Truthful Incentive Mechanisms for Crowdsourcing

Xiang Zhang, Guoliang Xue, Ruozhou Yu, Dejun Yang, Jian Tang

***Abstract*—With the prosperity of smart devices, crowdsourcing has emerged as a new computing/networking paradigm. Through the crowdsourcing platform, service requesters can buy service from service providers. An important component of crowdsourcing is its incentive mechanism. We study three models of crowdsourcing, which involve cooperation and competition among the service providers. Our simplest model generalizes the well-known user-centric model studied in a recent Mobicom paper. We design an incentive mechanism for each of the three models, and prove that these incentive mechanisms are individually rational, budget-balanced, computationally efficient, and truthful.**

1. Introduction

In 2005, Jeff Howe, an editor at Wired Magazine, coined the term "crowdsourcing" to represent "*the act of taking a job traditionally performed by a designated agent (usually an employee) and outsourcing it to an undefined, generally large group of people in the form of an open call*" [14]. Crowdsourcing takes advantage of the wisdom of individuals, teams, and communities to accomplish tasks. One famous example is Wikipedia. Instead of hiring professional writers and editors to manage millions of articles, Wikipedia grants the crowd the ability of creating information on their own. In recent years, crowdsourcing has been widely used in many fields, including environmental monitoring, e.g., Smart Citizen [1], healthcare, e.g., HealthMap [2], transportation, e.g., Uber [3], online marketplace, e.g., Amazon Mechanical Turk [4], and Q&A website, e.g. Quora [5]. In the networking research field, a large body of research has been conducted on *crowdsensing*, which applies the concept of crowdsourcing to mobile phone sensing [6, 9, 10, 12, 21, 25]. Regardless of the fields, all these crowdsourcing applications have one component in common: *incentives*. Crowdsourcing participants may have associated costs while performing crowdsourcing tasks. Therefore it is necessary to provide appropriate incentives to compensate their costs. To this end, incentive mechanisms are designed to determine who should be awarded and how much should be awarded. A typical crowdsourcing application provides a platform for the service requesters, who have crowdsourcing jobs that need to be done and are willing to pay for the service, and the service providers, who would like to perform crowdsourcing tasks in exchange for rewards. A lot of research effort [7, 8, 13, 15, 19, 22, 23, 25, 26] has been focused on developing such incentive mechanisms to encourage users to participate in crowdsourcing. However, most of the studies assume that one crowdsourcing job can be completed by one service provider. In some cases, a single service provider may not have enough resource to finish a job, and it may require the cooperation from other service providers. For example, a company wants to have a WiFi map to depict the WiFi signal strength in different buildings located in a shopping area. Collecting the signal strength in one building can be considered as one job. Obviously, this job cannot be completed by a single service provider, but needs the collective effort from a group of providers. Unfortunately, there are no off-the-shelf incentive mechanisms designed in the literature to tackle this problem, while satisfying several desirable economic properties: individual rationality, budget-balance, computational efficiency, and truthfulness. We will elaborate these properties in Section 3-D.

Zhang, Xue, and Yu are with Arizona State University, Tempe, AZ 85287. Email: {xzhan229, xue, ruozhouy}@asu.edu. Yang is with Colorado School of Mines, Golden, CO 80401. Email: djyang@mines.edu. Tang is with Syracuse University, Syracuse, NY 13244. Email: jtang02@syr.edu. This research was supported in part by ARO grant W911NF-09-1-0467 and NSF grants 1217611 and 1218203. The information reported here does not reflect the position or the policy of the federal government.

In this paper, we consider the scenarios where a crowdsourcing job may require the collective effort of multiple service providers. We assume that each job consists of a number of small tasks, and each winning service provider can work on a subset of these tasks. Each requester submits the jobs and the corresponding reward to the platform. Each provider submits the tasks it is willing to work on along with the price it requires. The platform acts as an auctioneer, decides which jobs can be completed and who should work on what tasks, as well as the corresponding payments.

The main contributions of this paper are as follows:

- To the best of our knowledge, we are the first to design incentive mechanisms for crowdsourcing where cooperation of service providers is required to finish the jobs.
- We consider three different models: single-requester single-bid model, single-requester multiple-bid model, and multiple-requester multiple-bid model. Note that the first model is a generalization of existing work [25].
- We design an incentive mechanism for each of these models, and prove that these incentive mechanisms are individually rational, budget-balanced, computationally efficient, and truthful.

The remainder of this paper is organized as follows. In Section 2, we review the state-of-art research on incentive mechanism design for crowdsourcing. In Section 3, we present three system models for crowdsourcing and list some important economic properties. We design incentive mechanisms for the three models in Sections 4, 5, and 6, respectively. In Section 7, we present our performance evaluation results. We draw our conclusions in Section 8.

2. Related Work

In this section, we review the state-of-art research on incentive mechanism design for crowdsourcing, including crowdsensing.

As pioneers on designing incentive mechanism specifically for crowdsourcing, DiPalantino and Vojnovic [7] proposed an all-pay auction crowdsourcing model where users could select the job they want to work on crowdsourcing platform. Considering the online arrival of users, Singer and Mittal [22] proposed two truthful online incentive mechanisms for crowdsourcing, where the first one maximizes the number of tasks with budget-balance, and the second one minimizes the total payments if the number of winning jobs is given and fixed. Along this line, Singla and Krause [23] proposed BP-UCB, a no-regret online truthful auction mechanism, while achieving near-optimal utility for all service providers in crowdsourcing. Besides monetary incentives, Zhang and van der Schaar [27] built a reputation-based incentive protocol to discard free-riding and false-reporting in crowdsourcing systems.

In the context of crowdsensing, Yang *et al.* [25] proposed two models, platform-centric model and user-centric model, to encourage service providers working on crowdsensing tasks. Using randomized techniques, Koutsopoulos [15] presented a mechanism which determines the user participation level and allocates payments. For the case where providers may arrive in an online manner, Zhao *et al.* [28] designed two online incentive mechanisms OMZ and OMG using a multi-stage sampling-accepting process. Zhang *et al.* [26] proposed three online auction-based incentive mechanisms, TBA, TOIM, and TOIM-AD, where TBA maximizes the platform utility and others achieve truthfulness. In addition to incentives, Li and Cao [17] considered privacy protection in the mechanism design.

However, none of the above works considered the case when a provider lacks enough resource to finish a job single-handedly on a crowdsourcing platform. Xu *et al.* [24] proposed DATA, which considers users cooperation in crowdsourcing, as a truthful double auction mechanism. It simplifies the model by assuming that a job can be finished as long as it is assigned with enough number of providers. However, it does not distinct each provider by their capabilities. In [18], Li *et al.* studied single set-cover truthful and individually rational auctions, which can be applied into the scenario of crowdsourcing, with an approximation ratio to the social-welfare. However, the requester does not have a budget-constraint, which means the requester might have a negative utility.

3. System Model and Economic Properties

In this section, we present three models of crowdsourcing, whose truthful *incentive mechanisms* (IMs) are presented in the next three sections. Our first model contains the *user-centric model* of [25] as a special case. Our second model is a generalization of the first. The first two models assume a single requester and multiple providers, engaging competition only among the providers. Our third model assumes multiple requesters and providers, engaging competition among both the providers and the requesters. At the end of this section, we present some important economic properties.

A. The SS-Model: Single-requester Single-bid

There is a universal set of *tasks*: $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_l\}$, where the l tasks are distinct and each task is indivisible. There is a set of *jobs*: $\mathcal{J} = \{J_1, J_2, \ldots, J_m\}$, where each of the m jobs is a *multi-subset* of $\mathcal{T}$. For job J_i, there is a corresponding *valuation* $v_i \geq 0$, which is a *private type* [16] and known only to the job owner. For example, we may have $J_6 = \{\tau_1, \tau_1, \tau_1, \tau_4, \tau_5\}$. Job J_6 is *finished* iff three copies of τ_1, one copy of τ_4, and one copy of τ_5 are completed.

The SS-model assumes a single requester, who owns all m jobs. There is a set of n providers: $\mathcal{P} = \{P_1, P_2, \ldots, P_n\}$, where P_j is capable of working on T_j, which is a multi-subset of $\mathcal{T}$. For P_j to work on T_j, there is an associated *cost* $c_j \geq 0$, which is a private type, and known only to P_j.

