Incentivizing the Biased Requesters: Truthful Task Assignment Mechanisms in Crowdsourcing

Jia Xu[1,2], Hui Li[1,2], Yanxu Li[1,2], Dejun Yang[3] and Tao Li[1,2]
[1]School of Computer, Nanjing University of Posts and Telecommunications, Nanjing, Jiangsu 210023 China
[2]Jiangsu Key Laboratory of Big Data Security and Intelligent Processing, Nanjing, Jiangsu 210023 China
[3]Department of Computer Science, Colorado School of Mines, Golden, CO 80401 USA
Email: xujia@njupt.edu.cn, lihuimx2016@163.com, halyx@126.com, djyang@mines.edu, towerlee@njupt.edu.cn

***Abstract*—Crowdsourcing has become an effective tool to utilize human intelligence to perform tasks that are challenging for machines. In the integrated crowdsourcing systems, the requesters are non-monopolistic and may show preferences over the workers. We are the first to design the incentive mechanisms, which consider the issue of stimulating the biased requesters in the competing crowdsourcing market. In this paper, we explore truthful task assignment mechanisms to maximize the total value of accomplished tasks for this new scenario. We present three models of crowdsourcing, which take the preferences of the requesters and the workload constraints of the workers into consideration. We design a task assignment mechanism, which follows the matching approach to solve the *Valuation Maximizing Assignment* (*VMA*) problem for each of the three models. Through both rigorous theoretical analyses and extensive simulations, we demonstrate that the proposed assignment mechanisms achieve computational efficiency, workload feasibility, preference (universal) truthfulness and constant approximation.**

I. Introduction

Crowdsourcing represents a problem-solving model that elicits solutions, ideas, data, etc. from an undefined or generally large group of people in the form of an open call [1]. Wikipedia [2], Semantic Web [17], Freebase [12], and other knowledge repositories were created by volunteers who contributed knowledge about a wide variety of topics. In recent years, crowdsourcing has been widely used in many fields, including video analysis [14], knowledge discovery from web tables [9] and Smart Citizen [3], conducting human-robot interaction studies, e.g., RMS [18], image quality assessment [8], online marketplace, e.g., Amazon Mechanical Turk (AMT) [4]. With the rapid proliferation of smartphones, mobile crowdsourcing has become an efficient approach to meeting the demands in large scale sensing applications.

Incentive mechanisms are crucial to crowdsourcing while the workers may have associated cost for performing tasks. The incentive, which compensates workers' cost, is also helpful to achieve good crowdsourcing service quality. A lot of research efforts [11,20,21,22] have been focused on developing such incentive mechanisms to entice workers to participate in crowdsourcing. However, existing mechanisms mainly consider the scenarios where all crowdsourcing tasks belong to a monopolistic requester, and there is no preference over the workers. In this paper, we consider the crowdsourcing system with the following two distinctive characteristics, which are different from most existing crowdsourcing systems.

- Competing requesters

Most of existing mechanisms ignored the situation where multiple requesters coexist and compete for limited available crowdsourcing capacity or qualified workers. In addition to AMT, some integrated crowdsourcing platforms such as Medusa [13] and gPS [7] have been designed in recent years. In such integrated crowdsourcing platforms, a variety of crowdsourcing applications are available and different requesters can publicize multiple crowdsourcing campaigns concurrently [16]. Crowdsourcing with competing requesters is a common scenario in these crowdsourcing systems since most of existing crowdsourcing systems provide no or little reward to the workers, leading to the shortage of workers. Another reason of bringing about the competition among the requesters is that multiple requesters would compete for the professional workers. For example, an important proportion of Human Intelligence Tasks (HITs) in AMT require the qualified workers, who have passed the corresponding tests.

- Biased requesters

The requesters usually have preferences over the workers in practical crowdsourcing systems. In other words, the requesters only require the workers, who conform with their preference perform the tasks rather than all workers. There are many crowdsourcing scenarios with biased requesters: The requesters intend to hire native Hungarian speakers to order Hungarian to English translations; the advisors wish to allocate the research projects to the students whose interests and skills can match the projects; The tasks in mobile crowdsourcing are usually location dependent and expected to be assigned to the workers who are close by the specific locations.

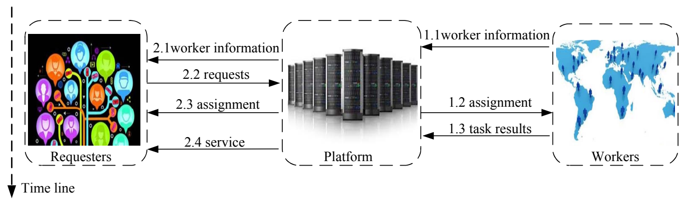


Fig.1. General crowdsourcing process with biased requesters

Unfortunately, there is no off-the-shelf task assigment mechanism designed in the literature for the crowdsourcing system with competing and biased requesters. In this paper, we aim at designing task assignment mechanisms to maximize the total value of accomplished tasks, and stimulating each requester to submit its preference truthfully for this new scenario. In our system model, the workers submit their information to the crowdsourcing platform. Then the platform desensitizes the worker information and publicizes it to all requesters. As a response, each requester submits a request to the platform. The request consists of a task and a preference set, which is a set of compatible workers of the requester. Specifically, if there is no preference over the workers, the requester can simply submit the preference set including all workers. The platform calculates the assignment between the tasks and workers, and notifies requesters and workers of the decision. The workers perform the assigned tasks and send the results back to the platform. Finally, the platform provides the service to the requesters. The crowdsourcing process is illustrated by Fig.1.

The problem of designing truthful assignment mechanisms to maximize the total value for such crowdsourcing system is very challenging. First, the requesters have the preference over the workers. This means that we should consider not only the value of the allocation but also the compatibility of the allocation. Second, a requester can take a strategic action by submitting a dishonest preference set to maximize his own value. Moreover, the workers are associated with workload constraints, and the aggregate workload of the assigned tasks to each worker should be within its workload constraint.

The main contributions of this paper are as follows:

- To the best of knowledge, this is the first work to design truthful assignment mechanisms for the crowdsourcing systems with competing and biased requesters.
- We formulate the *Valuation Maximizing Assignment* (*VMA*) problem and consider three different models: Identical workload Identical value (II-Model), Identical workload Non-identical value (IN-Model), and Non-identical workload Nonidentical value (NN-Model).
- We design an assignment mechanism for each of these models to solve the *VMA* problem. We show that the designed mechanisms satisfy four desirable properties: computational efficiency, workload feasibility, preference (universal) truthfulness, and constant approximation.

The rest of the paper is organized as follows. Section II formulates three system models for crowdsourcing and lists some desirable properties. Section III to Section V present the detailed design of our assignment mechanisms for the three models. Performance evaluation is presented in Section VI. We review the state-of-art research in Section VII, and conclude this paper in Section VIII.

