2019 IEEE 35th International Conference on Data Engineering (ICDE)

Cooperation-Aware Task Assignment in Spatial Crowdsourcing

Peng Cheng
HKUST
Hong Kong, China
pchengaa@cse.ust.hk

Lei Chen
HKUST
Hong Kong, China
leichen@cse.ust.hk

Jieping Ye
AI Labs, DiDi Chuxing
Beijing, China
yejieping@didichuxing.com

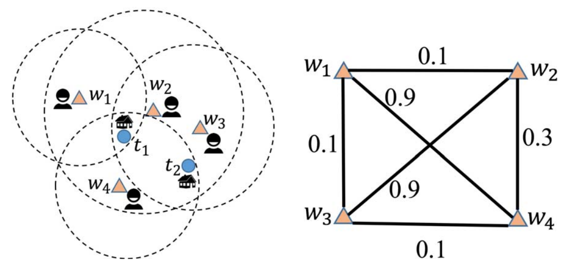
***Abstract*—With the popularity of smart devices and the development of high-speed wireless networks, the spatial crowdsourcing has attracted much attention from both academia and industry (e.g., Uber and TaskRabbit). Specifically, a spatial crowdsourcing platform assigns workers to location-based tasks according to their current positions, then the workers need to physically move to the specified locations to conduct the assigned tasks. In this paper, we consider an important spatial crowdsourcing problem, namely cooperation-aware spatial crowdsourcing (CA-SC), where spatial tasks (e.g., collecting the Wi-Fi signal strength in one building) are time-constrained and require more than one worker to complete thus the cooperation among assigned workers is essential to the result. Our CA-SC problem is to assign workers to spatial tasks such that the overall cooperation quality is maximized. We prove that the CA-SC problem is NP-hard by reducing from the k-set packing problem, thus intractable. To tackle the CA-SC problem, we propose *task-priority greedy* (TPG) approach and *game theoretic* (GT) approach with two optimization methods to quickly solve the CA-SC problem and achieve high total cooperation quality scores. Through extensive experiments, we demonstrate the efficiency and effectiveness of our proposed approaches over both real and synthetic datasets.**

***Index Terms*—Spatial Crowdsourcing, Game Theoretic Algorithm, Task Assignment**

I. Introduction

With the popularity of the smart devices and the development of high-speed wireless networks, people nowadays can easily participant in spatial tasks that are close to their current locations through online services and then contribute to the tasks in real world, such as taking photos/videos, cleaning rooms, and moving heavy stuff. To support these activities, a new framework, namely *spatial crowdsourcing* [11], is proposed and attracts much attention from both academia and industry (e.g., TaskRabbit [1]). Specifically, a spatial crowdsourcing system assigns moving workers to spatial tasks under the constraints of the locations, deadlines and capacities [5], [6], [11], [22], [26], [27].

In spatial crowdsourcing, some tasks need more than one worker to complete, such as moving heavy stuff, doing catering work for a wedding, and passing out leaflets at multiple nearby locations to advertise a shop [14]. When workers are assigned to such kind of tasks, they need to cooperate and communicate with each other to avoid free riders and accomplish the task well, then the relationship between the workers may play a crucial role in affecting the quality of



(a) Locations of tasks and workers (b) Cooperation Relationships
Fig. 1. An Example of Cooperation-Aware Spatial Crowdsourcing Problem.

the work. Then reputation or monetary awards of workers may also be affected by the cooperation quality. Good task cooperation quality can benefit workers, requesters and the platform. Existing studies [5], [11], [12], [22], however, have not focused on the cooperation relationship of workers.

With the consideration of the cooperation among workers, in this paper, we study an important problem in spatial crowdsourcing, namely *cooperation-aware spatial crowdsourcing* (CA-SC), which assigns *cooperation-aware moving workers* to *spatial tasks* to maximize the overall cooperation quality revenue. We first illustrate the motivation of the CA-SC problem with an example of doing catering work for a wedding.

Example 1. *In the example of cooperation-aware spatial crowdsourcing problem shown in Figure 1, there are two spatial tasks, t_1 and t_2, and four cooperation-aware workers, $w_1 \sim w_4$. The locations of the tasks and workers are shown in Figure 1(a), where workers are represented by yellow triangles and tasks are indicated with blue circles, and the dash circles show the working areas of workers. Specifically, worker w_2 can accept tasks t_1 and t_2 and worker w_1 only prefers task t_1. Assume each task needs two workers to complete the job of doing catering work for a wedding. In addition, based on the historical cooperation records, we estimate the cooperation qualities of worker pairs shown in Figure 1(b) with Equation 1, where each straight line indicates a cooperation relationship between the connected two workers and the number close to it denotes the cooperation quality of the two workers.*

Given the information above, the crowdsourcing system needs to assign each task with two workers under the working areas constraints of workers with a goal of completing the task with high quality. Since in spatial crowdsourcing, workers need to physically move to the required location of the as-

2375-026X/19/$31.00 ©2019 IEEE
DOI 10.1109/ICDE.2019.00130

signed task, where they will cooperate with the other workers assigned to the same task. In this example, if one worker loafs on the job, the other worker will need to work more and feel unfair, which may lead to a bad service quality. The system needs to determine a group of workers for each task such that they can cooperate well. In this example, we can assign workers w_1 and w_2 to task t_1, and workers w_3 and w_4 to task t_2 resulting in the total cooperation score of 0.2 (calculated by Equation 3). However, we can achieve a better assignment by dispatching workers w_1 and w_4 to task t_1, and workers w_2 and w_3 to task t_2, whose total cooperation quality score is 1.8 (obviously higher than 0.2).

In general, we handle CA-SC with a batch-based task assignment process, which means the platform periodically assigns the available workers to unfinished tasks. In this paper, we first prove that CA-SC is NP-hard by reducing from the k-set packing problem (k-SP) [29]. Thus, the CA-SC problem is not tractable and very hard to achieve the optimal result for problems with real world scales (e.g., hundreds of tasks and thousands of workers). Existing studies [5], [6], [11], [25], [33] of task assignment in spatial crowdsourcing do not take the cooperation scores of workers into consideration, thus cannot be used directly on our CA-SC problem. In this paper, we propose one greedy based approach and one novel game theoretic approach with two optimization methods to greedily handle CA-SC in each batch process.

Specifically, we make the following contributions.

- We formally define the *cooperation-aware spatial crowdsourcing* (CA-SC) problem proved NP-hard in Section II.
- We propose *task-priority greedy* approach in Section IV.
- We propose one game theoretic approach with two optimization methods in Section V.
- We conduct extensive experiments on real and synthetic data sets, and show the efficiency and effectiveness of our game theoretic approach in Section VI.

Section III introduces the batch-based framework of CA-SC problem. Section VII reviews previous studies. Finally, Section VIII concludes this paper.

II. PRELIMINARIES

Definition 1. ***(Cooperation-Aware Moving Workers)*** *Let $\mathbb{W} = \{w_1,\ w_2, ..., w_m\}$ be a set of m cooperation-aware moving workers. Each worker w_i ($1 \le i \le m$) is located at a location l_i having a moving speed of v_i at timestamp φ_i, and specifies a radius r_i of his/her working area. Moreover, the system knows the cooperation quality $q_i(w_k)$ between worker w_i and the other worker w_k.*

Each worker w_i comes to the spatial crowdsourcing system at timestamp φ_i and reveals his/her location l_i to the system. The worker may prefer to only accept the tasks within a working area with a radius r_i avoiding moving too much. Most importantly, the platform knows the cooperation quality score $q_i(w_k) \in [0, 1]$ between worker w_i and the other worker w_k. Intuitively, the cooperation quality score $q_i(w_k)$ reflects how good the two workers can work together. The higher $q_i(w_k)$ is, the better the service provided by the two workers is. Usually, platforms allow task requesters to rate the results. Let T_{ik} be a set of tasks that workers w_i and w_k both contributed to. Then the cooperation quality score $q_i(w_k)$ between two workers w_i and w_k can be estimated as:

$$q_i(w_k) = \alpha \cdot \omega + (1 - \alpha) \cdot \frac{\sum_{t_j \in T_{ik}} s_j}{|T_{ik}|}, \qquad (1)$$

where ω is a base cooperation quality (configured by the platform, such as 0.5), $s_j \in [0, 1]$ is the rating score of task t_j and $|T_{ik}|$ is the number of tasks in T_{ik}. In addition, α is a parameter to reconcile the basic quality score and the historical average quality score. The intuition of Equation 1 is that it reflects the balance of the historical performance (e.g., $\frac{\sum_{t_j \in T_{ik}} s_j}{|T_{ik}|}$) and the priori assumption (i.e., the average cooperation quality between any two workers, such as ω).

Definition 2. ***(Spatial Tasks)*** *Let $\mathbb{T} = \{t_1, t_2, ..., t_n\}$ be a set of spatial tasks. Each task t_j ($1 \le j \le n$) is created in the system at timestamp φ_j and requires a set of at most a_j workers at location l_j before its deadline τ_j.*

To guarantee tasks can be finished, at least B workers are required for each task. Specifically, for task t_j assigned with a set W_j of workers, we define the cooperation quality revenue $Q(W_j)$ of the workers as:

$$Q(W_j) = \begin{cases} 0, & |W_j| < B \\ \frac{\sum_{w_i \in W_j} \sum_{w_k \in W_j \wedge k \neq i} q_i(w_k)}{\min(|W_j|, a_j) - 1}, & B \le |W_j| \le a_j \end{cases} \qquad (2)$$

where $|W_j|$ indicates the number of workers in W_j. Note that, when the number of the assigned workers exceeds the capacity a_j of task t_j, we only consider a subset of a_j workers with the maximum overall cooperation quality. The reason is that the requester needs to pay each worker and the total budget to finish the task is limited, then limited workers are allowed.