Crowdsourcing in the SS-model works as follows. The platform posts the job set $\mathcal{J}$. Each provider P_j submits its *bid* $A_j = (T_j, a_j)$ to the platform, where a_j is P_j's *ask*, i.e., P_j is willing to work on T_j iff the *payment* is no smaller than a_j. Upon receiving the bids from the providers, the platform selects the set of *winning providers* $\mathcal{P}_w \subseteq \mathcal{P}$; and decides the *payment* $p_j \geq a_j$ for each $P_j \in \mathcal{P}_w$. Each winning provider P_j will complete T_j and receive the payment p_j from the platform.

We say a set of jobs is *completed* by $\mathcal{P}_w$ (hence every job in the set is completed) if the multi-set union of this set is a subset of $\biguplus_{P_j \in \mathcal{P}_w} T_j$, i.e., the set of winning providers can collectively work on all tasks in the union. The *job indicator* y_i is defined to be 1 if J_i is completed, and 0 otherwise. We will use these concepts for all three models, and will not repeat for the SM-model and the MM-model.

As a technical convention, we set $p_j = 0$ for $P_j \notin \mathcal{P}_w$. We set the *provider indicator* $x_j = 1$ if $P_j \in \mathcal{P}_w$, and 0 otherwise.

The *utility of provider* P_j is defined as

$$u_j^{SS} = p_j - x_j c_j, \tag{3.1}$$

which is the payment received minus the cost incurred.

The *utility of the platform* is defined as

$$U^{SS} = \sum_{i=1}^{m} y_i v_i - \sum_{j=1}^{n} p_j, \tag{3.2}$$

which is the total value added minus the total payment made.

Remark 3.1: By restricting each job in the SS-model to one distinct task, we obtain the user-centric model studied in [25]. Therefore our SS-model is a generalization of the user-centric model studied in [25]. □

For ease of presentation, we define some notations here. The *valuation vector* is denoted as $\mathbf{v} = (v_1, v_2, \ldots, v_m)$. The *cost vector* is $\mathbf{c} = (c_1, c_2, \ldots, c_n)$. The *ask vector* is $\mathbf{A} = (A_1, A_2, \ldots, A_n)$. The *payment vector* is $\mathbf{p} = (p_1, p_2, \ldots, p_n)$. The *provider indicator vector* is $\mathbf{x} = (x_1, x_2, \ldots, x_n)$. The *job indicator vector* is $\mathbf{y} = (y_1, y_2, \ldots, y_m)$.

B. The SM-Model: Single-requester Multiple-bid

In the SS-model presented in Section 3-A, each provider can submit only one bid. In the SM-model, we relax this constraint by allowing a provider to submit multiple bids, but will be assigned to work on at most one of the sets of tasks it bids.

The definitions of $\mathcal{T}$, $\mathcal{J}$, $\mathbf{v}$, $\mathcal{P}$, $\mathcal{P}_w$, $\mathbf{y}$, and $\mathbf{p}$ are the same as in Section 3-A. Rather than submitting a single bid (T_j, a_j) as in the SS-model, in the SM-model, each P_j submits *a set of bids*: $A_j = \{(T_j^1, a_j^1), (T_j^2, a_j^2), \ldots, (T_j^{\sigma_j}, a_j^{\sigma_j})\}$, where T_j^k is a multi-subset of $\mathcal{T}$, and a_j^k is the corresponding ask, i.e., P_j is willing to work on T_j^k if the corresponding payment is at least a_j^k. Note that A_j in the SM-model denotes *a set of bids*, while A_j in the SS-model denotes *a single bid*. This abuse of notation simplifies presentation without creating confusion, since its meaning should be clear from the context.

For each $j \in \{1, 2, \ldots, n\}$ and each $k \in \{1, 2, \ldots, \sigma_j\}$, there is a *cost* $c_j^k \geq 0$ for P_j to work on T_j^k. As in Section 3-A, this cost is a private type which is known only to P_j. The *cost vector* $\mathbf{c}$ has the form $(c_1^1, \ldots, c_1^{\sigma_1}; c_2^1, \ldots, c_2^{\sigma_2}; \ldots; c_n^1, \ldots, c_n^{\sigma_1})$. The *ask vector* $\mathbf{A}$ has the form $(A_1, A_2, \ldots, A_n)$.

Crowdsourcing in the SM-model works as follows. The platform posts $\mathcal{J}$. Each provider P_j submits its bids $A_j = \{(T_j^1, a_j^1), \ldots, (T_j^{\sigma_j}, a_j^{\sigma_j})\}$ to the platform, where P_j is willing to work on T_j^k iff the reward is no smaller than c_j^k. The platform selects the set of winning providers $\mathcal{P}_w \subseteq \mathcal{P}$; assigns each $P_j \in \mathcal{P}_w$ to work on $T_j^{\delta(j)}$ where $\delta(j) \in \{1, \ldots, \sigma_j\}$ (indicated by $x_j^{\delta(j)} = 1$); and determines the payment $p_j \geq a_j^{\delta(j)}$. Each winning provider P_j will complete $T_j^{\delta(j)}$ and receive the payment p_j from the platform.

Since each provider in the SM-model submits multiple bids, the *provider indicator vector* $\mathbf{x}$ has the form $(x_1^1, \ldots, x_1^{\sigma_1}; x_2^1, \ldots, x_2^{\sigma_2}; \ldots, x_n^1, \ldots, x_n^{\sigma_n})$, where x_j^k is 1 iff P_j is assigned to work on T_j^k, and 0 otherwise. The indicator vector $\mathbf{x}$ defines a *task assignment* function δ such that $\delta(j) = k$ if $x_j^k = 1$ for some $k \in \{1, \ldots, \sigma_j\}$, and $\delta(j) = 0$ otherwise.

The *utility of provider* P_j is defined as

$$u_j^{SM} = p_j - \sum_{k=1}^{\sigma_j} x_j^k c_j^k, \tag{3.3}$$

which is the payment received minus the cost incurred.

The *utility of the platform* is defined as

$$U^{SM} = \sum_{i=1}^{m} y_i v_i - \sum_{j=1}^{n} p_j, \tag{3.4}$$

which is the total value added minus the total payment made.

Remark 3.2: When $\sigma_j = 1$ for all $j = 1, 2, \ldots, n$, the SM-model reduces to the SS-model. Hence the SM-model is a generalization of the SS-model. □

C. The MM-Model: Multiple-requester Multiple-bid

In the SS-model and the SM-model, there is only one requester. With the increasing popularity of crowdsourcing, there may be more requesters, hence generating competition among the requesters, in addition to the providers. Our MM-model is designed to cope with such scenarios.

The definitions of $\mathcal{T}, \mathcal{J}, \mathbf{v}, \mathcal{P}, \mathcal{P}_w, \mathbf{c}, \mathbf{A}, \delta, \mathbf{x}$, and $\mathbf{y}$ are the same as in Section 3-B. In addition, there is a set of m requesters $\mathcal{R} = \{R_1, R_2, \ldots, R_m\}$. Requester R_i owns job J_i, which has a *valuation* $v_i \geq 0$ known only to R_i.

The requesters act as *buyers* who buy service from providers, and providers act as *sellers* who sell their service. In the rest of the paper, we use the terminology of buyers and requesters, sellers and providers, auctioneer and platform interchangeably.

Crowdsourcing in the MM-model works as follows. The buyers and sellers submit their bids to the platform, where the bid for buyer R_i has the form $B_i = (J_i, b_i)$, and the bids for seller P_j has the form $A_j = \{(T_j^1, a_j^1), \ldots, (T_j^{\sigma_j}, a_j^{\sigma_j})\}$. Working as the auctioneer, the platform selects the set of winning buyers $\mathcal{R}_w \subseteq \mathcal{R}$ and the set of winning sellers $\mathcal{P}_w \subseteq \mathcal{P}$; assigns each $P_j \in \mathcal{P}_w$ to work on $T_j^{\delta(j)}$ where $\delta(j) \in \{1, \ldots, \sigma_j\}$ (indicated by $x_j^{\delta(j)} = 1$); determines $q_i \leq b_i$ to be collected from R_i for each $R_i \in \mathcal{R}_w$, and $p_j \geq a_j^{\delta(j)}$ to be paid to each $P_j \in \mathcal{P}_w$. Each winning provider P_j will complete $T_j^{\delta(j)}$ and receive the payment p_j from the platform. Each winning buyer R_i will receive the service from the winning providers, and pay the platform the fee of q_i. The indicator vectors $\mathbf{x}$ and $\mathbf{y}$ are the same as in the SM-model.