II. SYSTEM MODEL AND DESIRABLE PROPERTIES

A. II-Model: Identical workload Identical value

We consider a crowdsourcing system consisting of a platform, a set of requesters R=$\{1, 2,..., n\}$ and a set of workers W=$\{1, 2,..., m\}$.The platform publicizes the worker set W to all requesters. We assume the worker information is processed by data desensitization such as removing all identifiable columns straightforwardly beforehand. Each requester i submits a request B_i=(t_i, P_i), where t_i is the task requester i wishes to be performed by the workers on the crowdsourcing system. We assume t_i is a small task, which can be completed by single worker. The small tasks abound in existing crowdsourcing systems such as labeling the product or recognizing specific values displayed in an image, translations for an article, noise level collection and gathering air pollution readings in a specific location. Each task t_i is associated with the preference set $Q_i \subseteq W$, which is a set of compatible workers of requester i. We consider that Q_i is the private information and known only to requester i. $P_i \subseteq W$ is the claimed preference set of requester i. The value of task t_i to requester i is defined as v_i^k, $i \in R$, $k \in W$. We assume all v_i^k are identical in this model.

We denote $B = \{B_1, B_2, ..., B_n\}$ as the set of all requests. The assignment mechanism $\mathcal{M}(W, B)$ outputs an assignment $\mathcal{A}$ between the tasks and workers, where each task can be assigned to at most one worker and each worker can perform at most one task. To simplify the notation, we reuse the notation i and R to denote the task submitted by requester i and the task set submitted by all requesters, respectively. Given the assignment $\mathcal{A}$, let $\mathcal{A}_i$ be the worker that task i is assigned to and $\mathcal{A}^k = \{i|\mathcal{A}_i = k\}$ be the set of tasks that are assigned to the worker k. Let $v(\mathcal{A}) = \sum_{(i,k)\in\mathcal{A}} v_i^k$ be the total value of the assignment, $v_i(\mathcal{A}) = v_i^{\mathcal{A}_i}$ be the value of task i in the assignment $\mathcal{A}$, and $v(\mathcal{A}^k) = \sum_{(i\in\mathcal{A}^k)} v_i^k$ be the total value of tasks which are assigned to the worker k, respectively.

Since we consider the requesters are selfish individuals, each requester i can behave strategically by submitting a dishonest preference set to maximize his own value $v_i(\mathcal{A})$, $i \in R$. We assume that the truthfulness of submitted task can be achieved since it can be verified by the platform. However, P_i can be different from the real preference set Q_i.

The assignment mechanism $\mathcal{M}(W, B)$ is run on the platform with the objective of maximizing the total valuation. The *VMA* problem in the II-Model can be formulated as follows:

$$max_{\mathcal{A}} v(\mathcal{A}) = \sum_{k\in W} |\mathcal{A}^k| = \sum_{i\in R} |\mathcal{A}_i|$$

$$\begin{aligned} \textit{s.t.}\ & (1)|\mathcal{A}^k| \le 1, \forall k \in W \\ & (2)|\mathcal{A}_i| \le 1, \forall i \in R \\ & (3)\mathcal{A} \in \{(i, k)|i \in R, k \in P_i\} \end{aligned}$$

B. IN-Model: Identical workload Non-identical value

We relax the constraints of identical value in this model by allowing v_i^k to be non-identical for $\forall i \in R$, $\forall k \in W$. So IN-Model is a generalization of II-Model. The definitions of R, W, t_i, P_i, Q_i, v_i^k are the same as those in Section II-A. For each worker $k \in W$, there is an effort indicator $I_k \ge 0$, which can be obtained by evaluating the quality of the historic crowdsourcing tasks performed by k. Many metrics or methods can be used to estimate the quality of performed tasks, such as completion rate and approval rate for AMT crowdsourcing tasks [4], the EM algorithm [5] for estimating mobile crowdsourcing data [15], etc.

Crowdsourcing in the IN-Model works as follows: The platform publicizes a two-tuple (W, I) to all requesters, where $I = (I_1, I_2, ..., I_m)$. Each requester i submits a request $B_i = (t_i, a_i, P_i)$, where a_i is the type of task t_i.The type of the tasks can be selected from the task type set predefined by the crowdsourcing system, such as transcription, content generation, content rewriting, object classification, website feedback, etc. We assume that the truthfulness of submitted task and the workload of task can be achieved since they can be verified by the platform. However, P_i also can be different from the real preference set Q_i of performing task i.

There is a difficulty coefficient D_i for each task t_i. We assume that D_i only depends on the type of task a_i straightforwardly, i.e., $D_i = \mathcal{F}(a_i)$, where $\mathcal{F}()$ is a mapping function from a_i to D_i. The value of task t_i to requester i is defined as $v_i^k = D_i * I_k, i \in R, k \in W$.

We denote $B = \{B_1, B_2, ..., B_n\}$ as the set of all requests. The assignment mechanism $\mathcal{M}(W, I, B)$ outputs an assignment $\mathcal{A}$ between the tasks and workers, where each task can be assigned to at most one worker and each worker can perform at most one task. The *VMA* problem in the IN-Model can be formulated as follows:

$$max_{\mathcal{A}} v(\mathcal{A}) = \sum_{(i,k)\in\mathcal{A}} v_i^k$$

$$\begin{aligned} s.t.\ & (1)|\mathcal{A}^k| \leq 1, \forall k \in W \\ & (2)|\mathcal{A}_i| \leq 1, \forall i \in R \\ & (3)\mathcal{A} \in \{(i,k)|i \in R, k \in P_i\} \end{aligned}$$

C. NN-Model: Non-identical workload Non-identical value

We further relax the constraints of identical workload in this model. The definitions of R, W, t_i, P_i, Q_i, v_i^k are the same as those in Section II-A, and the definitions of I, a_i are the same as those in Section II-B. We consider that each worker $k \in W$ submits a workload constraint $C_k > 0$, which is the maximum available work time it can spend.

The platform publicizes a triple (W, C, I) to all requesters, where $C = (C_1, C_2, ..., C_m)$. Each requester i submits a request $B_i = (t_i, c_i, a_i, P_i)$, where $c_i > 0$ is the workload of task t_i. We define short hand notation for $c()$ similarly to all $v()$ in Section II-A. We assume that the truthfulness of submitted task, workload and type of task can be achieved since they can be verified by the platform. However, P_i also can be different from the real preference set Q_i of performing task i in the NN-Model.

We denote $B = \{B_1, B_2, ..., B_n\}$ as the set of all requests. The assignment mechanism $\mathcal{M}(W, C, I, B)$ outputs an assignment $\mathcal{A}$ between the tasks and workers, where each task can be assigned to at most one worker and the aggregate workload of the assigned tasks to each worker is within the worker's workload constraint. The *VMA* problem in the NN-Model can be formulated as follows:

$$max_{\mathcal{A}} v(\mathcal{A}) = \sum_{(i,k)\in\mathcal{A}} v_i^k$$

$$\begin{aligned} s.t.\ & (1) \sum_{i\in\mathcal{A}^k} c_i \leq C_k, \forall k \in W \\ & (2)|\mathcal{A}_i| \leq 1, \forall i \in R \\ & (3)\mathcal{A} \in \{(i,k)|i \in R, k \in P_i\} \end{aligned}$$

Remark: Although II-Model and IN-Model are special cases of NN-Model, we will show that the corresponding truthful task assignment mechanisms can achieve better approximation ratio than that of NN-Model.

