This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.2987881, IEEE Transactions on Mobile Computing

Exploiting Multi-Dimensional Task Diversity in Distributed Auctions for Mobile Crowdsensing

Zhipeng Cai, *Senior Member*, IEEE, Zhuojun Duan, Wei Li, *Member*, IEEE

***Abstract*—To promote development of Mobile Crowdsensing Systems (MCSs), numerous auction schemes have been proposed to motivate mobile users' participation. But, task diversity of MCSs has not been fully explored by most existing works. To further exploit task diversity and improve performance of MCSs, in this paper, we investigate *the joint problem of sensing task assignment and schedule with considering multi-dimensional task diversity, including partial fulfillment, bilaterally-multi-schedule, attribute diversity, and price diversity*. First, task owner-centric auction model is formulated and two distributed auction schemes (CPAS and TPAS) are proposed such that each task owner can locally process auction procedure. Then, mobile user-centric auction model is established and two distributed auction schemes (VPAS and DPAS) are developed to facilitate local auction implementation. These four auction schemes differ in their approaches to determine winners and compute payments. We further rigorously prove that all the four auction schemes (CPAS, TPAS, VPAS, and DPAS) are computationally-efficient, individually-rational, and incentive-compatible and that both CPAS and TPAS are budget-feasible. Finally, we comprehensively evaluate the effectiveness of CPAS, TPAS, VPAS, and DPAS via comparing with the state-of-the-art in real-data experiments.**

***Index Terms*—Mobile crowdsensing system; truthful auction; task assignment; task schedule; distributed algorithm.**

I. Introduction

The recent years have witnessed the extraordinary progresses of *Mobile Crowdsensing Systems* (MCSs) which significantly advance data collection and sharing via motivating mobile users' participation in sensing activities. As the core component of MCSs, mobile users process sensing activities using various mobile devices (e.g., smartphones and tablets) and are paid rewards to compensate their costs such as resource consumption (*e.g.*, energy and bandwidth) and privacy leakage (*e.g.*, location exposure [1]), and others. Since present mobile devices (smartphones, tablets and vehicle-embedded sensing devices (GPS)) are embedded more computing, communication, and storage resources than traditional mote-class sensors [2], the major superiorities of MCSs over the traditional mote-class sensor networks lie in the *reduced cost* to deploy specialized sensing infrastructures as well as the *enhanced applicability* to a variety of real applications that demand resources and sensing modalities beyond the current mote-class sensor processes. MCSs have been widely applied to traffic congestion detection, wireless indoor localization, pollution monitoring, *etc* [2]–[4]. Such power of MCSs roots at the active participation of mobile users to collect and share sensory data. Thus, to become more efficient and applicable, MCSs should first deal with the critical problem: "*how to motivate mobile users to perform sensing tasks?*"

To encourage mobile users to join MCSs, a number of auction-based incentive mechanisms have been developed from various aspects to assign and schedule sensing tasks [3], [5]–[18]. Nevertheless, the following crucial issues have been overlooked in most of existing works. (i) In many auction models [13], [15]–[20], a task is assigned to a mobile user only if the task can be completely implemented by the mobile user, which is impractical in many scenarios. As a matter of fact, as a mobile user's available working duration in an MCS is limited, a task may not be completed by one mobile user at a time (*e.g.*, pollution monitoring within a specific area during a long-term period). (ii) Task diversity in MCSs is not fully investigated. On one hand, sensing tasks may have different requirements in terms of location, implementation duration, types of sensors, and so on; on the other hand, mobile users also vary in their locations, available working duration, equipped sensors. *Ignoring task diversity in MCSs may lead to inefficient task assignment and ineffective task implementation.* (iii) Moreover, due to the aforementioned diversities of task requirement and user availability, the prices demanded by a mobile user to process different tasks are also different. Since auction is a kind of market-based scheme, this price diversity should be well formulated in auction models. (iv) Last but not least, with the dramatic increase in the scale of MCSs, it becomes more difficult and expensive to find a centralized institution authorized by third party to dominate the auction process [3], [21].

Inspired by the above challenges, the intent of this paper is to solve *the joint problem of task assignment and scheduling in MCSs taking into task diversity from different dimensions*. (i) **Partial fulfillment**, which means a task can get assignment if it can be partially completed by mobile users in time domain. For instance, a task requests sensory data at a certain location from 9:00am to 11:00am, a mobile user who is the only user can collect the required data from 9:30am to 10:30am, and then the task is assigned to the mobile user. (ii) **Bilaterally-multi-schedule**, where one mobile user can process multiple tasks in both the time and space domains while a task can be scheduled to multiple mobile users in time domain, further improving task assignment efficiency. Partial fulfillment and bilaterally-multi-schedule together reflect diverse assignment in time and space domains. (iii) **Attribute diversity**, which indicates that the task requirement and the user availability vary in task attributes in terms of location, time duration, and

Zhipeng Cai and Wei Li are with the Department of Computer Science, Georgia State University, Atlanta, GA 30303, USA. E-mail: *zcai@gsu.edu, wli28@gsu.edu*.

Zhunjun Duan is with the Computing Sciences Department, University of Hartford, West Hartford, CT 06117, USA. E-mail: *duan@hartford.edu*.

types of sensors. (iv) **Price diversity**, which says that every mobile user could require different prices to carry out different tasks. The existing auction schemes [3], [13]–[18], [22] do not consider task schedule in the time domain and thus cannot be applied to solve our problem. More importantly, extending the existing auctions to enable partial fulfillment, bilaterally-multi-schedule, attribute diversity, and price diversity is nontrivial. As a result, it is challenging to develop auction schemes to possess the above four features.

To tackle this challenge, four auction schemes are elaborately designed to satisfy different application requirements. First, we model the proposed joint problem as a reverse auction with task owners being auctioneers, where partial fulfillment, attribute diversity, and price diversity are considered. Within such a framework, two distributed auction schemes, cost-preferred auction scheme (CPAS) and time schedule-preferred auction scheme (TPAS), are proposed. Then, to achieve bilaterally-multi-schedule, we mathematically formulate the proposed joint problem to be an auction for mobile users working as auctioneers. Correspondingly, two distributed auction schemes, valuation-preferred auction scheme (VPAS) and distance-preferred auction scheme (DPAS), are developed. These four auction schemes differ in their approaches to determine winners and compute payments. By conducting thorough theoretical analysis, we prove that all of CPAS, TPAS, VPAS, and DPAS can achieve computational-efficiency, individual-rationality, and incentive-compatibility and that both of CPAS and TPAS can also ensure budget-balance. Meanwhile, these four auction schemes differ in their approaches to determine winners and payments as well as their computational complexities, so that they can adapt to different application requirements, for which the major properties of the four proposed auction schemes are summarized in Table I. Furthermore, via intensive real-data experiments, the performance of our proposed auction schemes are validated. Our innovative contributions are addressed below:

- To the best of our knowledge, this is the first work to study auction models to achieve partial fulfillment, bilaterally-multi-schedule, attribute diversity, and price diversity for task assignment and schedule in MCSs.
- Distributed auction framework is designed to facilitate task owners/mobile users to locally control their auctions without collecting global information in MCSs, enhancing scalability of MCSs and reducing communication cost at the side of cloud platform.
- A cost-preferred auction scheme (CPAS) is proposed to assign each winning mobile user multiple working durations and a time schedule-preferred auction scheme (TPAS) is proposed to allocate each winning mobile user one continuous working duration.
- A valuation-preferred auction scheme (VPAS) and a distance-preferred auction scheme (DPAS) are developed, in which each mobile user can schedule a series of tasks for implementation according to task values and locations, respectively.
- In-depth theoretical analysis is performed to prove the properties of our proposed auction schemes in terms of computational-efficiency, individual-rationality, budget-feasibility, and truthfulness.
- A comprehensive comparison with the state-of-the-art are well conducted in real-data experiments to evaluate the performance of our proposed auction schemes in terms of allocation efficiency, working time utilization, STOs' cost, MUDs' valuation, and truthfulness.

TABLE I
SUMMARY OF AUCTION SCHEMES PROPOSED IN THIS PAPER.

Auction schemes	Individual-rationality	Budget-balance	Incentive-compatibility	Time complexcity
CPAS	✓	✓	✓	$O(n^3 \log(n))$
TPAS	✓	✓	✓	$O(n^2 \log(n))$
VPAS	✓	N/A	✓	$O(m^2 \log(m) + nl)$
DPAS	✓	N/A	✓	$O(m \log(m) + nl)$

The rest of this paper is organized as follows. We briefly summarize the related work in Section II. At the side of task owners who act as auctioneers, the system model and problem formulation are presented in Section III, and two auction schemes are proposed in Section IV and Section V, respectively. Next, we formulate the problem for mobile users acting as auctioneers in Section VI and present two auction schemes in Section VII and Section VIII, respectively. After evaluating the performance of the four proposed auction schemes in Section IX, we conclude this paper in Section X.

II. Related Work

The existing auction-based task assignment mechanisms for mobile crowdsensing are briefly summarized in this section.

Traditionally, auction mechanism can be controlled by an auctioneer in a centralized fashion [13]–[17], [23]–[28]. In [13], a reverse auction was proposed for the cloud platform to minimize cost, in which the service coverage of smartphones are taken into account. Jin *et al.* [14] designed a single-minded and a multi-mined reverse combinatorial auctions that can obtain sub-optimal social welfare by considering the information quality of mobile users. In [15], with considering mobile user dynamics and randomness of tasks, an offline auction and an online auction were proposed. In [16] three auction schemes were developed for three different scenarios of mobile crowdsensing, including single-requester single-bid model, single-requester multiple-bid model, and multiple-requester multiple-bid model. Ji *et al.* [17] investigated the discretization in crowdsensing systems and designed two auction mechanisms, in which each user has a uniform subtask length. In [23], two incentive mechanisms were designed to reduce the waste of sensory data due to the spatial correlation of different mobile participants. In [24], the quality of sensing is considered in platform's valuation function, and a budget-feasible auction was designed with approximating ratio approaching $\frac{2e}{e-1}$. Tang *et al.* [25] developed an integrated framework by combining a double auction-based incentive mechanism and a data aggregation mechanism, which not only achieves truthfulness, individual rationality, computational efficiency, and non-negative social welfare, but also generates high accuracy in the data aggregation results. In [26], the incentive mechanism maximized the number of recruited users and the utility functions. Khaledi *et al.* [27] proposed a multi-dimensional auction to make task allocation among a set mobile nodes in mobile cloud computing. In [28], a truthful double auction mechanism was proposed to reach max-min fairness. But, each of the above auction models just simply considered one assignment condition, such as task's location [13], [23], [26], [27], quality requirement [14], demand computing resource [25] and required sensing time [17], and has limitation on fully exploiting task diversity for practical applications.

To enhance the performance of mobile crowdsensing, some distributed incentive mechanisms have been designed [3], [21], [29]. In [3], the authors first formulated the problem of task selection for mobile users as a non-cooperative task selection game and then investigated the equilibriums and convergence of the game. In the proposed game, the objective is to maximize each mobile user's utility by finding an order to complete one or more sensing tasks that locate at different places. To maximize social welfare, Duan *et al.* [21] proposed a distributed algorithm, in which both task's demand and mobile user's time allocation strategy change over the prices published by a centralized cloud platform, and the centralized cloud platform updated the prices according to the demands of tasks and supplies of mobile users. In [29], a multi-stage stochastic programming approach is designed based on distributed game theoretic methodology under the multi-platform and multi-user scenario, which stops once Walrasian equilibrium is reached.

Different from the prior works, in this paper, we propose to design more practical distributed truthful auction schemes for task assignment and schedule in MCSs by fully exploring multi-dimensional task diversity, including *partial fulfillment*, *attribute diversity*, *price diversity*, and *bilaterally-multi-schedule*, which can further advance multi-dimensional diversity in mobile crowdsensing systems. More importantly, when the auction economic properties should be simultaneously achieved, performing a distributed auction to satisfy the needs of multi-dimensional diversity is not a trivial problem.

III. Auction Formulation for Task Owners

A. System Model

The MCS consists of a set of task owners (STOs) to demand sensing service, a set of mobile users who are equipped with smart devices (MUDs) to supply sensory data, and a cloud platform that provides connection and information announcement for STOs and MUDs.

There exist m STOs, each of which requests a task implementation, denoted by $\Pi = \{\pi_1, \pi_2, \cdots, \pi_m\}$ the set of tasks. Since every STO has only one task request, "STO i's task" and "task π_i" are interchangeable in this paper. Each task π_i has four attributes, including *locations*, *starting time*, *ending time*, and *resources* (e.g., camera and gyroscope), which imply the STOs' requirements to process task. Each STO i's *sensing task information* can be expressed by $f_i^\pi = (L_i^\pi, [\alpha_i^\pi, \beta_i^\pi], R_i^\pi)$, where L_i^π is the required location of task π_i, α_i^π and β_i^π are respectively the starting time and the ending time of task π_i, and R_i^π is a set of required sensor resources to process task π_i. Each STO i also has a budget b_i to complete task π_i per unit time. Let $\Gamma = \{\gamma_1, \gamma_2, \cdots, \gamma_n\}$ be the set of MUDs, in which each MUD γ_j is allowed to work for at most one STO. Similarly, each MUD γ_j has initial location L_j^γ, available starting time α_j^γ, available ending time β_j^γ, a set of sensor resources R_j^γ in smart device, and an average moving rate λ_j^γ. The formal form of each MUD γ_j's *sensing service information* is represented as $f_j^\gamma = (L_j^\gamma, [\alpha_j^\gamma, \beta_j^\gamma], R_j^\gamma, \lambda_j^\gamma)$. The cost of each MUD γ_j to implement task π_i contains two parts: i) moving cost from location L_j^γ to location L_i^π; and ii) resource consumption cost on smart device. To cover the task implementation cost, each MUD γ_j submits a price vector $A_j =< a_{1j}, a_{2j}, \cdots, a_{mj} >$ where a_{ij} ($1 \le i \le m$) is an asking price per unit time slot.