The *utility of buyer* R_i is defined as

$$\mu_i^{MM} = y_i v_i - q_i. \tag{3.5}$$

The *utility of seller* P_j is defined as

$$u_j^{MM} = p_j - \sum_{k=1}^{\sigma_j} x_j^k c_j^k. \tag{3.6}$$

The *utility of the auctioneer* is defined as

$$U^{MM} = \sum_{i=1}^{m} q_i - \sum_{j=1}^{n} p_j. \tag{3.7}$$

D. Desirable Economic Properties for Incentive Mechanisms

Apart from serving as the auctioneer, another important functionality of the crowdsourcing platform is to determine prices that are fair to all requesters and providers. Auctions with one buyer and multiple sellers are called *single-auctions*, and auctions with multiple buyers and multiple sellers are called *double-auctions*. A keen concern about an auction is that some requester or provider may gain a higher utility by dishonest behavior. This makes the mechanism vulnerable to malicious price manipulation. Therefore, there are several desirable economic properties that a good auction should possess. We list them in the following.

- **Individual rationality**: An auction is individually rational if for each buyer (resp. seller), its utility is non-negative when reporting its true valuation (resp. cost).
- **Budget-balance**: An auction is budget-balanced if by the end of an auction, the auctioneer's utility is non-negative.
- **Computational efficiency**: An auction mechanism is computationally efficient if it can be executed within polynomial time.

- **Truthfulness**: An auction is truthful if for each buyer (resp. seller), it cannot increase its utility by bidding a value deviating from its true valuation (resp. cost), no matter what others bid.

For each of the three models, we will design an IM that satisfies all of the above properties.

Among these four properties, truthfulness is the most crucial property in auction theory. In [20], Myerson proved that a single auction is truthful if it satisfies *monotonicity* and charges each individual the corresponding *critical value*. A *monotonic auction* is an auction that for any buyer (resp. seller), if it wins the auction by bidding (resp. asking) a value, it can still win the auction by bidding (resp. asking) a higher (resp. lower) value. The *critical value* for a buyer (resp. seller) is the maximum (resp. minimum) value, such that the buyer (resp. seller) would lose the auction if it bids less (resp. asks more) than this value.

4. Incentive Mechanism for the SS-Model

In this section, we present an incentive mechanism for crowdsourcing in the SS-model, named **IMC-SS**. We also prove that **IMC-SS** is individually rational, budget-balanced, computationally efficient, and truthful.

A. Design Rationale

In our design, **IMC-SS** consists of three steps: job selection, winning provider selection, and pricing. In the first step, we carefully select the jobs such that no monopoly provider (see Section 4-B) exists based on the selection, and the providers can finish all the selected jobs. In the second step, we choose winning providers in a greedy manner according to certain criterion (explained later). Finally, we determine the payment for each winning provider by calculating its critical value.

B. Design Details

Now we present **IMC-SS** in details.

Job Selection: We first select a subset of jobs that satisfies two conditions: 1) selected jobs can be finished by all of the providers; and 2) no monopoly provider exists over the selected jobs. A provider is called a *monopoly provider* if its elimination would make the rest of providers unable to finish the jobs. The second condition is essential to guarantee truthfulness, as *a monopoly provider can ask an arbitrarily high price* if the goal of the platform is to finish all the selected jobs. Ideally, one should select those jobs with the maximum total valuation while satisfying the above two conditions. However, it can be shown that the Knapsack problem, a well-known NP-hard problem [11], is a special case of this optimization problem. Thus it is unlikely to find an optimal subset of jobs in polynomial time. Therefore, we iteratively select jobs in a greedy manner, as illustrated in Algorithm 1. In each iteration, we select the job with the largest valuation without violating the above two conditions.

In Line 5 of Algorithm 1, we need to check whether a set of jobs can be completed by a set of providers, given their bids. We elaborate such a procedure in Algorithm 2. The pseudo code is written in a way so that it can be used for all three models: Lines 6-7 are used for the SS-model, and Lines 9-10 are used for the SM-model and the MM-model. The main rationale of Algorithm 2 is that each time a provider with the smallest bid is selected, the unfinished task set $\mathcal{T}$ is updated. If $\mathcal{T}$ is empty, which means these providers can cover all the tasks in $\mathcal{J}$, Algorithm 2 will return **TRUE**. Algorithm 2 returns **FALSE** iff the providers cannot cover all all the tasks in $\mathcal{J}$.

Algorithm 1: $JobSel(\mathcal{J}, \mathbf{v}, \mathbf{A})$

```
1  $\mathcal{J}^* \leftarrow \emptyset$;
2  WLOG, assume $v_1 \geq \cdots \geq v_m$ for simplicity;
3  for $i \leftarrow 1$ to $m$ do
4      if $Feasible(\mathcal{P}\setminus\{P_j\}, \mathcal{J}^*\cup\{J_i\}, \mathbf{A})$ for all $P_j \in \mathcal{P}$ then
5          $\mathcal{J}^* \leftarrow \mathcal{J}^* \cup \{J_i\}$;
6      end
7  end
8  $\nu \leftarrow \sum_{J_i \in \mathcal{J}^*} v_i$;
9  return $\mathcal{J}^*, \nu$;
```

Algorithm 2: $Feasible(\mathcal{P},\ \mathcal{J},\ \mathbf{A})$

```
1  $\mathcal{T} \leftarrow \biguplus_{J_i \in \mathcal{J}} J_i$, $\mathcal{P}' \leftarrow \mathcal{P}$;
2  while $\mathcal{T} \neq \emptyset$ do
3      if no provider in $P'$ can work on any task in $\mathcal{T}$ then
4          return false
5      else if in the SS-model then
6          $T_{j^*} \leftarrow \arg\min_{T_j}\{a_j | P_j \in \mathcal{P}', T_j \cap \mathcal{T} \neq \emptyset\}$;
7          $\mathcal{T} \leftarrow \mathcal{T} \setminus T_{j^*}$, $\mathcal{P}' \leftarrow \mathcal{P}' \setminus \{P_{j^*}\}$;
8      else
9          $T_{j^*}^{k^*} \leftarrow \arg\min_{T_j^k}\{a_j^k | P_j \in \mathcal{P}', T_j^k \cap \mathcal{T} \neq \emptyset\}$;
10         $\mathcal{T} \leftarrow \mathcal{T} \setminus T_{j^*}^{k^*}$, $\mathcal{P}' \leftarrow \mathcal{P}' \setminus \{P_{j^*}\}$;
11     end
12 end
13 return true;
```

Winning Provider Selection: We next determine the winning providers for completing the selected jobs, as shown in Algorithm 3. The selection algorithm proceeds iteratively. In each iteration (Lines 3 to 4), we select a provider according to the minimum ratio of ask to the number of tasks that can be used to complete the jobs. Due to Condition 1) from the previous step, it is guaranteed that such a set of winning providers exists.

Pricing: As shown in Algorithm 4, here we decide the payment for each winning provider. According to Myerson's Theorem [20], the payment of each winning provider should be set to its critical value to guarantee truthfulness. To find its critical value, we temporarily put each winning provider P_j aside and run a process similar to $WinProviderSel$ over the rest of the providers. For every selected provider, we compute the value such that if P_j asks this value, it would replace the selected provider as the winner for this iteration

Algorithm 3: $WinProviderSel(\mathcal{P},\ \mathcal{J}^*,\ \mathbf{A})$

```
1  $\mathcal{P}_w \leftarrow \emptyset$, $\mathcal{T}^* \leftarrow \biguplus_{J_i \in \mathcal{J}^*} J_i$;
2  while $\mathcal{T}^* \neq \emptyset$ do
3      $P_{j^*} \leftarrow \arg\min_{P_j}\{\frac{a_j}{|\mathcal{T}^* \cap T_j|} | P_j \in \mathcal{P} \setminus \mathcal{P}_w\}$;
       // $\frac{a_j}{0}$ is taken as $+\infty$
4      $\mathcal{P}_w \leftarrow \mathcal{P}_w \cup \{P_{j^*}\}$, $\mathcal{T}^* \leftarrow \mathcal{T}^* \setminus T_{j^*}$;
5  end
6  return $\mathcal{P}_w$;
```

(Line 8). Eventually, we take the maximum of these values as the payment for P_j. Algorithm 4 also computes $p'_j \geq p_j$, which is the value that providers outside $\mathcal{P}_w$ should bid in order to take P_j's place in $\mathcal{P}_w$.