D. Desirable Properties for Assignment Mechanisms

Our objective is to design the assignment mechanisms satisfying the following four desirable properties:

- **Computational efficiency:** An assignment mechanism $\mathcal{M}$ is computationally efficient if the assignment $\mathcal{A}$ can be computed in polynomial time.
- **Workload feasibility:** An assignment mechanism $\mathcal{M}$ is workload feasible if the aggregate workload of the assigned tasks to each worker is within the worker's workload constraint in the assignment $\mathcal{A}$.
- **Preference truthfulness:** An assignment mechanism is preference truthful if no requester can improve its valuation by submitting a preference set different from its real preference set, no matter what others submit. In other words, reporting the real preference set is a dominant strategy for all requesters. Specifically, we say the assignment mechanism is preference universally truthful if it takes a random distribution over deterministic preference truthful assignment mechanisms.
- **Constant approximation:** We attempt to find the approximation algorithms with constant approximation ratio, i.e., the ratio between approximation solution and the optimal solution is within a constant.

The importance of the first two properties is obvious, because they together assure the feasibility of the assignment mechanism. The last two properties are indispensable for guaranteeing the compatibility and high performance. Being truthful, the assignment mechanisms can eliminate the fear of market manipulation and the overhead of strategizing over others for the requesters.

III. TASK ASSIGNMENT MECHANISM FOR THE II-MODEL

A. Mechanism Design

In this section, we present a truthful task assignment mechanism for crowdsourcing in the II-Model, named *TAM-II*. The *VMA* problem in the II-Model is equivalent to the *MBM (Maximum Bipartite Matching)* problem [19] on the bigraph *G(R, W, E)*, where $E = \{(i,k)|\forall i \in R, k \in P_i\}$. It is well known that the *MBM* problem can be solved via *FFA (Ford Fulkerson Algorithm)* [6] with running time $O(|E|(|R| + |W|))$. Unfortunately, given the bigraph, there may be more than one optimal solution to the *MBM* problem, which leads to untruthfulness of the assignment mechanism due to the inconsistent tiebreaking rule. To address this problem, we define a consistent tiebreaking rule in order to make mechanism outputs the unique assignment between all optimal solutions.

In *TAM-II*, the pairs in E are placed in the sequence $\mathcal{H}$ according to the lexicographic order. We first compute the size of maximum matching through function *MBM( )*. Then we process each pair$(i,k) \in E$ in the order they appear in the sequence $\mathcal{H}$ iteratively. In each iteration, we check whether removing (i,k) decreases the size of the maximum matching by solving the problem on $E \backslash \{(i,k)\}$. If so we keep (i,k) in E, else we discard (i,k) by setting $E = E \backslash \{(i,k)\}$. When finished, E is the unique maximum matching. The whole process is illustrated in Algorithm 1.

B. Mechanism Analysis

In the following, we present the theoretical analysis, demonstrating that *TAM-II* can achieve the desired properties.

Lemma 1. *TAM-II is computationally efficient, workload feasible and optimal for VMA problem in the II-Model.*

Proof: The running time of *TAM-II* is dominated by the for-loop (Line 5-10), which takes $O((\sum_{i\in R}|P_i|)^2(|R|+|W|))$. Since $|P_i| \leq m$, for $\forall i \in R$, the running time of *TAM-II* is bounded by $O(n^2m^2(n+m))$. The workload feasibility is obvious since we assign at most one task to each worker. Obviously, *TAM-II* can output an optimal solution since we use *FFA* to solve the *MBM* problem, which is equivalent to the *VMA* problem in the II-Model. ■

Algorithm 1 :TAM-II

Input: Worker Set *W*, Request Set *B*

1: $\mathcal{A} \leftarrow \emptyset$;
2: Represent all pairs (i,k), $i \in R$, $k \in P_i$ as $(1,1),(1,2),...,(2,1),(2,2),...,(n,m)$ and the sequence is denoted by $\mathcal{H}$;
3: $\mathcal{H}' \leftarrow \mathcal{H}$;
4: $N \leftarrow MBM(\mathcal{H})$;
5: **for all** $j \in \mathcal{H}$ in order **do**
6: $\quad N' \leftarrow MBM(\mathcal{H}' \backslash \{j\})$;
7: $\quad$**if** $N' \geq N$ **then**
8: $\quad\quad$Remove j from $\mathcal{H}'$;
9: $\quad$**end if**
10: **end for**
11: $\mathcal{A} \leftarrow \mathcal{H}'$;
12: return($\mathcal{A}$);

Lemma 2. *TAM-II is preference truthful.*

Proof: Considering an arbitrary requester $i \in R$, let $E_i = \{(i,k)|k \in Q_i\}$, $E_i' = \{(i,k)|k \in P_i\}$ and $E_{-i} = \{(j,k)|\forall j \in R, j \neq i, k \in Q_j\}$. Assume $\mathcal{A}$ and $\mathcal{A}'$ are the assignments obtained from *TAM-II* on (E_i, E_{-i}) and (E_i', E_{-i}), respectively. We assume to the contrary that *TAM-II* is not preference truthful, i.e., $v_i(\mathcal{A}') > v_i(\mathcal{A})$. Since the task i can only be assigned to at most one worker, i will be matched by $e \in \mathcal{A}'$ and doesn't be matched by $\mathcal{A}$ at all, $e \in E_i \cap E_i'$. Now we run *TAM-II* on$((E_i \cap E_i'), E_{-i})$, and denote the output as $\mathcal{A}''$. There must be $v_i(\mathcal{A}'') \nless v_i(\mathcal{A}')$ and $e \in \mathcal{A}''$ since the pairs in $E_i' \backslash (E_i \cap E_i')$ cannot improve the value to the requester i in fact. Note that $e \notin \mathcal{A}$ and $E_i \cap E_i' \subseteq E_i$, we have $e \notin \mathcal{A}''$ because *TAM-II* applies consistent tie breaking rule to find the unique maximum matching, this yields a contradiction. ■

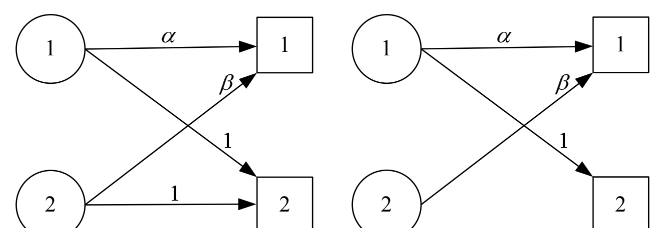
The above two lemmas prove the following theorem.

Theorem 1. *TAM-II is computationally efficient, workload feasible, preference truthful and optimal for VMA problem in the II-Model.*

IV. TASK ASSIGNMENT MECHANISM FOR THE IN-MODEL

A. Mechanism Design

The *VMA* problem in the IN-Model is equivalent to the *MWBM (Maximum Weighted Bipartite Matching)* problem [19] on the bigraph *G(R,W,E)*, where $E = \{(i,k)|\forall i \in R, k \in P_i\}$ with value $v_i^k = D_i * I_k$. It is known that the *Hungarian Method* [10] can obtain the optimal solution of the *MWBM* problem within polynomial time. Unfortunately, the optimal solution of the *VMA* problem in the IN-Model is not preference truthful. We use an example in Fig.2 to show the statement. In this example, $R = \{1,2\}, W = \{1,2\}, Q_1 = \{1,2\}, v_1^1 = \alpha, v_1^2 = 1, Q_2 = \{1,2\}, v_2^1 = \beta, v_2^2 = 1$. We assume that $\beta + 1 > \alpha > \beta > 1$. We first consider the case where both requesters submit their preference sets truthfully. Since $\alpha > \beta$, the optimal solution in this case is $\{(1,1),(2,2)\}$, and the value to requester 2 is 1. We now consider the case where requester 2 lies by submitting $P_2 = \{1\}$. Since $\beta + 1 > \alpha$, the optimal solution in this case is $\{(1,2),(2,1)\}$, and the value to requester 2 is β. Note that requester 2 increases its value from 1 to β by lying about its preference set.