For each worker $w_i \in W_j$ (here $|W_j| \ge B$), his/her average quality score can be calculated as $q_i(W_j) = \frac{\sum_{w_k \in W_j \wedge k \neq i} q_i(w_k)}{\min(|W_j|, a_j) - 1}$, which can be viewed as the expected revenue from hiring worker w_i. Then, the cooperation quality revenue $Q(W_j)$ can be treated as the expected qualified revenue received by the requester from hiring W_j workers.

Definition 3. ***(Valid Worker-and-Task Pairs)*** *For a set, $\mathbb{W}$, of m workers and a set, $\mathbb{T}$, of n tasks, a valid worker-and-task pair $\langle w_i, t_j \rangle$ indicates a pair of worker w_i and task t_j that: 1) worker w_i comes to the system after the task t_j is created; 2) the location l_j of task t_j is in the range of the working area of worker w_i; 3) the worker w_i can arrive at the location l_j before the deadline of task t_j.*

For each valid worker-and-task pair $\langle w_i, t_j \rangle$, we need to guarantee worker w_i can arrive at the required location of task t_j, that is $\frac{d(l_i, l_j)}{v_i} \le \tau_j - \varphi$, where $d(l_i, l_j)$ is the distance between w_i and t_j, v_i is the speed of w_i, τ_j is the deadline of t_j and φ is the current timestamp. Then, an assignment $\mathbb{A}$ of a CA-SC instance is a set of valid worker-and-task pairs satisfying the capacity constraint of tasks. Then, we can formally define the cooperation-aware spatial crowdsourcing (CA-SC) problem.

Definition 4. ***(CA-SC Problem)*** *Given a set $\mathbb{W}$ of workers and a set $\mathbb{T}$ of tasks, the CA-SC problem is to assign tasks to workers such that:*

1) the capacity constraints are satisfied; and
2) the working area constraints of workers are satisfied; and
3) the deadline constraints of tasks are satisfied; and
4) the overall cooperation quality revenue of all the tasks in $\mathbb{T}$ is maximized, which is:

$$Q(\mathbb{T}) = \sum_{t_j \in \mathbb{T}} Q(W_j) \tag{3}$$

where W_j is a set of workers assigned to task t_j and $Q(W_j)$ is their cooperation quality revenue defined in Equation 2.

Table I lists the commonly used symbols in this paper.

We prove that CA-SC is NP-hard by reducing from a well-known NP-hard problem, *k-set packing* (k-SP) problem [29].

Theorem II.1. ***(Hardness of CA-SC)*** *CA-SC is NP-hard.*

Proof. We prove the theorem through a reduction from the k-set packing problem (k-SP). A k-SP problem can be described as follows: given a universe of elements $\mathcal{U} = \{e_1, e_2, ..., e_{|\mathcal{U}|}\}$, a collection of subsets $\mathcal{C} = \{C_1, C_2, ..., C_{|\mathcal{C}|}\}$, where $C_i \subseteq \mathcal{U}$, and a number k. For each subset C_i, it is associated with a weight $w(C_i)$. The k-SP problem is to select some subsets $\mathcal{C}^* \subseteq \mathcal{C}$, satisfying that any two subsets $C_i, C_j \in \mathcal{C}^*$ are disjoint (e.g., $C_i \cap C_j = \emptyset$) and the size of $C_i \in \mathcal{C}^*$ is at most k (e.g., $|C_i| \leq k$), to maximize $\sum_{C_i \in \mathcal{C}^*} w(C_i)$.

For a given k-SP instance, we can transform it to a CA-SC instance within polynomial time as follows: each task requires k workers to complete. We configure that each worker can arrive at every task before its deadline. Next, for each element e_i, we create one worker w_i. For each subset C_j, we create one task t_j and associate it with a set W_j of workers such that $\forall e_i \in C_j, w_i \in W_j$ and $\forall e_i \notin C_j, w_i \notin W_j$, and the cooperation quality of the workers in W_j equals to $w(C_j)$ (i.e., $Q(W_j) = w(C_j)$). In addition, we set $B = \min(|W_j|), \forall t_j \in T$, which means every task t_j can be finished by its corresponding set W_j of workers. Then, for this CA-SC instance, we want to select a set of tasks to complete such that the overall cooperation quality score is maximized, which is same to maximize the total weight of the corresponding subsets in the original k-SP problem instance.

Given this polynomial-time mapping method, it is easy to show that the k-SP problem instance can be solved if and only if the transformed CA-SC problem can be solved.

This way, we reduce k-SP to the CA-SC problem. Since k-SP is well known to be NP-hard [29], CA-SC is also NP-hard, which completes our proof. □

Due to the NP-hardness of our CA-SC problem, in the next section, we propose a game theoretic approach with two optimization methods to solve CA-SC.

III. Framework for handling the CA-SC problem

In this section, we propose a batch-based framework as shown in Algorithm 1, which iteratively assigns workers to tasks for multiple batches. Specifically, for a batch starting at timestamp φ, we first retrieve a set, $\mathbb{T}_\varphi$, of available tasks and a set, $\mathbb{W}_\varphi$, of available workers (lines 2 - 3). Here, available tasks $\mathbb{T}_\varphi$ include the tasks that are not assigned with enough workers during the last batch and the newly appeared tasks after the last batch. Moreover, available workers $\mathbb{W}_\varphi$ includes the workers that are not assigned with any tasks in the last batch, the workers that have finished the previous assigned tasks, and the new workers after the last batch. As the tasks need workers to cooperate together, workers need to arrive at the location of task and start to conduct the task collaboratively. Each task needs a period of time to finish, then for different tasks the finishing time should be different as the requirement and the assigned workers are not same.

TABLE I
Symbols and Descriptions

Symbol	Description
w_i	a cooperation-aware worker
t_j	a spatial task
$\mathbb{W}$	a set of m cooperation-aware workers
$\mathbb{T}$	a set of n spatial tasks
$q_i(w_k)$	the cooperation quality between worker w_i and worker w_k
v_i	the moving speed of worker w_i
r_i	the radius of the working area of worker w_i
a_j	the capacity of task t_j
B	the minimum # of required workers to finish a task t_j
φ_i (φ_j)	the timestamp when worker w_i appears (task t_j is created) in the system
τ_j	the deadline of task t_j
W_j	the set of workers assigned to task t_j

Algorithm 1: Batch-based Framework

Input: A time interval Φ
Output: A set of worker-and-task assignments within the time interval Φ

1 **while** *current time φ is in Φ* **do**
2   retrieve all the available spatial tasks to $\mathbb{T}_\varphi$
3   retrieve all the available workers to $\mathbb{W}_\varphi$
4   **foreach** $w_i \in \mathbb{W}_\varphi$ **do**
5   obtain a set, T_i, of valid tasks for worker w_i
6   use our *task-priority greedy* or *game theoretic* approach to obtain a good assignment $\mathbb{A}$
7   **foreach** $\langle w_i, t_j \rangle \in \mathbb{A}$ **do**
8   inform worker w_i to conduct task t_j

To obtain the valid tasks T_i for each worker w_i, we can utilize the spatial index (e.g., R-Tree [24]) to conduct a range query with a range of r_i and a center at the current location l_i of w_i, then we remove the tasks that the worker w_i cannot arrive at the required locations of them and only report the tasks T_i that within the working area of worker w_i and worker w_i can arrive at the require locations of the tasks (lines 4 - 5). With the available tasks $\mathbb{T}_\varphi$, the available workers $\mathbb{W}_\varphi$ and the valid tasks T_i for each worker w_i, we can apply our proposed approaches, including *task-priority greedy* (TPG) and *game theoretic* (GT) with optimization methods, to achieve a good assignment $\mathbb{A}$ with a high total cooperation quality score (line 6). In the end of each iteration, for every pair $\langle w_i, t_j \rangle$ in $\mathbb{A}$, we inform the worker w_i to conduct task t_j (lines 7 - 8).

Specifically, TPG first greedily assigns the most suitable B-worker set to each task with the most potential candidate workers, then keeps choosing the "best" worker-and-task pair with the maximum total cooperation quality increase, which

Algorithm 2: Task-Priority Greedy Approach

Input: A set $\mathbb{W}_{(\varphi)}$ of available workers and a set $\mathbb{T}_{(\varphi)}$ of tasks at timestamp φ

Output: An Assignment $\mathbb{A}$ for this timestamp φ

1 $\mathbb{T} \leftarrow \emptyset$
2 **while** $\mathbb{T}_{(\varphi)} \neq \emptyset$ **do**
3   **foreach** $t_j \in \mathbb{T}_{(\varphi)}$ **do**
4   find a set of B workers $W_{j,B}$ with the highest cooperation quality score for task t_j
5   select the workers W_B^* with the highest cooperation quality score for the remaining tasks in $\mathbb{T}_{(\varphi)}$
6   **if** *more than one task competes for* W_B^* **then**
7   assign W_B^* to the task t_j with the most potential workers
8   **else**
9   assign W_B^* to the task t_j who owns it
10     $\mathbb{A} \leftarrow \mathbb{A} + \cup_{w_i \in W_B^*}\{\langle w_i, t_j\rangle\}$
11     $\mathbb{W}_{(\varphi)} \leftarrow \mathbb{W}_{(\varphi)} - W_B^*$
12     $\mathbb{T}_{(\varphi)} \leftarrow \mathbb{T}_{(\varphi)} - \{t_j\}$
13     $\mathbb{T} \leftarrow \mathbb{T} + \{t_j\}$
14 $\mathbb{T}_{(\varphi)} \leftarrow \mathbb{T}$
15 **while** $\mathbb{T}_{(\varphi)} \neq \emptyset$ ***and*** $\mathbb{W}_{(\varphi)} \neq \emptyset$ **do**
16   select a worker-and-task pair $\langle w_i, t_j\rangle$ with the highest utility value from workers in $\mathbb{W}_{(\varphi)}$ and tasks in $\mathbb{T}_{(\varphi)}$
17     $\mathbb{A} \leftarrow \mathbb{A} + \{\langle w_i, t_j\rangle\}$
18     $\mathbb{W}_{(\varphi)} = \mathbb{W}_{(\varphi)} - \{w_i\}$
19   **if** $|W_j| = a_j$ **then**
20         $\mathbb{T}_{(\varphi)} = \mathbb{T}_{(\varphi)} - \{t_j\}$
21 **return** *the assignment* $\mathbb{A}$ *of all the tasks*

is local optimal and may be unfair for some workers as they may have better choices if they are allowed to select tasks by themselves. Furthermore, GT utilizes the best-response strategy to iteratively adjust the "best" task for each worker until a Nash equilibrium state is met, which means no single worker can improve his/her cooperation quality score by unilaterally switching to any other tasks when other workers stay in the assigned tasks in the Nash equilibrium state. The corresponding assignment of the Nash equilibrium state usually has a high total cooperation quality score and is fair to every worker, as each single worker is assigned with his/her optimal strategy upon the other workers' current choices.