Algorithm 4: $Pricing\text{-}SS(\mathcal{J}^*,\ \mathcal{P},\ \mathcal{P}_w,\ \mathbf{A})$

```
1  $p_j \leftarrow 0$, $p'_j \leftarrow 0$ for all $P_j \in \mathcal{P}$;
2  $\mathcal{T}^* \leftarrow \biguplus_{J_i \in \mathcal{J}^*} J_i$;
3  for all $P_j \in \mathcal{P}_w$ do
4      $\mathcal{P}'_w \leftarrow \emptyset$, $\mathcal{T}' \leftarrow \mathcal{T}^*$;
5      while $\mathcal{T}' \neq \emptyset$ do
6          $P_{j^*} \leftarrow \arg\min_{P_k}\{\frac{a_k}{|\mathcal{T}' \cap T_k|} | P_k \in \mathcal{P} \setminus \mathcal{P}'_w,\ P_k \neq P_j\}$;
7          $P_{j'} \leftarrow \arg\min_{P_k}\{\frac{a_k}{|\mathcal{T}' \cap T_k|} | P_k \in \mathcal{P} \setminus \{\mathcal{P}_w \cup \mathcal{P}'_w\}\}$;
8          $p_j \leftarrow \max\{p_j, \frac{a_{j^*}|\mathcal{T}' \cap T_j|}{|\mathcal{T}' \cap T_{j^*}|}\}$;
9          $p'_j \leftarrow \max\{p'_j, \frac{a_{j'}|\mathcal{T}' \cap T_j|}{|\mathcal{T}' \cap T_{j'}|}\}$;
10         $\mathcal{P}'_w \leftarrow \mathcal{P}'_w \cup \{P_{j^*}\}$, $\mathcal{T}' \leftarrow \mathcal{T}' \setminus T_{j^*}$;
11     end
12 end
13 $p' \leftarrow \sum_{j=1}^{n} p'_j$;
14 return $\mathbf{p}, p'$;
```

Finally, we check if the auction result satisfies budget-balance. Here we compute an upper bound p' of $\sum_{j=1}^{n} p_j$, such that *p' cannot be controlled by any provider without losing the auction*, to guarantee budget-balance. The main algorithm of **IMC-SS** is illustrated in Algorithm 5.

Algorithm 5: IMC-SS$(\mathcal{J},\ \mathbf{v},\ \mathcal{P},\ \mathbf{A})$

```
1  $(\mathcal{J}^*, \nu) \leftarrow JobSel(\mathcal{J}, \mathbf{v}, \mathbf{A})$;
2  $\mathcal{P}_w \leftarrow WinProviderSel(\mathcal{P}, \mathcal{J}^*, \mathbf{A})$;
3  $(\mathbf{p}, p') \leftarrow Pricing(\mathcal{J}^*, \mathcal{P}, \mathcal{P}_w, \mathbf{A})$;
4  if $\nu < p'$ then
5      $\mathcal{P}_w \leftarrow \emptyset$, $\mathbf{p} \leftarrow \mathbf{0}$;
6  end
```

Remark 4.1: If we simply replace the condition $\nu < p'$ in Line 4 of Algorithm 5 with $\nu < \sum_{j=1}^{n} p_j$, the resulting auction is not truthful. Due to space limitation, our counter-example is not presented here. Our novel use of p' is essential to achieving budget-balance without sacrificing truthfulness. $\square$

C. A Walk-Through Example

We use the example in Fig. 1 to illustrate how **IMC-SS** works.

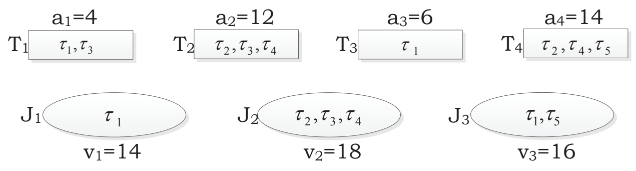


Fig. 1. A walk-through example for SIS

Job Selection: $\mathcal{J}^* = \emptyset$, and $v_2 > v_3 > v_1$. If we add J_2 into $\mathcal{J}^*$, the providers can finish it with no monopoly provider, and $\mathcal{J}^* = \{J_2\}$. Next, if we add J_3 into $\mathcal{J}^*$, P_4 becomes a monopoly provider, so J_3 can not be added into $\mathcal{J}^*$. Following the same rule, we add J_1 into $\mathcal{J}^*$, $\mathcal{J}^* = \{J_1, J_2\}$, and $\nu = 32$.

Winning Provider Selection: $\mathcal{P}_w = \emptyset$ and $\mathcal{T}^* = J_1 \biguplus J_2 = \{\tau_1, \tau_2, \tau_3, \tau_4\}$. $\frac{a_1}{|T_1 \cap \mathcal{T}^*|} = 2$, $\frac{a_2}{|T_2 \cap \mathcal{T}^*|} = 4$, $\frac{a_3}{|T_3 \cap \mathcal{T}^*|} = 6$, and $\frac{a_4}{|T_4 \cap \mathcal{T}^*|} = 7$. Thus, $\mathcal{P}_w = \{P_1\}$ and $\mathcal{T}^* = \{\tau_2, \tau_4\}$. Next, we add P_2 into $\mathcal{P}_w$, such that $\mathcal{P}_w = \{P_1, P_2\}$ and $\mathcal{T}^* = \emptyset$.

Pricing:

- p_1 and p'_1: At first $p_1 = p'_1 = 0$, $\mathcal{P}'_w = \emptyset$, and $\mathcal{T}' = \{\tau_1, \tau_2, \tau_3, \tau_4\}$. $\frac{a_2}{|T_2 \cap \mathcal{T}'|} = 4$, $\frac{a_3}{|T_3 \cap \mathcal{T}'|} = 6$, and $\frac{a_4}{|T_4 \cap \mathcal{T}'|} = 7$. So P_2 is selected as P_{j^*} and P_3 is selected as $P_{j'}$. We update $p_1 = \max\{p_1, 4 * |T_1 \cap \mathcal{T}'|\} = 8$ and $p'_1 = \max\{p'_1, 6 * |T_1 \cap \mathcal{T}'|\} = 12$. Then we add P_2 into $\mathcal{P}'_w$ and $\mathcal{T}' = \{\tau_1\}$. In the next round, P_3 is selected as P_{j^*} and $P_{j'}$. So $p_1 = \max\{8, 6\} = 8$, $p'_1 = \max\{12, 6\} = 12$, $\mathcal{P}'_w = \{P_2, P_3\}$, and $\mathcal{T}' = \emptyset$.
- p_2 and p'_2: With the same method, $p_2 = 14$ and $p'_2 = 18$.

In the end, $p' = p'_1 + p'_2 = 30$, $p_1 + p_2 = 22$, and $\nu > p'$. Thus, $\mathcal{P}_w = \{P_1, P_2\}$, $p_1 = 8$, $p_2 = 14$, and $U^{SS} = (14 + 18) - (8 + 14) = 10$.

D. Analysis

We now prove that **IMC-SS** satisfies the properties mentioned in Section 3-D with the help of the following four lemmas.

