCom), 2016, pp. 1–8.
[30] H. To, C. Shahabi, L. Kazemi, A server-assigned spatial crowdsourcing framework, ACM Trans. Spatial Algorithms Syst., 2015, pp. 2:1–2:28.
[31] C. Shan, N. Mamoulis, R. Cheng, G. Li, X. Li, Y. Qian, An end-to-end deep RL framework for task arrangement in crowdsourcing platforms, Proc. ICDE (2020) 49–60.
[32] L. Zheng, L. Chen, Maximizing acceptance in rejection-aware spatial crowdsourcing, IEEE TKDE (2017) 1943–1956.
[33] L. Zheng, L. Chen, J. Ye, Order dispatch in price-aware ridesharing, Proc. VLDB Endowment (2018) 853–865.
[34] C. Long, R.C.-W. Wong, P.S. Yu, M. Jiang, On optimal worst-case matching, Proc. SIGMOD (2013) 845–856.
[35] Z. Chen, P. Cheng, Y. Zeng, L. Chen, Minimizing maximum delay of task assignment in spatial crowdsourcing, Proc. ICDE (2019) 1454–1465.
[36] J. Pilourdault, S. Amer-Yahia, S.B. Roy, D. Lee, Task relevance and diversity as worker motivation in crowdsourcing, Proc. ICDE (2018) 365–376.
[37] Y. Zhao, J. Xia, G. Liu, H. Su, D. Lian, S. Shang, K. Zheng, Preference-aware task assignment in spatial crowdsourcing, Proc. AAAI (2019) 2629–2636.
[38] A.S. Fonteles, S. Bouveret, J. Gensel, in: Heuristics for task recommendation in spatiotemporal crowdsourcing systems, 13th International Conference on Advances in Mobile Computing and Multimedia, MoMM, ACM, 2015, pp. 1–5.
[39] A.S. Fonteles, S. Bouveret, J. Gensel, Towards matching improvement between spatio-temporal tasks and workers in mobile crowdsourcing market systems, in: Proceedings of the Third ACM SIGSPATIAL International Workshop on Mobile Geographic Information Systems, MobiGIS, ACM, 2014, pp. 43–50.
[40] A.S. Fonteles, S. Bouveret, J. Gensel, Trajectory recommendation for task accomplishment in crowdsourcing – a model to favour different actors, J. Locat. Based Serv. 10 (2016) 125–141.
[41] G.L. Nemhauser, L.A. Wolsey, M.L. Fisher, An analysis of approximations for maximizing submodular set functions, Mathematical Programming, 1978, pp. 265–294.
[42] J.F. Nash, Equilibrium points in n-person games, in: the National Academy of Sciences of the United States of, America (1950) 48–49.
[43] D. Monderer, L.S. Shapley, Potential games, in: Games and economic behavior, vol. 14, 1996, pp. 124–143.
[44] L.A. Wolsey, Mixed integer programming, in: Wiley Encyclopedia of Computer Science and Engineering, Wiley Online Library, 2007, pp. 1–10.

Yuan Xie is currently working for a Ph.D. degree in computer science and technology with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, China. Her research interests includes spatial–temporal data management, spatial crowdsourcing, trajectory.

Yongheng Wang received the Ph.D degree in computer science and technology from National University of Defense Technology, Changsha, China, in 2006. He is currently a research specialist in Big Data Intelligence Research Center of Zhejiang Lab. His research interests include Big data analytics, machine learning and intelligent decision making. His current research is on intelligent interactive data analysis and simulation-based intelligent decision making in the economic field.



Kenli Li received the Ph.D. degree in computer science from Huazhong University of Science and Technology, China, in 2003. His major research areas include parallel computing, highperformance computing, grid and cloud computing. He has published more than 200 research papers in international conferences and journals such as IEEE-TC, IEEE-TPDS, and ICPP. He serves on the editorial board of the IEEE-TC.

Xu Zhou received the master's degree in the College of Computer Science and Electronic Engineering, Hunan University, in 2009. She is currently an associate professor with the Department of Information Science and Engineering, Hunan University, Changsha, China. Her research interests include parallel computing and data management.

Zhao Liu received the Ph.D. in computer science and technology from the College of Computer Science and Engineering, Hunan University, Changsha, China. His research interests include spatio-temporal data management and AI accelerators.

Keqin Li is a SUNY Distinguished Professor of computer science. His current research computing and high-performance computing, distributed computing, energy-efficient computing and communication, heterogeneous computing systems, cloud computing, big data computing, wireless communication networks, sensor networks, mobile computing, service computing, Internet of things and cyberphysical systems. He has published over 690 journal articles, book chapters, and refereed conference papers, and has received several best paper awards. He is currently or has served on the editorial boards of IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Computers, IEEE Transactions on Cloud Computing, IEEE Transactions on Services Computing, IEEE Transactions on Sustainable Computing. He is an IEEE Fellow.