Optimal Complex Task Assignment in Service Crowdsourcing

Feilong Tang
Department of Computer Science and Engineering, Shanghai Jiao Tong University, China
tang_feilong@sjtu.edu.cn

Abstract

Existing schemes cannot assign complex tasks to the most suitable workers because they either cannot measure skills quantitatively or do not consider assigning tasks to workers who are the most suitable but unavailable temporarily. In this paper, we investigate how to realize optimal complex task assignment. Firstly, we formulate the multiple-skill-based task assignment problem in service crowdsourcing. We then propose a *weighted multi-skill tree (WMST)* to model multiple skills and their correlations. Next, we propose the *acceptance expectation* to uniformly measure the probabilities that different categories of workers will accept and complete specified tasks. Finally, we propose an *acceptance-expectation-based task assignment* (*AE-TA*) algorithm, which reserves tasks for the most suitable workers even unavailable temporarily. Comprehensive experimental results demonstrate that our WMST model and AE-TA algorithm significantly outperform related proposals.

1 Introduction

Service crowdsourcing provides a convenient platform to collaboratively perform complex tasks (*i.e., macrotasks*) [Haas et al., 2015], such as literature translation, document editing and product design, besides traditional simple tasks like labeling [Liu *et al.*, 2018]. These macrotasks usually require multiple skills [Alon *et al.*, 2015] and have specified deadlines. Specialized translation, for example, requires not only ability in at least two languages but also domain knowledge.

The most desirable goal of macrotask assignment is to "*assign tasks to the most suitable workers*" [Zheng *et al.*, 2015; Qiu *et al.*, 2016]. However, it is very challenging in multi-worker and multi-task service crowdsourcing. Essentially, there are two key challenges. The first is how to *precisely measure multiple skills* of workers because different workers usually possess different skills and diverse degree of proficiency in the same skill [Xu *et al.*, 2018]. The second is how to *assign tasks to the most suitable workers* in the case that states of workers are uncertain, i.e., some of more suitable workers may be temporarily unavailable at the time of task assignment. In realistic scenarios, although the most suitable workers often are not available during task assignment, they potentially can complete tasks before tasks' deadlines.

Some researchers have proposed skill models [Roy et al., 2015; Salek *et al., 2013*]. However, these models have not solved the above first challenge because they cannot reflect correlations among multiple skills. The existing hierarchical skill models also cannot quantitatively capture multiple skills. On the other hand, few existing proposals considered workers who are more suitable for specified tasks but unavailable currently. In [Boutsis and Kalogeraki, 2013; Boutsis and Kalogeraki, 2014], for example, the authors only estimated whether the currently available workers can complete tasks. In summary, from either precise skill estimation or optimal task assignment point of view, existing work cannot achieve to "assign tasks to the most suitable workers".

In this paper, we propose a multi-skill-based assignment approach for complex macrotasks in service crowdsourcing, which addresses the above both challenges simultaneously. Firstly, we propose a *Weighted Multi-Skill Tree (WMST)* to model multiple skills of workers and correlations among these skills, and define the *match quality* to capture how well works fit for tasks quantitatively. To assign tasks to the most suitable workers, we divide workers into three categories: *available*, *busy* and *offline workers*, and then quantitatively estimate their *acceptance-and-completion expectation*. Based on this expectation, we propose an adaptive algorithm that dynamically decides how many tasks should be reserved for workers more suitable but temporarily unavailable.

We focus on real macrotask-oriented service crowdsourcing systems with the following features. Firstly, these macrotasks, e.g., translating a book on Java programming, have professional skill requirements and should be finished before given deadlines. Secondly, workers have professional skills and their workloads are relatively stable even predictable. Obviously, senior software engineers can provide higher translation quality than taxi drivers. So, if workers with more related skills but unavailable temporarily have enough time to complete tasks, it is more desirable to assign such tasks to them. Our main contributions can be summarized as follows.

- We formally define the *MS-TA* (*Multi-Skill-based Task Assignment)* problem in service crowdsourcing systems, with the objective of maximizing global task assignment quality. We then prove that this problem is NP-hard.
- We propose the multi-skill model *WMST (Weighted*

Multi-Skill Tree) that can quantitatively capture multiple skills and correlations among them.

- We propose a new *acceptance expectation* model and design an *Acceptance-Expectation-based Task Assignment (AE-TA)* algorithm. Experimental results demonstrate the high performance of our MS-TA and AE-TA.

2 Related Work

2.1 Online Task Assignment

Push-based task assignment currently is predominant methods. In [Ho and Vaughan, 2012], the authors proposed a two-phase exploitation assignment algorithm. [Ho et al., 2013] proposed a near-optimal algorithm. In [Goel, Nikzad, and Singla, 2014], the authors modeled the task-to-worker assignment as a bipartite graph matching problem and proposed a mechanism for maximizing the utility. In [Assadi, Hsu, and Jabbari, 2015], the authors extended the problem by considering the online aspect. To increase the efficiency of task completion, [Yuen, King, and Leung, 2011] proposed a matching mechanism that keeps workers working over the long run. [Chandra *et al.*, 2015] proposed an interaction model in which workers are strategic about service cost.

2.2 Knowledge-Intensive Skill Models

Most expert crowdsourcing platforms are both time-sensitive and skill-related [Tang, 2019]. [Desmarais and Baker, 2012] proposed a review of techniques for learner skill assessment. Few work on crowdsourcing has discussed workers' skills in details. Most of them used keyword vectors as a simple representation of the skills [Fan *et al.*, 2015; Cheng *et al.*, 2016]. As an alternative, [El Maarry *et al.*, 2014] proposed an ontology-based skill model. [Mavridis *et al., 2016*] proposed a skill model with a hierarchical structure, but this model can only cope with tasks related to single skill and cannot reflect proficiency degree.