As MUD γ_j may need to move from L_j^γ to L_i^π to process the required task, we should calculate the actual starting time. Given the Euclidean distance $d(L_i^\pi, L_j^\gamma)$ and the moving rate λ_j^γ, MUD γ_j arrives at L_i^π at time $t_{ij}^\alpha = \frac{d(L_i^\pi, L_j^\gamma)}{\lambda_j^\gamma} + \alpha_j^\gamma$. For simplicity, we assume that an MUD can start working as soon as it arrives at a task's required location. Let T_{ij} be MUD γ_j's maximum available duration for performing π_i and $|T_{ij}|$ be the number of time slots within T_{ij}. Then T_{ij} can be calculated according to the following six cases:

- If $t_{ij}^\alpha \ge \beta_i^\pi$, task π_i is finished when MUD γ_j arrives and thus γ_j cannot process π_i, *i.e.*, $T_{ij} = \emptyset$;
- If $\beta_j^\gamma \le \alpha_i^\pi$, task π_i starts when MUD γ_j's available duration ends. Therefore, γ_j cannot process π_i, and $T_{ij} = \emptyset$;
- If $\alpha_i^\pi \le t_{ij}^\alpha < \beta_j^\gamma < \beta_i^\pi$, MUD γ_j arrives when/after task π_i begins, and MUD γ_j's available duration ends before task π_i's ending time. Thus, $T_{ij} = [t_{ij}^\alpha, \beta_j^\gamma]$;
- If $\alpha_i^\pi \le t_{ij}^\alpha < \beta_i^\pi \le \beta_j^\gamma$, MUD γ_j arrives when/after task π_i begins and MUD γ_j's available duration ends when/after task π_i's ending time. Therefore, $T_{ij} = [t_{ij}^\alpha, \beta_i^\pi]$;
- If $t_{ij}^\alpha < \alpha_i^\pi < \beta_j^\gamma < \beta_i^\pi$, MUD γ_j arrives before task π_i begins and MUD γ_j's available duration ends before task π_i's ending time. In this case, $T_{ij} = [\alpha_i^\pi, \beta_j^\gamma]$;
- If $t_{ij}^\alpha < \alpha_i^\pi < \beta_i^\pi \le \beta_j^\gamma$, MUD γ_j arrives before task π_i begins and MUD γ_j's available duration ends when/after task π_i's ending time. As a result, we have $T_{ij} = [\alpha_i^\pi, \beta_j^\gamma]$.

Finally, the notations used in this paper are presented in Table II.

B. Problem Formulation

Suppose MUD γ_j obtains an assigned duration T_{ij}^s for task π_i and the number of time slots within T_{ij}^s is $|T_{ij}^s|$. Since T_{ij} is the maximum available duration of MUD γ_j for task π_i, we have $T_{ij}^s \subseteq T_{ij}$ and $0 \le |T_{ij}^s| \le |T_{ij}|$. The task assignment is indicated by $x_{ij} \in \{0, 1\}$; that is, $x_{ij} = 1$ if and only if π_i is assigned to γ_j. As each MUD γ_j works for at most one STO, there must be $\sum_{i=1}^m x_{ij} \le 1$. If MUD γ_j implements task π_i, γ_j can obtain a payment p_{ij} from STO i and receive a utility U_j^γ:

$$U_j^\gamma = \sum_{i=1}^m u_{ij}^\gamma = \sum_{i=1}^m x_{ij}(p_{ij} - a_{ij}|T_{ij}^s|). \quad (1)$$

TABLE II
LIST OF NOTATIONS USED IN THIS PAPER.

Notation	Description
m	Number of tasks
n	Number of MUDs
Π	Set of tasks
Γ	Set of MUDs
π_i	Task π_i
γ_j	MUD γ_j
f_i^π	π_i's sensing task information
f_j^γ	γ_j's sensing service information
L_i^π	π_i's required location
L_j^γ	γ_j's location
$[\alpha_i^\pi, \beta_i^\pi]$	Starting time and ending time of π_i
$[\alpha_j^\gamma, \beta_j^\gamma]$	Starting time and ending time of γ_j
R_i^π	Set of sensors resources required by π_i
b_i	Budget for π_i
R_j^γ	Set of sensors resources provided by γ_j
λ_j^γ	γ_j's moving rate
a_{ij}	γ_j's asking price per time unit to process π_i
$d(L_i^\pi, L_j^\gamma)$	Euclidean distance between L_i^π, L_j^γ
T_{ij}	γ_j's maximum available duration for performing π_i
T_{ij}^s	γ_j's assigned duration T_{ij}^s for task π_i
$\lvert T_{ij}\rvert$	Number of time slots of T_{ij}
$\lvert T_{ij}^s\rvert$	Number of time slots of T_{ij}^s
x_{ij}	Task assignment indicator

A practical crowdsensing scenario is taken into account, in which each STO i independently decides the winning MUDs and schedules their working time slots. In other words, each STO i works as an auctioneer of its local task auction that is formulated as a *reverse auction*:

$$\min \quad \sum_{j=1}^{n} x_{ij} a_{ij} |T_{ij}^s|; \tag{2a}$$

$$\text{s.t.} \quad \bigcup_{j=1}^{n} x_{ij} T_{ij}^s \subseteq [\alpha_i^\pi, \beta_i^\pi]; \tag{2b}$$

$$\sum_{j=1}^{n} x_{ij} |T_{ij}^s| \leq |\beta_i^\pi - \alpha_i^\pi|; \tag{2c}$$

$$x_{ij} \in \{0, 1\}, 1 \leq j \leq n; \tag{2d}$$

$$T_{ij}^s \subseteq T_{ij}, 1 \leq j \leq n. \tag{2e}$$

In Eq. (2), the objective is to minimize the cost for task assignment while ensuring the following constraints: i) Eq. (2b) requests that the union of scheduled working durations cannot exceed the task's duration; ii) Eq. (2c) implies that the total allocated time slots cannot be more than the number of slots of the task's duration; iii) Eqs. (2d) and (2e) represent the ranges of x_{ij} and T_{ij}^s, respectively.

C. Auction Economic Properties

In this paper, we aim to achieve the following economic properties in each STO's auction [24], [27], [30]:

- **Individual-rationality.** No MUD obtains a negative utility, *i.e.*, $U_j^\gamma \geq 0$ for all $\gamma_j \in \Gamma$, which can encourage MUDs to join the auction
- **Budget-balance.** In each STO i's local auction, budget-balance means $\sum_{j=1}^{n} x_{ij} b_i |T_{ij}^s| - \sum_{j=1}^{n} x_{ij} p_{ij} \geq 0$ for all $1 \leq i \leq m$, which ensures that each STO has enough payment paid to the winning MUDs.

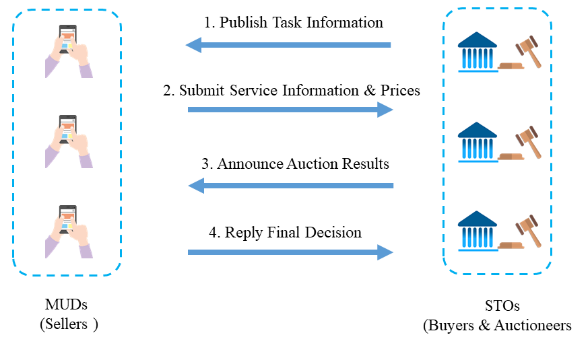


Fig. 1. STOs' distributed auction framework.

- **Incentive-compatibility.** In each STO i's auction, incentive-compatibility ensures that each MUD $\gamma_j \in \Gamma$'s utility can be maximized if and only if the asking price is $a_{ij} = \bar{a}_{ij}$ for all $\pi_i \in \Pi$, where $\bar{a}_{ij}$ is the true asking price of MUD γ_j for task π_i. This is also called "truthfulness" making sure that no MUD can increase utility by manipulating bidding prices.

If an auction can simultaneously satisfy the above three properties, it is *economic robust* and thus can attract more mobile users' participation.

D. Task Owner's Distributed Auction Framework

As shown in Fig. 1, there are four major stages in our proposed auction framework:

- **Stage 1: Publish Task Information.** STOs announce their task information and deadline of accepting bids from MUDs on the cloud platform.
- **Stage 2: Submit Service Information & Price.** MUDs submit their service information and asking prices to STOs.
- **Stage 3: Announce Auction Results.** After collecting service information and asking prices from MUDs, STOs compute the potential winners, working time, and payments, and then announce auction results and deadline of accepting final decision from MUDs.
- **Stage 4: Reply Final Decision.** If an MUD is selected as a potential winner by one or more STOs, it should reply final decision to STOs.

If a STO is rejected by the selected MUDs, the STO continues its task auction to schedule the unassigned working time slots in a multi-round manner until the task duration is completely assigned or no MUD can be selected. In this paper, the essence of auction implementation at the side of STOs is performing a single-side auction iteratively until a termination condition is satisfied, and any two rounds of the auction are independent to each other. Since each round of the auction is treated as a new one, it does not matter whether MUDs change their bid prices or not. On the other hand, if an MUD accepts a task, it exits auction; it continues to compete for tasks until no task auction is conducted. Within the proposed auction framework for STOs, two different policies can be adopted for task assignment : i) **cost-preferred policy:** i) STOs compute auction results according to the non-decreasing order of MUDs' asking prices; and ii) **time schedule-preferred policy:** STOs compute auction results

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.2987881, IEEE Transactions on Mobile Computing

Algorithm 1 Auction Scheme for STO i

1: **Input**: f_i^π, b_i, and ξ.
2: **Output**: $\{x_{ij}\}$, $\{T_{ij}^s\}$.
3: Set $\{x_{ij}\} = \{0\}$, $\{T_{ij}^s\} = \{\emptyset\}$, and $T_i^u = [\alpha_i^\pi, \beta_j^\pi]$;
4: Publish task information f_i^π;
5: Receive service information $\{f_j^\gamma\}$ and asking prices $\{a_{ij}\}$ from the MUDs;
6: **repeat**
7:   **if** $\xi = cpas$ **then**
8:     Run Alg. 2 to compute auction results;
9:   **end if**
10:   **if** $\xi = tpas$ **then**
11:     Run Alg. 3 to compute auction results;
12:   **end if**
13:   Collect replies from MUDs, record $\{x_{ij}\}$, and update $T_i^u = T_i^u \setminus \bigcup_{j=1}^n x_{ij} T_{ij}^s$;
14: **until** $T_i^u = \emptyset$ or no potential winner is selected.

based on the first-come-first-serve manner. The adoption of policies and deadlines for publishing task information and announcing auction results are determined through MUDs' negotiation before the auction. The implementation of the auction schemes is presented in Algorithm 1, and the auction schemes are detailed in Section IV and Section V, respectively.

IV. Cost-Preferred Auction Scheme

In this section, a Cost-Preferred Auction Scheme (**CPAS**) is proposed (i.e., $\xi = cpas$ in line 7 of Algorithm 1), where each STO i greedily schedule task according to the non-decreasing order of the MUDs' asking prices. Since CPAS is performed in a multi-round manner and the auction procedure of each round is the same, we just demonstrate the auction procedure of a round in the following part of this section.

A. Potential Winner Determination & Payment Calculation

After receiving service information $\{f_j^\gamma\}$ and asking price $\{a_{ij}\}$ from MUDs, STO i determines a set of available MUDs as: $\Gamma^c(\pi_i) = \{\gamma_j | (T_{ij} \cap T_i^u) \neq \emptyset, R_j^\gamma \subseteq R_i^\pi$, and $a_{ij} \leq b_i\}$, where T_i^u is the currently unassigned time duration of task π_i. This is implemented in lines 2-7 of Algorithm 2.

1) Potential Winner Determination: The set of potential winners is $W(\pi_i) = \emptyset$ initially. To schedule working time, STO i first sorts MUDs in $\Gamma^c(\pi_i)$ in a non-decreasing order in terms of their asking prices and obtains a sorted set $\Gamma^{c\prime}(\pi_i)$ (line 8 in Algorithm 2). Next, STO i scans MUDs in $\Gamma^{c\prime}(\pi_i)$ and allocates unassigned time slots in a greedy fashion. If MUD γ_j's current available duration $(T_{ij} \cap T_i^u)$ is not fully scheduled to other available MUDs, *i.e.*, $(T_{ij} \cap T_i^u) \cap (\bigcup_{\gamma_{j'} \in W(\pi_i)} T_{ij'}^s) \neq (T_{ij} \cap T_i^u)$, MUD γ_i is selected as a potential winner and assigned a set of time slots that are not allocated to current potential winners in $W(\pi_i)$, *i.e.*, $T_{ij}^s = (T_{ij} \cap T_i^u) \backslash (T_{ij} \cap T_i^u \cap (\bigcup_{\gamma_{j'} \in W(\pi_i)} T_{ij'}^s))$ (lines 9-14 in Algorithm 2).