(a) Both requesters submit their preference sets truthfully (b) Requester 2 lies by submitting $P_2 = \{1\}$

Fig.2. An example showing the untruthfulness of the optimal solution of the *VMA* problem in the IN-Model, where the disks represent requesters, the squares represent workers and the arrows represent the preference sets. The numbers above the arrows represent the value to the requesters. We assume that $\beta + 1 > \alpha > \beta > 1$.

Although the *Hungarian Method* can obtain the optimal solution of our *VMA* problem in the IN-Model, the failure of guaranteeing preference truthfulness makes it less attractive. To address this issue, we design a 2-approximate deterministic truthful task assignment mechanism, named *TAM-IN*, which follows a greedy approach. The basic idea is that each requester is assigned based on the value-first strategy, i.e., the requester is assigned to the worker in its preference set with largest possible value. By this way, claiming $P_i \subseteq Q_i$ would not increase the value to requester i.

Illustrated in Algorithm 2, *TAM-IN* calculates the value for each pair in *E*, and sorts the pairs based on their value in nonincreasing order. Then *TAM-IN* selects the pair j with largest value into the assignment if $\mathcal{A}\cup\{j\}$ is still a matching. The process repeated until all pairs in *E* have been considered.

Algorithm 2 :TAM-IN

```
Input: Worker Set W, Request Set B, Effort Indicators I
1:  for all i ∈ R do
2:      for all k ∈ P_i do
3:          v_i^k ← F(a_i) * I_k;
4:      end for
5:  end for
6:  Sort all pairs (i,k), i ∈ R, k ∈ P_i based on v_i^k in
    nonincreasing order and the sequence is denoted by J;
7:  A ← ∅;
8:  for all j ∈ J in order do
9:      if A ∪ {j} is a matching on G(R,W,J) then
10:         A ← A ∪ {j};
11:     end if
12: end for
13: return(A)
```

B. Mechanism Analysis

We present the theoretical analysis, demonstrating that *TAM-IN* can achieve the desired properties.

Lemma 3. *TAM-IN is computationally efficient and workload feasible.*

Proof: Calculating the value for all pairs (Line 1-5) takes $O(nm)$ time. Sorting the pairs (Line 6) takes $O(nm\cdot log(nm))$ time. Since there are at most $min(n,m)$ pairs in the assignment $\mathcal{A}$, the for-loop (Line 8-12) takes $O(nm\cdot min(n,m))$. Hence the running time of *TAM-IN* is bounded by $O(nm\cdot max(log(nm),min(n,m)))$. The workload is feasibile since *TAM-IN* assigns at most one task to each worker. ■

Lemma 4. *TAM-IN is preference truthful.*

Proof: Considering an arbitrary requester $i\in R$, let $E_i=\{(i,k)|k\in Q_i\}, E_i^{'}=\{(i,k)|k\in P_i\}$ and $E_{-i}=\{(j,k)|\forall j\in R, j\neq i, k\in Q_j\}$. Assume $\mathcal{A}$ and $\mathcal{A}^{'}$ are the assignments obtained from *TAM-IN* on (E_i,E_{-i}) and $(E_i^{'},E_{-i})$, respectively. We assume to the contrary that *TAM-IN* is not preference truthful, i.e., $v_i(\mathcal{A}^{'})>v_i(\mathcal{A})$. In such case, task i must be matched by $e\in A^{'}$. If $e\in E_i^{'}\backslash E_i$, there is no value to requester i. $\mathcal{A}$ contradiction arises since $v_i(\mathcal{A})\geq v_i(\mathcal{A}^{'})$. If $e\in E_i^{'}\cap E_i$, we have $e\in E_i$. Note that *TAM-IN* assigns the requester to the worker in its preference set with largest possible value, requester i must be matched by some $e^{'}\in E_i$ in $\mathcal{A}$, which makes $v_i(\mathcal{A})\geq v_i(\mathcal{A}^{'})$. A contradiction arises. ■

Lemma 5. *TAM-IN can approximate the optimal solution within a factor of 2.*

Proof: We denote $\mathcal{A}_{opt}$ as the optimal assignment of our *VMA* problem in the IN-Model. Without loss of generality, consider that $(i,k), i\in R, k\in P_i$ is a matching in the assignment $\mathcal{A}$, then any matching $(j,k),\forall j\in R, j\neq i$ would not appear in $\mathcal{A}$ since the worker k has been matched by the requester i. On the other hand, for the $\mathcal{A}_{opt}$, considering $(i,k)\notin\mathcal{A}_{opt}$, then the worker k is matched at most one pair $(j,k),\forall j\in R, j\neq i$. This means the impact of requester $i's$ matching on the whole assignment is at most one other requester j. Hence we can observe the difference between $\mathcal{A}$ and $\mathcal{A}_{opt}$ via the generalized 2-requester assignment problem, which is illustrated in Fig.3. In the generalized 2-requester assignment problem, $R=\{1,2\}, W=\{1,2\}, P_1=\{1,2\}, v_1^1=\alpha, v_1^2=\chi, P_2=\{1,2\}, v_2^1=\beta, v_2^2=\delta$. We assume that $\alpha>\beta,\alpha>\chi,\alpha>\delta$ and $\beta+\chi>\alpha+\delta$. It is easy to see that $\mathcal{A}=\{(1,1),(2,2)\}$ with $v(\mathcal{A})=\alpha+\delta$ and $\mathcal{A}_{opt}=\{(1,2),(2,1)\}$ with $v(\mathcal{A}_{opt})=\beta+\chi$.The upper bound of the approximation ratio 2 can be obtained if we let β and χ be sufficiently close to α, and let δ be sufficiently close to 0. ■

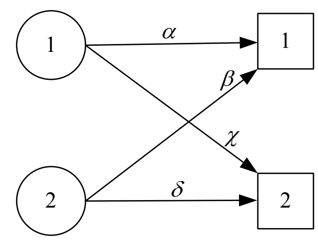


Fig.3. An 2-requester assignment, where the disks represent requesters, the squares represent workers and the arrows represent the preference sets. The numbers above the arrows represent the value to the requesters. We assume that $\alpha>\beta,\alpha>\chi,\alpha>\delta$ and $\beta+\chi>\alpha+\delta$.

The above three lemmas prove the following theorem.

Theorem 2. *TAM-IN is computationally efficient, workload feasible, preference truthful and 2-approximate in the IN-Model.*

V. TASK ASSIGNMENT MECHANISM FOR THE NN-MODEL

A. Mechanism Design

First of all, we attempt to find an optimal algorithm for the *VMA* problem in the NN-Model. Unfortunately, the following theorem shows that it is NP-hard to find the optimal solution.