3 System Model and Problem Statement

3.1 System Model

Let the service crowdsourcing system run in a set of equal timeslots $Q = \{q_k|1 \le k \le N_Q\}$. There are a set of workers $W = \{w_i|1 \le i \le N_W\}$ and a set of tasks $T = \{t_j|1 \le j \le N_T\}$, where N_W and N_T are the number of workers and tasks entering the system during N_Q timeslots, respectively.

Each worker $w_i \in W$ possesses a set of attributes $\{S_{w_i}, V_{w_i}\}$, where S_{w_i} is the set of skills of w_i and $V_{w_i} = < v_{w_i}^{(q_1)}, ..., v_{w_i}^{(q_{N_Q})} >$ is a vector of states. $v_{w_i}^{(q_k)}$ is the state of w_i in q_k such that $v_{w_i}^{(q_k)}$=1 if w_i is online in q_k; otherwise, $v_{w_i}^{(q_k)}$=0. A worker is allowed to accept multiple tasks simultaneously. We use $L_{w_i}^{(q_k)}$ to represent the list of current tasks that w_i has accepted but has not yet completed in q_k. Each task $t_j \in T$ is associated with an attribute set $\{b_{t_j}, d_{t_j}, g_{t_j}, R_{t_j}\}$, where b_{t_j} and d_{t_j} respectively are the starting time and deadline of t_j with $q_1 \le b_{t_j} < d_{t_j} \le q_{N_Q}$; g_{t_j} is the number of workers required for t_j; and R_{t_j} denotes the set of skills required for t_j. Note that g_{t_j} and R_{t_j} are predetermined by task requesters. $e(w_i, t_j)$ refers to the execution time of t_j with w_i. Finally, $||X||$ denotes the size of a set X.

3.2 Problem Formulation

Let $a_{w_i,t_j}^{(q_k)}$ denote a match that t_j is assigned to w_i in q_k. We propose a general model *match quality* $m(a_{w_i,t_j}^{(q_k)})$ to measure the degree of task matching, which will be discussed in the next section. The *global assignment* $A = \{a_{w_i,t_j}^{(q_k)} | \forall q_k \in Q, \forall w_i \in W, \forall t_j \in T\}$ specifies the set of matches in all timeslots [Tang *et al.*, 2019]. Accordingly, the *global assignment quality* $\Psi(A)$ can be defined as

$$\Psi(A) = \sum_{\forall a_{w_i,t_j}^{(q_k)} \in A} m(a_{w_i,t_j}^{(q_k)}) \quad (1)$$

We now formulate the *MS-TA* problem as follows

$$max \quad \Psi(A) \quad (2)$$

$$s.t. \quad v_{w_i}^{(q_k)} = 1, \forall a_{w_i,t_j}^{(q_k)} \in A \quad (3)$$

$$(a_{w_i,t_j}^{(q_{k1})} \in A) \wedge (a_{w_i,t_j}^{(q_{k2})} \in A) = false, q_{k1} \neq q_{k2} \quad (4)$$

$$||\{a_{w,t}^{(q)} \in A | t = t_j, w \in W, q \in Q\}|| \le n_{t_j}, \forall t_j \in T \quad (5)$$

$$q_k + e(w_i, t_j) \le d_{t_j}, \forall a_{w_i,t_j}^{(q_k)} \in A \quad (6)$$

(2) defines the optimization objective. Constraint (3) ensures that any w_i can be assigned tasks only in timeslots q_k when he is available. (4) ensures that a worker cannot be assigned tasks that he has already worked on, and (5) ensures that the number of workers assigned to any task cannot exceed the number n_{t_j} required for that task. Finally, (6) guarantees that when a task is assigned to a worker, he will have sufficient time to complete it.

Theorem 1. *The MS-TA problem is NP-hard.*

Proof. We prove the above theorem by reducing the multiple knapsack problem to the MS-TA problem. The former is a more complex version of the 0-1 knapsack problem, and both problems have been proofed to be NP-hard [Magazine and Chern, 1984]. The multiple knapsack problem can be described as follows. Given x items, each item has an associated value v_p and a weight c_p such that $p<x$. Let m knapsacks have a set of capacities C_q such that $q \le m$. The problem is to put items into those knapsacks such that the total value among all knapsacks is maximized without exceeding any knapsack's capacity.

The above multiple knapsack problem is just a special case of our MS-TA problem. Specifically, for any task, the execution times and match qualities for all workers are assumed to be equal; i.e., $m(a_{w_i,t_j}^{(q_k)})=v_j$ and $e(w_i, t_j)=c_j$ for any $w_i \in W$, $t_j \in T$ and $q_k \in Q$. Let workers be available at consecutive times; we denote the length of available time by C_i. Moreover, let $b_{t_j}=q_1$, $d_{t_j}=q_{N_Q}$, and $g_{t_j}=1$ for all $t_j \in T$, meaning that all tasks can be assigned only once. We regard the workers as analogous to knapsacks and the tasks as analogous to items, and we wish to find an assignment that maximizes the total match quality $\sum_{\forall(w,t)\in W \times T} m(a_{w_i,t_j}^{(q_k)})$ subject to $\sum_{t_j \in T} e(w_i, t_j) \le C_i$ for every $w_i \in W$.