2) Payment Calculation: STO i computes payments for each selected MUD γ_j by identifying γ_j's *critical neighbor* who is an MUD γ_k in $\Gamma^c(\pi_i)$ such that γ_j can not be selected if a_{ij} is higher than a_{ik}. Different from the existing works [13], [15]–[20] in which each winner has only one

Algorithm 2 Cost-Preferred Task Scheduling & Pricing for Task π_i

Input: $f_i^\pi, b_i, T_i^u, \Gamma, \{f_j^\gamma\}, \{a_{ij}\}$.
Output: $W(\pi_i), \{T_{ij}^s\}, \{p_{ij}\}$.
1: Set $\Gamma^c(\pi_i) = \emptyset, W(\pi_i) = \emptyset$, $\{T_{ij}^s\} = \{\emptyset\}$, and $\{p_{ij}\} = \{0\}$;
2: **for** each $\gamma_j \in \Gamma$ with submitted f_j^γ and a_{ij} **do**
3:   Calculate T_{ij};
4:   **if** $(T_{ij} \cap T_i^u) \neq \emptyset$, $R_j^\gamma \subseteq R_i^\pi$, and $a_{ij} \leq b_i$ **then**
5:     $\Gamma^c(\pi_i) = \Gamma^c(\pi_i) \cup \gamma_j$;
6:   **end if**
7: **end for**
8: Sort all MUDs in $\Gamma^c(\pi_i)$ in non-decreasing order based on $\{a_{ij}\}$ and obtain the sorted set $\Gamma^{c\prime}(\pi_i)$;
9: **for** $j = 1$ to $|\Gamma^{c\prime}(\pi_i)|$ **do**
10:   **if** $(T_{ij} \cap T_i^u) \cap (\bigcup_{\gamma_{j'} \in W(\pi_i)} T_{ij'}^s) \neq (T_{ij} \cap T_i^u)$ **then**
11:     $W(\pi_i) = W(\pi_i) \cup \gamma_j$;
12:     $T_{ij}^s = (T_{ij} \cap T_i^u) \backslash (T_{ij} \cap T_i^u \cap (\bigcup_{\gamma_{j'} \in W(\pi_i)} T_{ij'}^s))$;
13:   **end if**
14: **end for**
15: **for** each $\gamma_j \in W_i^\pi$ **do**
16:   Set $\{T_{ik}^{s\prime}\} = \{\emptyset\}$ and $T_{ij}^{s\prime} = T_{ij}^s$;
17:   Sort all the MUDs in $\Gamma^c(\pi_i) \backslash \gamma_j$ in a non-decreasing order based on $\{a_{ik}\}$ and obtain the sorted set $\Gamma_{-\gamma_j}^{c\prime}(\pi_i)$;
18:   Set $k = 1$ and $W_{-\gamma_j}(\pi_i) = \emptyset$;
19:   **while** $k \leq |\Gamma_{-\gamma_j}^{c\prime}(\pi_i)|$ and $T_{ij}^{s\prime} \neq \emptyset$ **do**
20:     **if** $(T_{ik} \cap T_i^u) \cap (\bigcup_{\gamma_{j'} \in W_{-\gamma_j}(\pi_i)} T_{ij'}^{s\prime}) \neq (T_{ik} \cap T_i^u)$ **then**
21:       $W_{-\gamma_j}(\pi_i) = W_{-\gamma_j}(\pi_i) \cup \gamma_k$,
22:       $T_{ik}^{s\prime} = (T_{ik} \cap T_i^u) \backslash (T_{ik} \cap T_i^u \cap (\bigcup_{\gamma_{j'} \in W_{-\gamma_j}(\pi_i)} T_{ij'}^{s\prime}))$;
23:       **if** $T_{ik}^{s\prime} \cap T_{ij}^{s\prime} \neq \emptyset$ **then**
24:         $p_{ij} = p_{ij} + a_{ik} |T_{ik}^{s\prime} \cap T_{ij}^{s\prime}|$,
25:         $T_{ij}^{s\prime} = T_{ij}^{s\prime} \backslash (T_{ik}^{s\prime} \cap T_{ij}^{s\prime})$;
26:       **end if**
27:     **end if**
28:     $k = k + 1$;
29:   **end while**
30:   **if** $T_{ij}^{s\prime} \neq \emptyset$ **then**
31:     $p_{ij} = p_{ij} + b_i |T_{ij}^{s\prime}|$.
32:   **end if**
33: **end for**

critical neighbor, each selected MUD γ_j in CPAS has one or more critical neighbors because the time slots of T_{ij}^s may be assigned to one or more other MUDs if MUD γ_j does not join the auction (lines 17 - 32 of Algorithm 2). Therefore, the payment is calculated according to γ_j's all critical neighbors. To identify the critical neighbors, STO i sorts MUDs in $\Gamma_{-\gamma_j}^c(\pi_i) = \Gamma^c(\pi_i) \setminus \gamma_j$ in the non-decreasing order in terms of their asking prices, selects winners again in the sorted set $\Gamma_{-\gamma_j}^{c\prime}(\pi_i)$, and allocates working time slots. An MUD γ_k is a critical neighbor of MUD γ_j if their allocated time durations are overlapping, *i.e.*, $T_{ik}^{s\prime} \cap T_{ij}^{s\prime} \neq \emptyset$ where $T_{ik}^{s\prime}$ is the time duration assigned to MUD γ_k and $T_{ij}^{s\prime}$ records the remaining time duration in T_{ij}^s that is not allocated to others. Accordingly, the critical payment is $a_{ik} |T_{ik}^{s\prime} \cap T_{ij}^{s\prime}|$. But, if no critical neighbor is found for MUD γ_j, the critical payment is STO i's budget $b_i |T_{ij}^{s\prime}|$.

Remark: In Algorithm 2, each selected MUD receives a working duration T_{ij}^s that contains one or more sub-durations. For example, the duration of task π_i is from 1:00pm to 5:00pm, γ_j's working duration is $T_{ij}^s = \{$[2:00pm, 3:00pm], [4:30pm, 5:00pm]$\}$ containing two sub-durations, and the number of working time slots is $|T_{ij}^s| = 90$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.2987881, IEEE Transactions on Mobile Computing

minutes.

B. Final Service Decision

MUDs make their service decision when the auction results are published. Let $\Pi(\gamma_j)$ be the set of tasks, of which their owners select MUD γ_j as a potential winner, *i.e.,* $\Pi(\gamma_j) = \{\pi_i | \gamma_j \in W(\pi_i) \text{ and } \pi_i \in \Pi\}$. The decision is determined as follows.

- If $|\Pi(\gamma_j)| = 0$, MUD γ_j loses all STOs' local auctions, does not need to reply, and remains in the auction until no task auction is conducted.
- If $|\Pi(\gamma_j)| = 1$, MUD γ_j is a potential winner in a STO i's location auction, accepts the service request, and exits the auction.
- If $|\Pi(\gamma_j)| > 1$, MUD γ_j is selected by multiple STOs and chooses the task which yields the maximum utility, *i.e.,* $\pi_i = \arg \max_{\pi_h \in \Pi(\gamma_j)} \{(p_{hj} - a_{hj}|T^s_{hj}|)\}$.

C. Property Analysis

In this subsection, we mathematically prove the performance of CPAS.

Lemma 1: Algorithm 2 can terminate within $O(n^2 \log(n))$.

Proof: From line 2 to line 7, the running time of forming set $\Gamma^c(\pi_i)$ is at most n that is the number of MUDs in set Γ. In line 8, sorting the MUDs in $\Gamma^c(\pi_i)$ costs at most $n\log(n)$ time. The potential winner determination, in lines 9-14, has a time complexity of $O(n)$. Similarly, the time of sorting process of line 17 is $O(n\log(n))$, and critical neighbors can be found within $O(n)$. The "for" loop from line 15 to line 33 has at most n iterations and stops within $O(n^2\log(n))$. In a summary, the time complexity of Algorithm 2 is $O(n^2\log(n))$. ■

Theorem 1: The time complexity of Algorithm 1 with $\xi = cpas$ is $O(n^3\log(n))$.

Proof: In Algorithm 1, each STO i stops if and only if either of the two conditions satisfies: i) $T^u_i = \emptyset$; and ii) no potential winner is selected. In the worst case, STO i picks only one potential winner at each round but is rejected by the potential winner. Under this situation, the potential winner definitely accepts another STO's task request and then exits the auction. Thus, after at most n rounds, STO i ends its auction as no potential winner can be selected. From Lemma 1, we can conclude that the time complexity of Algorithm 1 with $\xi = cpas$ is $O(n^3\log(n))$. ■

Theorem 2: The auction scheme CPAS is individually-rational to all MUDs.

Proof: If MUD γ_j loses all STOs' auctions, $\sum_{i=1}^{m} x_{ij} = 0$ and $U^\gamma_j = 0$. If MUD γ_j wins task π_i, $x_{ij} = 1$ and $T^s_{ij} > 0$. Moreover, $a_{ik} \geq a_{ij}$ for γ_j's every critical neighbor γ_k and $b_i \geq a_{ij}$ for STO i, thus $p_{ij} \geq a_{ij}|T^s_{ij}|$. As a result, $U^\gamma_j = \sum_{i=1}^{m} u^\gamma_{ij} = \sum_{i=1}^{m} x_{ij}(p_{ij} - a_{ij}|T^s_{ij}|) \geq 0$. ■

Theorem 3: The auction scheme CPAS achieves budget-balance for all STOs

Proof: In Algorithm 2, all the potential winners are selected from $\Gamma^c(\pi_i)$ and $a_{ij} \leq b_i$ for all $\gamma_j \in \Gamma^c(\pi_i)$. Line 24 and line 31 of Algorithm 2 show that $b_i \geq p_{ij}$ for each winner γ_j. Therefore, $\sum_{j=1}^{n} x_{ij} b_i |T^s_{ij}| - \sum_{j=1}^{n} x_{ij} p_{ij} \geq 0$, *i.e.,* CPAS achieves budget-balance for each STO i. ■

Lemma 2: In each STO i's auction CPAS, if MUD γ_j is selected as a potential winner with a price a_{ij}, it can still be a potential winner with a smaller price $a^1_{ij} < a_{ij}$ and $T^s_{ij} \subseteq T^{s1}_{ij}$, where T^{s1}_{ij} is the assigned working duration corresponding to a^1_{ij}.

Proof: Suppose that $pos(a^1_{ij})$ and $pos(a_{ij})$ are the positions of MUD γ_j in the sorted set $\Gamma^{c\prime}(\pi_i)$ when bidding with a^1_{ij} and a_{ij}, respectively. Since $a^1_{ij} < a_{ij}$, $pos(a^1_{ij}) \leq pos(a_{ij})$. According to lines 8 to 14 of Algorithm 2, MUD γ_j submitting a^{s1}_{ij} can be successfully scheduled a time duration T^{s1}_{ij} and $T^s_{ij} \subseteq T^{s1}_{ij}$. ■

Theorem 4: The auction scheme CPAS is incentive compatible to all MUDs.

Proof: Proving this theorem is equivalent to prove that in each STO i's local auction CPAS, each MUD $\gamma_j \in \Gamma$ cannot enhance utility by submitting $a_{ij} \neq \bar{a}_{ij}$, which is analyzed from the following cases.

Case 1: $a_{ij} < \bar{a}_{ij}$ (or $a_{ij} > \bar{a}_{ij}$) and MUD γ_j loses the auction with both a_{ij} and $\bar{a}_{ij}$. In this case, γ_j's utility received from STO i's auction is zero.

Case 2: $a_{ij} < \bar{a}_{ij}$ and MUD γ_j wins the auction with both a_{ij} and $\bar{a}_{ij}$. According to Lemma 2, $\bar{T}^s_{ij} \subseteq T^s_{ij}$ and $|\bar{T}^s_{ij}| \leq |T^s_{ij}|$, where $\bar{T}^s_{ij}$ and $|\bar{T}^s_{ij}|$ respectively denote the assigned time duration and the number of time slots of $\bar{T}^s_{ij}$ corresponding to $\bar{a}_{ij}$. Accordingly, the payment p_{ij} can be re-computed via two parts: i) the payments $\bar{p}_{ij}$ paid for time duration $\bar{T}^s_{ij}$ that is the same for both $\bar{a}_{ij}$ and a_{ij}; and ii) payment Δp_{ij} paid for time duration $T^s_{ij} \setminus \bar{T}^s_{ij}$, in which $a_{ij}|T^s_{ij} \setminus \bar{T}^s_{ij}| \leq \Delta p_{ij} \leq \bar{a}_{ij}|T^s_{ij} \setminus \bar{T}^s_{ij}|$ as $a_{ij} \leq a_{ik} \leq \bar{a}_{ij}$ for γ_j's every critical neighbor γ_k. Correspondingly, the received utility is $u^\gamma_{ij} = p_{ij} - \bar{a}_{ij}|T^s_{ij}| = (\bar{p}_{ij} - \bar{a}_{ij}|\bar{T}^s_{ij}|) + (\Delta p_{ij} - \bar{a}_{ij}|T^s_{ij} \setminus \bar{T}^s_{ij}|)$. Since $a_{ij}|T^s_{ij} \setminus \bar{T}^s_{ij}| \leq \Delta p_{ij} \leq \bar{a}_{ij}|T^s_{ij} \setminus \bar{T}^s_{ij}|$, we have $(\Delta p_{ij} - \bar{a}_{ij}|T^s_{ij} \setminus \bar{T}^s_{ij}|) \leq 0$. As a result, we obtain $u^\gamma_{ij} = p_{ij} - \bar{a}_{ij}|T^s_{ij}| \leq \bar{p}_{ij} - \bar{a}_{ij}|\bar{T}^s_{ij}|$, *i.e.,* MUD γ_j cannot get a higher utility by bidding a_{ij}.

Case 3: $a_{ij} < \bar{a}_{ij}$ and MUD γ_j wins with a_{ij} but loses with $\bar{a}_{ij}$. In this case, $\bar{a}_{ij}$ is higher than its critical neighbors' asking prices $\{a_{ik}\}$ or is higher than STO i's budget b_i. That is, $\bar{a}_{ij}|\bar{T}^s_{ij}| \geq p_{ij}$. Therefore, $u^\gamma_{ij} = p_{ij} - \bar{a}_{ij}|\bar{T}^s_{ij}| \leq 0$.

Case 4: $a_{ij} > \bar{a}_{ij}$ and MUD γ_j wins with $\bar{a}_{ij}$ but loses with a_{ij}. Thus, $u^\gamma_{ij} = 0$ which cannot be higher than the utility corresponding to $\bar{a}_{ij}$.

Case 5: $a_{ij} > \bar{a}_{ij}$ and MUD γ_j wins with both $\bar{a}_{ij}$ and a_{ij}. Similar to the analysis of Case 2, we have $T^s_{ij} \subseteq \bar{T}^s_{ij}$ and $|T^s_{ij}| \leq |\bar{T}^s_{ij}|$. The payment $\bar{p}_{ij}$ consists of two parts: i) p_{ij} paid for duration T^s_{ij} that is the same for both $\bar{a}_{ij}$ and a_{ij}; and ii) $\Delta\bar{p}_{ij}$ paid for time duration $\bar{T}^s_{ij} \setminus T^s_{ij}$, in which $\bar{a}_{ij}|\bar{T}^s_{ij} \setminus T^s_{ij}| \leq \Delta\bar{p}_{ij} \leq a_{ij}|\bar{T}^s_{ij} \setminus T^s_{ij}|$ as $\bar{a}_{ij} \leq a_{ik} \leq a_{ij}$ for γ_j's every critical neighbor γ_k. Thus, the received utility is $u^\gamma_{ij} = p_{ij} - \bar{a}_{ij}|T^s_{ij}| \leq (\bar{p}_{ij} - \bar{a}_{ij}|T^s_{ij}|) + (\Delta\bar{p}_{ij} - \bar{a}_{ij}|\bar{T}^s_{ij} \setminus T^s_{ij}|)$; that is, MUD γ_j's utility cannot be enhanced by submitting $a_{ij} > \bar{a}_{ij}$.