Note that, the framework just greedily resolves the "current" CA-SC problem in each batch, which does not result in the global optimal for the entire time interval Φ.

IV. The Task-Priority Greedy Approach

In this section, we propose a task-priority greedy (TPG) approach, which first iteratively assigns the set of B workers with the highest cooperation quality score to the most suitable task having not be assigned with any workers, then keeps selecting the "best" worker-and-task pair with the maximum total cooperation quality increase until all the tasks are assigned with enough workers or every available worker has been assigned to his/her most suitable task. The intuition of TPG is to first greedily finish as many tasks as possible, then greedily assign workers to tasks with the maximum total cooperation score increase. The TPG approach can be applied to the batch process in line 6 of the batch-based framework shown in Algorithm 1.

The Cooperation Quality Increase of Assigning Worker w_i to Task t_j. Before we present the TPG algorithm, we first define the total cooperation quality increase, $\Delta Q(w_i, t_j)$, of assigning worker w_i to task t_j as following:

$$\Delta Q(w_i, t_j) = Q(W_j) - Q(W_j - \{w_i\}), \tag{4}$$

where W_j is the assigned workers of task t_j (including worker w_i), and $Q(W_j)$ indicates the cooperation quality calculated with Equation 2.

We propose a two-stage task-priority greedy approach shown in Algorithm 2. In the first stage, we assign each task a set of B workers with a high cooperation score (lines 2 - 13); in the second stage, we iteratively select a new worker-and-task pair with the highest total cooperation quality score increase (lines 15 - 20). Specifically, we first initial a temp set $\mathbb{T}$ as an empty set to store the tasks having been assigned with B workers (line 1). In each iteration of the first stage, we first calculate the "best" set of B workers for each task (lines 3 - 4), then select the set W_B^* of B workers with the highest cooperation score among all the tasks in $\mathbb{T}_{(\varphi)}$ (line 5). If the worker set W_B^* is the "best" pair for more than one task, we assign them to the task with the most potential candidate workers to guarantee the task has a wider range to select other workers in the second stage (lines 6 - 9). In each iteration of the second stage, we first select a worker-and-task pair $\langle w_i, t_j\rangle$ with the highest total cooperation quality score increase (defined in Equation 4) from the remaining available workers $\mathbb{W}_{(\varphi)}$ and tasks $\mathbb{T}_{(\varphi)}$. Then, we put the selected pair $\langle w_i, t_j\rangle$ in to the assignment set $\mathbb{A}$ (line 17). In addition, if we find task t_j has been assigned with enough workers, we remove it from the set, $\mathbb{T}_{(\varphi)}$, of available tasks (lines 19 - 20). Finally, we return the achieved assignment $\mathbb{A}$ for the system to notify the selected workers about their assigned tasks.

The Time Complexity. Next, we analyze the time complexity of the TPG approach shown in Algorithm 2. Assume that each worker is valid to $\bar{n}$ tasks and each task is valid to $\bar{m}$ workers and require $\bar{a}$ workers to complete. Specifically, in each iteration of the first stage (lines 2 - 13), to greedily find the B-worker set with the highest cooperation quality score for each one of n tasks, it needs $O(\bar{m}n)$ time (lines 3 - 4). To select one B-worker set with the highest cooperation score for all tasks needs $O(n)$ (line 5). To solve the competing for the best twined worker-pair, it needs at most $O(\bar{m})$ (line 6 - 9). As in each iteration at least one task will be assigned with a B-worker set, there are at most n iterations. Thus, the time complexity of the first stage is $O(\bar{m}n^2)$. In the second stage, there are at most $\max(O(m\bar{n}), O(\bar{m}n))$ valid worker-and-pairs, thus to select one "best" worker-and-task pair needs $\max(O(m\bar{n}), O(\bar{m}n))$ time (line 16). In addition, as we just need at most $(\bar{a}-2)n$ more worker-and-task pairs to complete all the n tasks and each iteration we will assign at least one worker-and-task pair, there are at most $(\bar{a}-2)n$ iterations. Then, the second stage needs $\max(O(mn\bar{n}), O(\bar{m}n^2))$. Thus, the overall time complexity of the TPG approach is $\max(O(mn\bar{n}), O(\bar{m}n^2))$.

V. The Game Theoretic Approach

Although TPG can solve CA-SC problem approximately, it assumes the authority of the centralized server: workers will follow the instructions of the server to conduct the assigned tasks. The fundamental nature of the CA-SC problem is that each worker needs to interact with other workers during conducting tasks. For each worker, he/she may prefer to cooperate with other workers who have high cooperation quality scores with him/her such that the tasks can be quickly finished with high quality. Then he/she may receive either better reputational or monetary awards. To conform to the individual incentives of workers and achieve better societal welfare, numerous game theoretic models are developed in economics, politics and networks studies [3], [18], [20]. Based on the existing studies in game theoretic models, in this section we propose a game theoretic solution that can iteratively adjust a valid assignment of workers to tasks until the *Nash equilibrium* is met [16]. Intuitively, in a Nash equilibrium assignment, any single worker cannot improve his/her cooperation quality score by unilaterally switching from the assigned task to other tasks when other workers stay in their assigned tasks, which means workers will voluntarily select the assigned tasks when they have freedom to do so. In addition, usually a Nash equilibrium assignment can result in a high total cooperation quality score since each worker is assigned to his/her "best" task in the stable assignment, which is confirmed in our experimental study in Section VI. Note that, a Nash equilibrium assignment does not guarantee the global optimal result, and we prove the theoretical quality of the results in Section V-C.

We first introduce the basics of game theory, then introduce the game theoretic approach with two optimization methods.

A. Game Theory

Game theory is the study of mathematical models of conflict and cooperation between intelligent rational decision-makers, which is widely used in economics, political science, and computer science [9]. In *strategic games*, there are a set of *players* $\mathcal{N}$ competing or cooperating for some resources in order to optimize their individual objective functions (*utilities*). For each player $i \in \mathcal{N}$, he/she can choose one *strategy* s_i (i.e., conducting task t_x) out from the set of his/her possible strategies S_i, and has a *utility function* U_i, whose value depends on the strategy of player i as well as the strategies of other players. The input of the utility function U_i is a given *joint strategy* $\mathbf{S} \in \mathbb{S}$, where $\mathbb{S}$ is the Cartesian product of the actions of all players (i.e., $\mathbb{S} = S_1 \times S_2 \times \cdots \times S_{|\mathcal{N}|}$). Let s_i be the strategy of player i in the joint strategy $\mathbf{S}$ and s_{-i} be the joint strategies of all other players except for player i. A strategic game has a pure Nash equilibrium [19] $\mathbf{S}^* \in \mathbb{S}$, if and only if for every player $i \in \mathcal{N}$ we have the following conditions:

$$U_i(s_i^*, s_{-i}^*) \geq U_i(s_i, s_{-i}^*), \forall s_i \in S_i$$

Algorithm 3: Game Theoretic Approach

Input: A set $\mathbb{W}_{(\varphi)}$ of available workers and a set $\mathbb{T}_{(\varphi)}$ of tasks at timestamp φ
Output: Nash Equilibrium

```
1 Apply TPG approach to achieve an initial assignment
2 while Not Nash Equilibrium do
3     foreach w_i ∈ W_(φ) do
4         select the best-response task t_j for worker w_i
5         assign worker w_i to task t_j
6 return the assignment of each worker
```

In other words, in a Nash equilibrium no player can improve his/her utility by unilaterally changing his/her strategy when other players persist in their current strategies.

One common used framework to search a Nash equilibrium $\mathbf{S}^* \in \mathbb{S}$ for a given strategic game $\mathcal{G} = \langle \mathcal{N}, \mathbb{S}, \{U_i\}_{i\in\mathcal{N}} \mapsto \mathbb{N}\rangle$ is the *best-response framework* [21], which first randomly selects a strategy for each player, then iteratively selects the "best" strategy for each player i based on the current strategies of other players until a Nash equilibrium is found (i.e., no one will change the selected strategy).

There are several issues related to the best-response framework: a) **Stability**: whether the best-response frame can find a Nash equilibrium; b) **Convergence**: how fast it can converge; c) **Quality**: how good is the found solution. We will first propose one game theoretic approach to solve our CA-SC problem, then answer the three issues related to the approach one-by-one.