The above reduction demonstrates that the MS-TA problem is more complex than the multiple knapsack problem. Therefore, the MS-TA is a NP-hard problem. □

4 Multiple Skill Model

4.1 Weighted Multi-Skill Tree (WMST)

Hierarchical structure can effectively describe multiple skills of a worker [Mavridis *et al., 2016*]. We define the *weighted multi-skill tree WMST* $S=\{s, [S[1], S[2], ..., S[k]]\}$ to model not only multiple skills but also their correlations of a worker, where s is a root node and $S[1], S[2], ..., S[k]$ are sub-trees at lower levels. Each node $s \in S$ corresponds to a key-value pair $<key(s), d(s)>$, where $key(s)$ is a skill and $d(s)$ is the weight of that skill. Let $dep(s)$ be the depth of a node s in the WMST. We use $fth^{(n)}(s)$ $(0 \leq n \leq dep(s))$ to denote a up-level node that is n levels higher than s. We set $fth^{(0)}(s)=s$. Finally, $chd(s)$ denotes the set of child nodes of s.

Our WMST can model multiple skills for complex tasks. Here, a leaf node denotes an atomic skill (e.g., Java language skill), and a middle node captures a composite skill (e.g., coding skill). Each worker $w_i \in W$ has an associated WMST S_{w_i}.

Weights represent the proficiency of skills of a worker. The weight of any node except a leaf node is the sum of weights of its child nodes. Let $d_{w_i}(s)$ denote the weight of s for w_i. We have

$$d_{w_i}(s) = \sum_{\forall s' \in chd(s)} d_{w_i}(s') \tag{7}$$

4.2 Match Quality

Workers with required skills are more suitable for a given task [Roy et al., 2015]. However, other related skills also contribute to the task execution [Choi, Yoo, and Lee, 2018].

Complexities of tasks and skill proficiency of workers are unobserved random variables. Instead, observed variables are responses from workers. We employ the Rasch model [Rasch, 1993] to estimate initial skill in leaf nodes of a WMST, using questionnaire as a pre-test. The weights of middle nodes are calculated using (7). We update weights in a WMST of a worker whenever he finishes a new task. So, the WMST can capture the proficiency degree more and more accurately.

Each task $t_j \in T$ is associated with a set of skill requirements $R_{t_j}=\{r_{t_j}(s_1), ..., r_{t_j}(s_n)\}$, where s_i represents a non-root skill node in the WMST, with the constraints that $fth^{(x)}(s_j) \neq s_i$ for any $x>0$ and $dep(s_i)<dep(s_j)$ for $s_i, s_j \in S$. $r_{t_j}(s_i)$ is the proportion of the total skill requirement for t_j that is associated with s_i, under the constraint such that $\sum_{\forall s_i} r_{t_j}(s_i)=1$. The values of these proportions are determined by the task requester.

We use the *match quality*, $m(a^{(q_k)}_{w_i,t_j})$, to represent how well a worker w_i is fit for a task t_j, considering both the required skill and related skills. The principle is that the larger the distance between a required skill and a related skill is, the less contribution the related skill will have. For example, a software engineer is more suitable for a software development than a good driver, however, a software engineer adept to both programming languages (required skill) and software engineering methodology (related skill) will be more qualified than that only adept to a programming language. So, we can formulate the match quality as

$$m(a^{(q_k)}_{w_i,t_j}) = \sum_{\forall s \in R_{t_j}} r_{t_j}(s) \times \left(d_{w_i}(s) + \sum_{n=1}^{dep(s)-1} \frac{\sigma^n (d_{w_i}(fth^{(n)}(s)) - d_{w_i}(fth^{(n-1)}(s)))}{\|chd(fth^{(n)}(s))\| - 1} \right) \tag{8}$$

where σ $(0<\sigma<1)$ is the attenuation coefficient. All influences are exerted through common ancestors.

5 Adaptive Task Assignment Algorithm

5.1 Acceptance Expectation

We propose the *acceptance expectation* model, which calculates the expected number of workers who will *accept* and *complete* a task before its deadline. We estimate the execution time for a task using a power law distribution [Boutsis and Kalogeraki, 2014; Ipeirotis, 2010]. Here, we assume that workers be permanent, like [Kobren *et al.*, 2015].

Let $p_{w_i,t_j}(x)$ be the probability that w_i will spend an amount of time x on t_j. The corresponding CDF $P_{w_i,t_j}(x)$ represents the probability that w_i will spend time more than x. Mathematically, we have $p_{w_i,t_j}(x) = \frac{\alpha-1}{x_{min}} \left(\frac{x}{x_{min}}\right)^{-\alpha}$, where $\alpha>0$ is a constant and x_{min} is a lower time bound, and $P_{w_i,t_j}(x) = Pr(X > x) = \int_x^\infty p(X)dX = \left(\frac{x}{x_{min}}\right)^{-\alpha+1}$, where $\alpha=1+n\left(\sum_{i=1}^n \ln \frac{x_i}{x_{min}}\right)$. For each worker, we initially set x_{min} to his lowest previous execution time.

We divide workers $U^{(q_k)}$ into three pairwise disjoint subsets: *available workers* $U_A^{(q_k)}$ who are online and free, i.e., $v_{w_i}^{(q_k)}=1$ and $||L_{w_i}^{(q_k)}||=0$; *busy workers* $U_B^{(q_k)}$ who are online but working, i.e., $v_{w_i}^{(q_k)}=1$ and $||L_{w_i}^{(q_k)}|| \geq 1$; and *offline workers* $U_O^{(q_k)}$ who are offline, i.e, $v_{w_i}^{(q_k)}=0$. Note that in any timeslot q_k, we have $U^{(q_k)}= U_A^{(q_k)} \cup U_B^{(q_k)} \cup U_O^{(q_k)}$.