Lemma 4.1: IMC-SS is individually rational. $\square$

Proof: To prove individual rationality, we need to prove that for each $P_j \in \mathcal{P}$, if $a_j = c_j$, $u_j^{SS} \geq 0$. If $P_j \notin \mathcal{P}_w$, $u_j^{SS} = 0$. Now assume $P_j \in \mathcal{P}_w$. Hence $u_j^{SS} = p_j - c_j$. At the iteration where P_j is selected to enter $\mathcal{P}_w$, $\frac{a_j}{|\mathcal{T}^* \cap T_j|} = \frac{c_j}{|\mathcal{T}^* \cap T_j|}$ is less than or equal to $\frac{a_{j'}}{|\mathcal{T}^* \cap T_{j'}|}$ for any $j' \neq j$ and $P_{j'} \in \mathcal{P} \setminus \mathcal{P}_w$. By Line 8 of $Pricing\text{-}SS$, $p_j \geq \frac{a_{j'}|\mathcal{T}^* \cap T_j|}{|\mathcal{T}^* \cap T_{j'}|}$, where $\mathcal{T}' = \mathcal{T}^*$ in that iteration. Hence $p_j \geq c_j$ and $u_j^{SS} \geq 0$. $\blacksquare$

Lemma 4.2: IMC-SS is budget-balanced. $\square$

Proof: If $\mathcal{P}_w = \emptyset$, $U^{SS} = 0$. Note that, for any $P_j \in \mathcal{P}_w$, $p'_j \geq p_j$ since $\mathcal{P} \setminus \{\mathcal{P}_w \cup \mathcal{P}'_w\} \subseteq \mathcal{P} \setminus \mathcal{P}'_w$. If $\mathcal{P}_w \neq \emptyset$, $U^{SS} = \sum_{i=1}^{m} y_i v_i - \sum_{P_j \in \mathcal{P}_w} p_j \geq \nu - p' \geq 0$, where the last inequality follows from the if-condition in Line 4 of **IMC-SS**. $\blacksquare$

Lemma 4.3: IMC-SS is computationally efficient. $\square$

Proof: Time complexity of **IMC-SS** is upper-bounded by $O(mn^2\lambda)$, where $\lambda = \max_{J_i \in \mathcal{J}} |J_i|$. $\blacksquare$

Lemma 4.4: IMC-SS is truthful. □

Proof: We prove this lemma using Myerson's Theorem [20] In other words, we need to show that the provider selection rule is monotonic and the payment to each winning provider is its critical value. Assume that P_j is selected as a winning provider by reporting a_j. The monotonicity of the selection rule is obvious, as reporting a smaller ask value will make P_j selected as well.

Next, we prove that, for any $P_j \in \mathcal{P}_w$, p_j is its critical value, which means that p_j is the smallest value such that if $a_j > p_j$, P_j loses the auction. If P_j asks $a_j > p_j$, since $p_j \geq \frac{a_{j^*}|\mathcal{T}^* \cap T_j|}{|\mathcal{T}^* \cap T_{j^*}|}$ at each iteration, we have $\frac{a_j}{|\mathcal{T}^* \cap T_j|} > \frac{a_{j^*}}{|\mathcal{T}^* \cap T_{j^*}|}$ at each iteration, where $P_{j^*} = \arg\min_{P_k}\{\frac{a_k}{|\mathcal{T}^* \cap T_k|} | P_k \in \mathcal{P} \setminus \{\mathcal{P}_w \cup \{P_j\}\}$. Therefore, P_j loses the auction. If P_j asks $a_j < p_j$, there will be an iteration in which P_j is added (as P_{j^*} in the algorithm) into $\mathcal{P}_w$. Thus, p_j is the critical value for P_j, and the truthfulness is proved. ■

With Lemmas 4.1 to 4.4, we have Theorem 1 as follows.

Theorem 1: IMC-SS is individually rational, budget-balanced, computationally efficient, and truthful. □

5. Incentive Mechanism for the SM-model

In this section, we present an incentive mechanism for crowdsourcing in the SM-model, named **IMC-SM**. We also prove that **IMC-SM** is individually rational, budget-balanced, computationally efficient, and truthful.

A. Design Rationale

Similar to **IMC-SS**, **IMC-SM** consists of three steps: job selection, winning provider selection, and pricing. The difference lies in the second and third steps. In the SM-model, each provider can submit multiple bids with multiple sets of tasks that it is willing to work on. In the winning provider selection step, we not only select the winning providers, but also determine the specific task set that each winning provider will work on. In the pricing step, we determine the payment for each winning provider. We take a similar strategy as in *Pricing-SS* while deciding the payment for each winning provider.

B. Design Details

Job Selection: This step is the same as that in **IMC-SS**.

Algorithm 6: $WinProviderSel\text{-}SM(\mathcal{J}^*,\ \mathcal{P},\ \mathbf{A})$

```
1 $\delta(j) \leftarrow 0$, for all $P_j \in \mathcal{P}$;
2 $\mathcal{P}_w \leftarrow \emptyset$, $\mathcal{T}^* \leftarrow \biguplus_{J_i \in \mathcal{J}^*} J_i$;
3 while $\mathcal{T}^* \neq \emptyset$ do
4     $T_{j^*}^{k^*} \leftarrow \arg\min_{T_j^k}\{a_j^k | P_j \in \mathcal{P} \setminus \mathcal{P}_w, T_j^k \cap \mathcal{T}^* \neq \emptyset\}$;
5     $\mathcal{P}_w \leftarrow \mathcal{P}_w \cup \{P_{j^*}\}$;
6     $\mathcal{T}^* \leftarrow \mathcal{T}^* \setminus T_{j^*}^{k^*}$;
7     $\delta(j^*) \leftarrow k^*$;
8 end
9 return $\mathcal{P}_w, \delta(\cdot)$;
```

Winning Provider Selection: The selection algorithm proceeds iteratively. In each iteration, we select the task set $T_{j^*}^{k^*}$ with the minimum ask value and take its owner P_{j^*} as a winning provider. Meanwhile, P_{j^*} is assigned to work on $T_{j^*}^{k^*}$ by setting $\delta(j^*)$ to k^*. The details are given in Algorithm 6. Due to Condition 1) discussed in Section 4-B, it is guaranteed that the algorithm can compute a set of winning providers.

Pricing: We compute the critical value for each winning provider as its payment. We temporarily put each winning provider P_γ aside and run a process similar to $WinProviderSel\text{-}SM$ over the rest of providers (Lines 3 to 11). For every selected task set $T_{\bar{j}}^{\bar{k}}$, we compute the smallest value such that if P_γ asks more than this value for working on $T_\gamma^{\delta(\gamma)}$, it would not replace $P_{\bar{j}}$ as the winner in this iteration. Finally, we take the maximum of these values as the payment for P_γ. The detailed algorithm is shown in Algorithm 7.

Algorithm 7: $Pricing\text{-}SM(\mathcal{P},\ \mathcal{P}_w,\ \mathcal{J}^*,\ \mathbf{A}, \delta(\cdot))$

```
1 $p_\gamma \leftarrow 0$ and $p'_\gamma \leftarrow 0$ for all $P_\gamma \in \mathcal{P}$;
2 for all $P_\gamma \in \mathcal{P}_w$ do
3     $\mathcal{P}'_w \leftarrow \emptyset,\ \mathcal{T}^* \leftarrow \biguplus_{J_i \in \mathcal{J}^*} J_i,\ \mathcal{P}_{-\gamma} \leftarrow \mathcal{P} \setminus \{P_\gamma\}$;
4     while $\mathcal{T}^* \neq \emptyset$ do
5         $T_{\bar{j}}^{\bar{k}} \leftarrow \arg\min_{T_j^k}\{a_j^k | T_j^k \cap \mathcal{T}^* \neq \emptyset, P_j \in \mathcal{P}_{-\gamma} \setminus \mathcal{P}'_w\}$;
6         $T_{\hat{j}}^{\hat{k}} \leftarrow \arg\min_{T_j^k}\{a_j^k | T_j^k \cap \mathcal{T}^* \neq \emptyset, P_j \in \mathcal{P}_{-\gamma} \setminus \{\mathcal{P}'_w \cup \mathcal{P}_w\}\}$;
7         if $T_\gamma^{\delta(\gamma)} \cap \mathcal{T}^* \neq \emptyset$ then
8             $p_\gamma \leftarrow \max\{p_\gamma, a_{\bar{j}}^{\bar{k}}\}$, $p'_\gamma \leftarrow \max\{p'_\gamma, a_{\hat{j}}^{\hat{k}}\}$;
9         end
10        $\mathcal{P}'_w \leftarrow \mathcal{P}'_w \cup \{P_{\bar{j}}\}$, $\mathcal{T}^* \leftarrow \mathcal{T}^* \setminus T_{\bar{j}}^{\bar{k}}$;
11    end
12 end
13 $p' \leftarrow \sum_{P_\gamma \in \mathcal{P}_w} p'_\gamma$;
14 return $\mathbf{p}$, $p'$;
```

In the end, same as **IMC-SS**, we check if the auction result satisfies budget-balance. Putting all these steps together, we have the main algorithm of **IMC-SM** as shown in Algorithm 8.