B. The Game Theoretic Approach

In this section, we first model a CA-SC problem instance as a strategic game $\mathcal{G}$ then propose a game theoretic approach (GT) based on the best-response framework to find a Nash equilibrium joint strategy for the strategic game $\mathcal{G}$, where every worker is assigned to his/her "best" task such that a high total cooperation quality 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 cooperation score (utility) for him/her. For each player w_i, his/her strategy set S_i indicates all the possible strategies that he/she can choose (e.g., all the valid tasks he/she can conduct). Then, a joint strategy $\mathbf{S}$ for the strategic game $\mathcal{G}$ corresponds to an assignment $\mathbb{A}$ for the CA-SC problem instance.

We first define the utility function U_i of worker w_i in the joint strategy $\mathbf{S}$ as the cooperation quality score increase:

$$U_i(\mathbf{S}) = U_i(s_i, s_{-i}) = \Delta Q(w_i, t_j) = Q(W_j) - Q(W_j - \{w_i\}) \quad (5)$$

where W_j is the assigned workers of task t_j (worker w_i selects task t_j). Then, we propose a basic game theoretic approach, shown in Algorithm 3, to achieve a Nash equilibrium for a set of workers $\mathbb{W}_{(\varphi)}$ and a set of tasks $\mathbb{T}_{(\varphi)}$ at timestamp φ. Specifically, we first apply TPG approach to achieve an initial assignment (lines 1). Next, we iteratively adjust each worker's strategy to his/her best-response strategy that maximizes his/her utility function U_i (as defined in Equation 5) according to the other workers' current joint strategy until a

Nash equilibrium is found (i.e., no worker will change his/her strategy in the best-response framework) (lines 2 - 5). Here, each iteration of the WHILE loop (lines 2-5) is called a *round*.

C. Analysis of the Game Theoretic Approach

In this section, we analyze the three important issues related to the game theoretic approach (Algorithm 3): a) whether it can find a Nash equilibrium (**Stability**); b) how fast it can converge (**Convergence**); c) how good is the found result (**Quality**).

The Stability of the Approach. We prove that Algorithm 3 can finally result in a Nash equilibrium. To prove the stability of Algorithm 3, we first introduce the theory of Potential Games [17]. A strategic game $\mathcal{G} = \langle \mathcal{N}, \mathbb{S}, \{U_i\}_{i\in\mathcal{N}} \mapsto \mathbb{R}\rangle$ is called an *exact potential game* if and only if there exists a *potential function* $F(\mathbf{S}) : \mathbb{S} \mapsto \mathbb{R}$ such that:

$$U_i(s_i, s_{-i}) - U_i(s_i', s_{-i}) = F(s_i, s_{-i}) - F(s_i', s_{-i}), \forall s_i, s_i' \in S_i$$

where s_i and s_i' are the strategies that worker w_i can select, and s_{-i} is the joint strategy of the other workers except for worker w_i. Intuitively, in an exact potential game, the change in a single player's utility due to his/her own strategy deviation results in exactly the same amount of change in the potential function. The most important property of potential games is that the best-response framework always converges to a pure Nash equilibrium for finite-strategy potential games [17].

Next, we prove the stability of the basic game theoretic approach by proving the strategic game of our CA-SC problem is an exact potential game in the following theorem. Specifically, we define the potential function F as the objective function in Equation 3.

Theorem V.1. *The strategic game of the CA-SC problem is an exact potential game.*

Proof. Let s_i and s_i' be any other response strategy of worker w_i. Here a given joint strategy s_{-i} is for the other workers except for worker w_i. We note the task selected in strategies s_i and s_i as tasks t_j and t_k, respectively, then we have:

$$\begin{aligned}
& F(s_i, s_{-i}) - F(s_i, s_{-i}) \\
= \quad & Q(W_j) + Q(W_k - \{w_i\}) + \sum_{t_x \in \mathbb{T}-\{t_j,t_k\}} Q(W_x) \\
& -\Big(Q(W_k) + Q(W_j - \{w_i\}) + \sum_{t_x \in \mathbb{T}-\{t_j,t_k\}} Q(W_x)\Big) \\
= \quad & Q(W_j) + Q(W_k - \{w_i\}) - \big(Q(W_k) + Q(W_j - \{w_i\})\big) \\
= \quad & Q(W_j) - Q(W_j - \{w_i\}) - \big(Q(W_k) - Q(W_k - \{w_i\})\big) \\
= \quad & U_i(s_i, s_{-i}) - U_i(s_i, s_{-i}) \qquad (6)
\end{aligned}$$

Thus, according to the definition of potential games [17], the strategic game of the CA-SC problem is an exact potential game. □

Based on the theory of potential games, with Theorem V.1, we achieve the conclusion that the best-response based game theoretic approach (shown in 3) can finally result in a pure Nash equilibrium since the strategic game of the CA-SC problem is a potential game and has finite strategies.

The Convergence of the Approach. To answer the convergence speed of the GT approach (Algorithm 3), we need to know how many rounds it needs to find a stable result (a pure Nash equilibrium) and the time complexity of each round.

To estimate the upper bound of the total rounds that the GT approach needs to achieve a pure Nash equilibrium, we consider a scaled version of the problem where the objective function takes integer values. Specifically, for the corresponding potential game of a CA-SC problem instance, $\langle \mathcal{N}, \mathbb{S}, \{U_i\}_{i\in\mathcal{N}} \mapsto \mathbb{R}\rangle$, we assume that there is an equivalent game with potential function $F_{\mathbb{Z}}(\mathbf{S}) = d \cdot F(\mathbf{S})$, where d is a positive multiplicative factor chosen such that $F_{\mathbb{Z}}(\mathbf{S}) \in \mathbb{Z}, \forall \mathbf{S} \in \mathbb{S}$. With this scaled potential function, we show that the GT approach executes at most $F_{\mathbb{Z}}(\mathbf{S}^*)$ rounds, where $\mathbf{S}^*$ is the best strategy the workers can select in this potential CA-SC game (i.e., $F_{\mathbb{Z}}(\mathbf{S}^*)$ is the product of optimal value of the objective function $Q(\mathbb{T}_{(\varphi)})$ and the positive multiplicative factor d).

Lemma V.1. *The GT approach requires at most $F_{\mathbb{Z}}(\mathbf{S}^*)$ rounds to achieve a pure Nash equilibrium, where $F_{\mathbb{Z}}(\mathbf{S}^*)$ $(= d \cdot F(\mathbf{S}^*))$ is a scaled potential function with only integer values and $\mathbf{S}^*$ is the optimal joint strategy the workers can select in the potential CA-SC game.*

Proof. The GT approach (Algorithm 3) converges when no workers deviates from their current strategies, which means in each round (lines 3-6 in Algorithm 3) there is at least one worker deviating from his/her current strategy. In addition, the change of each worker w_i from his/her current strategy s_i to a better strategy s_i' will improve the scaled potential function at least 1 (i.e., $F_{\mathbb{Z}}(s_i', s_{-i}) - F_{\mathbb{Z}}(s_i, s_{-i}) \geq 1$), which is because 1) for potential games, the change in a single player's utility due to his/her own strategy deviation results in exactly the same amount of change in the potential function; 2) $F_{\mathbb{Z}}(\mathbf{S}) \in \mathbb{Z}$. Thus, in each round, the value of the scaled potential function will increase at least 1. Since the maximum value of the scaled potential function is $F_{\mathbb{Z}}(\mathbf{S}^*)$ and the total cooperation quality score is always positive, the GT approach needs at most $F_{\mathbb{Z}}(\mathbf{S}^*)$ rounds to converge to a pure Nash equilibrium. □

However, we do not know the optimal joint strategy of the workers unless we enumerate all the possible strategies, which is impractical due to that the CA-SC problem is NP-hard (as proven in Theorem II.1). Thus, we need to estimate the upper bound of $F_{\mathbb{Z}}(\mathbf{S}^*)$ $(= d \cdot \sum_{t_j \in \mathbb{T}_{(\varphi)}} Q(W_j^*)$, where W_j^* is the set of workers assigned to task t_j in the optimal joint strategy). We notice that if we can calculate the optimal value of the cooperation quality score of each task t_j, noted as $Q_{max}(t_j)$, we can estimate the upper bound of the scaled potential function as $F_{\mathbb{Z}}(\mathbf{S}^*) \leq d \cdot \sum_{t_j \in \mathbb{T}_{(\varphi)}} Q_{max}(W_j)$. However, to find a_j workers having the highest cooperation quality score for task t_j is also a NP-hard problem, which is called the Maximum Weight Connected k-Induced Subgraph Problem and is proved NP-hard by reducing from the CLIQUE problem [2]. Nevertheless, we can estimate the upper bound of the cooperation quality score $Q(W_j^*)$ of task t_j in the optimal joint strategy with the lemma below.