We use $\widetilde{P}_{U,w_i,t_j}$ to denote probability that $w_i \in U$ will can complete t_j before its deadline, where U is a set of workers. ρ_{t_j} denotes probability that t_j will be accepted. The acceptance expectation $E^{(q_k)}_{U,t_j}$ of workers in U for t_j in q_k will be

$$E^{(q_k)}_{U,t_j} = \sum_{w_i \in U} \rho_{t_j} \widetilde{P}_{U,w_i,t_j} \tag{9}$$

Available Workers

The probability that t_j will be completed by available workers before its deadline is calculated by

$$\widetilde{P}_{U_A,w_i,t_j} = 1 - P_{w_i,t_j}(d_{t_j} - q_{w_i,t_j}), \tag{10}$$

where q_{w_i,t_j} is the time when t_j is assigned to w_i.

Busy Workers

Let w_i be busy on $t_b \in L_{w_i}^{(q_k)}$. We call the list exclusive of the first element as *pending task list* $L_{w_i}^{'(q_k)}$. The probability $\widetilde{P}^{(q_a,q_b)}_{U_B,w_i,t_j}$ that w_i completes t_j during $[q_a, q_b]$ such that $b_{t_j} \leq q_{w_i,t_j} \leq q_a < q_b \leq d_{t_j}$ is

$$\begin{aligned} \widetilde{P}^{(q_a,q_b)}_{U_B,w_i,t_j} &= Pr\left(q_a - q_{w_i,t_j} < X < q_b - q_{w_i,t_j}\right) \\ &= \frac{P_{w_i,t_j}\left(q_a - q_{w_i,t_j}\right) - P_{w_i,t_j}(q_b - q_{w_i,t_j})}{1 - P_{w_i,t_j}\left(q_a - q_{w_i,t_j}\right)} \end{aligned} \tag{11}$$

When a new task t_j is assigned to a busy worker w_i, he has to have sufficient time to complete t_j after he has completed all the previous tasks in his current list. Therefore, we have

$$\widetilde{P}_{U_B,w_i,t_j} = \sum_{q_l=q_k}^{d_{t_j}-1} \widetilde{P}_{U_B,w_i,t_b}^{(q_l,q_{l+1})} \widetilde{P}_{U_B,w_i,t_j}^{(q_{l'},d_{t_j})} \tag{12}$$

where $q_{l'} = q_l + 1 + e_{w_i}||L_{w_i}^{(q_k)} - t_b||$ and e_{w_i} is the average execution time of w_i, with $\widetilde{P}_{U_B,w_i,t_j}^{(q_{l'},d_{t_j})}$=0 if $q_{l'}$>d_{t_j}.

Offline Workers

We use a binary state vector to represent a worker's s-tate, where 0 and 1 represent the offline and online s-tates, respectively. We estimate the probability using Markov model. Let $M^{(q_k)}$ denote the transition matrix from q_k to q_{k+1}, where $1{\le}k{<}n$. We assume a period of timeslot-s $\widetilde{Q}$=$\{q_1, q_2, ..., q_m\}{\subseteq}Q$ such that $M^{(q_k)}$=$M^{(q_{k+m})}$ for any $1{\le}k{<}n{-}m$. Formally, we have $M^{(q_k)}{=}\begin{pmatrix} p_{00}^{(q_k)} & p_{01}^{(q_k)} \\ p_{10}^{(q_k)} & p_{11}^{(q_k)} \end{pmatrix}$, where $p_{ij}^{(q_k)}$ is the probability from state i in q_k to j in q_{k+1}. This probability can be computed by counting the frequency of state transitions from i in q_k to j in q_{k+1}. Let $I_{w_i}^{(q_k)}$=$[p^{(q_k)}(x{=}0), p^{(q_k)}(x{=}1)]$ denote the initial state of w_i in q_k. We use the l previous states to estimate the future states. The probability of a given state in q_{k+s} can be formulated as

$$I_{w_i}^{(q_{k+s})} = \sum_{n=0}^{l} \zeta_n I_{w_i}^{(q_{k-n})} \prod_{m=-n}^{s-1} M^{(q_{k+m})} \tag{13}$$

where ζ_n is the weight of the state in q_{k-n} subject to the constraint $\sum_{n=0}^{l} \zeta_n$=1. We use $\widehat{P}_{U_O,w_i}^{(q_k,q_{k+s})}$ to denote probability that an offline worker w_j will login during $[q_k, q_{k+s}]$, using

$$\widehat{P}_{U_O,w_i}^{(q_k,q_{k+s})} = p^{(q_{k+s})}(x=1) \prod_{m=0}^{s-1} p^{(q_{k+m})}(x=0) \tag{14}$$

There are two possible situations for any offline worker: the current task list is empty, or there may be one or more tasks in his current task list. So, we have $\widetilde{P}_{U_O,w_i,t_j}$=

$$\begin{cases} \sum_{s=1}^{d_{t_j}-q_k} \widehat{P}_{U_O,w_i}^{(q_{k+s})} \widetilde{P}_{U_O,w_i,t_j}^{(q_{k+s+1},d_{t_j})}, & ||L_{w_i}^{(q_k)}||=0 \\ \sum_{s=1}^{d_{t_j}-q_k} \widehat{P}_{U_O,w_i}^{(q_{k+s})} \sum_{q_l=q_{k+s}}^{d_{t_j}-1} \widetilde{P}_{U_O,w_i,t_b}^{(q_l,q_{l+1})} \widetilde{P}_{U_O,w_i,t_j}^{(q_{l'},d_{t_j})}, & ||L_{w_i}^{(q_k)}||\ge \quad 1 \end{cases} \tag{15}$$

Combining the formulas (9)-(15), when we assign t_j to workers $U^{(q_k)}{\subseteq}W$ in q_k, the acceptance expectation of t_j is

$$E_{U,t_j}^{(q_k)} = E_{U_A,t_j}^{(q_k)} + E_{U_B,t_j}^{(q_k)} + E_{U_O,t_j}^{(q_k)} \tag{16}$$