Algorithm 8: IMC-SM$(\mathcal{J}, \mathbf{v}, \mathcal{P}, \mathbf{A})$

```
1 $(\mathcal{J}^*, \nu) \leftarrow JobSel(\mathcal{J}, \mathbf{v}, \mathbf{A})$;
2 $(\mathcal{P}_w, \delta(\cdot)) \leftarrow WinProviderSel\text{-}SM(\mathcal{J}^*, \mathcal{P}, \mathbf{A})$;
3 $(\mathbf{p}, p') \leftarrow Pricing\text{-}SM(\mathcal{P}, \mathcal{P}_w, \mathcal{J}^*, \mathbf{A}, \delta(\cdot))$;
4 if $\nu < p'$ then
5     $\mathcal{P}_w \leftarrow \emptyset$; $\mathbf{p} \leftarrow \mathbf{0}$; $\delta(j) \leftarrow 0$, for all $P_j \in \mathcal{P}$;
6 end
```

C. Analysis

In this section, we prove the desired properties of **IMC-SM**.

Lemma 5.1: IMC-SM is individually rational. □

Proof: Let P_j be any provider that bids truthfully, i.e., $a_j^k = c_j^k$ for $k = 1, \ldots, \sigma_j$. We need to prove that $u_j^{SM} \geq 0$.

If $P_j \notin \mathcal{P}_w$, $u_j^{SM} = 0$. Now assume that $P_j \in \mathcal{P}_w$. Hence $u_j^{SM} = p_j - c_j^{\delta(j)}$. At the iteration where P_j is selected to enter $\mathcal{P}_w$, $a_j^{\delta(j)} = c_j^{\delta(j)}$ is less than or equal to $a_{j'}^{k'}$ for any $j' \neq j$ and k' such that $P_{j'} \in \mathcal{P}\backslash\mathcal{P}_w$ and $T_{j'}^{k'} \cap \mathcal{T}^* \neq \emptyset$. By *Pricing–SM*, $p_j \geq \min\{a_{j'}^{k'} | j' \neq j, P_{j'} \in \mathcal{P} \setminus \mathcal{P}_w, T_{j'}^{k'} \cap \mathcal{T}^* \neq \emptyset\}$. Hence $p_j \geq c_j^{\delta(j)}$. Therefore $u_j^{SM} \geq 0$. ■

Lemma 5.2: IMC-SM is budget-balanced. □

Proof: If the auction fails, $U^{SM} = 0$. If the auction succeeds, $\nu \geq p' \geq \sum_{j=1}^{n} p_j$ guarantees budget-balance. ■

Lemma 5.3: IMC-SM is computationally efficient. □

Proof: The time complexity of **IMC-SM** is upper-bounded by $O(m^2 n^2 \kappa\lambda)$, where $\kappa = \max_{P_j \in \mathcal{P}} \sigma_j$ and $\lambda = \max_{J_i \in \mathcal{J}} |J_i|$. ■

Lemma 5.4: IMC-SM is truthful. □

Proof: We show that for any provider P_j, it cannot have a higher utility by unilaterally changing its bid from $\bar{A}_j = \{(T_j^1, c_j^1), \ldots, (T_j^{\sigma_j}, c_j^{\sigma_j})\}$ to $\hat{A}_j = \{(T_j^1, a_j^1), \ldots, (T_j^{\sigma_j}, a_j^{\sigma_j})\}$ with $(a_j^1, \ldots, a_j^{\sigma_j}) \neq (c_j^1, \ldots, c_j^{\sigma_j})$. We use case analysis.

Case 1: *P_j loses the auction by bidding $\bar{A}_j$*, which implies that at any iteration during the winner selection stage we have $a_{j^*}^{k^*} \leq \min_k\{c_j^k | T_j^k \cap \mathcal{T}^* \neq \emptyset\}$. If by changing the bid, P_j remains a loser, its utility does not increase. Now suppose that P_j unilaterally changes its bid from $\bar{A}_j$ to $\hat{A}_j$, and wins the auction for working on $T_j^{\delta(j)}$ with a payment $\hat{p}_j$. By *Pricing–SM*, we have $\hat{p}_j = \max\{a_{\bar{j}}^{\bar{k}}\} = \max\{a_{j^*}^{k^*}\} \leq c_j^{\delta(j)}$. Thus its resulting utility is $u_j^{SM} = \hat{p}_j - c_j^{\delta(j)} \leq 0$.

Case 2: *P_j wins the auction by bidding $\bar{A}_j$*, earning a payment of p_j to work on $T_j^{\delta(j)}$. Hence at the iteration that P_j is selected to enter $\mathcal{P}_w$, $c_j^{\delta(j)}$ is the smallest ask among all the asks $a_{j'}^{k'}$, such that $T_{j'}^{k'} \cap \mathcal{T}^* \neq \emptyset$. By Lemma 5.1, $u_j^{SM} = p_j - c_j^{\delta(j)} \geq 0$.

Case 2.1: *By changing its bid, P_j becomes a loser, or P_j still works on $T_j^{\delta(j)}$*. If P_j becomes a loser, P_j's utility becomes 0. If P_j still works on $T_j^{\delta(j)}$, by Algorithm 7, the payment p_j does not change, hence P_j's utility does not change either.

Case 2.2: *By changing its bid, P_j wins to work on $T_j^{k'}$* with $k' \neq \delta(j)$, earning a payment $\hat{p}_j$. Since in the pricing stage, the payment for P_j does not depend on the bid of P_j, we have $\hat{p}_j = p_j$. The second statement in Case 2 implies that $c_j^{\delta(j)} \leq c_j^{k'}$. Therefore $\hat{p}_j - c_j^{k'} \leq p_j - c_j^{\delta(j)}$, i.e., the utility for P_j cannot be increased by changing its bid from $\bar{A}_j$ to $\hat{A}_j$ unilaterally. This proves the lemma. ■

By Lemmas 5.1 to 5.4, we have the following theorem.

Theorem 2: IMC-SM is individually rational, budget-balanced, computationally efficient, and truthful. □

6. Incentive Mechanism for the MM-model

In this section, we present an incentive mechanism for crowd-sourcing in the MM-model, named **IMC-MM**. We also prove that **IMC-MM** is individually rational, budget-balanced, computationally efficient, and truthful.

A. Design Rationale

IMC-MM is divided into three steps: winning requester selection, winning provider selection, and pricing. Compared with **IMC-SM**, the difference lies in the first step.

In the MM-model, each job is owned by a requester, who claims a bid as the maximum amount it is willing pay the platform. In winning requester selection, we greedily select a requester, while making sure that the set of jobs we select can be finished without monopoly providers. The winning buyer selection step and pricing step are the same as **IMC-SM**.

B. Design Details

Winning Requester Selection: The selection algorithm proceeds iteratively. In each iteration, we greedily select a job with the maximum bid per task value, under the condition that the set of selected jobs can be completed by the providers without a monopoly provider. After the greedy selection, we end up with a set of jobs. Among the selected jobs, we discard the one with the minimum bid per task, and use its bid per task value as a threshold payment to ensure truthfulness. Algorithm 9 presents the detailed algorithm of this step.