Theorem 3. *The VMA problem in the NN-Model is NP-hard.*

Proof: We consider the *MKP* (*Multiple Knapsack Problem*): there are a set of n items R and a set of m knapsacks W. Each knapsack $k\in W$ has a capacity C_k, and each item $i\in R$ has a size c_i and a value v_i. The objective is finding a subset $L\subseteq R$ of maximum value such that L has a feasible packing in W. We can see that the *VMA* problem in the NN-Model is a generalization of the *MKP* where each item $i\in R$ can only be assigned to the knapsacks within its preference set and v_i is a function of the knapsack. Since the *MKP* is NP-hard, the *VMA* problem in the NN-Model is NP-hard. ■

Since the *VMA* problem in the NN-Model is NP-hard, we turn our attention to developing an approximation algorithm in user selection phase. We first give two greedy assignment mechanisms, named *GREEDY-VALUE* and *GREEDY-DENSITY*, respectively. We denote $\mathcal{G}$ and $\widetilde{C_k}$ as the set of compatible task-worker pairs and the remaining workload of worker k, respectively. *GREEDY-VALUE* selects the task-worker pair $(i^{'}, k^{'})$ with the maximum value from $\mathcal{G}$ iteratively. If $\mathcal{A} \cup (i^{'}, k^{'})$ is workload feasible, the allocation is added to the assignment $\mathcal{A}$, and the task-worker pairs including $i^{'}$ are removed from $\mathcal{G}$ since each task can be assigned to at most one worker, otherwise, $(i^{'}, k^{'})$ is removed from $\mathcal{G}$. It is easy to see that *GREEDY-VALUE* decides the allocation according to the nonincreasing order of value, i.e., the task of requester i is allocated to the worker who is compatible and can bring the maximum value to requester i if the allocation is workload feasible. The process is illustrated in Algorithm 3. *GREEDY-DENSITY* works similarly with *GREEDY-VALUE*. While in *GREEDY-DENSITY*, the allocation is decided according to the nonincreasing order of the value density: the ratio of the value to the workload, $\frac{v_i^k}{c_i}, (i,k) \in \mathcal{G}$.

Algorithm 3 :GREEDY-VALUE

Input: Worker Set *W*, Request Set *B*, Effort Indicators $\mathbf{I}$, Workload Constraints C

for all $i \in R$ **do**
  for all $k \in P_i$ **do**
    $v_i^k \leftarrow \mathcal{F}(a_i) * I_k$;
    $\widetilde{C_k} \leftarrow C_k$;
  end for
end for
$\mathcal{A} \leftarrow \emptyset$;
$\mathcal{G} \leftarrow \{(i,k) | \forall i \in R, k \in P_i\}$;
while $\mathcal{G} \neq \emptyset$ **do**
  $(i^{'}, k^{'}) \leftarrow argmax_{(i,k) \in \mathcal{G}} v_i^k$;
  if $\widetilde{C_k^{'}} - c_{i'} \geq 0$ **then**
    $\widetilde{C_k^{'}} \leftarrow \widetilde{C_k^{'}} - c_{i'}$;
    $\mathcal{A} \leftarrow \mathcal{A} \cup \{(i^{'}, k^{'})\}$;
    $\mathcal{G} \leftarrow \mathcal{G} \backslash \{(i^{'}, k^{''}) | k^{''} \in P_{i'}\}$;
  else
    $\mathcal{G} \leftarrow \mathcal{G} \backslash \{(i^{'}, k^{'})\}$;
  end if
end while
return($\mathcal{A}$);

Although the above two greedy based assignment mechanisms can obtain the feasible solutions of our *VMA* problem in the NN-Model, they can work arbitrarily badly.

Lemma 6. *There is no upper bound of the approximation ratio for either GREEDY-VALUE or GREEDY-DENSITY.*

Proof: We denote $\mathcal{A}_{opt}$, $\mathcal{A}_{value}$, $\mathcal{A}_{density}$ as the optimal solution, solution obtained from *GREEDY-VALUE* and *GREEDY-DENSITY*, respectively. We first show *GREEDY-VALUE* works arbitrarily badly for the example illustrated in Fig.4: There are many identical tasks with value $\alpha - \varepsilon$ and workload ε each. In addition, there is a task with value 1 and workload 1. There is only one worker with workload constraint 1. All tasks tend to be allocated to the worker. We assume that $\alpha > 1, \alpha - \varepsilon < 1, \varepsilon \in (0, \frac{1}{2}]$. It is easy to see that $v(\mathcal{A}_{value}) = 1$ since $1 > \varepsilon$. The optimal solution would assign $\lfloor \frac{1}{\varepsilon} \rfloor$ tasks with value $\alpha - \varepsilon$ each to the worker and obtain $v(\mathcal{A}_{opt}) = \lfloor \frac{1}{\varepsilon} \rfloor (\alpha - \varepsilon)$. If we let ε be sufficiently close to 0, the approximation ratio of *GREEDY-VALUE* tends to infinite. Then we show *GREEDY-DENSITY* works arbitrarily badly for the example illustrated in Fig.5: There is a task with value $\alpha\varepsilon$ and workload ε. In addition, there is another task with value 1 and workload 1. There is only one worker with workload constraint 1. Both tasks tend to be allocated to the worker. We assume that $\alpha > 1, \alpha\varepsilon < 1, \varepsilon \in (0, 1)$. It is easy to see that $v(\mathcal{A}_{density}) = \alpha\varepsilon$ since $\alpha > 1$, and $v(\mathcal{A}_{opt}) = 1$ since $\alpha\varepsilon < 1$. If we let ε be sufficiently close to 0, the approximation ratio of *GREEDY-DENSITY* tends to infinite. ■

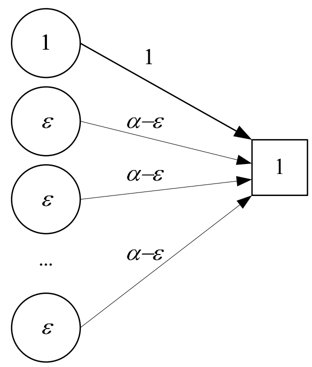


Fig.4. Example shows that GREEDY-VALUE works arbitrarily badly. The disks represent tasks, the square represents worker and the arrows represent the preference sets. The numbers above the arrows represent the value of the tasks. The numbers in the disks represent the workload of the tasks. The number in the square represents the workload constraint of the worker. We assume that $\alpha > 1, \alpha - \varepsilon < 1, \varepsilon \in (0, \frac{1}{2}]$.

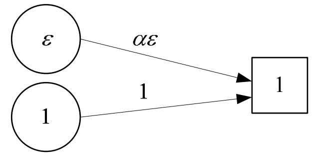


Fig.5. Example shows that GREEDY-DENSITY works arbitrarily badly. The disks represent tasks, the square represents worker and the arrows represent the preference sets. The numbers above the arrows represent the value of the tasks. The numbers in the disks represent the workload of the tasks. The number in the square represents the workload constraint of the worker. We assume that $\alpha > 1, \alpha\varepsilon < 1, \varepsilon \in (0, 1)$.