5.2 Task Assignment Algorithm (AE-TA)

We define the concept of *task urgency degree* $u_{t_j}^{(q_k)} = g_{t_j}^{(q_k)}/(g_{t_j}^{(b_j)}(d_{t_j}{-}q_k))$, where $g_{t_j}^{(q_k)}$ is the number of remaining positions for t_j in q_k. $u_{t_j}^{(q_k)}$ is proportional to the remaining

Algorithm 1: AE-TA Algorithm

```
Input: $W, T, Q, \{\rho_{t_j}^{(b_{t_j})} | t_j \in T\}$
Output: the set of matches $A$ and rejected matches $A'$
1: for $q_k = q_1$ to $q_n$:
2:     Clear pending task lists for all $w_i{\in}W$;
3:     $T^{(q_k)}{\leftarrow} \forall t_j \in T$ with $b_{t_j} < q_k < d_{t_j}, g_{t_j}^{(q_k)} > 0$;
4:     sort $T^{(q_k)}$ by $u_{t_j}^{(q_k)}$;
5:     for $t_j \in T^{(q_k)}$:
6:         $W_{t_j}^{(q_k)} \leftarrow \{w | a_{w,t_j}^{(q_s)} \notin A, q_s < q_k\}$;
7:         sort $W_{t_j}^{(q_k)}$ by $m(a_{w,t}^{(q_k)})$;
8:         $U_{t_j}{\leftarrow}\arg\min_{||U||} \left(E_{U,t_j}^{(q_k)} \ge g_{t_j}^{(q_k)}\right), U_{t_j}{\in}W_{t_j}^{(q_k)}$
9:         for $w_i \in U_{t_j}$:    /* $U_{t_j}{=}U_{A,t_j}{\cup}U_{B,t_j}{\cup}U_{O,t_j}$ */
10:            $A = A \cup a_{w_i,t_j}^{(q_k)}$ if $w_i \in U_{A,t_j}$ ;
11:            $L_{w_i}^{(q_k)} \leftarrow L_{w_i}^{(q_k)} \cup t_j$;
12:        end for
13:        for all $w_i'$ rejects $t_j$:        /* after feedback */
14:            $A' \leftarrow A' \cup a_{w_i,t_j}^{(q_k)}$;    $L_{w_i}^{(q_k)} \leftarrow L_{w_i}^{(q_k)} - t_j$;
15:        end for
16:        $g_{t_j}^{(q_{k+1})}{\leftarrow}\max(g_{t_j}^{(q_k)}{-}||U_{A,t_j}||{+}||A'_{t_j}||, 0)$;
17:        $\rho_{t_j}^{(q_{k+1})} \leftarrow \frac{\sum_{q=b_{t_j}}^{q_k} ||U_{A,t_j}^{(q)}||{-}||A'_{t_j}||}{\sum_{q=b_{t_j}}^{q_k} ||U_{A,t_j}^{(q)}||}$;
18:    end for
19: end for
```

number of required workers and inversely proportional to the remaining time. We propose the two types of assignments.

The first is *actual assignment for available workers*. In this case, we directly assign each of them a suitable task. If he accepts a task, his state will change to busy. Instead, the second is *dummy assignment for busy and offline workers*. Here, a task is added to the pending task list. Possibly, a busy worker completes his task more slowly than expected and an offline worker maybe login later than expected. So, the pending task list is updated at the beginning of every timeslot.

Our AE-TA is presented in Algorithm 1. In every timeslot, tasks are sorted by their urgency degrees in lines 3-4 before assignment. For each task, AE-TA finds the group of workers who are the most suitable for that task (lines 6-8) and assigns the task to them (lines 9-12). This process includes both *actual assignment* (line 10) and *dummy assignment* (line 11). The key point is to find the minimum set U_{t_j} of workers who are the most suitable for t_j; then, AE-TA directly assigns the task to the workers in U_{A,t_j}. In this way, AE-TA reserves positions for the workers who are the most suitable but temporarily u-navailable. After feedback, AE-TA records rejected matches in lines 13-15 and updates parameters in lines 16-17.

6 Performance Evaluations

6.1 Compared Algorithms and Skill Models

We compare our AE-TA with the four task assignment algorithms: (1) *Random*, which randomly assigns tasks to available workers only; (2) *MatchParticipantFirst* [Mavridis *et al.*, *2016*], in which tasks are assigned firstly to the workers with

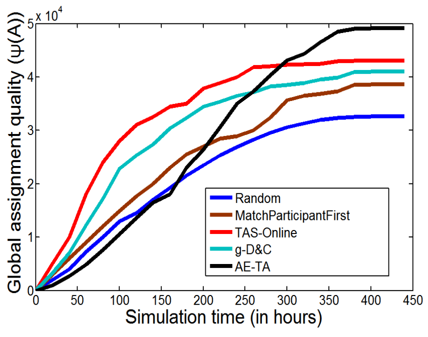

Figure 1: $\Psi(A)$ vs. time

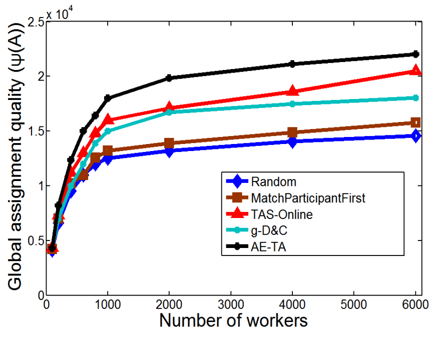

Figure 2: $\Psi(A)$ with the number of workers

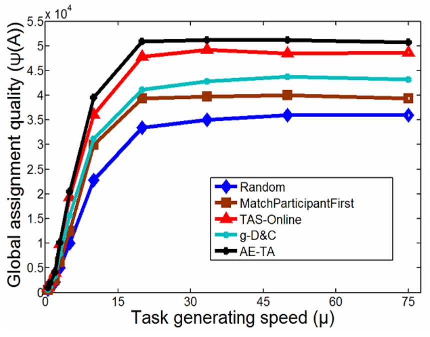

Figure 3: $\Psi(A)$ with task generating speed

the fewest skills; (3) *TAS-Online* [Lykourentzou and Schmitz, 2015], which finds the maximum-weight bipartite match among tasks and available workers; and (4) *g-D&C* [Cheng *et al.*, 2016], which decomposes tasks and workers into g groups and greedily chooses the maximum task-to-worker pairs.

