

# Quality-Assured Synchronized Task Assignment in Crowdsourcing

Jiayang Tu, Peng Cheng, Lei Chen, *Member, IEEE*

**Abstract**—With the rapid development of crowdsourcing platforms that aggregate the intelligence of Internet workers, crowdsourcing has been widely utilized to address problems that require human cognitive abilities. Considering great dynamics of worker arrival and departure, it is of vital importance to design a task assignment scheme to adaptively select the most beneficial tasks for the available workers. In this paper, in order to make the most efficient utilization of the worker labor and balance the accuracy of answers and the overall latency, we a) develop a parameter estimation model that assists in estimating worker expertise, question easiness and answer confidence; b) propose a *quality-assured synchronized task assignment scheme* that executes in batches and maximizes the number of potentially completed questions (MCQ) within each batch. We prove that MCQ problem is NP-hard and present two greedy approximation solutions to address the problem. The effectiveness and efficiency of the approximation solutions are further evaluated through extensive experiments on synthetic and real datasets. The experimental results show that the accuracy and the overall latency of the MCQ approaches outperform the existing online task assignment algorithms in the synchronized task assignment scenario.

**Index Terms**—Crowdsourcing, Scheduling Algorithm, Task Assignment

## 1 INTRODUCTION

Recently, crowdsourcing has gained significant attention and popularity in the field of academic research or real-world applications. Crowdsourcing is leveraged to solve questions that require human cognitive abilities, for example, schema matching [26] and entity resolutions [21]. The emergence of multiple well-established public crowdsourcing platforms such as Amazon Mechanical Turk (AMT) and Figure Eight has facilitated a manageable worker labor market. However, workers have different domain knowledge and are error-prone especially when assigned to questions they are unskilled at. To control the quality of answers, a common alternative for most requesters is to design a task assignment scheme that assigns the most beneficial tasks to the target workers.

There are many existing studies to effectively perform **task assignment** with quality assurance on an online basis [8], [15], [18], [32], where task selection is based on some pre-defined evaluation metrics such as Accuracy or F-score [32], and each coming worker is assigned with the top- $k$  tasks that maximize the evaluation metric. However, the number of required repetitions for tasks is usually set beforehand, thus it is unlikely for the requesters to augment or terminate the allocation of a task according to its answer confidence, which will either cause the return of an uncertain question or an unnecessary waste of worker labor (or a longer delay).

As mentioned above, a task assignment scheme that dynamically determines the number of repetitions of tasks is urgently needed. To improve the existing work, there are two challenges to deal with: 1) developing an efficient parameter estimation model. Task assignment is typically studied in isolation or together with parameter estimation in the online scenarios. In other words, the model must be able to efficiently keep track of important parameters like the confidence of answers that assist in the task assignment optimization; 2) balancing the accuracy and the overall latency. The accuracy of the answers can be improved if we assign more repetitions of tasks to the workers [10], unfortunately, which will increase the overall latency or needed cost. Therefore, providing a quality-assured task assignment strategy while not sacrificing the overall latency is of vital importance.

To address the first challenge, we devise a parameter estimation model that assumes **worker expertise** and **question easiness** are two potential parameters to collectively determine the **confidence of answers** [23]. Specifically, worker expertise denotes the average accuracy of a worker, confidence of answers determines the probability of each answer being the truth, and question easiness measures the certainty of current voted answers. Intuitively, answers of expert workers to easier questions have higher probabilities of being true. In addition, the parameters are adaptively updated in the question answering process, thus it requires no prior knowledge of workers. In our model, a question is noticed to become easier when the inferred answer is more certain. With this regard, we provide an easiness score threshold  $\delta$  denoting the certainty of returned answers, and requesters can set the threshold value according to their preferences to control the accuracy of answers.

To resolve the second challenge, based on the parameter estimation model stated above, we run our task assignment scheme batch by batch (*synchronized task assignment*) and

- Jiayang Tu is with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong, China; Lenovo Machine Intelligence Center, Hong Kong, China. Email: jtuaa@connect.ust.hk.
- Lei Chen is with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong, China. Email: leichen@cse.ust.hk.
- Peng Cheng is with the School of Computer Science and Software Engineering, East China Normal University, Shanghai, China. Email: pcheng@sei.ecnu.edu.cn. Corresponding author: Peng Cheng

TABLE 1: Easiness Score Increase

$W \backslash Q$	$q_1(c_1 = 0.35, rep_1 = 0)$	$q_2(c_2 = 0.25, rep_2 = 2)$
$w_1$	0.2	0.2
$w_2$	0.1	0.1

renew the model parameters at the end of each batch. At the very beginning of every batch, we carefully consider simultaneous task processing situation and assign each idle worker to one task, aiming at maximizing the number of completed questions without wasting worker labor. In order to fulfill this objective, workers are assigned to a question set  $Q_T$  that is as packed as possible (i.e.,  $|Q_T|$  is minimized). The advantages of batch processing and the optimization goal are two-fold: 1) worker labor is concentrated on the smallest set of questions, then for each batch it is likely to gain more completed questions, thus the overall latency can be controlled; 2) fewer questions are occupied for worker processing, which reserves more available questions for next batch workers and enables further optimization.