Lemma V.2. *Given a group W_x of $|W_x| \geq B$ workers, for any worker $w_i \in W_x$, we have $\frac{\sum_{w_k \in W_x \wedge k \neq i} q_i(w_k)}{|W_x|-1} \leq \frac{\sum_{w_k \in \hat{W}^i_{B-1} \wedge k \neq i} q_i(w_k)}{B-1} = \hat{q}_{i,B}$, where $\hat{W}^i_{B-1}$ is a set of $B-1$ workers who have the $B-1$ highest cooperation quality scores to worker w_i. $\hat{q}_i$ represents the highest average quality score of worker w_i in a group of not less than B workers.*

Proof.

$$\begin{aligned}
&\frac{\sum_{w_k \in W_x \wedge k \neq i} q_i(w_k)}{|W_x| - 1} \\
= &\frac{\sum_{w_k \in W_x \cap \hat{W}^i_{B-1} \wedge k \neq i} q_i(w_k) + \sum_{w_k \in W_x - \hat{W}^i_{B-1} \wedge k \neq i} q_i(w_k)}{|W_x| - 1} \\
\leq &\frac{\sum_{w_k \in W_x \cap \hat{W}^i_{B-1} \wedge k \neq i} q_i(w_k) + |W_x - \hat{W}^i_{B-1}| q^{B-1}_{i,max}}{|W_x| - 1} \\
\leq &\frac{\sum_{w_k \in \hat{W}^i_{B-1} \wedge k \neq i} q_i(w_k)}{B - 1} = \hat{q}_{i,B}
\end{aligned} \tag{7}$$

where $q^{B-1}_{i,max}$ is the $(B-1)$th maximum cooperation quality score of worker w_i. □

Then, we have the cooperation quality score of task t_j in the optimal joint strategy is upper bounded by

$$\hat{Q}_{t_j} = \sum_{w_x \in \hat{W}_j} \hat{q}_x, \tag{8}$$

where $\hat{W}_j$ is a set of a_j workers who have top a_j highest value of $\hat{q}_x$. On the other hand, the upper bound should be lower than the summation of the highest average quality scores $\hat{q}_{i,B}$ of all the workers. Thus the upper bound of the total cooperation quality revenue can be calculated as

$$\hat{Q}_{(\varphi)} = \min(\sum_{t_j \in \mathbb{T}_{(\varphi)}} \hat{Q}_{t_j}, \sum_{w_i \in \mathbb{W}_{(\varphi)}} \hat{q}_{i,B}) \tag{9}$$

Then, the GT approach requires at most $F_{\mathbb{Z}}(\mathbf{S}^*) \leq d \cdot \hat{Q}_{(\varphi)}$ rounds to reach a Nash equilibrium

Similarly, we have the lemma below about the lowest average quality score $\check{q}_{i,B}$ of a worker w_i in a group of not less than B workers.

Lemma V.3. *Given a group W_x of $|W_x| \geq B$ workers, for any worker $w_i \in W_x$, we have $\frac{\sum_{w_k \in W_x \wedge k \neq i} q_i(w_k)}{|W_x|-1} \geq \frac{\sum_{w_k \in \check{W}^i_{B-1} \wedge k \neq i} q_i(w_k)}{B-1} = \check{q}_{i,B}$, where $\check{W}^i_{B-1}$ is a set of $B-1$ workers who have the $B-1$ lowest cooperation quality scores to worker w_i.*

Time Complexity. With the upper bound of the total rounds to reach a Nash equilibrium, we can analyze the time complexity of GT algorithm as $O(mn \sum_{t_j \in \mathbb{T}_{(\varphi)}} \hat{Q}_{t_j})$, where $\hat{Q}_{t_j}$ is the upper bound of the cooperation quality score of task t_j. Specifically, in Algorithm 3, to randomly assign each worker to a task needs $O(n)$ (lines 1 - 2). In each round of lines 3 - 7, it needs $O(mn)$ to select the best action for every workers (lines 4 - 6). As GT needs at most $d \cdot \sum_{t_j \in \mathbb{T}_{(\varphi)}} \hat{Q}_{t_j}$ rounds to find a Nash equilibrium, the whole time complexity of it is $O(mn \sum_{t_j \in \mathbb{T}_{(\varphi)}} \hat{Q}_{t_j})$.

The Quality of the Approach. For any strategic game, there may be many Nash equilibriums with different qualities w.r.t. the global objective functions [21]. In existing studies about Game theory [9], [17], [21], three measures are widely used to evaluate the quality of equilibria: 1) *social optimum* (OPT); 2) *price of stability* (PoS); 3) *price of anarchy* (PoA). The social optimum is the result that achieves the global optimal joint strategy w.r.t the given global objective function (i.e., the overall utility (or cost) is maximum (or minimum), which is also the goal of the related optimization problem.). PoS of a strategic game indicates the ratio of the best utility value achieved among all the equilibria to the OPT (i.e., PoS = value of the best equilibrium / OPT). For another, PoA of a strategic game describes the ratio of the worst utility value achieved by equilibria to the OPT (i.e., PoA = value of the worst equilibrium / OPT). Intuitively, PoS and PoA reflect the upper and lower bounds of the ratio of the utility value achieved by equilibria to the OPT, respectively. We show the upper bound of PoS and the lower bound of PoA as follows.

Theorem V.2. *In the strategic game of CA-SC, the lower bound of PoA is $\frac{N_{init} B \check{q}}{\hat{Q}_{(\varphi)}}$ and the upper bound of PoS is 1, where N_{init} is the number of finished tasks in the initialization stage of GT, and $\hat{q}$ is the minimum average cooperation quality score of any worker in a set of B workers.*

Proof. Let $Q(\mathbf{S})$ be the overall cooperation quality score of the corresponding assignment $\mathbb{A}$ of the joint strategy $\mathbf{S}$ (i.e., $Q(\mathbf{S}) = F(\mathbf{S})$). In addition, we note the global optimal joint strategy as $\hat{\mathbf{S}}$, the equilibrium with the best total cooperation quality score as $\mathbf{S}^*$, and the equilibrium with the worst total cooperation quality score as $\mathbf{S}^{\#}$.

As for any strategy $\mathbf{S}$, we have $Q(\mathbf{S}) = F(\mathbf{S})$, we have: $Q(\hat{\mathbf{S}}) = F(\hat{\mathbf{S}})$ and $Q(\mathbf{S}^*) = F(\mathbf{S}^*)$. As $\hat{\mathbf{S}}$ is the joint strategy that results in the global optimum and $\mathbf{S}^*$ is the equilibrium that achieves the maximum total cooperation quality score, we have $OPT = Q(\hat{\mathbf{S}}) = F(\hat{\mathbf{S}}) \geq F(\mathbf{S}^*) = Q(\mathbf{S}^*)$. As a result, we directly have:

$$PoS = \frac{Q(\mathbf{S}^*)}{OPT} \leq 1 \tag{10}$$

To estimate the lower bound of PoA, we first analyze the lower bound of the worst total cooperation quality revenue achieved by an equilibrium. We can note that in the best-response iteration, the overall cooperation quality revenue only can increase, thus the overall cooperation quality revenue must be larger than that of the result achieved by TPG in the initialization stage. Let T_{init} be the set of finished tasks in the initialization stage and N_{init} be the number of tasks in T_{init}, we have:

$$\begin{aligned}
Q(\mathbf{S}^{\#}) \quad &\geq \quad \sum_{t_j \in T_{init}} \sum_{w_i \in W_j} \frac{\sum_{w_k \in W_j - \{w_i\}} q_i(w_k)}{\min(|W_j|, a_j) - 1} \\
&\geq \quad \sum_{t_j \in T_{init}} \sum_{w_i \in W_j} \check{q}_{i,B} \\
&\geq \quad \sum_{t_j \in T_{init}} B\check{q} = N_{init} B \check{q}
\end{aligned} \tag{11}$$

where $\check{q}_{i,B}$ is the lower bound of the average cooperation quality of worker w_i in a set of B workers and $\check{q} = \min_{w_i \in \mathbb{W}_{(\phi)}} \check{q}_i$. For the upper bound of $Q(\hat{\mathbf{S}})$, we have $Q(\hat{\mathbf{S}}) \leq \hat{Q}_{(\varphi)}$ (defined in Equation 9). Thus, $PoA = \frac{Q(\mathbf{S}^{\#})}{Q(\hat{\mathbf{S}})} \geq \frac{N_{init} B \check{q}}{\hat{Q}_{(\varphi)}}$. □

D. Optimization Methods

The game theoretic algorithm shown in Algorithm 3 can keep adjusting the strategy of each worker until a Nash equilibrium is found, where each worker cannot improve his/her own utility (defined in Equation 5) through unilaterally changing his/her strategy. However, the game theoretic algorithm may be slow in solving large scale CA-SC problem instances. Then, we propose two optimization methods to improve the efficiency of the Algorithm 3.

Threshold Stop of the Iteration (TSI). The game theoretic (GT) approach shown in Algorithm 3 is an anytime algorithm (interruptible algorithm) [23], which means it can be interrupted at anytime and a valid solution can still be returned. GT is expected to achieve a joint strategy with a higher total cooperation score, when the more time it keeps running. One observation in our experiments is that the increase of the total cooperation score for each round (lines 4 - 6 in Algorithm 3) will become smaller and smaller until convergence. For real world applications, we may stop the iteration when the round increase of the total cooperation score is less than a small value $\epsilon \cdot Q_c$, where Q_c denotes the current total cooperation score and ϵ is a given parameter, which will dramatically reduce the running time of GT approach however only slightly hurt the total cooperation score.

Lazy-Updating of the Best-Responses of Players (LUB). In the line 5 of Algorithm 3, we need to find the best-response strategy of each worker. However, not all the workers' best-response strategies need to be recalculated. Then, one direct optimization of the game theoretic algorithm (shown in Algorithm 3) is to lazy-update the best-response strategies of workers. Here, lazy-updating the best-response strategies means we only recalculate the best-response strategies of workers when their best-response strategies will possibly change. We give two theorems to guide Algorithm 3 to recalculate the best-response strategy of a given worker w_i only when necessary.

Theorem V.3. *Given a worker w_i and his/her current best-response strategy: joining task t_j, if a new worker w_x is assigned to t_j. Worker w_i is possible to change his/her current best-response only when the new worker w_x crowds out anther worker w_y from task t_j and $q_i(w_y) > q_i(w_x)$, which means worker w_i prefers to cooperate with w_y than w_x.*

Proof. Let $\bar{W}_j$ be the workers who are currently assigned to task t_j and $W_j = \bar{W}_j \cup \{w_i\}$. As joining task t_j is the best-response of worker w_i, for any other task t_k, we have $Q(W_j) - Q(W_j - \{w_i\}) > Q(W_k) - Q(W_k - \{w_i\})$.