Although *GREEDY- VALUE* and *GREEDY- DENSITY* can obtain the feasible solutions, the failure of bounding the approximation ratio makes them less attractive. We present a random mechanism, named *TAM-NN*, which runs either *GREEDY-VALUE* or *GREEDY- DENSITY* with equal probability. The process is illustrated in Algorithm 4.

Algorithm 4 :TAM-NN

Input:Worker Set W, Request Set B, Effort Indicators $\mathbf{I}$, Workload Constraints $\mathbf{C}$

```
Generate a random number o from the uniform distribution on the interval [0,1];
A ← ∅;
if o ≤ 1/2 then
    A ← GREEDY − VALUE(W, B, I, C);
else
    A ← GREEDY − DENSITY(W, B, I, C);
end if
return(A);
```

B. Mechanism Analysis

We present the theoretical analysis, demonstrating that *TAM-NN* can achieve the desired properties.

Lemma 7. *TAM-NN is computationally efficient and workload feasible.*

Proof: Since *GREEDY-VALUE* and *GREEDY-DENSITY* have the same time complexity. It suffices to show that *GREEDY-VALUE* is computationally efficient. The running time of *GREEDY-VALUE* is dominated by the while-loop (Line 9-18), which runs at most nm times since there are at most nm task-worker pairs in $\mathcal{G}$. In the while-loop, finding the task-worker pair $(i^{'}, k^{'})$ with the maximum value from $\mathcal{G}$ takes $O(nm)$ time, and removing the task-worker pairs including $i^{'}$ from $\mathcal{G}$ takes $O(nm)$ time. Hence the running time of *TAM-NN* is bounded by $O(n^2m^2)$. The workload is feasible since the workload feasibility of every allocation is checked both in *GREEDY- VALUE* and *GREEDY- DENSITY*. ■

Lemma 8. *TAM-NN is preference universally truthful.*

Proof: It is well known that a nondeterministic mechanism is universally truthful if it takes a random distribution over deterministic truthful mechanisms. Since *TAM-NN* is a nondeterministic mechanism, it suffices to show that both *GREEDY-VALUE* and *GREEDY-DENSITY* are preference truthful. We consider that there is an arbitrary requester $i \in R, Q_i \neq P_i$. Requester i is allocated by $k^{'} \in P_i$ when i reports P_i and is allocated by $k \in Q_i$ when i reports Q_i. If $k^{'} \in P_i \backslash Q_i$, there is no value to requester i, and reporting $P_i \neq Q_i$ cannot improve the value to the requester. Otherwise, if $k^{'} \in Q_i$, there must be $v_i^k \geq v_i^{k^{'}}$ since *GREEDY-VALUE* decides the allocation according to the nonincreasing order of the value. Hence *GREEDY-VALUE* is preference truthful. The proof for the preference truthfulness of *GREEDY-DENSITY* is similar with that of *GREEDY-VALUE* since the workload of the task only relies on the type of the task itself. ■

Lemma 9. *The approximation ratio of TAM-NN is 4.*

Proof: We denote $\mathcal{V}, \mathcal{D}$ as the assignments got from *GREEDY-VALUE* and *GREEDY-DENSITY* of our *VMA* problem in the NN-Model, respectively, and denote $\mathcal{O}$ as the optimal assignment. The expected value of *TAM-NN* is given by $\frac{1}{2}v(\mathcal{V}) + \frac{1}{2}v(\mathcal{D})$. We want to show that

$$4(\frac{1}{2}v(\mathcal{V}) + \frac{1}{2}v(\mathcal{D})) = 2v(\mathcal{V}) + 2v(\mathcal{D}) \geq v(\mathcal{O})$$

We denote S as the set of tasks i for which $v_i(\mathcal{O}) > v_i(\mathcal{V})$ and $v_i(\mathcal{O}) > v_i(\mathcal{D})$. This means there must be $v_i(\mathcal{V}) > v_i(\mathcal{O})$ or $v_i(\mathcal{D}) > v_i(\mathcal{O})$ for each $i \notin S$. Then we get

$$\sum_{i\in R} v_i(\mathcal{V}) + \sum_{i\in R} v_i(\mathcal{D}) \geq \sum_{i\notin S} v_i(\mathcal{V}) + \sum_{i\notin S} v_i(\mathcal{D}) \geq \sum_{i\notin S} v_i(\mathcal{O}) \tag{1}$$

Further, we have

$$v(\mathcal{O}) = \sum_{i\notin S} v_i(\mathcal{O}) + \sum_{i\in S} v_i(\mathcal{O}) = \sum_{i\notin S} v_i(\mathcal{O}) + \sum_{k\in W} v(\mathcal{O}^k \cap S) \tag{2}$$

Now consider any task $i_0 \in S$, let $\mathcal{O}_{i_0} = k$. By the definition of S, we know that task i_0 prefers worker k to its assignments in $\mathcal{V}$ and $\mathcal{D}$. Therefore, by the greedy rule of $\mathcal{V}$, we have

$$v(\mathcal{V}^k) \geq v_i^k = v_i(\mathcal{O}), \forall i \in O^k \cap S \tag{3}$$

By the value density greedy rule of $\mathcal{D}$, for each $i \in \mathcal{O}^k \cap S$, we have $\frac{v_i(\mathcal{O})}{c_i(\mathcal{O})} \leq \frac{v_{i^{'}}(\mathcal{D})}{c_{i^{'}}(\mathcal{D})}, \forall i^{'} \in \mathcal{D}^k$ Further, we have

$$\frac{\sum_{i\in \mathcal{O}^k\cap S} v_i(\mathcal{O})}{\sum_{i\in \mathcal{O}^k\cap S} c_i(\mathcal{O})} \leq \frac{\sum_{i^{'}\in \mathcal{D}^k} v_{i^{'}}(\mathcal{D})}{\sum_{i^{'}\in \mathcal{D}^k} c_{i^{'}}(\mathcal{D})} = \frac{v(\mathcal{D}^k)}{c(\mathcal{D}^k)}$$

For the considered task $i_0 \in \mathcal{O}^k \cap S$, we can write

$$v((\mathcal{O}^k \cap S)\backslash\{i_0\}) = \sum_{i\in \mathcal{O}^k\cap S, i\neq i_0} v_i(\mathcal{O}) \leq c((\mathcal{O}^k \cap S)\backslash\{i_0\}) \frac{v(\mathcal{D}^k)}{c(\mathcal{D}^k)} \tag{4}$$

For any $i \in \mathcal{O}^k \cap S$, since i doesn't be allocated to k in $\mathcal{D}$, we have $C_k < c(\mathcal{D}^k) + c_i(\mathcal{O})$. Thus we have

$$c((\mathcal{O}^k \cap S)\backslash\{i_0\}) \leq C_k - c_{i_0}(\mathcal{O}) < c(\mathcal{D}^k) \tag{5}$$

Applying formula (5) and (4), we have

$$v((\mathcal{O}^k \cap S)\backslash\{i_0\}) \leq c(\mathcal{D}^k)\frac{v(\mathcal{D}^k)}{c(\mathcal{D}^k)} = v(\mathcal{D}^k) \tag{6}$$

Combining formula (3) and (6), we have
$v(\mathcal{V}^k) + v(\mathcal{D}^k) \geq v((\mathcal{O}^k \cap S)\backslash\{i_0\}) + v_{i_0}(\mathcal{O}) = v(\mathcal{O}^k \cap S)$
Hence

$$\sum_{k\in W} v(V^k) + \sum_{k\in W} v(\mathcal{D}^k) \geq \sum_{k\in W} v(\mathcal{O}^k \cap S) \tag{7}$$

Combining formula (1), (2) and (7), we have

$$2v(\mathcal{V}) + 2v(\mathcal{D}) = \sum_{i\in R} v_i(\mathcal{V}) + \sum_{i\in R} v_i(\mathcal{D}) + \sum_{k\in W} v(\mathcal{V}^k)$$
$$+ \sum_{k\in W} v(\mathcal{D}^k) \geq \sum_{i\notin S} v_i(\mathcal{O}) + \sum_{k\in W} v(\mathcal{O}^k \cap \mathcal{S}) = v(\mathcal{O}).$$

We derive the lemma. ■

The above three lemmas prove the following theorem.