Current task assignment assumes an online task assignment scenario, where tasks are set with a fixed number of repetitions beforehand and assigned to the workers once they join the platform. Such settings can have the following problems: (1) uncertain answers are likely to return if the number of repetitions is limited, resulting in less reliable data quality; (2) short-sighted task assignment may happen due to the unconsciousness of the overall situation, which may cause exceeding assignment of the questions, waste worker labor and lead to longer delay. To improve the above problems, in some real applications like Didi Chuxing [27], batch by batch assignment (synchronized task assignment) [27], [9] is widely utilized in which the joint benefit of multiple workers is optimized. Next, we show the difference between synchronized task assignment and top- $k$  online assignment regarding problem (1) with a concrete example as follows.

**Motivating Example.** Suppose two workers  $W = \{w_1, w_2\}$  arrive sequentially in the question pool  $Q = \{q_1, q_2\}$ . And we judge whether a question can be returned by verifying whether its easiness score has reached the threshold  $\delta$ . Intuitively, easiness score measures the current certainty of the voted answers regarding a question and is calculated from confidences of answers, while  $\delta$  is the requirement of the question certainty set by the requesters. For binary questions  $q$ ,  $\delta = 0.6$  requires the easiness score of its two answers should have an absolute difference of 0.6, meaning the easiness score distribution is 0.8 and 0.2. And for multiple-choice questions, it measures the average easiness score distances among answers. Assume  $\delta$  and the current estimated easiness score of  $q_j$ , denoting by  $sc.d(q_j)$  are known beforehand, thus the remaining easiness score of  $q_j$  is equal to  $c_j = \delta - sc.d(q_j)$ . In this example, we assume  $c = \{0.35, 0.25\}$ . Besides, the expected easiness score increase of all workers to all tasks is shown in TABLE 1.

**a) Synchronized task assignment.** In synchronized task assignment, we aim at making the most efficient utilization of worker labor and minimize  $|Q_T|$  (the number of assigned tasks), so the remaining easiness score of questions  $c$  cannot be overused. We show 4 possible cases of question assignment as the following:

**Case 1:**  $w_1, w_2 \rightarrow q_1$ : feasible and  $|Q_T| = 1$

**Case 2:**  $w_1 \rightarrow q_1, w_2 \rightarrow q_2$ : feasible and  $|Q_T| = 2$

**Case 3:**  $w_1, w_2 \rightarrow q_2$ : overuse

**Case 4:**  $w_1 \rightarrow q_2, w_2 \rightarrow q_1$ : feasible and  $|Q_T| = 2$

In the 4 above cases, case 1 is a feasible assignment because  $0.35 - 0.2 - 0.1 > 0$  (similar with case 2 and 4) and case 3 is not adopted since it overuses the remaining easiness score of  $q_2$  ( $0.2 + 0.1 > 0.25$ ) and results in a waste of worker labor. According to the optimization goal of synchronized task assignment, we tend to select case 1 to minimize  $|Q_T|$ . And two repetitions of  $q_1$  are created immediately and assigned to  $w_1$  and  $w_2$ .

**b) Online task assignment** (assume  $k = 1$ ). According to online task assignment, workers arrive in the order of  $w_1, w_2$ , and each of them should be allocated with one question. For each worker, assume the evaluation metric is to assign him/her to one question whose remaining easiness score is minimized. As noted in Table 1, suppose the number of repetitions left for  $q_1, q_2$  are 0, 2 respectively (denoted as  $rep_1 = 0$  and  $rep_2 = 2$ ) and  $c = \{0.35, 0.25\}$  stays the same. The reason why  $rep_1 = 0$  but  $c_1$  still has a large value is that the answers of  $q_1$  contributed by previous workers are so diverse that the remaining easiness score (uncertainty) is kept high. Since  $rep_1 = 0$ , the only task assignment method is case 3 ( $w_1, w_2 \rightarrow q_2$ ). As a result,  $q_1$  is returned but far from certainty while the assignment has exceeded the remaining easiness score  $c_2$  of  $q_2$  and wastes worker labor.