Algorithm 9: *WinRequesterSel-MM*$(\mathcal{R},\ \mathcal{J},\ \mathbf{b},\ \mathbf{A})$

```
1  $\mathcal{J}^* \leftarrow \emptyset$, $\mathcal{R}_w \leftarrow \emptyset$, $\nu \leftarrow 0$;
2  $w_i \leftarrow \frac{b_i}{|J_i|}$ for $i \leftarrow 1$ to $m$;
3  WLOG, assume $w_1 \geq \cdots \geq w_m$ for simplicity;
4  for $i \leftarrow 1$ to $m$ do
5      if $Feasible(\mathcal{P}\backslash\{P_j\}, \mathcal{J}^* \cup \{J_i\}, \mathbf{A})$ for all $P_j \in \mathcal{P}$ then
6          $\mathcal{J}^* \leftarrow \mathcal{J}^* \cup \{J_i\}$;
7      end
8  end
9  $J_{i^*} \leftarrow \arg\min_{J_i \in \mathcal{J}^*} w_i$;
10 $\mathcal{J}^* \leftarrow \mathcal{J}^* \setminus \{J_{i^*}\}$;
11 for all $J_i \in \mathcal{J}^*$ do
12     $\mathcal{R}_w \leftarrow \mathcal{R}_w \cup \{R_i\}$, $q_i \leftarrow w_{i*}|J_i|$, $\nu \leftarrow \nu + q_i$;
13 end
14 return $\mathcal{R}_w, \mathcal{J}^*, \mathbf{q}, \nu$
```

Winning Provider Selection & **Pricing:** These steps are the same as those of **IMC-SM**.

Algorithm 10: IMC–MM$(\mathcal{R},\ \mathcal{J},\ \mathcal{P},\ \mathbf{b},\ \mathbf{A})$

```
1 $(\mathcal{R}_w, \mathcal{J}^*, \mathbf{q}, \nu) \leftarrow WinRequesterSel\text{-}MM(\mathcal{R}, \mathcal{J}, \mathbf{b}, \mathbf{A})$;
2 $(\mathcal{P}_w, \delta(\cdot)) \leftarrow WinProviderSel\text{-}SM(\mathcal{J}^*, \mathcal{P}, \mathbf{A})$
3 $(\mathbf{p}, p') \leftarrow Pricing\text{-}SM(\mathcal{P},\ \mathcal{P}_w, \mathcal{J}^*,\ \mathbf{A}, \delta(\cdot))$;
4 if $\nu < p'$ then
5     $\mathcal{P}_w \leftarrow \emptyset$; $\mathcal{R}_w \leftarrow \emptyset$; $\mathbf{q} \leftarrow \mathbf{0}$; $\mathbf{p} \leftarrow \mathbf{0}$;
6     $\delta(j) \leftarrow 0$, for all $P_j \in \mathcal{P}$;
7 end
```

At the end of the main algorithm of **IMC-MM**, as shown in Algorithm 10, we check the budget-balance constraint.

C. Analysis

In this section, we prove the desired properties of **IMC-MM**.

Lemma 6.1: IMC-MM is individually rational. □

Proof: The individual rationality of providers in **IMC-MM** can be proved similarly to the individual rationality of providers in **IMC-SM** (see Lemma 5.1). It suffices to prove that $\mu_i^{MM} \geq 0$ if requester R_i bids (J_i, v_i).

If R_i loses the auction, we have $\mu_i^{MM} = 0$. Now assume that R_i wins the auction. Since R_i wins the auction by bidding (J_i, v_i), we have (see Algorithm 8) $w_{i^*} \leq \frac{v_i}{|J_i|}$, and R_i will be charged $q_i = w_{i^*}|J_i|$. Hence $\mu_i^{MM} = v_i - w_{i^*}|J_i| \geq 0$. ■

Lemma 6.2: IMC-MM is budget-balanced. □

Proof: It follows from the winning buyer selection algorithm that $\nu = \sum_{i=1}^{m} q_i$. It follows from *Pricing–SM* that $p' \geq \sum_{j=1}^{n} p_j$. Hence we have $U^{MM} = \sum_{i=1}^{m} q_i - \sum_{j=1}^{n} p_j \geq \nu - p' \geq 0$. This proves that **IMC-MM** is budget-balanced. ■

Lemma 6.3: IMC-MM is computationally efficient. □

Proof: The running time of **IMC-MM** is upper-bounded by $O(m^2 n^2 \kappa\lambda)$, where $\kappa = \max_{P_j \in \mathcal{P}} \sigma_j$ and $\lambda = \max_{J_i \in \mathcal{J}} |J_i|$. ■

Lemma 6.4: IMC-MM is truthful. □

Proof: The truthfulness of the providers in **IMC-MM** follows from the truthfulness of the providers in **IMC-SM**.

We prove the truthfulness of requesters by showing that **IMC-MM** satisfies monotonicity and that q_i is the critical value for R_i (see the classical result of Myerson in Section 3-D). If requester R_i wins the auction by bidding (J_i, b_i), we have $\frac{b_i}{|J_i|} \leq w_{i^*}$. When it changes its bid into (J_i, b_i') with $b_i' \geq b_i$, we have $\frac{b_i'}{|J_i|} \geq \frac{b_i}{|J_i|} \geq w_{i^*}$, hence J_i still wins the auction. This shows that **IMC-MM** guarantees monotonicity.

If R_i bids (J_i, b_i') such that $b_i' < q_i = w_{i^*}|J_i|$, then the new bid per task of R_i is $w_i' = \frac{b_i'}{|J_i|} < \frac{q_i}{|J_i|} = w_{i^*}$. Thus, J_i becomes the job with the minimum bidding per task value in $\mathcal{T}^*$, and it is eliminated in Line 10 of Algorithm 9. Therefore, q_i is the critical value for J_i. It follows from Myerson's result [20] that **IMC-MM** is truthful for the requesters. ■

Lemmas 6.1 to 6.4 lead to the following.

Theorem 3: IMC-MM is individually rational, budget-balanced, computationally efficient, and truthful. □

7. Performance Evaluation

For performance evaluation, we have implemented our incentive mechanisms for the SS-model (denoted by **IMC-SS**), the SM-model (denoted by **IMC-SM**), and the MM-model (denoted by **IMC-MM**), and run extensive tests on a Linux PC with Intel Core I7-4770 3.5Hz processor and 16GB memory.

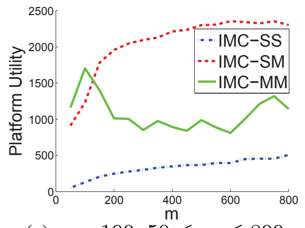
A. Simulation Setup and Performance Metrics

Simulation Setup: We evaluate each metric by varying the number of providers n and the number of jobs m from 50 to 800 with an increment of 50, respectively. To evaluate the impact of n, we fix $m = 100$. Similarly, to evaluate the impact of m, we fix $n = 100$. We set $|\mathcal{T}| = 10$, and generate each job as a random multi-subset of $\mathcal{T}$ with a maximum size of 10. The valuation of each job is uniformly distributed over $(0, 80]$. In **IMC-SS**, each provider can only submit one subset of tasks; in **IMC-SM** and **IMC-MM**, each provider can submit up to 5 random multi-subsets of $\mathcal{T}$. The size of each multi-subset does not exceed 5, and the ask value for each subset is uniformly distributed over $(0, 10]$. The results are averaged over 100 instances for each parameter configuration.

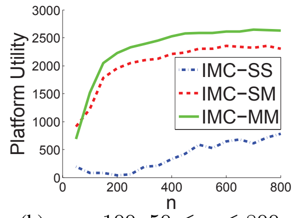
Performance Metrics: We study *platform utility*, *average provider utility*, *running time*, and the *relationship between* $p = \sum_{P_j \in \mathcal{P}_w} p_j$ *and* $p' = \sum_{P_j \in \mathcal{P}_w} p_j'$.

B. Simulation Results

Fig. 2 shows the impact of m and n on platform utility for each of the three models. Here there is no relationship among the values for **IMC-SS**, **IMC-SM**, and **IMC-MM**. We plot them together to save space (this philosophy applies to Figs. 2-4).