Therefore, each STO i's auction CPAS is truthful for all MUDs. Furthermore, from Subsection IV-B, it can be found that each MUD γ_j cannot increase the value of $\max_{\pi_i \in \Pi(\gamma_j)} \{(p_{ij} - a_{ij}|T^s_{ij}|)\}$ by cheating on a_{ij} for each task π_i. Therefore, the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.2987881, IEEE Transactions on Mobile Computing

Algorithm 3 Time-Preferred Task Scheduling for Task π_i

1: **Input**: f_i^π, T_i^u, Γ, $\{f_j^\gamma\}$.
2: **Output**: $\{T_{ij}^s\}$.
3: Set $\Gamma^t(\pi_i) = \emptyset$ and $\{T_{ij}^s\} = \{\emptyset\}$ for $\forall \gamma_j \in \Gamma(\pi_i)$;
4: **for** each $\gamma_j \in \Gamma$ with submitted f_j^γ and a_{ij} **do**
5: Calculate t_{ij}^α and T_{ij};
6: **if** $(T_{ij} \cap T_i^u) \neq \emptyset$ and $R_j^\gamma \subseteq R_i^\pi$ **then**
7: $\Gamma^t(\pi_i) = \Gamma^t(\pi_i) \cup \gamma_j$;
8: **end if**
9: **end for**
10: Sort all MUDs in $\Gamma^t(\pi_i)$ in the non-decreasing order based on $\{t_{ij}^\alpha\}$ and get the sorted set $\Gamma^{t\prime}(\pi_i)$;
11: Set $Start = \alpha_i^\pi$;
12: **for** $j = 1$ to $|\Gamma^{t\prime}(\pi_i)|$ **do**
13: **if** $Start < \min\{\beta_j^\gamma, \beta_i^\pi\}$ and $(T_{ij} \cap T_{ij}^u) \cap (\bigcup_{j'=1}^{j-1} T_{ij'}^s) \neq (T_{ij} \cap T_{ij}^u)$ **then**
14: $T_{ij}^s = [Start, \min\{\beta_j^\gamma, \beta_i^\pi\}]$;
15: $Start = \min\{\beta_j^\gamma, \beta_i^\pi\}$.
16: **end if**
17: **end for**

auction scheme CPAS can achieve truthfulness for all MUDs. ■

V. Time-Preferred Auction Scheme

Notice that in the auction scheme CPAS, an MUD's assigned working duration contains one or more sub-durations. While, to allocate one single continuous duration to MUDs, we propose a time schedule-preferred auction scheme (**TPAS**) (i.e., $\xi = tpas$ in line 10 of Algorithm 1), where STOs schedule MUDs based on a first-come-first-serve manner in the time domain and then computes their payments.

A. Potential Winner Determination & Payment Calculation

Before determining the potential winners, each STO i computes the set of available MUDs $\Gamma^t(\pi_i) = \{\gamma_j | (T_{ij} \cap T_i^u) \neq \emptyset$ and $R_j^\gamma \subseteq R_i^\pi\}$.

1) Potential Winner Determination: With the first-come-first-serve policy, each STO i greedily assigns a working duration to each available MUD γ_j according to the non-decreasing order of MUDs' arrival time $t_{ij}^\alpha = \{\frac{d(L_i^\pi, L_j^\gamma)}{\lambda_j^\gamma} + \alpha_j^\gamma\}$ until no available MUD can be selected or the unassigned working duration T_i^u becomes empty. To schedule a working duration that is as long continuous as possible, STO i assigns each available MUD $\gamma_j \in \Gamma^t(\pi_i)$ a duration from γ_j's prior MUD's ending working time to the time $\min\{\beta_i^\pi, \beta_j^\gamma\}$ if this time duration is unassigned. The pseudo-code of the scheduling scheme is presented in Algorithm 3.

2) Payment Calculation: To compute payments of potential winners, each STO i sorts all available MUDs' based on their asking prices in the non-decreasing order. Without loss of generality, we assume that $a_1 \leq a_2 \leq \cdots \leq a_{|\Gamma^t(\pi_i)|}$. Then, each STO i finds a maximum index k_i^π such that $a_{k_i^\pi} \leq b_i < a_{k_i^\pi+1}$ and determines winners according to the following two cases.

- Case 1: $T_{ik_i^\pi}^s \neq \emptyset$. If $\gamma_j \in \Gamma^t(\pi_i)$, $1 \leq j \leq k_i^\pi$ and $T_{ij}^s \neq \emptyset$, set $W(\pi_i) = W(\pi_i) \cup \gamma_j$ and $p_{ij} = b_i |T_{ij}^s|$, where b_i is the *critical price* of all MUDs in STO i's auction.
- Case 2: $T_{ik_i^\pi}^s = \emptyset$. If $\gamma_j \in \Gamma^t(\pi_i)$, $1 \leq j < k_i^\pi$ and $T_{ij}^s \neq \emptyset$, set $W(\pi_i) = W(\pi_i) \cup \gamma_j$ and $p_{ij} = a_{ik_i^\pi} |T_{ij}^s|$, in which MUD $\gamma_{k_i^\pi}$ and $a_{ik_i^\pi}$ are the *critical neighbor* and the *critical price* of all MUDs in STO i's auction, respectively.

B. Final Service Decision

In the scheme TPAS, MUDs make their final decision using the method the same as that in Subsection IV-B.

C. Property Analysis

In this subsection, the performance of the auction scheme TPAS is rigorously analyzed.

Lemma 3: The computational complexity of Algorithm 3 is $O(n \log(n))$.

Proof: From line 4 to line 9, the construction of set $\Gamma^t(\pi_i)$ can be done within $O(n)$. In line 10, the sorting process can be completed within $O(n \log(n))$. From line 12 to line 17, the scheduling process contains at most n iterations, each of which has time complexity of $O(1)$. Therefore, the computational complexity of Algorithm 3 is $O(n \log(n))$. ■

Lemma 4: The computational complexity of payment calculation in TPAS is $O(n)$.

Proof: To compute the payments, each STO i identifies a maximum index k_i^π by scanning $\Gamma^t(\pi_i)$. As $|\Gamma^t(\pi_i)| \leq n$, the computation complexity of payment calculation is $O(n)$. ■

Theorem 5: The time complexity of Algorithm 1 with $\xi = tpas$ is $O(n^2 \log(n))$.

Proof: From Lemma 3, Lemma 4 and Theorem 1, this theorem can be proved. ■

Theorem 6: The auction scheme TPAS is individually-rational for all MUDs.

Proof: If an MUD γ_j is a loser, $\sum_{i=1}^{m} x_{ij} = 0$. Thus, for any $\pi_i \in \Pi$, $p_{ij} = 0$ and $|T_{ij}^s| = 0$, indicating $U_j^\gamma = 0$. If an MUD γ_j is a winner, $\exists \pi_i \in \Pi$ such that $x_{ij} = 1$. So, $U_j^\gamma = u_{ij}^\gamma = p_{ij} - a_{ij}|T_{ij}^s| \geq 0$. Therefore, TPAS achieves individual-rationality for all MUDs. ■

Theorem 7: The auction scheme TPAS ensures budget-balance for all STOs.

Proof: If task π_i is successfully assigned, we have $\sum_{j=1}^{n} x_{ij} \geq 1$, $\sum_{j=1}^{n} x_{ij}|T_{ij}^s| > 0$, and $p_{ij} \leq b_i |T_{ij}^s|$ for each MUD γ_j in $W(\pi_i)$. Thus, for STO i, $\sum_{j=1}^{n} x_{ij} b_i |T_{ij}^s| - \sum_{j=1}^{n} x_{ij} p_{ij} \geq 0$. ■

Lemma 5: For each STO i, the scheduling results $\{T_{ij}^s\}$ of Algorithm 3 are independent of all MUDs' asking prices $\{a_{ij}\}$.

Proof: As shown in lines 12 to 16 of Algorithm 3, the calculation of γ_j's working duration T_{ij}^s does not depend on any a_{ij}. Therefore, this theorem holds. ■

Lemma 6: In each STO i's local auction TPAS, if MUD γ_j is a potential winner with a_{ij}, it can also become a potential winner with $a_{ij}^1 < a_{ij}$.

Proof: When MUD γ_j submits a smaller price a_{ij}^1, γ_j's sorted position in $\Gamma^t(\pi_i)$ changes from j to j^1. Since $a_{ij}^1 < a_{ij}$, $j^1 \leq j \leq k_i^\pi$.Lemma 5 shows that the assigned working

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.2987881, IEEE Transactions on Mobile Computing

duration T_{ij}^s remains the same for MUD γ_j. Thus, γ_j can be selected as a potential winner by STO i. ■

Theorem 8: The auction scheme TPAS can guarantee incentive compatibility for all MUDs.

Proof: Proving this theorem is equivalent to prove that in each STO i's local auction TPAS, each MUD $\gamma_j \in \Gamma$ cannot enhance u_{ij}^{γ} by asking for a price $a_{ij} \neq \bar{a}_{ij}$, for which there are five cases to be considered.

Case 1: $a_{ij} < \bar{a}_{ij}$ (or $a_{ij} > \bar{a}_{ij}$) and MUD γ_j loses the auction with both a_{ij} and $\bar{a}_{ij}$. In this case, γ_j's utility is zero.

Case 2: $a_{ij} < \bar{a}_{ij}$ and MUD γ_j wins with both a_{ij} and $\bar{a}_{ij}$. From Lemma 5, the assigned working duration is T_{ij}^s for MUD γ_j with both a_{ij} and $\bar{a}_{ij}$. The pricing approach in TPAS and Lemma 6 imply that $a_{ij} < \bar{a}_{ij} \leq a_{ik_i^\pi} \leq b_i$, *i.e.*, $a_{ij}|T_{ij}^s| < \bar{a}_{ij}|T_{ij}^s| \leq p_{ij}$. Therefore, the utility is unchanged, *i.e.*, $u_{ij}^{\gamma} = p_{ij} - \bar{a}_{ij}|T_{ij}^s|$.

Case 3: $a_{ij} < \bar{a}_{ij}$ and MUD γ_j wins with a_{ij} but loses with $\bar{a}_{ij}$, indicating $\bar{a}_{ij}$ is higher than the critical price $a_{ik_i^\pi}$ or STO i's budget b_i. Thus, we have $\bar{a}_{ij}|T_{ij}^s| \geq p_{ij}$ and $u_{ij}^{\gamma} = p_{ij} - \bar{a}_{ij}|T_{ij}^s| \leq 0$.

Case 4: $a_{ij} > \bar{a}_{ij}$ and MUD γ_j wins with $\bar{a}_{ij}$ but loses with a_{ij}. In this case, $u_{ij}^{\gamma} = 0$ which cannot be higher than the utility corresponding to $\bar{a}_{ij}$.

Case 5: $a_{ij} > \bar{a}_{ij}$ and MUD γ_j wins with both $\bar{a}_{ij}$ and a_{ij}. Similar to Case 2, we have: i) T_{ij}^s for MUD γ_j with both a_{ij} and $\bar{a}_{ij}$; and ii) $\bar{a}_{ij} < a_{ij} \leq a_{ik_i^\pi} \leq b_i$. Thus, the utility keeps the same, *i.e.*, $u_{ij}^{\gamma} = p_{ij} - \bar{a}_{ij}|T_{ij}^s|$.

The above five cases prove that each STO i's auction is incentive-compatible. Moreover, from Subsection IV-B, one can see that each MUD γ_j cannot increase $\max_{\pi_i \in \Pi(\gamma_j)} \{(p_{ij} - a_{ij}|T_{ij}^s|)\}$ through cheating on a_{ij} for any task π_i. Therefore, the auction scheme TPAS can ensure incentive-compatibility for all MUDs. ■

Remark: In TPAS, the process of task scheduling is independent of MUDs' asking prices. As a result, an MUD that has been assigned a non-empty working duration cannot win the auction if the MUD's asking price is higher than the critical price. In fact, any price-independent scheduling algorithm can be applied in TPAS to obtain $\{T_{ij}^s\}$ without impact on truthfulness for the MUDs.

VI. Auction Formulation for Mobile Users

A. Problem Formulation

In the aforementioned auction schemes (including CPAS and TPAS), task owners work as auctioneers to schedule tasks, and each mobile user is assigned at most one task. To further enhance crowdsensing efficiency, in this section, a distributed auction framework is proposed for mobile users where MUDs can work as auctioneers to handle task assignment. Notably, besides *partial fulfillment*, *attribute diversity* and *price diversity*, *bilaterally-multi-schedule* is also taken into consideration, i.e., *each mobile user can process multiple tasks while each task can be scheduled to multiple mobile users*. Different from CPAS and TPAS, there are two challenging issues in MUDs' auctions.

- Due to location diversity, MUDs need enough time to arrive at the requested location of next scheduled task. To be concrete, suppose MUD γ_j schedules working durations $T_{(i-1)j}^s = [\alpha_{(i-1)j}^s, \beta_{(i-1)j}^s]$, $T_{ij}^s = [\alpha_{ij}^s, \beta_{ij}^s]$ and $T_{(i+1)j}^s = [\alpha_{(i+1)j}^s, \beta_{(i+1)j}^s]$ for tasks π_{i-1}, π_i and π_{i+1}, respectively. We have $\beta_{(i-1)j}^s + \frac{d(L_{i-1}^\pi, L_i^\pi)}{\gamma_j^\gamma} \leq \alpha_{ij}^s$ and $\beta_{ij}^s + \frac{d(L_i^\pi, L_{i+1}^\pi)}{\gamma_j^\gamma} \leq \alpha_{(i+1)j}^s$.
- To guarantee incentive-compatibility for STOs, MUDs calculate payments for each scheduled time slot rather than each sub-duration. When MUD γ_j schedules working duration T_{ij}^s to task π_i, task owner STO i should make a payment $p_{ij}(t)$ to γ_j for each time slot $t \in T_{ij}^s$. Accordingly, STO i's utility can be computed via Eq. (3).

$$U_i^{STO} = \sum_{j=1}^{n} u_{ij}^{STO} = \sum_{j=1}^{n} \sum_{t \in T_{ij}^s} y_{ij}(t)[b_i - p_{ij}(t)], \tag{3}$$

where $y_{ij}(t) \in \{0, 1\}$ indicates whether MUD γ_j is accepted by STO i to work for π_i in time slot t, *i.e.*, $y_{ij}(t) = 1$ if and only if γ_j implements π_i in time slot t.