Moreover, we compared our WMST with (1) *Taxonomy-based skill model* [Mavridis *et al., 2016*] based on a non-weight tree structure, where each task is associated with only one main skill, and (2) *Keyword-based skill model* [Fan *et al.*, 2015], where skills are modeled as a set of keywords.

6.2 Performance on Synthetic Data

The timeslot was set as one hour. We simulated 1500 workers, and built one WMST for each worker. In each WMST, weights of skills in leaf nodes were randomly set in $[0, 1]$ and weights of non-leaf nodes were computed using formula (7). All WMSTs had the same structure, with a maximum depth of 5, and each node except the leaf nodes had five children.

Tasks were generated at an average speed of μ=10 $tasks/hour$. The required number g_t of workers for each task t was randomly initialized such that $g_t \in [5, 50]$. The deadline d_t was set by adding b_t with a time drawn from a normal distribution with a mean of 50 and a variance of 20. Any task randomly requires 1 to 5 skills in leaf nodes of WMSTs of the workers who involved in this task.

Let task execution time follow the power law distributions $p(x)=cx^{-\alpha}$, where c and α were randomly selected in $c\in[10, 30]$ and $\alpha\in[1.5, 2.5]$. To obtain the distribution of active workers, we extracted the active time of 300 workers from Weibo within one week. Following this distribution of active workers, we generated worker state vectors. We set ρ_{t_j}=0.8 for all tasks. Four thousand tasks were randomly generated over 15 days. The experiment ended when all tasks had expired, after 18 days in total.

We tested how global assignment quality $\Psi(A)$ changes with time. Figure 1 shows that after 280 hours, our AE-TA algorithm can achieve the highest $\Psi(A)$ among all algorithms, although its $\Psi(A)$ is lower in the early stage. The reason is that our AE-TA algorithm reserves tasks for workers who are more suitable but currently unavailable, whereas the other algorithms always assign tasks to available workers only.

To evaluate how $\Psi(A)$ is affected by the number of workers, we changed the worker pool size from 100 to 6000 with the fixed task generating speed (μ=10 $tasks/hour$) and execution time (18 days). As shown in Figure 2, $\Psi(A)$ increases with the number of workers if there are not enough workers. When the number of workers reaches saturation, however, $\Psi(A)$ tends to be stable. The results demonstrate that our AE-TA algorithm always outperforms the other algorithms in terms of global assignment quality. Similarly, Figure 3 illustrates how $\Psi(A)$ changes with the number of tasks. Here, we fixed the number of workers as 1500 and generated tasks with a speed distribution of $\mu\in[0.5, 75]$ $tasks/hour$. As illustrated in Figure 3, saturation is eventually reached as tasks increase.

6.3 Performance on Real Data

To evaluate real performances of our task assignment algorithm AE-TA and skill model WMST, we recruited workers with different skills, conducted *one-shot* and *long-run evaluations*. In the one-shot evaluation, the workers were asked to complete a questionnaire consisting of single-option questions. For the long-run evaluation, we designed specialized translation tasks and assigned them over 7 days. In both parts of evaluations, WMSTs of all the workers have the same structure with different weights and a maximum depth of 5.

One-Shot Performance

For this evaluation, we designed a questionnaire that consists of 70 single-option questions. Each question is associated with 4 alternative answers, which requires 1 to 5 atomic skills described by leaf nodes of the WMST. To built WMSTs of 80 recruited workers, we carefully selected and assigned 20 questions for initial estimation of skill weights of these workers. In case that a worker did not finish all 20 questions, the corresponding weights in his WMST were set as 0.

Based on these WMSTs, we then evaluated the relationship between the proposed *match quality* and the correctness of the answers to demonstrate the effectiveness of our approach. Here, we asked the 20 workers with individual WMSTs built above to complete the remaining 50 questions, which generated 1000 answers. Figure 4 presents histograms showing the numbers of correct and incorrect answers. Among these answers, the average match quality scores for correct and incorrect answers are 3.46 and 2.12, respectively. The results also reveal that these workers have different fields of expertise, and the workers who perform higher match quality achieve higher correct ratios of answers.

Next, we arranged other 60 workers to evaluate different assignment algorithms and skill models, using the 50 questions. We set g_t=5 for all tasks. Since the tasks were assigned