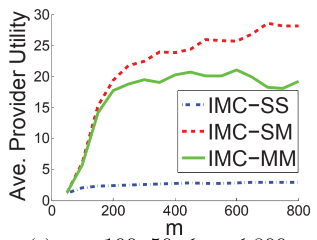
(a) $n = 100$, $50 \leq m \leq 800$



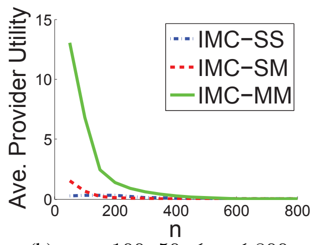
(b) $m = 100$, $50 \leq n \leq 800$

Fig. 2. Platform Utility

In Fig. 2(a), with the increase of m, platform utilities in **IMC-SS** and **IMC-SM** increase as well. This is because providers can work on jobs with higher valuations when m increases. With more providers, more jobs can be finished. That explains why with the increase of n, platform utilities in **IMC-SS**, **IMC-SM**, and **IMC-MM** increase in Fig. 2(b). However, in **IMC-MM**, we observe that the platform utility does not display a well-observed trend with the increase of m. There are complicated reasons for this, since the platform utility in the MM-model depends on the payments to the providers and the charges to the requesters, who are only interested in maximizing their own utilities. We explain this phenomena as follows. At first, with the increase of m, providers can finish more jobs and increase the platform utility. After a certain point, providers could not finish more jobs as m increases. Therefore the total payment from the requesters decreases and remains at a level as there are more jobs with low bids per task values.



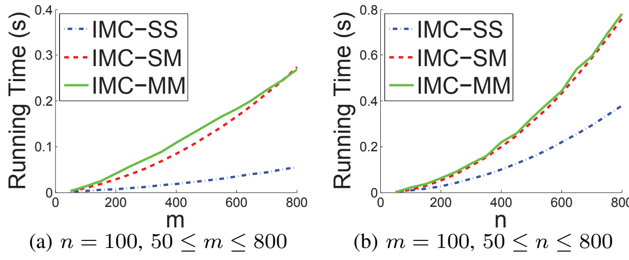
(a) $n = 100$, $50 \leq m \leq 800$



(b) $m = 100$, $50 \leq n \leq 800$

Fig. 3. Average Provider Utility

Fig. 3 shows the average provider utility, defined as the total utilities of all providers over the number of providers. With the increase of m, average provider utility rises and then remains steady. The reason is that each service provider has more opportunities to get a higher payment as there are more high-valued jobs, before the labor market is saturated. With the increase of n, the average provider utility drops dramatically for all three mechanisms. This is because with more providers, the competition among providers becomes more fierce and as a result, the payment for each provider decreases.

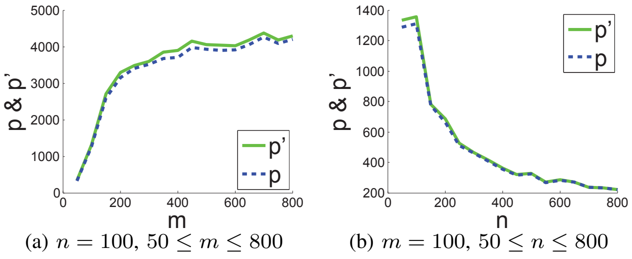


(a) $n = 100$, $50 \le m \le 800$ (b) $m = 100$, $50 \le n \le 800$

Fig. 4. Running Time

Fig. 4 shows the impact of m and n on running time (in seconds) of the three mechanisms. With the increment of m and n, the running time of **IMC-SM** and **IMC-MM** increases with an approximated speed of $O(n^2)$ and $O(m^2)$, respectively. The running time of **IMC-SS** is almost linear in m and an $O(n^2)$ time-consumption in terms of n. These results match our analysis in Lemmas 4.3, 5.3, and 6.3.

In **IMC-SS**, **IMC-SM**, and **IMC-MM**, we compare p' and ν to guarantee budget-balance instead of comparing p and ν. Theoretically, we know that p' is an upper bound of p. How close are the two values? In our tests, these two values are very close in all cases. Due to space limit, we only present the results for the SM-model (the results for the other two models are similar), as shown in Fig. 5.



(a) $n = 100$, $50 \le m \le 800$ (b) $m = 100$, $50 \le n \le 800$

Fig. 5. Values of p and p' in **IMC-SM**

We notice that both p and p' go up with the increase of m, but decline with the increase of n. This pattern is very similar to that of the average provider utility in Fig. 3. Another observation is that the gap between q and q' is relatively small, which means that by comparing ν and p' instead of ν and p, we do not sacrifice too many auctions to guarantee budget-balance.

8. Conclusion

We studied three models of crowdsourcing. Our SS-model contains, as a special case, the user-centric model of [25]. Our SM-model is a generalization of the SS-model. Our MM-model further involves competition among the requesters, in addition to the providers. We presented incentive mechanisms for all three models, and proved that they are individually rational, budget-balanced, computationally efficient, and truthful. Extensive numerical results are presented to verify our analysis.

References

[1] [Online]. Available: http://www.smartcitizen.me/
[2] [Online]. Available: http://healthmap.org/
[3] [Online]. Available: https://www.uber.com/
[4] [Online]. Available: https://www.mturk.com/mturk/welcome
[5] [Online]. Available: http://www.quora.com/
[6] Y. Chon, N. D. Lane, F. Li, H. Cha, and F. Zhao, "Automatically characterizing places with opportunistic crowdsensing using smartphones," in *Proc. ACM UbiComp'12*.
[7] D. DiPalantino and M. Vojnovic, "Crowdsourcing and all-pay auctions," in *Proc. ACM EC'09*.
[8] L. Duan, T. Kubo, K. Sugiyama, J. Huang, T. Hasegawa, and J. Walrand, "Incentive mechanisms for smartphone collaboration in data acquisition and distributed computing," in *Proc. IEEE INFOCOM'12*.
[9] Z. Feng, Y. Zhu, Q. Zhang, L. Ni, and A. Vasilakos, "TRAC: Truthful auction for location-aware collaborative sensing in mobile crowdsourcing," in *Proc. IEEE INFOCOM'14*.
[10] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges," *IEEE Comm. Mag.*, vol. 49, no. 11, pp. 32–39, 2011.
[11] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman Co., 1990.
[12] S. He, D.-H. Shin, J. Zhang, and J. Chen, "Toward optimal allocation of location dependent tasks in crowdsensing," in *Proc. IEEE INFOCOM'14*.
[13] C.-J. Ho and J. W. Vaughan, "Online task assignment in crowdsourcing markets," in *Proc. AAAI'12*.
[14] J. Howe, *Crowdsourcing: Why the Power of the Crowd Is Driving the Future of Business*. Crown Publishing Group, 2008.
[15] I. Koutsopoulos, "Optimal incentive-driven design of participatory sensing systems." in *Proc. IEEE INFOCOM'13*.
[16] V. Krishna, *Auction Theory*. Academic Press, 2002.
[17] Q. Li and G. Cao, "Providing privacy-aware incentives for mobile sensing," in *Proc. IEEE PerCom'13*.
[18] X.-Y. Li, Z. Sun, and W. Wang, "Cost sharing and strategyproof mechanisms for set cover games," in *Proc. STACS 2005*.
[19] W. Mason and D. J. Watts, "Financial incentives and the 'performance of crowds'," in *Proc. ACM HCOMP'09*.
[20] R. Myerson, "Optimal auction design," *Mathematics of Operations Research*, 1981.
[21] M.-R. Ra, B. Liu, T. F. La Porta, and R. Govindan, "Medusa: A programming framework for crowd-sensing applications," in *Proc. MobiSys'12*.
[22] Y. Singer and M. Mittal, "Pricing mechanisms for online labor markets," in *Proc. AAAI HCOMP'11*.
[23] A. Singla and A. Krause, "Truthful incentives in crowdsourcing tasks using regret minimization mechanisms," in *Proc. WWW '13*.
[24] W. Xu, H. Huang, Y. Sun, F. Li, Y. Zhu, and S. Zhang, "DATA: A double auction based task assignment mechanism in crowdsourcing systems," in *Proc. IEEE CHINACOM'13*.
[25] D. Yang, G. Xue, X. Fang, and J. Tang, "Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing," in *Proc. ACM MobiCom'12*.
[26] X. Zhang, Z. Yang, Z. Zhou, H. Cai, L. Chen, and X. Li, "Free market of crowdsourcing: Incentive mechanism design for mobile sensing," *IEEE TPDS*, vol. PP, no. 99, pp. 1–1, 2014.
[27] Y. Zhang and M. van der Schaar, "Reputation-based incentive protocols in crowdsourcing applications." in *Proc. IEEE INFOCOM'12*.
[28] D. Zhao, X.-Y. Li, and H. Ma, "How to crowdsource tasks truthfully without sacrificing utility: Online incentive mechanisms with budget constraint," in *Proc. IEEE INFOCOM'14*.