Thus, the local auction of each MUD γ_j can be formulated in Eq. (4).

$$\max \quad \sum_{i=1}^{m} \sum_{t \in T_{ij}^s} y_{ij}(t) b_i; \tag{4a}$$

$$\text{s.t.} \quad \sum_{i=1}^{m} \sum_{t \in T_{ij}^s} y_{ij}(t) \leq |\beta_j^\gamma - \alpha_j^\gamma|; \tag{4b}$$

$$\sum_{j=1}^{n} \sum_{t \in T_{ij}^s} y_{ij}(t) \leq |\beta_i^\pi - \alpha_i^\pi|; \tag{4c}$$

$$y_{ij}(t) \in \{0, 1\},\ 0 \leq i \leq m, t \in [\alpha_j^\gamma, \beta_j^\gamma]; \tag{4d}$$

$$T_{ij}^s \subseteq T_{ij}, 0 \leq i \leq m. \tag{4e}$$

In Eq. (4), the objective is to maximize valuation of sensing service assignment and schedule with considering the following constraints: i) Eq. (4b) requires that the union of scheduled working durations for all tasks cannot exceed γ_j's available duration; ii) Eq. (4c) shows that the union of scheduled working durations cannot exceed task's duration; and iii) Eq. (4d) and Eq. (4e) indicate the ranges of $y_{ij}(t)$ and T_{ij}^s, respectively.

B. Auction Economic Properties

In each MUD's local auction, two economic properties are required [30]:

- **Individual-rationality.** In this paper, $U_i^{STO} \geq 0$ for all $\pi_i \in \Pi$.
- **Incentive-compatibility.** In each MUD γ_j's auction, incentive-compatibility ensures that each STO i can receive a maximum utility if and only if $b_i = \bar{b}_i$ for all $\gamma_j \in \Gamma$, where $\bar{b}_i$ is the true value of STO i's budget.

C. Mobile User's Distributed Auction Framework

As shown in Fig. 2, both the distributed auction frameworks of mobile users and task owners contain four stages, including *publish service/task information*, *submit task/service information & budgets/prices*, *announce auction results*, and *replay final decision*, in which the main difference is that in MUDs'

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.2987881, IEEE Transactions on Mobile Computing

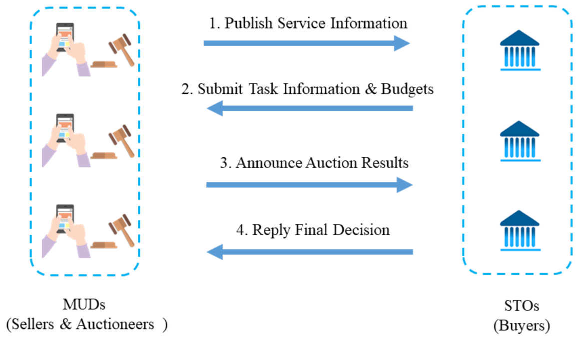


Fig. 2. MUDs' distributed auction framework.

Algorithm 4 Auction Scheme for MUD γ_j

Input: f_j^γ and A_j, and ξ.
Output: $\{y_{ij}(t)\}$, $\{T_{ij}^s\}$, and $\{p_{ij}(t)\}$.
Set $\{y_{ij}(t)\} = \{0\}$, $\{T_{ij}^s\} = \{\emptyset\}$, and $\{p_{ij}(t)\} = \{0\}$;
Publish sensing service information f_j^γ;
Receive sensing task information $\{f_i^\pi\}$ and budget $\{b_i\}$ from the sensing task owners;
if $\xi = vpas$ **then**
  Run Alg. 5 to compute auction results;
end if
if $\xi = dpas$ **then**
  Run Alg. 7 to compute auction results;
end if
Collect replies from the STOs using Alg. 6.

auctions, MUDs compute auction results and STOs make final decisions. In the proposed auction framework for MUDs, we also design two different policies for task assignment and schedule: i) **valuation-preferred policy:** MUDs determine potential winners according to the non-increasing order of the differences between STOs' budgets and MUDs' prices; and ii) **distance-preferred policy:** MUDs determine potential winners based on a nearest-task-first manner. The adoption of policy could be negotiated among MUDs before starting auctions. The implementation of auction schemes within the distributed auction framework is presented in Algorithm 4. Since each MUD in Algorithm 4 can process multiple tasks, the auction schemes, Alg. 5 and Alg. 7, just need to run once. The details of Alg. 5 and Alg. 7 are illustrated in Section VII and Section VIII, respectively.

VII. Valuation-Preferred Auction Scheme

When $\xi = vpas$ in Algorithm 4, VPAS is performed for each MUD, which is demonstrated in this section.

A. *Potential Task Assignment & Payment Calculation*

When MUD γ_j obtains task information $\{f_i^\pi\}$ and budgets $\{b_i\}$ from STOs, it identifies a set of available tasks as $\Pi^v(\gamma_j) = \{\pi_i | T_{ij} \neq \emptyset, R_j^\gamma \subseteq R_i^\pi, \text{and } a_{ij} \leq b_i\}$ where the computation of T_{ij} is described in Section III-A. This is shown in lines 3-8 of Algorithm 5.

1) Potential Task Assignment: In Algorithm 5, $W(\gamma_j)$ records the potential assigned tasks and is empty initially. To schedule tasks, MUD γ_j sorts all available tasks in $\Pi^v(\gamma_j)$ in a non-increasing order according to their budgets and receives a sorted set $\Pi^{v\prime}(\gamma_j)$ (*i.e.*, line 9 of Algorithm 5). Then, MUD γ_j scans tasks in $\Pi^{v\prime}(\gamma_j)$ and greedily schedules working

Algorithm 5 Valuation-Preferred Task Scheduling & Pricing for MUD γ_j

Input: $f_j^\gamma, A_j, \Pi, \{\text{STO } i\}, \{f_i^\pi\}, \{b_i\}$.
Output: $W(\gamma_j), \{T_{ij}^s\}, \{p_{ij}(t)\}$.
1: Set $\Pi^v(\gamma_j) = \emptyset, W(\gamma_j) = \emptyset$, $\{T_{ij}^s\} = \{\emptyset\}$, and $\{p_{ij}(t)\}$=$\{0\}$;
2: $L_j^{\gamma^{current}} = L_j^\gamma$, $\alpha_j^{\gamma^{current}} = \alpha_j^\gamma$, and $\beta_j^{\gamma^{current}} = \beta_j^\gamma$;
3: **for** each STO i with submitted f_i^π and b_i **do**
4:   Calculate T_{ij};
5:   **if** $T_{ij} \neq \emptyset$, $R_j^\gamma \subseteq R_i^\pi$, and $a_{ij} \leq b_i$ **then**
6:     $\Pi^v(\gamma_j) = \Pi^v(\gamma_j) \cup \pi_i$;
7:   **end if**
8: **end for**
9: Sort all tasks in $\Pi^v(\gamma_j)$ in non-increasing order based on $\{b_i\}$ and obtain the sorted set $\Pi^{v\prime}(\gamma_j)$;
10: **for** $i = 1$ to $|\Pi^{v\prime}(\gamma_j)|$ **do**
11:   Update $T_{ij}^{current}$;
12:   **if** $T_{ij}^{current} \cap (\bigcup_{\pi_{i'} \in W(\gamma_j)} T_{i'j}^s) \neq T_{ij}^{current}$ **then**
13:     $W(\gamma_j) = W(\gamma_j) \cup \pi_i$;
14:     $T_{ij}^s = T_{ij}^{current} \backslash (T_{ij}^{current} \cap (\bigcup_{\pi_{i'} \in W(\gamma_j)} T_{i'j}^s))$;
15:     Set $L_j^{\gamma^{current}} = L_i^\pi$ and $\alpha_j^{\gamma^{current}} =$ ending time of T_{ij}^s;
16:   **end if**
17: **end for**
18: **for** each $\pi_i \in W(\gamma_j)$ **do**
19:   Set $L_j^{\gamma^{current}} = L_j^\gamma$;
20:   Sort all the tasks in $\Pi^v(\gamma_j) \backslash \pi_i$ in a non-increasing order based on $\{b_k - a_{kj}\}$ and obtain the sorted set $\Pi_{-\pi_i}^{v\prime}(\gamma_j)$, where $(0 \leq k \leq |\Pi^v(\gamma_j) \backslash \pi_i|)$;
21:   Set $\{T_{kj}^{s\prime}\} = \{\emptyset\}$ and $T_{ij}^{s\prime} = T_{ij}^s$;
22:   Set $k = 1$ and $W_{-\pi_i}(\gamma_j) = \emptyset$;
23:   **while** $k \leq |\Pi_{-\pi_i}^{v\prime}(\gamma_j)|$ and $T_{ij}^{s\prime} \neq \emptyset$ **do**
24:     Update $T_{kj}^{current}$;
25:     **if** $T_{kj}^{current} \cap (\bigcup_{\pi_{i'} \in W_{-\pi_i}(\gamma_j)} T_{i'j}^{s\prime}) \neq T_{kj}^{current}$ **then**
26:       $W_{-\pi_i}(\gamma_j) = W_{-\pi_i}(\gamma_j) \cup \pi_k$,
27:       $T_{kj}^{s\prime} = T_{kj}^{current} \backslash (T_{kj}^{current} \cap (\bigcup_{\pi_{i'} \in W_{-\pi_i}(\gamma_j)} T_{i'j}^{s\prime}))$;
28:       **if** $T_{kj}^{s\prime} \cap T_{ij}^{s\prime} \neq \emptyset$ **then**
29:         **for** each $t \in T_{kj}^{s\prime} \cap T_{ij}^{s\prime}$ **do**
30:           $p_{ij}(t) = a_{ij} + (b_k - a_{kj})$;
31:         **end for**
32:         $T_{ij}^{s\prime} = T_{ij}^{s\prime} \backslash (T_{kj}^{s\prime} \cap T_{ij}^{s\prime})$;
33:       **end if**
34:       Set $L_j^{\gamma^{current}} = L_k^\pi$;
35:     **end if**
36:     $k = k + 1$;
37:   **end while**
38:   **if** $T_{ij}^{s\prime} \neq \emptyset$ **then**
39:     **for** each $t \in T_{ij}^{s\prime}$ **do**
40:       $p_{ij}(t) = a_{ij}$;
41:     **end for**
42:   **end if**
43: **end for**

durations. A task $\pi_i \in \Pi^v(\gamma_j)$ is selected if MUD γ_j has enough time to move to the location of π_i and provides valid working duration, *i.e.*, $T_{ij}^{current} \cap (\bigcup_{\pi_{i'} \in W(\gamma_j)} T_{i'j}^s) \neq T_{ij}^{current}$. $T_{ij}^{current}$ for π_i is determined by its current location $L_j^{\gamma^{current}}$ and current available time period $[\alpha_j^{\gamma^{current}}, \beta_j^{\gamma^{current}}]$ with $\beta_j^{\gamma^{current}} = \beta_j^\gamma$, in which the computation of $T_{ij}^{current}$ is the same as that of T_{ij} described in Section III-A. The assigned duration of MUD γ_j for task π_i is the duration that has not been scheduled to other tasks, *i.e.*, $T_{ij}^s = T_{ij}^{current} \backslash (T_{ij}^{current} \cap (\bigcup_{\pi_{i'} \in W(\gamma_j)} T_{i'j}^s))$ (see lines 10-17 of Algorithm 5). Next, update

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.2987881, IEEE Transactions on Mobile Computing

Algorithm 6 Final Decision of STO i

1: **Input**: $\Gamma(i)$, f_i^{π}, $\{T_{ij}^s\}$, and $\{p_{ij}(t)\}$.
2: **Output**: $y_{ij}(t)$.
3: Set $\{y_{ij}(t)\} = \{0\}$;
4: **for** Each time slot $t \in [\alpha_i^{\pi}, \beta_i^{\pi}]$ **do**
5: $\gamma_{j'} = \arg \min_{\gamma_{j''} \in \{\gamma_h | \gamma_h \in \Gamma(i), t \in T_{ih}^s\}} \{p_{ij''}(t)\}$;
6: Set $y_{ij'}(t) = 1$;
7: **end for**

$\alpha_j^{\gamma^{current}}$ = ending time of T_{ij}^s and $T_{ij}^{current}$ based on the Euclidean distance $d(L_i^{\pi}, L_j^{\gamma})$ and the moving rate λ_j^{γ}.

2) Payment Calculation: After the potential task assignment, each MUD γ_j calculates payments for the corresponding STOs. For each time slot $t \in T_{ij}^s$, STO i pays $p_{ij}(t)$ to MUD γ_j, which is decided by STO i's *critical neighbor*. To find critical neighbor of task π_i in each time slot, MUD γ_j sorts all tasks in $\Pi^v(\gamma_j)\backslash\pi_i$ in the non-increasing order in terms of $b_k - a_{kj}$ with $b_k - a_{kj} \geq 0$ (see line 20 of Algorithm 5). Next, MUD γ_j chooses tasks again from $\Pi^v(\gamma_j)\backslash\pi_i$ and schedules working durations. Any task $\pi_k \in \Pi^v(\gamma_j)\backslash\pi_i$ is a critical neighbor of task π_i if their scheduled working durations overlap, *i.e.*, $T_{kj}^{s\prime} \cap T_{ij}^{s\prime} \neq \emptyset$ (see line 28 of Algorithm 5). Thus, STO i makes a payment $p_{ij}(t) = a_{ij} + (b_k - a_{kj})$ to MUD γ_j for each time slot $t \in T_{kj}^{s\prime} \cap T_{ij}^{s\prime}$, which is computed in lines 23-37 of Algorithm 5. If $t \in T_{ij}^s$ does not overlap with any scheduled working duration of $\pi_i \in \Pi^v(\gamma_j)\backslash\pi_i$, task π_i does not have any critical neighbor in t and $p_{ij}(t) = a_{ij}$ (see lines 38-42 of Algorithm 5).

B. Final Schedule Decision

STOs need to make their final decisions to maximize utilities since each of them might be selected by more than one MUD. Let $\Gamma(i)$ represent the set of MUDs who select π_i at the stage of potential task assignment, *i.e.*, $\Gamma(i) = \{\gamma_j | \pi_i \in W(\gamma_j) \text{ and } \gamma_j \in \Gamma\}$. For each time slot $t \in [\alpha_i^{\pi}, \beta_i^{\pi}]$, STO i selects MUD $\gamma_j \in \Gamma(i)$ who requests the minimum payment as shown in line 5 of Algorithm 6, accepts the service provided by γ_j in time slot t, and sets $y_{ij}(t) = 1$.

C. Properties Analysis

In this section, the performance of auction scheme VPAS is analyzed theoretically in terms of computational efficiency, individual-rationality, and truthfulness.

Theorem 9: The time complexity of Algorithm 4 with $\xi = vpas$ is $O(m^2 \log(m) + nl)$, where l is the maximum number of time units in a sensing task's available time duration.