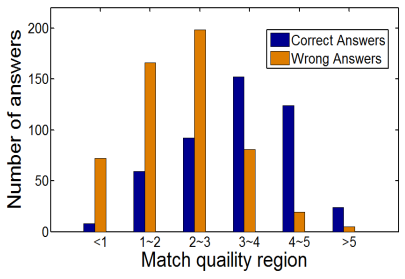

Figure 4: Match quality with answers

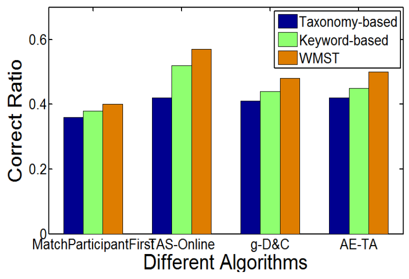

Figure 5: One-shot performance

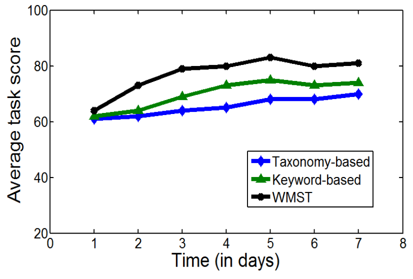

Figure 6: Long-run performance of models

in a batch in each timeslot, every worker was regarded as available and could only be assigned one task. Therefore, not all tasks could be fully completed, but we counted the number of correct answers and computed the corresponding ratio of correct answers with respect to the number of questions answered. According to Figure 5, our WMST always significantly outperforms the taxonomy- and keyword-based skill models. The taxonomy-based model exhibits the worst performance, indicating that it is not suitable for the multi-skill scenario. The keyword-based model cannot capture the correlations among skills so that it cannot ensure that workers are assigned the most suitable tasks. By contrast, our WMST is designed to consider multiple skills and their correlations.

From Figure 5, we also find that our AE-TA algorithm can achieve higher correct answer ratios except TAS-Online. TAS-Online strives to achieve the maximum-weight bipartite match, which corresponds to the theoretical maximum performance; however, *it is only suitable for application in a single timeslot*. Real-world scenarios are more complex, and worker states can change over time. Meanwhile, it is easy to see that both the one-shot version of AE-TA and g-D&C perform greedy matching so that they achieve similar performances.

Long-Run Performance

We also conducted long-run experiments over 7 days. 84 workers were recruited for the long-run performance tests. They were evenly divided into four groups in MatchParticipantFirst, TAS-Online, g-*D&C* and our AE-TA. Their WMSTs were built by the 20 above single-option questions.

We chose 70 articles, which varied in length from 500 to 5000 words. Each article translation was treated as a task. We equally partitioned each article into several parts with 500 words. These tasks were released randomly at an average speed of μ=10 $tasks/day$ and were set to expire in 1 to 3 days. In Figure 6, we used AE-TA as the common assignment algorithm to evaluate different skill models. Based on the common skill model WMST, we then tested four task assignment algorithms, as shown in Figure 7. We requested more 20 independent experts to evaluate every translation with a score from 0 to 100. Since all tasks have almost the same length, the payment depends on only the quality of translations.

As shown in Figure 6, where we combined our AE-TA with the three skill models respectively, our WMST model always outperforms Taxonomy-based and Keyword-based skill models in multi-skill scenarios. Further, Figure 7 shows the average translation scores over 7 days. The results are similar to those in the synthetic experiments: our AE-TA exhibits the overall best performance among all four task assignment algorithms. Although TAS-Online has better performance in the first two days, its average score declines in subsequent days. The reason is that this algorithm assigns tasks to all available workers; and during the first few days, there are more available workers, whereas in later timeslots, not enough suitable workers are available. Consequently, AE-TA outperforms over TAS-Online. Thus, the long-run evaluation indicates that the AE-TA algorithm assigns tasks to more skill-qualified workers, thereby ensuring that more tasks are completed by workers with more suitable skills.

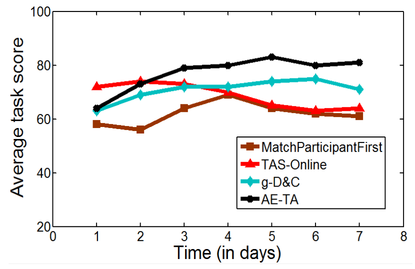

Figure 7: Long-run performance of algorithms

7 Conclusion and Future Work

We investigated the problem of multi-skill-based optimal complex task assignment in service crowdsourcing. We propose a *WMST* model to capture correlations among skills, and a match quality model to provide a comprehensive and quantitative description of how well a worker is suitable for a task. We design the *AE-TA* algorithm based on the proposed acceptance expectation. Among the most related task assignment algorithms and skill models, our AE-TA algorithm and WMST model exhibit the best performance on both synthetic and real data sets. We will investigate how to solve and prove the task assignment problem theoretically.

Acknowledgments

This work was supported by the National Natural Science Foundation of China projects (Nos. 61832013 and 61672351), the STCSM AI project (No. 19511120300), National Key R&D Program of China (No. 2019YFB2102204), Aerospace Fund (No. 20GFC-JJ02-012), Alibaba AIR project (No. SCCT622019010803), and Huawei Technologies Co., Ltd projects (Nos. YBN2019105155 and YBN2019075061). Feilong Tang is the corresponding author of this paper.

References

[Alon *et al.*, 2015] Noga Alon, Michal Feldman, Omer Lev, and Moshe Tennenholtz. How robust is the wisdom of the crowds? In *IJCAI*, pages 2055–2061, 2015.

[Assadi, Hsu, and Jabbari, 2015] Sepehr Assadi, Justin Hsu, and Shahin Jabbari. Online assignment of heterogeneous tasks in crowdsourcing markets. In *HCOMP*, pages 12–21, 2015.

[Boutsis and Kalogeraki, 2013] Ioannis Boutsis and Vana Kalogeraki. Crowdsourcing under real-time constraints. In *IPDPS*, pages 753–764, 2013.

[Boutsis and Kalogeraki, 2014] Ioannis Boutsis and Vana Kalogeraki. On task assignment for real-time reliable crowdsourcing. In *ICDCS*, pages 1–10, 2014.

[Chandra *et al.*, 2015] Praphul Chandra, Yadati Narahari, Debmalya Mandal, and Prasenjit Dey. Novel mechanisms for online crowdsourcing with unreliable, strategic agents. In *AAAI*, pages 1256–1262, 2015.