We assume there exists a worker w'_x, having $q_i(w_y) < q_i(w'_x)$, who replaces worker w_y from task t_j and the best-response strategy of worker w_i will change. We note $\bar{W}'_j = (\bar{W}_j - \{w_x\}) \cup \{w'_x\}$ and $W'_j = \bar{W}'_j \cup \{w_i\}$. Then the utility of assigning worker w_i to task t_j after the switching from worker w_y to worker w'_x is:

$$\begin{aligned}
& Q(W'_j) - Q(W'_j - \{w_i\}) \\
= \; & Q(W'_j - \{w'_x\}) + \sum_{w_z \in W'_j - \{w'_x\}} q_z(w'_x) \\
& -(Q(W'_j - \{w_i, w'_x\}) - \sum_{w_z \in W'_j - \{w_i, w'_x\}} q_z(w'_x)) \\
= \; & Q(W'_j - \{w'_x\}) - Q(W'_j - \{w_i, w'_x\}) + q_i(w'_x) \\
= \; & Q(W_j - \{w_y\}) - Q(W_j - \{w_i, w_y\}) + q_i(w'_x) \\
> \; & Q(W_j - \{w_y\}) - Q(W_j - \{w_i, w_y\}) + q_i(w_y) \\
= \; & Q(W_j) - Q(W_j - \{w_i\})
\end{aligned} \quad (12)$$

Thus, for any other task t_k, we have

$$Q(W'_j) - Q(W'_j - \{w_i\}) > Q(W_k) - Q(W_k - \{w_i\}), \quad (13)$$

which means after using worker w'_x to replace worker w_y from task t_j, the best-response strategy of worker w_i is still to join task t_j. Thus, worker w'_x cannot exist. □

Theorem V.3 tells us when will worker w_i be possibly crowded out from his/her current best-response. Next, we show another situation that worker w_i is attracted to another task.

Theorem V.4. *Given a worker w_i and his/her current best-response: joining task t_j, if a new worker w_x is assigned to another task t_k. Worker w_i is possible to change his/her current best-response strategy only when 1) no worker is crowded out from t_k; 2) worker w_x crowded out another worker w_y and $q_i(w_y) < q_i(w_x)$.*

The proof of Theorem V.4 is similar to Theorem V.3, as space limitation, we omit the proof here.

Thus, we only need to recalculate the best-response of worker w_i when his/her best-response strategy is possibly changed, which can reduce a lot of unnecessary calculation in line 5 of Algorithm 3.

VI. Experimental Study

In this section, we evaluate the effectiveness and efficiency of our CA-SC approaches through the experiments on both real and synthetic data sets.

A. Experimental Methodology

1) Data Sets: We use both real and synthetic data sets to test our proposed CA-SC approaches. Specifically, for real data set, we use Meetup data set from [13], which was crawled from meetup.com between Oct. 2011 and Jan. 2012. There are 5,153,886 users, 5,183,840 events, and 97,587 groups in Meetup, where each user is associated with a location, each group is associated with a set of users who joined in, and each event is associated with a location to held. Since workers are unlikely to move between two distant cities to conduct one spatial task, and the constraint of deadline of tasks also prevents workers from moving too far, we only consider those user and event records located in the same city. Specifically, we select one meetup popular city, Hong Kong, and extract Meetup records from the area of Hong Kong (with latitude from 22.209° to 22.609° and longitude from 113.843° to 114.283°), in which we obtain 1,282 tasks and

TABLE II
EXPERIMENTAL SETTINGS.

Parameters	Values
capacity a_j of tasks	**3**, 4, 5, 6
range $[v^-, v^+]$ of worker speeds (%)	[1, 3], [1, 5], **[1, 8]**, [1, 10]
range $[r^-, r^+]$ of areas of workers (%)	[1,5], [5, 10], **[10, 15]**, [15, 20]
remaining time τ_j of tasks	1, 2, **3**, 4, 5
threshold parameter ϵ	0, 0.01, 0.03, **0.05**, 0.08
number, m, of workers in each round	500, 800, **1K**, 2K, 5K
number, n, of tasks in each round	100, 300, **500**, 800, 1K
number, R, of total rounds	**10**
least required number, B, of workers	**3**

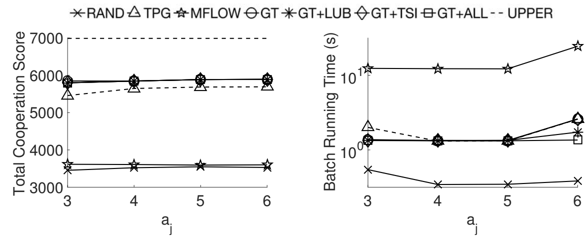
3,525 workers. We use the locations of users and events to initialize the locations of workers and tasks, respectively. For simplicity, we linearly map check-in locations from Gowalla and Foursquare into a $[0, 1]^2$ space. To estimate the cooperation quality scores of worker-pairs, for the cooperation quality score $q_i(w_k)$ of worker w_i to worker w_k, we configure it as $q_i(w_k) = 0.5 * 0.5 + 0.5 * \frac{c_{ik}}{C_{ik}}$ (i.e., let $\alpha = \omega = 0.5$ and $s_j = 1$ in Equation 1), where c_{ik} is the number of common attended groups by workers w_i and w_k and C_{ik} is the number of union groups attended by w_i or w_k. We assume that the more two workers commonly attend groups, the better they can cooperate. In each round, we uniformly sample required number of workers and tasks from the meetup dataset.

For synthetic data, we generate the locations of workers and tasks in a 2D data space $[0, 1]^2$ following either Uniform (UNIF) or Skewed (SKEW) distribution. For Uniform distribution, we uniformly generate the locations of workers/tasks in the 2D data space. As for the Skewed distribution, we first locate 80 % of them into a Gaussian cluster (centered at (0.5, 0.5) with variance = 0.2^2) and distribute the rest uniformly in the 2D data space.

For both real and synthetic data sets, we simulate the velocity of each worker with Gaussian distribution within range $[v^-, v^+]$, for $v^-, v^+ \in (0, 1)$. For the working range of each worker w_i, we generate r_i with Gaussian distribution within range $[r^-, r^+]$, for $r^-, r^+ \in (0, 1)$. Here, for the Gaussian distributions on different ranges, we linearly map data samples within $[-1, 1]$ of a Gaussian distribution $\mathcal{N}(0, 0.2^2)$ to a target ranges.

CA-SC Approaches and Measures. We conduct experiments to compare our approaches, TPG and GT with two baseline algorithms: a maximum-flow based method [11], namely MFLOW, and a random method, namely RAND. For the GT approach, we also test its three variants: GT+LUB, GT+TSI and GT+ALL by using two optimization methods separately and jointly.

Specifically, TPG first iteratively assigns a worker-pair with the highest cooperation quality score to the most suitable task having not be assigned with any workers, then keeps selecting the "best" worker-and-task pair with the maximum total cooperation quality increase until all the tasks are assigned with enough workers or every available workers has been assigned to his/her most suitable task. Then, the GT algorithm iteratively adjusts the workers' strategies based on their current best-response strategies until a Nash equilibrium is found, which leads to a high overall cooperation score. To improve the efficiency of GT, we propose two optimization methods, LUB and TSI in Section V-D. Here, LUB avoids unnecessary recalculation of best-response strategies of workers and TSI stops the iterations of GT when the total cooperation score increase ratio is less than a predefined value ϵ. GT+LUB and GT+TSI use LUB and TSI on GT separately and GT+ALL applies the two optimization methods jointly. To evaluate our approaches, we need to compare the achieved results with the optimal results. However, as proved in Theorem II.1, CA-SC is NP-hard, and infeasible to calculate the real optimal results. Alternatively, we show the effectiveness of our approaches by comparing with a maximum-flow based method [11], namely MFLOW, and a random (RAND) method. In MFLOW, each batch processing is transfered to a maximum flow problem, then an assignment with the maximum number of valid worker-and-task pairs is generated for the batch processing [11]. As for RAND, it randomly chooses a task, and then randomly assigns a set of valid workers to it. In addition, we report the upper bound (UPPER) estimated with Equation 9.



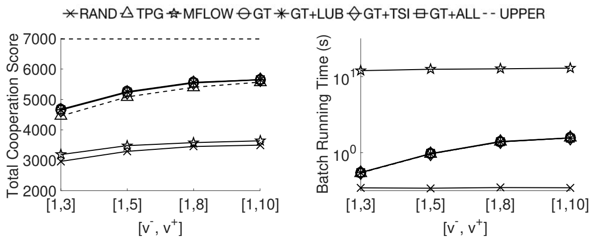
(a) Total Cooperation Score (b) Running Time
Fig. 2. Effect of the Capacity a_j of Tasks.

Table II summarizes our experimental settings, where the default values of parameters are in bold font. In each set of experiments, we vary one parameter, while keeping other parameters to their default values. For each experiment, we report the running time and the total cooperation quality revenue. All our experiments were run on an Intel Xeon X5675 CPU @3.07 GHZ with 32 GB RAM in Java.