Proof: We first analyze the time complexity of Algorithm 5. From line 3 to line 8, forming the set $\Pi^v(\gamma_j)$ costs m. In line 9, the running time of obtaining $\Pi^v(\gamma_j)$ is at most $m \log(m)$. The time complexity of potential task assignment, in lines 10-16, is $O(m)$. To calculate payments, the sorting process is performed again (see line 20). Since there are at most m selected tasks, lines 18-43 are iterated at most m times. In addition, the "while" loop (lines 23-37) terminate within $O(m)$. Thus, the time complexity of Algorithm 5 is $O(m^2 \log(m))$. The time complexity of Algorithm 6 is $O(nl)$, where l is the maximum number of time units in a sensing task's available time duration.

In Algorithm 4, the running time of lines 3-5 is $O(1)$, that of line 7 is $O(m^2(\log(m)))$, and that of line 12 is $O(nl)$. Thus, the time complexity of Algorithm 4 with $\xi = vpas$ is $O(m^2 \log(m) + nl)$. ■

Theorem 10: The auction scheme VPAS guarantees individual-rationality for all STOs.

Proof: If STO i loses in every MUD γ_j's local auction, $U_i^{STO} = 0$. If STO i is a winner in at least one MUD γ_j's auction, $T_{ij}^s > 0$. For STO i's each critical neighbor STO k, we have $b_i \geq b_k$. Since $p_{ij}(t)$ for each time slot $t \in T_{ij}^s$ is the critical price, $b_i \geq p_{ij}(t)$ for each time slot $t \in T_{ij}^s$. Therefore, $U_i^{STO} = \sum_{j=1}^{n} u_{ij}^{STO} = \sum_{j=1}^{n} \sum_{t \in T_{ij}^s} y_{ij}(t)(b_i - p_{ij}(t)) \geq 0$. ■

Lemma 7: In each MUD γ_j's VPAS auction, if task π_i is selected with b_i, task π_i can be still selected with $b_i^1 > b_i$ and $T_{ij}^s \subseteq T_{ij}^{s1}$, where T_{ij}^{s1} is working duration of γ_j for π_i.

Proof: We use $pos(b_i^1)$ and $pos(b_i)$ to denote the positions of task π_i in the sorted set $\Pi^{v'}(\gamma_j)$ when STO i's budget is b_i^1 and b_i, respectively. Since $b_i^1 > b_i$, there is $pos(b_i^1) \leq pos(b_i)$. By running Algorithm 5, MUD γ_j will decide to perform task π_i during T_{ij}^{s1} with $T_{ij}^s \subseteq T_{ij}^{s1}$. ■

Theorem 11: The auction scheme VPAS guarantees incentive-compatibility for all STOs.

Proof: To prove this theorem, we first show that each STO i cannot increase utility through submitting a budget $b_i \neq \overline{b}_i$ in any MUD γ_j's local auction, which is analyzed via the following cases.

Case 1: $b_i > \overline{b}_i$ (or $b_i < \overline{b}_i$) and STO i loses the auction with both b_i and $\overline{b}_i$. Consequently, the utility of STO i is zero in this auction.

Case 2: $b_i > \overline{b}_i$ and STO i wins the auction with both b_i and $\overline{b}_i$. According to Lemma 7, $\overline{T}_{ij}^s \subseteq T_{ij}^s$, in which $\overline{T}_{ij}^s$ and T_{ij}^s represent the working durations assigned by γ_j when STO i's budget is $\overline{b}_i$ and b_i, respectively. Formally, let $\overline{p}_{ij}(t)$ and $p_{ij}(t)$ be payments of π_i for each $t \in \overline{T}_{ij}^s$ and $t \in T_{ij}^s$, respectively. Thus, we have $\overline{p}_{ij}(t) = p_{ij}(t)$ for each $t \in \overline{T}_{ij}^s \cap T_{ij}^s$ and $b_i \geq p_{ij}(t) \geq \overline{b}_i$ for each $t \in T_{ij}^s \backslash \overline{T}_{ij}^s$. Thus, STO i's utility is $u_{ij}^{STO} = [\overline{b}_i |T_{ij}^s \backslash \overline{T}_{ij}^s| - \sum_{t \in T_{ij}^s \backslash \overline{T}_{ij}^s} p_{ij}(t)] + [\overline{b}_i |\overline{T}_{ij}^s| - \sum_{t \in \overline{T}_{ij}^s} \overline{p}_{ij}(t)]$. Additionally, $\overline{b}_i |T_{ij}^s \backslash \overline{T}_{ij}^s| - \sum_{t \in T_{ij}^s \backslash \overline{T}_{ij}^s} p_{ij}(t) = \sum_{t \in T_{ij}^s \backslash \overline{T}_{ij}^s} (\overline{b}_i - p_{ij}(t)) \leq 0$ because $b_i \geq p_{ij}(t) \geq \overline{b}_i$. Since $u_{ij}^{STO} \leq \overline{b}_i |\overline{T}_{ij}^s| - \sum_{t \in \overline{T}_{ij}^s} \overline{p}_{ij}(t)$, we can conclude that STO i cannot increase utility with b_i.

Case 3: $b_i > \overline{b}_i$ and STO i wins with b_i but loses with $\overline{b}_i$. In this case, $\overline{b}_i$ is smaller than its critical neighbors' budgets $\{b_k\}$ or MUD γ_j's price a_{ij}, indicating $\overline{b}_i |T_{ij}^s| \leq \sum_{t \in T_{ij}^s} p_{ij}(t)$. Thus, we have $\overline{b}_i |T_{ij}^s| - \sum_{t \in T_{ij}^s} p_{ij}(t) \leq 0$.

Case 4: $b_i < \overline{b}_i$ and STO i wins with $\overline{b}_i$ but loses with b_i. In this case, $u_{ij}^{STO} = 0$ which is not larger than the utility when STO i submits $\overline{b}_i$.

Case 5: $b_i < \overline{b}_i$ and STO i wins with both $\overline{b}_i$ and b_i. In this case, we have $T_{ij}^s \subseteq \overline{T}_{ij}^s$ according to Lemma 7 and $\overline{p}_{ij}(t) = p_{ij}(t)$ for each $t \in \overline{T}_{ij}^s \cap T_{ij}^s$. As a result, STO i's

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.2987881, IEEE Transactions on Mobile Computing

Algorithm 7 Distance-Preferred Task Scheduling & Pricing for MUD γ_j

Input: f_j^{γ}, A_j, Π, {STO i}, $\{f_i^{\pi}\}, \{b_i\}$.
Output: $W(\gamma_j), \{T_{ij}^s\}, \{p_{ij}(t)\}$.
1: Set $\Pi^d(\gamma_j) = \emptyset, W(\gamma_j) = \emptyset$, $\{T_{ij}^s\} = \{\emptyset\}$, $\{T_{ij}^{current}\} = \{\emptyset\}$, and $\{p_{ij}(t)\}$={0};
2: $L_j^{\gamma^{current}} = L_j^{\gamma}$, $\alpha_j^{\gamma^{current}} = \alpha_j^{\gamma}$, and $\beta_j^{\gamma^{current}} = \beta_j^{\gamma}$;
3: **for** each STO i with submitted f_i^{π} **do**
4: Calculate T_{ij};
5: **if** $T_{ij} \neq \emptyset$ and $R_j^{\gamma} \subseteq R_i^{\pi}$ **then**
6: $\Pi^d(\gamma_j) = \Pi^d(\gamma_j) \cup \pi_i$;
7: **end if**
8: **end for**
9: $\Pi^{d'}(\gamma_j) = \Pi^d(\gamma_j)$;
10: **while** $\Pi^{d'}(\gamma_j) \neq \emptyset$ and $\alpha_j^{\gamma^{current}} \neq \beta_j^{\gamma^{current}}$ **do**
11: $\pi_i = \arg \min_{\pi_{i'} \in \Pi^{d'}(\gamma_j)} \{\frac{d(L_{i'}^{\pi}, L_j^{\gamma^{current}})}{\lambda_j^{\gamma}}\}$;
12: Update $T_{ij}^{current}$;
13: **if** $T_{ij}^{current} \neq \emptyset$ **then**
14: Set $T_{ij}^s = T_{ij}^{current}$ and $L_j^{\gamma^{current}} = L_i^{\pi}$;
15: Set $\alpha_j^{\gamma^{current}}$ = ending time of T_{ij}^s;
16: **end if**
17: $\Pi^{d'}(\gamma_j) = \Pi^{d'}(\gamma_j) \backslash \pi_i$;
18: **end while**
19: Sort all tasks in $\Pi^d(\gamma_j)$ in a non-increasing order based on $\{b_k - a_{kj}\}$ and obtain the sorted set $\Pi^{d''}(\gamma_j)$, where $(0 \leq k \leq |\Pi^d(\gamma_j)|)$;
20: Find a maximum index k' such that $b_{k'} - a_{k'j} \geq 0$;
21: **if** $T_{k'j}^s \neq \emptyset$ **then**
22: **for** each $\pi_i \in \Pi^{d''}(\gamma_j)$ **do**
23: **if** $i \leq k'$ and $T_{ij}^s \neq \emptyset$ **then**
24: $W(\gamma_j) = W(\gamma_j) \cup \pi_i$;
25: **for** Each $t \in T_{ij}^s$ **do**
26: $p_{ij}(t) = a_{ij}$;
27: **end for**
28: **end if**
29: **end for**
30: **else**
31: **for** each $\pi_i \in \Pi^{d''}(\gamma_j)$ **do**
32: **if** $i < k'$ and $T_{ij}^s \neq \emptyset$ **then**
33: $W(\gamma_j) = W(\gamma_j) \cup \pi_i$;
34: **for** Each $t \in T_{ij}^s$ **do**
35: $p_{ij}(t) = a_{ij} + (b_{k'} - a_{k'j})$;
36: **end for**
37: **end if**
38: **end for**
39: **end if**

utility is $u_{ij}^{STO} = \bar{b}_i|T_{ij}^s| - \sum_{t \in T_{ij}^s} p_{ij}(t) \leq \bar{b}_i|T_{ij}^s| - \sum_{t \in T_{ij}^s} \overline{p}_{ij}(t) + (\bar{b}_i|\overline{T}_{ij}^s \backslash T_{ij}^s| - \sum_{t \in \overline{T}_{ij}^s \backslash T_{ij}^s} \overline{p}_{ij}(t)) = \bar{b}_i|\overline{T}_{ij}^s| - \sum_{t \in T_{ij}^s} \overline{p}_{ij}(t)$. That is, STO i's utility cannot be improved by setting $b_i < \bar{b}_i$.

To sum up, each MUD γ_j's VPAS auction is truthful for all STOs. Moreover, from Algorithm 6, one can see that each STO i cannot enlarge the value of $\min_{\gamma_{j''} \in \{\gamma_h | \gamma_h \in \Gamma(i), t \in T_{ih}^s\}} \{p_{ij''}(t)\}$ for each scheduled time slot by cheating on b_i. As a result, the auction scheme VPAS is incentive-compatible for all STOs. ■

VIII. Distance-Preferred Auction Scheme

To reduce moving distance of each mobile user, we in this section propose a distance-preferred auction scheme (**DPAS**), in which task schedule follows a nearest-task-first manner. The auction scheme DPAS for each γ_j is called in Algorithm 4 when $\xi = dpas$, and the procedure of DPAS is presented in Algorithm 7.

A. Potential Task Assignment & Payment Calculation

At this stage, each MUD γ_j decides a set of available tasks, *i.e.*, $\Pi^d(\gamma_j) = \{\pi_i | T_{ij} \neq \emptyset$ and $R_j^{\gamma} \subseteq R_i^{\pi}\}$. The corresponding implementation is described in lines 3-8 of Algorithm 7.

1) Potential Task Assignment: Each MUD γ_j iteratively and greedily selects tasks and schedules working durations according to the nearest-task-first manner. In each iteration, MUD γ_j selects the nearest task $\pi_i = \arg \min_{\pi_{i'} \in \Pi^{d'}(\gamma_j)} \{\frac{d(L_{i'}^{\pi}, L_j^{\gamma^{current}})}{\lambda_j^{\gamma}}\}$ from $\Pi^d(\gamma_j)$. Then, γ_j updates currently maximum available working duration $T_{ij}^{current}$ for π_i based on its current location $L_j^{\gamma^{current}}$ and current available time period $[\alpha_j^{\gamma^{current}}, \beta_j^{\gamma^{current}}]$ with $\beta_j^{\gamma^{current}} = \beta_j^{\gamma}$, in which the computation of $T_{ij}^{current}$ is the same as that of T_{ij} described in Section III-A. If $T_{ij}^{current}$ is not empty, MUD γ_j performs three actions: i) assign working duration to π_i with $T_{ij}^s = T_{ij}^{current}$; ii) set π_i's location as current location, *i.e.*, $L_j^{\gamma^{current}} = L_i^{\pi}$; and iii) update its current starting time to the time when γ_j finishes its scheduled working duration T_{ij}^s for π_i, *i.e.*, $\alpha_j^{\gamma^{current}}$ = ending time of T_{ij}^s. The computation of each iteration is shown in lines 10-18 of Algorithm 7. MUD γ_j terminates the iterations when no available task can be selected or its current available duration becomes empty.

2) Payment Calculation: Each MUD γ_j first sorts all available tasks in $\Pi^{d''}(\gamma_j)$ in a non-increasing order based on $b_k - a_{kj}$ with $b_k - a_{kj} \geq 0$ (see line 19 of Algorithm 7). Then, each MUD γ_j finds a maximum index k' such that $b_{k'} - a_{k'j} \geq 0$ and calculates payments as follows:

- Case 1: $T_{k'j}^s \neq \emptyset$. MUD γ_j selects each $\pi_i \in \Pi^{d''}(\gamma_j)$ with $0 \leq i \leq k'$ and $T_{ij}^s \neq \emptyset$ as the potential winner. In this case, the payment for each $t \in T_{ij}^s$ of $\pi_i \in W(\gamma_j)$ is $p_{ij}(t) = a_{ij}$.
- Case 2: $T_{k'j}^s = \emptyset$. MUD γ_j selects each $\pi_i \in \Pi^{d''}(\gamma_j)$ with $0 \leq i < k'$ and $T_{ij}^s \neq \emptyset$ as the potential winner and computes $p_{ij}(t) = a_{ij} + (b_{k'} - a_{k'j})$.

B. Final Schedule Decision

The approach of STOs to make final decisions is the same as that in Subsection VII-B.