Theorem 4. *TAM-NN is computationally efficient, workload feasible, preference universally truthful and 4-approximate in the NN-Model.*

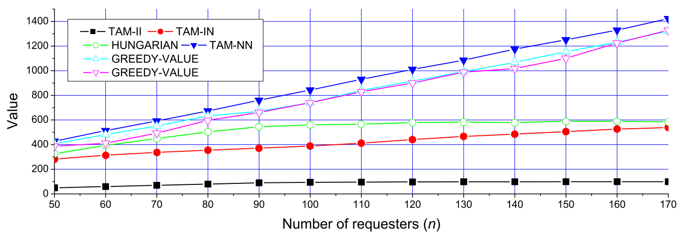
VI. PERFORMANCE EVALUATION

We have conducted thorough simulations to investigate the performance of *TAM-II*, *TAM-IN* and *TAM-NN*. We measure the value and running time with different number of requesters, number of workers and size of preference set. We also implement Hungarian [10] for IN-Model, *GREEDY-VALUE* and *GREEDY-DENSITY* for NN-Model, respectively. All the simulations were run on a Ubuntu 14.04.3 LTS machine with Intel Xeon CPU E5-2420 and 16 GB memory. Each measurement is averaged over 1000 instances.

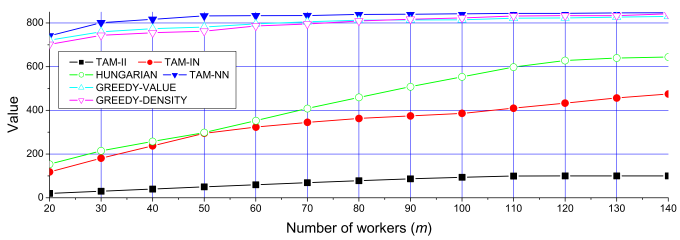
A. Simulation Setup

For our simulations, the workers in preference set are selected randomly from W and all requesters have the same size of preference set. We set $n = m = 100$, $|P_i| = 3, i \in R$ as the default setting, however we will vary them for exploring the impacts of these parameters respectively. For IN-Model and NN-Model, we set $\mathcal{F}(a_i) = a_i, i \in R$, which is uniformly distributed over [1,3], and the effort indicator $I_k, k \in W$ is uniformly distributed over [1,5]. For NN-Model, c_i is uniformly distributed over [1,10] and C_k is uniformly distributed over [1,100] for all $i \in R, k \in W$.

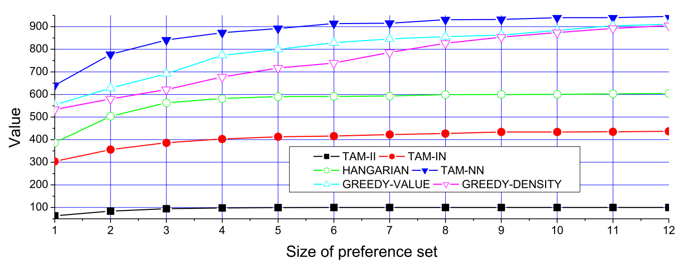
B. Value



(a) The value versus the number of requesters



(b) The value versus the number of workers

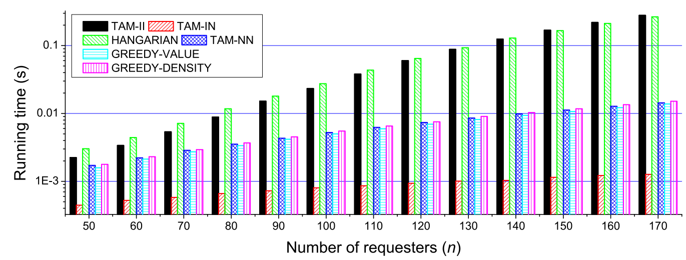


(c) The value versus the size of preference set

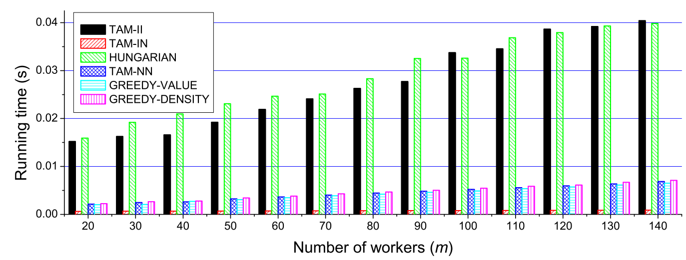
Fig.6. The value with different number of requesters, number of workers and size of preference set.

We first vary the number of requesters from 50 to 170. As shown in Fig.6(a), the value of all mechanisms increased with increasing number of requesters. In II-Model and IN-Model, the value becomes stable when the number of requesters grows to 100. In II-Model, the value of *TAM-II* approaches to 100, which is the maximum value can be obtained since we fix $m = 100$. However, the mechanisms in NN-Model don't show the convergence since each worker can perform more than one task. The average approximation ratio of *TAM-IN* is 1.27, which verifies that *TAM-IN* is within a factor of 2 of optimal. In all cases, the value of *TAM-NN* is more than those of *GREEDY-VALUE* and *GREEDY-DENSITY*. On average, *TAM-NN* can output 8.45% and 12.38% more value than *GREEDY-VALUE* and *GREEDY-DENSITY*, respectively.

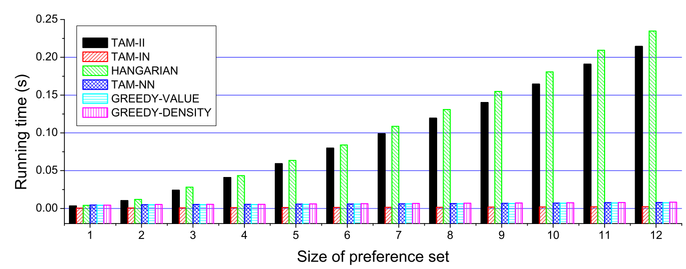
Fig.6(b) shows that the value of all mechanisms increases with increasing the number of workers. The average approximation ratio of *TAM-IN* is 1.28. Moreover, the improvement of *TAM-NN* with respect to the value is small. On average, *TAM-NN* only output 3.69% and 4.24% more value than *GREEDY-VALUE* and *GREEDY-DENSITY*, respectively. This is because each worker can perform 10 tasks on average based on the default setting. In this setting, increasing the workers cannot improve the value significantly.