B. Experiemnts on Real Data

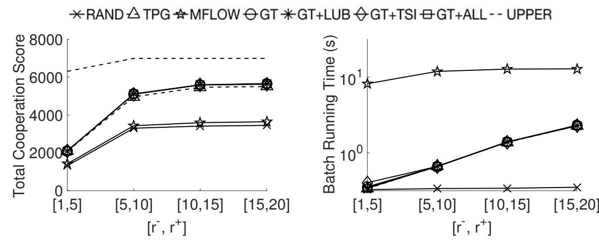
In this section, we present the effects of the capacity of tasks a_j, the range of workers' speeds $[v^-, v^+]$, the range of the working areas of workers $[r^-, r^+]$ and the remaining time of tasks τ_j on the real dataset.

Effect of the capacity a_j of tasks. Figure 2 shows the experimental results on different capacities of tasks from 3 to 5 while the other parameters are in the default values in Table II. In Figure 2(a), the total cooperation scores of all the tested approaches first increase then stay high when the capacities of tasks increases from 3 to 5. When the capacities of tasks become 4, each worker can cooperate with more workers, which leads to the average cooperation utility of each worker increases. However, there are only a limited number of workers in each batch and only a small number of more tasks can be finished when the capacities of tasks increases from 4 to 5. Thus, the total cooperation scores of approaches have only a slight increase when the capacities of tasks become higher than 4. In addition, we can see our CA-SC approaches, GT and GT variants (GT+LUB, GT+TSI and GT+ALL) achieve about 5% higher the total cooperation scores than TPG, which are all significantly higher than that of MFLOW and RAND. It shows the effectiveness of our TPG and the game theoretic



(a) Total Cooperation Score (b) Running Time

Fig. 3. Effect of the Range of Workers' Moving Speeds $[v^-, v^+]$.



(a) Total Cooperation Score (b) Running Time

Fig. 4. Effect of the Range of Working Areas $[r^-, r^+]$.

framework. The total cooperation scores of GT and its variants are close to each other, which shows our optimization methods in Section V-D almost do not decrease the total cooperation scores of the GT variants compared with that of GT. In addition, the total cooperation quality revenue achieved by our approaches are close to UPPER, which can show the effectiveness of our approaches. In Figure 2(b), we present the running times of all the approaches when the capacities of tasks increase. RAND runs fastest among all the tested approaches and TPG is slower than RAND (lower than 5 seconds for each batch). In addition, our GT and its variants are slower than RAND and TPG. In addition, MFLOW is the slowest one. Through comparing the running times of GT and its variants, we can see the two optimization methods in Section V-D can improve the running speed of GT when they are used separately. When the two optimization methods are used together, GT+ALL runs the fastest among GT related approaches. The results show the effectiveness of our GT framework. Most importantly, the two optimization methods reduce the running times of GT but almost do not reduce the total cooperation scores of GT.

Effect of the range of the moving speeds of workers $[v^-, v^+]$. In Figure 3(a), when the range of moving speeds of workers increases, the total cooperation scores of all the tested approaches increases. The reason is that the faster the workers move, the more tasks they can reach within fixed remaining time of tasks. Then the approaches can have more valid worker-and-task pairs to select and find the assignments with higher total cooperation scores. GT and its variants still can achieve about 5% higher total cooperation quality revenue than TPG, which are close to UPPER (from 50% to 78%) and much higher than that of MFLOW and RAND In Figure 3(b), when the range of the moving speeds of workers increases, the running times of all the tested approaches, except for MFLOW, also increase, which is because that the number of valid worker-and-task pairs increases when workers move faster. As the numbers of workers and tasks do not change, the complexity and running time of MFLOW, a maximum flow based algorithm, will not change. GT and its variants are slower than RAND but faster than MFLOW. All our approaches run fast enough to response to the task requests within 2 seconds in each batch.

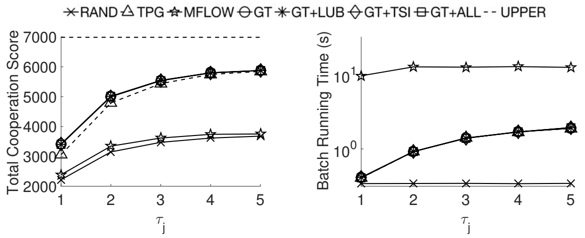
Effect of the range of the working areas of workers $[r^-, r^+]$. Figure 4 presents the experimental results of all approaches when the range of working areas of workers varying from [0.05, 0.1] to [0.2, 0.25]. In Figure 4(a), when the range of working areas of workers increases from [0.05, 0.1] to [0.1, 0.15], the total cooperation quality revenues achieved by the tested approaches first increase; then they stop growing when the range of the working areas increasing from [0.1, 0.15] to [0.2, 0.25]. The reason is that, at the beginning, with the increase of working areas range, workers can reach more tasks before their deadlines. However, since the moving speeds of workers and the remaining times of tasks stay unchanged, which prevents the workers from reaching tasks that are too far away. Similarly, TPG, GT and GT variants can achieve much higher total cooperation scores than MFLOW and RAND. In Figure 4(b), the running times of all the approaches increase when the working areas of workers get enlarged. GT and its variants run slower than RAND and TPG but faster than MFLOW. GT runs the slowest and still can find a Nash equilibrium within 4 seconds.

Effect of the remaining time τ_j of tasks. Figure 5 shows the experimental result of all the approaches when the remaining time τ_j of tasks varying from 1 to 5. Specifically, in Figure 5(a), when the remaining time of tasks increases from 1 to 3, the total cooperation scores achieved by the approaches first increase; then they stop growing for the remaining time of tasks increasing from 3 to 5. The reason is that, at the beginning, with the increase of remaining times of tasks, workers can reach more tasks before their deadlines. However, as the working areas of workers stay unchanged, which prevents the workers from reaching more tasks outside their working areas. TPG, GT and GT variants can achieve much higher total cooperation scores than RAND and MFLOW. In Figure 5(b), when the remaining times of tasks increase, the running times of GT and its variants increase slightly. RAND is faster than other approaches. GT+ALL is the fastest among GT related approaches, which shows the effectiveness of the optimization methods in Section V-D.

C. Experiments on Synthetic Data

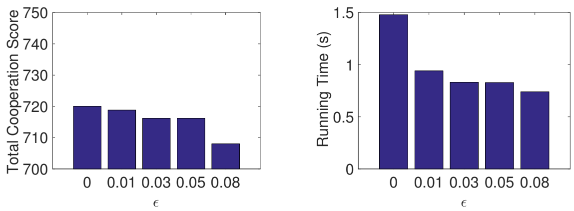
In this section, we show the effect of the threshold parameter ϵ of the TSI optimization method for GT. Then, we examine the effectiveness and scalability of our CA-SC approaches by varying the number of workers m and the number of tasks n on our synthetic data sets.

Effect of the threshold parameter ϵ. In Figure 6, we show the effect of the threshold parameter ϵ of the TSI optimization method for GT by showing the results of GT+TSI with different values of ϵ varying from 0 to 0.08. In Figure 6(a), the total cooperation scores of GT+TSI with different values of ϵ can achieve similar results. Particularly, only when $\epsilon = 0.08$,



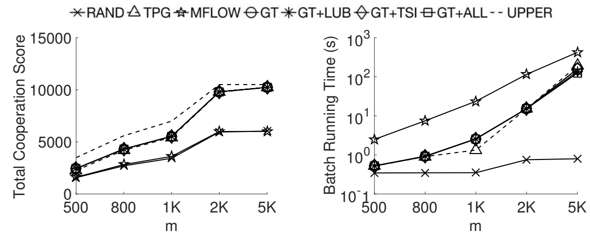
(a) Total Cooperation Score (b) Running Time

Fig. 5. Effect of the Remaining Time τ_j of Tasks.



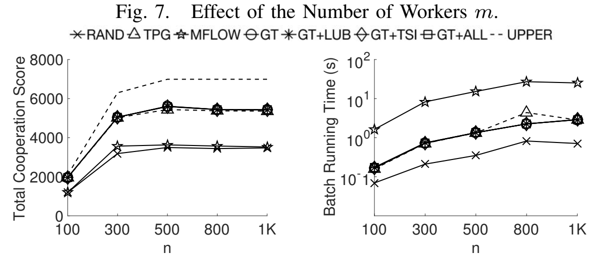
(a) Total Cooperation Score (b) Running Time

Fig. 6. Effect of the threshold parameter ϵ.



(a) Total Cooperation Score (b) Running Time

Fig. 7. Effect of the Number of Workers m.



(a) Total Cooperation Score (b) Running Time

Fig. 8. Effect of the Number of Tasks n.

there is a noticeable decrease on the total cooperation score. In Figure 6(b), the running time of GT+TSI will decrease when the threshold parameter ϵ increases. The reason is that the larger ϵ is, the less iterations GT+TSI runs. Thus, in the default setting, we configure $\epsilon = 0.05$, such that the running speed of GT+TSI is fast while the total cooperation scores achieved by GT+TSI are close to the scores achieved by GT.

Effect of the number of workers m. Figure 7 illustrates the effect of the number, m, of workers in each batch by varying it from 500 to 5,000 over synthetic data sets while the other parameters are set as their default values. In Figure 7(a), the tested approaches can achieve higher scores when the number of workers in each batch increases from 500 to 2,000. The reason is that, at the beginning, more workers can finish more tasks. However, when the number of workers reaches 2,000, it is already sufficient to complete all the tasks. GT and its variants can achieve almost similar total cooperation scores compared to each other but much higher results than RAND and MFLOW. We can also see that when m increases, our approaches can achieve total cooperation quality revenues closer to UPPER (97% in best). In Figure 7(b), when the number of workers increases, the running times of all tested approaches also increase, which is because more workers need to be checked by the tested approaches. RAND still runs the fastest and MFLOW runs the slowest.