C. Property Analysis

In this subsection, we theoretically prove the desired properties of the auction scheme DPAS.

Theorem 12: When $\xi = dpas$, the time complexity of Algorithm 4 is $O(m\log(m) + nl)$, where l is the maximum number of time units in a sensing task's available time duration.

Proof: The procedure of DPAS is outlined in Algorithm 7. From line 3 to line 8, the construction of set $\Pi^d(\gamma_j)$ costs $O(m)$. The scheduling process shown in lines 10-18 has a time complexity of $O(m)$. In line 19, the sorting process can be done within $O(m\log(m))$. From line 21 to line 39, the time complexity of potential winner determination and payment calculation is $O(m)$. In a summary, DPAS can be completed within $O(m\log(m))$. In the line 12 of Algorithm 4, the time complexity of Algorithm 6 is $O(nl)$, where l is the maximum number of time units in a sensing task's available time duration. Therefore, the time complexity of Algorithm 4 with $\xi = dpas$ is $O(m\log(m) + nl)$. ∎

Theorem 13: The auction scheme DPAS is individually-rational for all STOs.

Proof: Once the auction scheme DPAS is done, there are two cases for each STO i. If task π_i is not selected in all MUDs' auctions, STO i's utility is $U_i^{STO} = 0$. If task π_i is selected by at least one MUD γ_j and is assigned working duration $T_{ij}(s)$, we have $p_{ij}(t) \geq a_{ij}$ for each $t \in T_{ij}^s$. Therefore, $U_i^{STO} = \sum_{j=1}^{n} \sum_{t\in T_{ij}^s} y_{ij}(t)(b_i - p_{ij}(t)) \geq 0$. In conclusion, DPAS can achieve individual-rationality for all STOs. ∎

Lemma 8: In each MUD γ_j's local auction DPAS, the schedule results $\{T_{ij}^s\}$ of Algorithm 7 are independent of all STOs' budgets $\{b_i\}$.

Proof: From line 10 to line 18 of Algorithm 7, it is seen that the values of $\{T_{ij}^s\}$ are determined by task information $\{f_i^{\pi}\}$ and MUD's service information $\{f_i^{\gamma}\}$ instead of $\{b_i\}$. Thus, this lemma holds. ∎

Lemma 9: In each MUD γ_j's local auction DPAS, if task π_i is selected with b_i, π_i can be still selected with $b_i^1 > b_i$.

Proof: As analyzed by Lemma 8, the working duration T_{ij}^s assigned by MUD γ_j is independent of b_i. When STO i submits a larger budget $b_i^1 > b_i$, task π_i's position in the sorted set $\Pi(\gamma_j)^{d'}$ changes from i to i^1, and $i^1 \leq i \leq k'$ because of $b_i^1 > b_i$. Therefore, STO i can also become a potential winner for MUD γ_j. ∎

Theorem 14: The auction scheme DPAS is incentive-compatible for all STOs.

Proof: This theorem can hold if and only if each STO i cannot enhance u_{ij}^{STO} with $b_i \neq \overline{b}_i$ in any MUD γ_j's local auction. The analysis process is shown below.

Case 1: $b_i > \overline{b}_i$ (or $b_i < \overline{b}_i$) and STO i is a loser with both b_i and $\overline{b}_i$. In this case, STO i's utility received from MUD γ_j is zero.

Case 2: $b_i > \overline{b}_i$ and STO i wins with both b_i and $\overline{b}_i$. When STO i submits $\overline{b}_i$, we use $\overline{p}_{ij}(t)$ to represent STO i's unit payment in MUD γ_j's auction. From Lemma 9, $b_i - a_{ij} > \overline{b}_i - a_{ij} \geq b_{k'} - a_{k'j} \geq 0$. According to Lemma 8, MUD γ_j schedules the same working duration T_{ij}^s to STO i with both b_i and $\overline{b}_i$. If $T_{k'j}^s \neq \emptyset$, $p_{ij}(t) = \overline{p}_{ij}(t) = a_{ij}$ with $t \in T_{ij}^s$. Thus, $u_{ij}^{STO} = \overline{u}_{ij}^{STO} = (\overline{b}_i - a_{ij})|T_{ij}^s|$. If $T_{k'j}^s = \emptyset$, $p_{ij}(t) = \overline{p}_{ij}(t) = a_{ij} + (b_{k'} - a_{k'j})$ with $t \in T_{ij}^s$. Thus, $u_{ij}^{STO} = \overline{u}_{ij}^{STO} = (\overline{b}_i - a_{ij} - (b_{k'} - a_{k'j}))|T_{ij}^s|$. One can see that STO i's utility remains the same with both b_i and $\overline{b}_i$.

Case 3: $b_i > \overline{b}_i$ and STO i wins with $\overline{b}_i$ but loses with b_i. According to Algorithm 7, we have $b_i - a_{ij} \geq b_{k'} - a_{k'j} \geq 0 > \overline{b}_i - a_{ij}$. If $T_{k'j}^s \neq \emptyset$, $p_{ij}(t) = a_{ij}$ for each $t \in T_{ij}^s$ and $u_{ij}^{STO} = (\overline{b}_i - a_{ij})|T_{ij}^s| < 0$. If $T_{k'j}^s = \emptyset$, $p_{ij}(t) = a_{ij} + (b_{k'} - a_{k'j})$ for each $t \in T_{ij}^s$ and $u_{ij}^{STO} = [\overline{b}_i - a_{ij} - (b_{k'} - a_{k'j})]|T_{ij}^s| < 0$.

Case 4: $b_i < \overline{b}_i$ and STO i wins with $\overline{b}_i$ but loses with b_i. In this case, $u_{ij}^{STO} = 0$ which cannot exceed $\overline{u}_{ij}^{STO}$.

Case 5: $b_i < \overline{b}_i$ and STO i wins with both b_i and $\overline{b}_i$. Similar to Case 2, MUD γ_j schedules the same working duration T_{ij}^s to task π_i with both b_i and $\overline{b}_i$, and $\overline{b}_i - a_{ij} > b_i - a_{ij} \geq b_{k'} - a_{k'j} \geq 0$. Thus, we obtain $u_{ij}^{STO} = \overline{u}_{ij}^{STO}$.

The above five cases show that every MUD γ_j's auction is truthful for all STOs. Therefore, the auction scheme DPAS can ensure truthfulness for all STOs. ∎

IX. Performance Evaluation

In this section, the baseline schemes, the experiment settings, and experiment results are presented.

A. Baseline Schemes

Since there is no distributed auction for assigning sensing tasks in mobile crowdsensing, three centralized auction schemes are adopted for comparison, i.e., Task-SRC, MUD-SRC and TDAM. In each centralized scheme, a cloud platform (CP) acts as an auctioneer to compute the auction results.

Both of Task-SRC and MUD-SRC are modified based on a single-minded reverse combinatorial auction [14] to adapt to our considered scenarios. In Task-SRC, the CP owns all the sensing tasks and recruits MUDs to minimize the social cost with consideration of working time scheduling requirement instead of the QoI coverage constraint. In MUD-SRC, the CP publishes information of all MUDs and assigns all MUDs to work for sensing tasks according to the bids from STOs.

TDAM is a variant of max-min fairness-based truthful double auction [28] with an aim of maximizing the valuation of sensing service assignment and schedule.

B. Experiment Settings

We evaluate the performance of seven auction schemes, including CPAS, TPAS, VPAS, DPAS, Task-SRC, MUD-SRC, and TDAM, by utilizing real data from Google Maps. The crowdsensing scenarios are set as follows: i) the number of tasks varies from 5 to 30; ii) the number of MUDs varies from 10 to 35; iii) the locations of all tasks and MUDs are selected from restaurants, tourist sites, and shopping malls within downtown in Atlanta with area of four-square miles; iv) each MUD walks from one location to another location; and v) the moving time of each MUD between any two locations is calculated through Google Maps app. We consider 10 types of sensors and the number of each type of sensor is one. Each task requests a certain number of sensors, which is a random number uniformly picked from $[3, 10]$; similarly, each MUD is equipped with a certain number of sensors, which is a random number uniformly chosen from $[1, 10]$. Each STO's budget (and each MUD's asking price) is an integer that is uniformly selected from $[10, 25]$ at random. In the experiments, the unit time slot is one minute and the longest duration is 300 minutes. There are three performance metrics adopted in our evaluation:

- **Allocation Efficiency.** The allocation efficiency of a task is the ratio of the total number of assigned working time slots to the number of requested working time slots.
- **Working Time Utilization.** The working time utilization of an MUD is the ratio of the number of assigned working time slots to the number of available working time slots.
- **STOs' Cost.** The cost of a STO is defined in Eq. (2), and the average cost of all STOs will be evaluated.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.2987881, IEEE Transactions on Mobile Computing

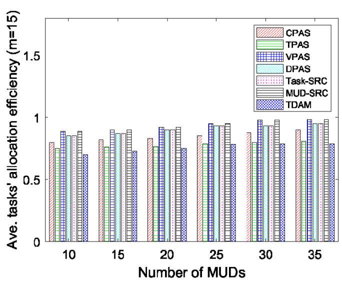


Fig. 3. Ave. tasks' allocation efficiency (m=15).

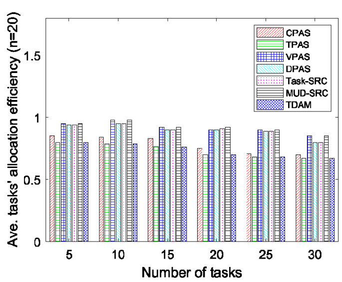


Fig. 4. Ave. tasks' allocation efficiency (n=20).

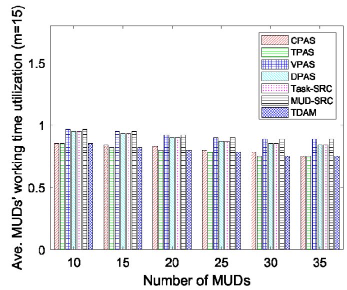


Fig. 5. Ave. MUDs' working time utilization (m=15).

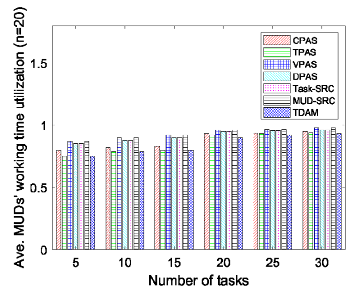


Fig. 6. Ave. MUDs' working time utilization (n=20).

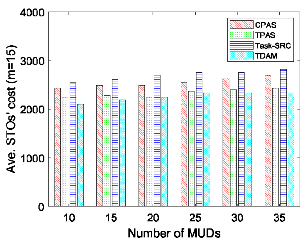


Fig. 7. Ave. STOs' cost (m=15).

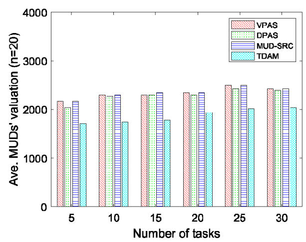


Fig. 8. Ave. MUDs' valuation (n=20).

- **MUDs' Valuation.** The valuation of an MUD is defined in Eq. (4), and the average valuation of all MUDs will be evaluated.
- **Truthfulness.** To examine the received utility when biding truthfully and untruthfully, at each time, one MUD (or one STO) is randomly selected with being set fake asking prices (or budgets).

C. Experiment Results and Analysis

The performance is evaluated under two scenarios: (i) the number of tasks is 15, and the number of MUDs increases from 10 to 35; and (ii) the number of tasks changes from 5 to 30, and the number of MUDs is fixed at 20. The results are shown in Fig. 3 through Fig. 8.

First, from the average allocation efficiency of all tasks in Fig. 3 and Fig. 4, we obtain the following observations:

1) As shown in Fig. 3, the average allocation efficiency increases when the number of MUDs increases. This is because if more MUDs participate in crowdsensing, each STO can recruit more MUDs to process task.
2) The results of Fig. 4 imply that the average allocation efficiency decreases when the number of tasks increases, because each STO may have less MUDs to work when more STOs compete in crowdsensing.
3) VPAS and DPAS perform batter than CPAS and TPAS. This is because each MUD in VPAS and DPAS can implement multiple tasks but in CPAS and TPAS is allowed to work for at most one task. That is, in VPAS and DPAS, a larger portion of required working duration can be scheduled to MUDs, bringing a higher allocation efficiency for each STO.
4) CPAS performs better than TPAS. The reason mainly lies in two aspects: i) compared with TPAS, CPAS could assign more working time slots to MUDs; ii) the working time schedule of TPAS is independent of MUDs' asking prices, leading to that some MUDs who have been scheduled a working duration may still lose if their prices are higher than the critical price.
5) The performance of VPAS is better than DPAS. This is due to that working time schedule of DPAS is independent of the MUDs' asking prices and some MUDs who have been scheduled a working time duration may become losers if their prices are higher than the critical price.
6) Even both Task-SRC and MUD-SRC implement task assignment and scheduling in a centralized manner, the performance of CPAS, TPAS, CPAS, and DPAS proposed in this paper is very close to the performance of Task-SRC and MUD-SRC (see DPAS *vs* Task-SRC and VPAS *vs* MUD-SRC). This can validate the effectiveness of our distributed auction schemes.
7) Moreover, all the four distributed schemes perform much better than TDAM. In TDAM, each mobile user bids for different tasks with one price, thus a mobile user either wins all workable tasks or losses all workable tasks, reducing the allocation efficiency and working time utilization.

Then, we analyze the average working time utilization of all MUDs, in which the results are presented in Fig. 5 and Fig. 6. From Fig. 5, one can observe that the average working time utilization of all MUDs decreases when the number of MUDs increases, as more intense competition exists among MUDs. On the other hand, as shown in Fig. 6, the average working time utilization of all MUDs is increased when more tasks are published, because it becomes easier for any MUD to be scheduled to process tasks. In addition, VPAS and DPAS perform better than CPAS and TPAS. This is because each MUD can work for at most one task in CPAS and TPAS but can work for multiple tasks in VPAS and DPAS. With respect to the average MUD's working time utilization, the performance of CPAS and TPAS is close to that of TDAM, the performance of VPAS is close to that of MUD-SRC, and the performance of DPAS is close to that of Task-SRC. The corresponding reasons are the same as those of observations from 4) to 7) for Fig. 3 and Fig. 4.