[Cheng *et al.*, 2016] Peng Cheng, Xiang Lian, Lei Chen, and Jizhong Zhao. Task assignment on multi-skill oriented s-patial crowdsourcing. *IEEE Transactions on Knowledge and Data Engineering*, 28(8):2201–2215, 2016.

[Choi, Yoo, and Lee, 2018] Jihun Choi, Kang Min Yoo, and Sang-goo Lee. Learning to compose task-specific tree structures. In *AAAI*, 5094–5101, 2018.

[Desmarais and Baker, 2012] Michel Desmarais and Ryan Baker. A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction*, 22(1-2):9–38, 2012.

[El Maarry *et al.*, 2014] Kinda El Maarry, Wolf-Tilo Balke, Hyunsouk Cho, Seung-won Hwang, and Yukino Baba. Skill ontology-based model for quality assurance in crowdsourcing. In *DASFAA*, pages 376–387, 2014.

[Fan *et al.*, 2015] Ju Fan, Guoliang Li, Beng Chin Ooi, Kian-lee Tan, and Jianhua Feng. icrowd: An adaptive crowdsourcing framework. In *SIGMOD*, pages 1015–1030, 2015.

[Goel, Nikzad, and Singla, 2014] Gagan Goel, Afshin Nikzad, and Adish Singla. Allocating tasks to workers with matching constraints: truthful mechanisms for crowdsourcing markets. In *WWW*, pages 279–280, 2014.

[Haas et al., 2015] Daniel Haas, Jason Ansel, Lydia Gu, and Adam Marcus. Argonaut: macrotask crowdsourcing for complex data processing. *Proceedings of the VLDB Endowment* 8(12):1642–1653, 2015.

[Ho and Vaughan, 2012] Chien-Ju Ho and Jennifer Wortman Vaughan. Online task assignment in crowdsourcing markets. In *AAAI*, pages 45–51, 2012.

[Ho et al., 2013] Chien-Ju Ho, Shahin Jabbari, and Jennifer Wortman Vaughan. Adaptive task assignment for crowdsourced classification. In *ICML*, pages 534–542, 2013.

[Ipeirotis, 2010] Panagiotis G Ipeirotis. Analyzing the amazon mechanical turk marketplace. *XRDS: Crossroads, The ACM Magazine for Students*, 17(2):16–21, 2010.

[Kobren *et al.*, 2015] Ari Kobren, Chun How Tan, Panagiotis Ipeirotis, and Evgeniy Gabrilovich. Getting more for less: Optimized crowdsourcing with dynamic tasks and goals. In *WWW*, pages 592–602, 2015.

[Liu *et al.*, 2018] Liyuan Liu, Jingbo Shang, Xiang Ren, Frank Fangzheng Xu, Huan Gui, Jian Peng, Jiawei Han Empower sequence labeling with task-aware neural language model. In *AAAI*, pages 5253–5260, 2018.

[Lykourentzou and Schmitz, 2015] Ioanna Lykourentzou, and Heinz Schmitz. An online approach to task assignment and sequencing in expert crowdsourcing. In *HCOMP*, pages 1–2, 2015.

[Magazine and Chern, 1984] Michael J Magazine and Maw-Sheng Chern. A note on approximation schemes for multidimensional knapsack problems. *Mathematics of Operations Research*, 9(2):244–247, 1984.

[Mavridis *et al., 2016]* Panagiotis Mavridis, David Gross-Amblard, and Zoltán Miklós. Using hierarchical skills for optimized task assignment in knowledge-intensive crowdsourcing. In *WWW*, 843–853, 2016.

[Qiu *et al.*, 2016] Chenxi Qiu, Anna Cinzia Squicciarini, Barbara Carminati, James Caverlee, and Dev Rishi Khare. Crowdselect: Increasing accuracy of crowdsourcing tasks through behavior prediction and user selection. In *CIKM*, 539–548, 2016.

[Rasch, 1993] Georg Rasch. *Probabilistic models for some intelligence and attainment tests.* ERIC, 1993.

[Roy et al., 2015] Senjuti Basu Roy, Ioanna Lykourentzou, Saravanan Thirumuruganathan, Sihem Amer-Yahia, and Gautam Das. Task assignment optimization in knowledge-intensive crowdsourcing. *The VLDB Journal*, 24(4):467–491, 2015.

[Salek *et al., 2013]* Mahyar Salek, Yoram Bachrach, and Peter Key. Hotspotting-a probabilistic graphical model for image object localization through crowdsourcing. In *AAAI*, pages 1156–1162, 2013.

[Tang, 2019] Feilong Tang. Bidirectional Active Learning with Gold-Instance-Based Human Training. In *IJCAL*, pages 5989–5996, 2019.

[Tang *et al.*, 2019] Feilong Tang, Heteng Zhang, and Laurence T. Yang. Multipath cooperative routing with efficient acknowledgement for LEO satellite networks. *IEEE Transactions on Mobile Computing*, 18(1): 179–192, 2019.

[Xu *et al.*, 2018] Tong Xu, Hengshu Zhu, Chen Zhu, Pan Li, and Hui Xiong. Measuring the popularity of job skills in recruitment market: A multi-criteria approach. In *AAAI*, pages 2572–2579, 2018.

[Yuen, King, and Leung, 2011] Man-Ching Yuen, Irwin K-ing, and Kwong-Sak Leung. Task matching in crowdsourcing. In *iThings/CPSCom 2011*, pages 409–412, 2011.

[Zheng *et al.*, 2015] Yudian Zheng, JiannanWang, Guoliang Li, Reynold Cheng, and Jianhua Feng. Qasca: a quality-aware task assignment system for crowdsourcing applications. In *SIGMOD*, pages 1031–1046, 2015.