(a) The running time versus the number of requesters



(b) The running time versus the number of workers



(c) The running time versus the size of preference set

Fig.7. The running time with different number of requesters, number of workers and size of preference set.

Then we vary the size of preference set from 1 to 12 further. As can be shown in Fig.6(c), when the size of preference set goes up, the value increases because large size of preference set can lessen the competition and reduce the conflict among the requesters. The value becomes stable for all mechanisms since there are upper bounds of value in theory. The average approximation ratio of *TAM-IN* is 1.40. On average, *TAM-NN* can output 14.89% and 18.17% more value than *GREEDY-VALUE* and *GREEDY-DENSITY*, respectively.

C. Running time

To investigate the computational efficiency of designed mechanisms, we vary the number of requesters, the number of workers and the size of preference set. Fig.7 shows the impact of these parameters on the running time of mechanisms. The running time of all mechanism increases with the increasing requester scale, worker scale and preference set size. Among our three truthful assignment mechanisms, *TAM-IN* runs fastest and *TAM-II* takes the longest time. The numeral results verify our theoretical analysis in Lemma1, Lemma 3 and Lemma 7. Note that *TAM-IN* only takes averagely 9.52% of time required by *HANGARIAN* in all cases.

VII. RELATED WORK

A. Crowdsourcing systems

The Hybrid Machine-Crowdsourcing System [14] exploits crowdsourcing for web table schema matching and adopts a hybrid machine-crowdsourcing based approach for a general solution to the matching problem. Toris et al. presented the Robot Management System (RMS) [3], a novel framework for bringing robotic experiments to the web, and described an initial trial of the RMS as a means of conducting user studies. Ghadiyaram *et al*. designed and implemented an online crowdsourcing system [18], which is used to conduct a large-scale, on-going, multi-month image quality assessment (IQA) subjective study, wherein a wide range of diverse observers record their judgments of image quality. Although there are many applications and systems on crowdsourcing, most of them are based on voluntary participation.

B. Incentive mechanisms

Yang *et al*. proposed two different models for smartphone crowdsourcing [21]: the platform-centric model where the platform provides a reward shared by participating users, and the user-centric model where users have more control over the payment they will receive. Xu *et al*. proposed truthful incentive mechanisms [20] for the mobile crowdsourcing system where the tasks are time window dependent, and the platform has strong requirement of data integrity. Furthermore, they studied the budget feasible mechanisms for the same mobile crowdsourcing system [11]. However, none of the above works considered the case where there are multiple requesters. Zhang [22] *et al*. proposed IMC, which consider the competition among the requesters in crowdsourcing. It simplifies the model by assuming that the value of tasks doesn't depend on the workers. In [16], Chen *et al*. studied mechanisms in a two-sided heterogeneous mobile crowdsourcing market with multiple requesters and users. However, they do not consider the compatibility between requesters and workers.

VIII. CONCLUSION

In this paper, we have investigated the task assignment incentive mechanisms for the crowdsourcing system with competing and biased requesters. We have studied three models of crowdsourcing and formulated the *VMA* problem for each model. We presented the task assignment mechanisms for all three models, and proved that they are computationally efficient, workload feasible, preference (universally) truthful and constant approximate. Extensive numerical results are presented to verify our theoretical analysis.

ACKNOWLEDGMENT

This work has been supported in part by the NSFC (No. 61472193, 61472192, 61373139, 61502251, 91646116), NSF (No. 1444059, 1461886) and NSF of Jiangsu Province (No.BK20141429, BK20151511).

REFERENCES

[1] [Online]. http://www.cbsnews.com/news/what-is-crowdsourcing.
[2] [Online]. wikipedia (n.d.). wikipedia: The free encyclopedia. On the internet. http://www.wikipedia.org.
[3] [Online]. Available: http://www.smartcitizen.me.
[4] [Online]. https://www.mturk.com/mturk/welcome.
[5] Dawid A P, Skene A M. "Maximum likelihood estimation of observer error-rates using the EM algorithm". *Applied statistics*, 1979:20-28.
[6] Ford L R, Fulkerson D R. "Maximal Flow Through a Network". *Classic Papers in Combinatorics*. Birkhuser Boston, 2009:243-248.
[7] Wu F J, Luo T. "A generic participatory sensing framework for multi-modal datasets". in *Proc. IEEE ISSNIP*. 2014.
[8] Ghadiyaram D, Bovik A C. "Crowdsourced study of subjective image quality". in *Proc. IEEE Asilomar Conference on Signals, Systems, and Computers*. 2015.
[9] Fan J, Lu M, Ooi B C, et al. "A hybrid machine-crowdsourcing system for matching web tables". in *Proc. IEEE ICDE*. 2014.
[10] Munkres J. "Algorithms for the assignment and transportation problems". *Journal of the Society for Industrial & Applied Mathematics*, 1957, 5(1):32-38.
[11] Xu J, Xiang J, Li Y. "Incentivize maximum continuous time interval coverage under budget constraint in mobile crowd sensing". *Wireless Networks*, 2016:1-14.
[12] Bollacker K, Tufts P, Pierce T, et al. "A Platform for Scalable, Collaborative, Structured Information Integration". *Iiweb*, 2007.
[13] Ra M R, Liu B, La Porta T F, et al. "Medusa: A programming framework for crowd-sensing applications". in *Proc. ACM Mobisys*. 2012.
[14] Tahboub K, Gadgil N, Delp E J. "An intelligent crowdsourcing system for forensic analysis of surveillance video". in *Proc. SPIE The International Society for Optical Engineering*, 2015.
[15] Peng D, Wu F, Chen G. "Pay as How Well You Do: A Quality Based Incentive Mechanism for Crowdsensing". in *Proc ACM MobiHoc*. 2015.
[16] Chen S, Liu M, Chen X. "A truthful double auction for two-sided heterogeneous mobile crowdsensing markets". *Computer Communications*, 2016:31-42.
[17] Berners-Lee T, Hendler J, Lassila O, et al. "The Semantic Web: A New Form of Web Content That is Meaningful to Computers Will Unleash a Revolution of New Possibilities". *Scientific American*, 2001:34-43.
[18] Toris R, Kent D, Chernova S. "The Robot Management System: A Framework for Conducting Human-Robot Interaction Studies Through Crowdsourcing". *Journal of Human-Robot Interaction*, 2014:25-49.
[19] Douglas W. B, "Introduction to Graph Theory (2nd ed.)", Prentice Hall, Chapter 3, 1999.
[20] Xu J, Xiang J, et al. "Incentive Mechanisms for Time Window Dependent Tasks in Mobile Crowdsensing". *IEEE TWC*, 2015 :6353-6364.
[21] Yang D, Xue G, et al. "Crowdsourcing to smartphones: incentive mechanism design for mobile phone sensing". in *Proc. ACM MobiCom*, 2012.
[22] Zhang X, Xue G, Yu R, et al. "Truthful incentive mechanisms for crowdsourcing". in *Proc. IEEE INFOCOM*, 2015.