Next, we compare the performance of CPAS, TPAS, Task-SRC, and TDAM in terms of average cost of all STOs and report the results in Fig. 7. According to Eq. (2), with more assigned time slots, a STO's auction cost is higher. As shown

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2020.2987881, IEEE Transactions on Mobile Computing

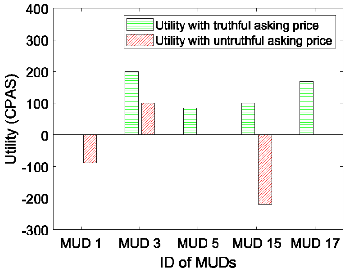


Fig. 9. MUD's truthfulness in CPAS.

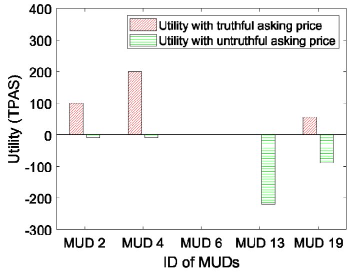


Fig. 10. MUD's truthfulness in TPAS.

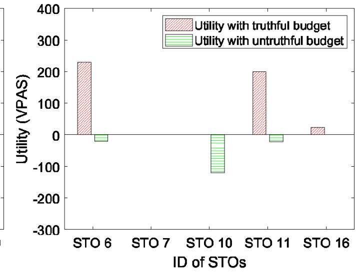


Fig. 11. STO's truthfulness in VPAS.

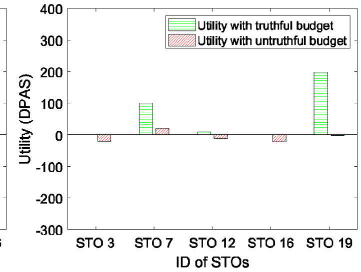


Fig. 12. STO's truthfulness in DPAS.

in Fig. 7, the average cost of all STOs increases when the number of MUDs increases because if more MUDs participate in crowdsensing, more time slots of a task are assigned to MUDs and more cost is spent to pay for MUDs' sensing service. Besides, Task-SRC has the largest average cost as it has the highest MUDs' working time utilization, and TDAM has the smallest average cost as it has the lowest MUDs' working time utilization. In particular, the average cost of CPAS is very close to that of Task-SRC, and the average cost of CPAS or TPAS is higher than that of TDAM, indicating that our distributed auctions can obtain effective solutions to the optimization problem in Eq. (2).

For each MUD's local auction, the average valuation of all MUDs in VPAS, DPAS, MUD-SRC, and TDAM is compared via Fig. 8. As defined in Eq. (4), an MUD's auction valuation is increased as the number of assigned time slots grows. The results in Fig. 8 show that the average MUDs' valuation increases when the number of tasks increases. This is because more tasks lead to more assigned time slots and higher valuation for MUDs. In addition, MUD-SRC performs best and TDAM performs worst; especially, the performance of VPAS and DPAS is close to that MUD-SRC. Thus, the effectiveness of our distributed auctions, VPAS and DPAS, can be validated.

Furthermore, we verify truthfulness of CPAS, TPAS, VPAS and DPAS, in which there are 20 tasks and 20 MUDs. In the experiments, we randomly selected one MUD in CPAS and TPAS (or one STO in VPAS and DPAS) at a time, set fake asking prices (or budgets) to the selected MUD (or STO), and compare received utilities when truthfully bidding and untruthfully bidding. We totally select five different MUDs and five different STOs and present the results in Figs. 9-12. Notice that all the selected MUDs (or STOs) cannot receive higher utilities via cheating on asking prices (or budgets). For example, in Fig. 9, the selected MUDs are the 1st MUD, the 3rd MUD, the 5th MUD, the 15th MUD, and the 17th MUD. The situations of these MUDs are illustrated in the following: i) the 1st MUD's utility is reduced to a negative value when cheating; ii) the 3rd MUD wins the auction when bidding truthfully and untruthfully, but its utility is reduced when bidding untruthfully; iii) the 5th MUD receives the reduced utility when bidding untruthfully; iv) the 15th MUD obtains a positive utility with truthful prices but a negative utility with fake prices; and v) the 17th MUD's utility is reduced to a negative value when cheating. Similarly, for the selected STOs in Fig. 11, there are four different situations: i) the utilities of the 6th and the 11th STOs are reduced from a positive value to a negative value; ii) the utility of the 7th STO remains zero; iii) the utility of the 10th STO is decreased from zero to a negative value; and iv) utility of the 16th STO is decreased from a positive value to zero. Therefore, the results of Figs. 9-12 demonstrate that no MUD or STO can enhance the received utility via cheating in our four proposed auction schemes.

From the above comparison, the adoption of CPAS, TPAS, VPAS, and DPAS in different crowdsensing scenarios can be briefly summarized as follows. Under the STOs' distributed auction framework, STOs should select CPAS to improve allocation efficiency and working time utilization, and should select TPAS if they would like small auction cost and low time complexity. On the other hand, under the MUDs' distributed auction framework, VPAS should be adopted is the goal of MUDs is to enhance allocation efficiency and working time utilization, and DPAS should be adopted if they prefer low time complexity.

X. Conclusion

To motivate mobile users to implement sensing tasks in MCSs, we propose two distributed auction frameworks and four novel distributed auction schemes, including CPAS, TPAS, VPAS, and DPAS, for task assignment and schedule. Our proposed auction schemes have the following major innovations: i) the auction model is practical by taking into account multi-dimensional task diversity, including partial fulfillment, bilaterally-multi-schedule, attribute diversity, and price diversity; ii) the four auction schemes can be implemented within well-designed distributed auction frameworks, in which each task owner/mobile user can locally and independently mange auction process; iii) all the four auction schemes possess nice properties, including computational-efficiency, individual-rationality and incentive-compatibility, and CPAS and TPAS can also satisfy budget-feasibility. Finally, comprehensive real-data experiments well confirm the effectiveness of our proposed auction schemes.

Acknowledgment

This work is partly supported by the U.S. National Science Foundation (NSF) under grant NOs. 1741277, 1829674, 1704287, and 1912753.

REFERENCES

[1] M. Han, J. Li, Z. Cai, and Q. Han, "Privacy reserved influence maximization in gps-enabled cyber-physical and online social networks," in *IEEE SocialCom*, October 2016, pp. 284–292.

[2] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges," *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32–39, 2011.

[3] M. H. Cheung, R. Southwell, F. Hou, and J. Huang, "Distributed time-sensitive task selection in mobile crowdsensing," in *ACM MobiHoc*, Hangzhou, China, June 2015, pp. 157–166.

[4] J. Li, Z. Cai, M. Yan, and Y. Li, "Using crowdsourced data in location-based social networks to explore influence maximization," in *IEEE INFOCOM*. IEEE, 2016, pp. 1–9.

[5] Q. Hu, S. Wang, L. Ma, R. Bie, and X. Cheng, "Anti-malicious crowdsourcing using the zero-determinant strategy," in *IEEE ICDCS*, Atlanta, USA, June 2017.

[6] C. Wang, C. Wang, Z. Wang, X. Ye, J. X. Yu, and B. Wang, "Deepdirect: Learning directions of social ties with edge-based network embedding," *IEEE Transactions on Knowledge and Data Engineering*, 2018.

[7] W. Li, J. Yu, X. Cheng, R. Bie, and F. Zhao, "An extensible and flexible truthful auction framework for heterogeneous spectrum markets," *to appear in IEEE Transactions on Cognitive Communications and Networking*, vol. 2, no. 4, pp. 427–441, 2016.

[8] Y. Wang, Z. Cai, G. Yin, Y. Gao, X. Tong, and G. Wu, "An incentive mechanism with privacy protection in mobile crowdsourcing systems," *Computer Networks*, vol. 102, pp. 157–171, 2016.

[9] W. Li, S. Wang, X. Cheng, and R. Bie, "Truthful multi-attribute auction with discriminatory pricing in cognitive radio networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 18, no. 1, pp. 3–13, January 2014.

[10] W. Li, S. Wang, and X. Cheng, "Truthful multi-attribute auction with discriminatory pricing in cognitive radio networks," in *ACM MobiCom Workshop on CRAB*, Miami, USA, October 2013, pp. 21–30.

[11] W. Li, M. Larson, C. Hu, R. Li, X. Cheng, and R. Bie, "Secure multi-unit sealed first-price auction mechanisms," *Security and Communication Networks, SI on cyber security, crime, and forensics of wireless networks and applications*, vol. 9, no. 16, pp. 3833–3843, 2016.

[12] Y. Wang, Z. Cai, X. Tong, Y. Gao, and G. Yin, "Truthful incentive mechanism with location privacy-preserving for mobile crowdsourcing systems," *Computer Networks*, vol. 135, pp. 32–43, 2018.

[13] Z. Feng, Y. Zhu, Q. Zhang, L. M. Ni, and A. V. Vasilakos, "Trac: Truthful auction for location-aware collaborative sensing in mobile crowdsourcing," in *IEEE INFOCOM*, Toronto, Canada, April 2014, pp. 1231–1239.

[14] H. Jin, L. Su, D. Chen, H. Guo, K. Nahrstedt, and J. Xu, "Thanos: Incentive mechanism with quality awareness for mobile crowd sensing," *IEEE Transactions on Mobile Computing*, 2018.

[15] Y. Zhu, Q. Zhang, H. Zhu, J. Yu, J. Cao, and L. M. Ni, "Towards truthful mechanisms for mobile crowdsourcing with dynamic smartphones," in *IEEE ICDCS*, Madrid, Spain, July 2014, pp. 11–20.

[16] X. Zhang, G. Xue, R. Yu, D. Yang, and J. Tang, "Truthful incentive mechanisms for crowdsourcing," in *IEEE INFOCOM*, HONG KONG, China, April 2015, pp. 2830–2838.

[17] S. Ji and T. Chen, "Incentive mechanisms for discretized mobile crowdsensings," *IEEE Transactions on Wireless Communications*, vol. 15, no. 1, pp. 146–161, Janary 2016.

[18] H. Shah-Mansouri and V. W. Wong, "Profit maximization in mobile crowdsourcing: A truthful auction mechanism," in *IEEE ICC*, London, UK, June 2015, pp. 3216–3221.

[19] Y. Zhang and M. van der Schaar, "Reputation-based incentive protocols in crowdsourcing applications," in *IEEE INFOCOM*, Orlando, Florida, USA, March 2012, pp. 2140–2148.

[20] H. Xie and J. Lui, "Modeling crowdsourcing systems: Design and analysis of incentive mechanism and rating system," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 42, no. 2, pp. 52–54, September 2014.

[21] X. Duan, C. Zhao, S. He, P. Cheng, and J. Zhang, "Distributed algorithms to compute walrasian equilibrium in mobile crowdsensing," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 5, pp. 4048–4057, 2017.

[22] D. Yang, G. Xue, X. Fang, and J. Tang, "Crowdsourcing to smartphones: incentive mechanism design for mobile phone sensing," in *ACM Mobicom*, Istanbul, Turkey, August 2012, pp. 173–184.

[23] J. Li, Z. Cai, J. Wang, M. Han, and Y. Li, "Truthful incentive mechanisms for geographical position conflicting mobile crowdsensing systems," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 2, pp. 324–334, 2018.

[24] B. Song, H. Shah-Mansouri, and V. W. Wong, "Quality of sensing aware budget feasible mechanism for mobile crowdsensing," *IEEE Transactions on Wireless Communications*, vol. 16, no. 6, pp. 3619–3631, 2017.

[25] L. Tang, S. He, and Q. Li, "Double-sided bidding mechanism for resource sharing in mobile cloud," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 2, pp. 1798–1809, 2016.

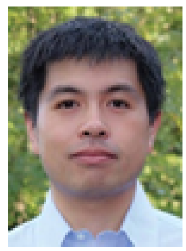
[26] X. Zhang, L. Jiang, and X. Wang, "Incentive mechanisms for mobile crowdsensing with heterogeneous sensing costs," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3992–4002, 2019.

[27] M. Khaledi, M. Khaledi, and S. K. Kasera, "Profitable task allocation in mobile cloud computing," in *Proceedings of the 12th ACM Symposium on QoS and Security for Wireless and Mobile Networks*. ACM, 2016, pp. 9–17.

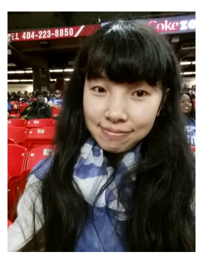
[28] H. Huang, Y. Xin, Y.-E. Sun, and W. Yang, "A truthful double auction mechanism for crowdsensing systems with max-min fairness," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2017, pp. 1–6.

[29] B. Cao, S. Xia, J. Han, and Y. Li, "A distributed game methodology for crowdsensing in uncertain wireless scenario," *IEEE Transactions on Mobile Computing*, 2019.

[30] V. Krishna, *Auction Theory*, 2nd ed. Elsevier, 2010.



Zhipeng Cai received his PhD and M.S. degrees in the Department of Computing Science at University of Alberta, and B.S. degree from Beijing Institute of Technology. Dr. Cai is currently an Assistant Professor in the Department of Computer Science at Georgia State University. Dr. Cai's research areas focus on Networking, Privacy and Big data. He has published more than 50 journals papers, including more than 20 IEEE/ACM Transactions papers, such as IEEE Transactions on Knowledge and Data Engineering, IEEE Transactions on Dependable and Secure Computing, IEEE/ACM Transactions on Networking, IEEE Transactions on Mobile Computing, etc. Dr. Cai is the recipient of an NSF CAREER Award. He is an editor/guest editor for Algorithmica, Theoretical Computer Science, Journal of Combinatorial Optimization, and IEEE/ACM Transactions on Computational Biology and Bioinformatics. He is a senior member of the IEEE.



Zhuojun Duan received her PhD degree in the Department of Computing Science at Georgia State University, and M.S. degree from Shaanxi Normal University. Dr. Duan is currently an Assistant Professor in the Department of Computer Science at University of Hartford. Her research interest is Internet of Things, game theory and social network.

Wei Li currently is an Assistant Professor in the Department of Computer Science at Georgia State University. Dr. Li received her Ph.D. degree in Computer Science from The George Washington University in 2016 and her M.S. degree in Computer Science from Beijing University of Posts and Telecommunications in 2011. She won the Best Paper Awards in ACM MobiCom Workshop CRAB 2013 and WASA 2011, respectively. Her current research mainly spans the areas of security and privacy for the Internet of Things and Cyber-Physical Systems, secure and privacy-aware computing, game theory, and algorithm design and analysis. She is a member of IEEE and ACM.