Effect of the number of tasks n. In Figure 8(a), all tested approaches can achieve higher cooperation scores when the number of tasks in each batch increases from 100 to 500; then the total cooperation scores have almost no increase when the number of tasks in each batch increases from 500 to 1,000. The reason is at the beginning, the increase of the number of tasks leads to more workers are assigned to tasks, which results in higher total cooperation scores. However, there are only 1,000 workers in each batch. When the number of tasks reaches 500, almost all the workers have been assigned to tasks. Thus, after the number of tasks reaches 500, more tasks will not lead to the increase of the total assigned workers. Similarly, TPG, GT and GT variants can achieve much higher total cooperation scores than RAND and MFLOW. In Figure 8(b), the increase of the number of tasks causes the running times of all the approaches increase. The reason is that more tasks need more time to maintain the valid worker-and-task pairs and select the best task for each worker. GT+ALL is faster than GT and only slightly slower than RAND.

In summary, on both real and synthetic data sets, our TPG, GT and GT variants can achieve much higher total cooperation scores than MFLOW and RAND. Comparing with UPPER, our approach can achieve from 50% to 97% of the upper bounded total cooperation quality revenue, which shows the effectiveness of our approaches. Most importantly, our optimization methods for GT can improve its running speed but only reduce a very small portion of its total cooperation score, which shows the effectiveness of GT and the two optimization methods.

VII. Related Work

Spatial Crowdsourcing. The spatial crowdsourcing systems [4], [11], [31] require workers to move to specific locations of spatial tasks to perform tasks subjected to the various constraints. In [11], based on the publishing models, there are two modes, *worker selected tasks* (WST) [8] mode and *server assigned tasks* (SAT) [5], [6], [11], [25], [33] mode, in which tasks are selected by or assigned to workers respectively. In addition, for the SAT mode, from processing styles, there are two kinds of server assigned tasks modes: online task assignment mode and batch-based task assignment mode. Specifically, in the online task assignment mode [25], [28], the spatial crowdsourcing servers need to immediately assign valid tasks to workers upon the reaching of workers in a one-by-one style. However, in the batch-based task assignment mode [5], [6], [11], [32], [33], the servers periodically assign a set of tasks to a set of workers.

Our CA-SC problem is in the batch-based SAT mode. Prior studies in the SAT mode [5], [6], [11], [32], [33] have different goals, such as maximizing the number of assigned tasks on the server side [11], maximizing the reliable-and-diversity score of assignments [6], maximizing the acceptance rate on the worker side [33], or maximizing the total assigned number of tasks while the workers can arrive at their destinations before their deadlines [32]. In contrast, our CA-SC problem in this paper assigns workers to spatial tasks under constraints of working areas of workers, and deadlines and capacities of tasks with a different goal of maximizing the overall cooperation score. Here, the higher the cooperation score is, the better the workers complete the tasks together. Thus, the existing solution cannot be applied to our problem directly. As a result, we propose a game theoretic approach with two optimization methods for our CA-SC problem that maximizes the total cooperation score under constraints of working areas, deadlines and capacities.

Ridesharing. Ridesharing allows passengers to share vehicles together to alleviate the public traffic congestion and to monetarily benefit drivers and passengers, when their travel routes are similar. Ridesharing requires the drivers to pick up passengers and send them to their destinations, however, in spatial crowdsourcing workers only need to conduct the tasks at the task-specific locations. In [15], they proposed one framework to handle the online taxi-sharing problem, where riders and taxis keep arriving and leaving. The framework will schedule a most suitable vehicle to serve the passenger when he/she joins the platform such that the time window and monetary constraints are satisfied. In [10], the researchers proposed a kinetic tree structure to trace the valid schedule plans for each vehicle. In [30], the authors consider the fairness in ridesharing, which means the payments of the passengers are according to the fair plan (e.g., a stable matching between riders), and often even lower. In [7], the authors focused on the utility of the ridesharing scheduling, which includes the vehicle-related utility, rider-related utility and trajectory-related utility. In [34], the authors focused on maximizing the profit of the ridesharing platform.

VIII. Conclusion

In this paper, we formalize the problem of cooperation-aware spatial crowdsourcing (CA-SC) problem, which assigns a set of moving workers to a set of time and capacity constrained spatial tasks, such that the tasks can be accomplished with high cooperation qualities. We prove that the CA-SC problem is NP-hard by reducing it from a well-known NP-hard problem, k-set packing problem (k-SP) and then propose a greedy based approach and a game theoretic approach with two optimization methods to solve it. Extensive experiments have been conducted to show the efficiency and effectiveness of our CA-SC approaches on both real and synthetic data sets.

IX. Acknowledgment

The work is partially supported by the Hong Kong RGC GRF Project 16207617, the National Science Foundation of China (NSFC) under Grant No. 61729201, Science and Technology Planning Project of Guangdong Province, China, No. 2015B010110006, Hong Kong ITC ITF grants ITS/425/16FX and ITS/212/16FP, and Microsoft Research Asia Collaborative Research Grant. Peng Cheng is the corresponding author.

References

[1] *TaskRabbit*. https://www.taskrabbit.com.
[2] E. Althaus, M. Blumenstock, et al. Algorithms for the maximum weight connected k-induced subgraph problem. In *COCOA*. Springer, 2014.
[3] T. Basar and G. J. Olsder. *Dynamic noncooperative game theory*, volume 23. Siam, 1999.
[4] Z. Chen, R. Fu, Z. Zhao, Z. Liu, L. Xia, L. Chen, and et al. gmission: A general spatial crowdsourcing platform. *PVLDB*, 2014.
[5] P. Cheng, X. Lian, L. Chen, J. Han, and J. Zhao. Task assignment on multi-skill oriented spatial crowdsourcing. *TKDE*, 2016.
[6] P. Cheng, X. Lian, Z. Chen, R. Fu, L. Chen, et al. Reliable diversity-based spatial crowdsourcing by moving workers. *PVLDB*, 2015.
[7] P. Cheng, H. Xin, and L. Chen. Utility-aware ridesharing on road networks. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1197–1210. ACM, 2017.
[8] D. Deng, C. Shahabi, et al. Maximizing the number of worker's self-selected tasks in spatial crowdsourcing. In *SIGSPATIAL GIS*, 2013.
[9] R. Gibbons. An introduction to applicable game theory. *Journal of Economic Perspectives*, 11(1):127–149, 1997.
[10] Y. Huang, F. Bastani, R. Jin, and X. S. Wang. Large scale real-time ridesharing with service guarantee on road networks. *PVLDB*, 2014.
[11] L. Kazemi and C. Shahabi. Geocrowd: enabling query answering with spatial crowdsourcing. In *SIGSPATIAL GIS*, 2012.
[12] Q. Liu, T. Abdessalem, H. Wu, et al. Cost minimization and social fairness for spatial crowdsourcing tasks. In *DASFAA*. Springer, 2016.
[13] X. Liu, Q. He, et al. Event-based social networks: linking the online and offline social worlds. In *ACM SIGKDD 2012*. ACM.
[14] A. C. Lozano, H. Li, and et al. Spatial-temporal causal modeling for climate change attribution. In *SIGKDD*. ACM, 2009.
[15] S. Ma, Y. Zheng, and O. Wolfson. Real-time city-scale taxi ridesharing. *TKDE*, 2015.
[16] E. Maskin. Nash equilibrium and welfare optimality*. *Harvard Institute of Economic Research Working Papers*, 66(1):23–38, 1998.
[17] D. Monderer and L. S. Shapley. Potential games. *Games and economic behavior*, 14(1):124–143, 1996.
[18] R. B. Myerson. *Game theory*. Harvard university press, 2013.
[19] J. F. Nash. Equilibrium points in n-person games. *PNAS*, 1950.
[20] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani. *Algorithmic game theory*. Cambridge university press, 2007.
[21] M. J. Osborne. *An introduction to game theory*, volume 3. Oxford university press New York, 2004.
[22] L. Pournajaf, L. Xiong, V. Sunderam, et al. Spatial task assignment for crowd sensing with cloaked locations. In *MDM*. IEEE, 2014.
[23] T. Rahwan, S. D. Ramchurn, et al. An anytime algorithm for optimal coalition structure generation. *JAIRs*, 34:521–567, 2009.
[24] H. Samet. *The design and analysis of spatial data structures*, volume 199. Addison-Wesley Reading, MA, 1990.
[25] S. Tianshu, T. Yongxin, W. Libin, S. Jieying, and et al. Trichromatic online matching in real-time spatial crowdsourcing. *ICDE*, 2017.
[26] H. To, L. Fan, et al. Real-time task assignment in hyperlocal spatial crowdsourcing under budget constraints. In *PerCom*. IEEE, 2016.
[27] H. To, C. Shahabi, and L. Kazemi. A server-assigned spatial crowdsourcing framework. *TSAS*, 2015.
[28] Y. Tong, J. She, B. Ding, L. Wang, and L. Chen. Online mobile micro-task allocation in spatial crowdsourcing. *ICDE*, 2016.
[29] V. V. Vazirani. *Approximation algorithms*. Springer Science & Business Media, 2013.
[30] O. Wolfson and J. Lin. Fairness versus optimality in ridesharing. In *MDM*, pages 118–123. IEEE, 2017.
[31] H. Yin, Y. Sun, B. Cui, and et al. Lcars: a location-content-aware recommender system. In *SIGKDD*. ACM, 2013.
[32] Y. Zhao, Y. Li, Y. Wang, H. Su, and K. Zheng. Destination-aware task assignment in spatial crowdsourcing. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 297–306. ACM, 2017.
[33] L. Zheng and L. Chen. Maximizing acceptance in rejection-aware spatial crowdsourcing. *TKDE*, 2017.
[34] L. Zheng, L. Chen, and J. Ye. Order dispatch in price-aware ridesharing. *PVLDB*, 11(8):853–865, 2018.