The motivation example reveals that compared to top- $k$  online task assignment, synchronized task assignment is able to make efficient utilization of worker labor. We keep tracking of the remaining easiness score of questions to carefully assign workers and decide whether to increase or decrease the allocation of a question. Therefore, the number of repetitions of every question can be determined on the fly and unnecessary expenses can be avoided for the requesters. In addition, answers can be returned with quality assurance.

In this paper, we focus on the multiple-choice questions, the most common kinds of questions in crowdsourcing, in which they have the limited number of choices and a consensus can be reached if workers act truthfully. We make the following contributions:

- We develop an efficient parameter estimation model that assists in estimating worker expertise, question easiness and answer confidence in Section 2.
- We propose a quality-assured synchronized task assignment scheme executing in batches and maximizing the number of completed questions (MCQ) in every batch (MCQ problem is proved NP-hard) in Section 3.
- We present two efficient approximation algorithms to address the MCQ problem in Section 4.
- We conducted extensive experiments on synthetic and real datasets to evaluate the proposed algorithms in Section 5.

Related literature is discussed in Section 6. We finally conclude the work in Section 7.

## 2 PARAMETER ESTIMATION MODEL

In this section, we begin with the modeling of basic crowdsourcing components and show the detailed development of the parameter estimation model. We lay the foundation of the optimization problem in next section and briefly introduce several standard crowdsourcing concepts commonly used throughout this paper.

- **Worker.** Workers arrive at the crowdsourcing platform and select human intelligence tasks to perform. In this

TABLE 2: Summary of Symbols

Symbol	Description
$c(a)$	the confidence of answer $a$
$d(q)$	the easiness of question $q$
$e(w)$	the expertise of worker $w$
$A(w)$	the set of answers provided by $w$
$W(a)$	the set of workers offering answer $a$
$A(q)$	all possible choices of question $q$
$l$	the number of answers of each question
$t$	the truth of a question
$\delta$	the easiness score threshold
$w_i$	the $i^{th}$ worker
$q_j$	the $j^{th}$ question
$c_j$	the remaining easiness score of question $q_j$
$Q_T$	the set of questions assigned to workers in each batch
$I$	the question assignment scheme in each batch
$E^2 I_{ij}$	the expected easiness score increase of $w_i$ to $q_j$
$W(q_j)$	the set of workers assigned to question $q_j$
$Q$	the set of questions in this batch
$W$	the set of available workers in this batch
$Q_o$	the set of current open questions
$U_{i,j}$	the remaining easiness score of $q_j$ after answered by $w_i$
$X_j$	the set of workers assigned by XM Greedy to $j$ th question

paper, workers are assumed to be rational, meaning that there exists no group of malicious workers that aims to dominate the answers and destroy the benefits of the crowdsourcing platform or the other workers.

- **Question.** Questions (or tasks) are created and published to crowdsourcing platforms by requesters. The task types are usually varying from multiple-choice questions to numeric questions, both of which are micro-tasks that require short processing time for workers. Note that *questions* and *tasks* are interchangeably used in this paper and they refer to the same entity.
- **Repetition.** Repetition is the most atomic unit of a question that a worker can respond to. The number of repetitions regarding a question is declared by the requester on task creation with the purpose of improving the answer reliability. For a single question, the repetitions are identical and are assigned to different workers.
- **Answer.** Answers are submitted worker votes. Answers to the same question are often various due to the distinct worker expertise. Note that *answers*, *choices* and *votes* are interchangeably used in this paper and they refer to the same entity.

## 2.1 Fundamental Definitions

**Definition 1 (Answer Confidence).** The confidence of an answer  $a$  of question  $q$ , denoted as  $c(a)$ , is the probability that  $a$  is the truth of  $q$ .

**Definition 2 (Question Easiness).** The easiness of a question  $q$ , denoted as  $d(q)$ , is the adjusted average of the total pairwise distances between confidences of different answers.  $0 < d(q) < 1$ , where a larger value of  $d(q)$  indicates that  $q$  is more certain to return the answer with a higher confidence as the truth.

**Definition 3 (Worker Expertise).** The expertise of a crowdsourcing worker  $w$ , denoted as  $e(w)$ , is the average expected confidence over all answers that  $w$  has voted.

Intuitively, without ground truths, Answer Confidence helps estimate the probability of each answer being the truth; Question Easiness is used to measure how far a question is away from certainty in order to guarantee the

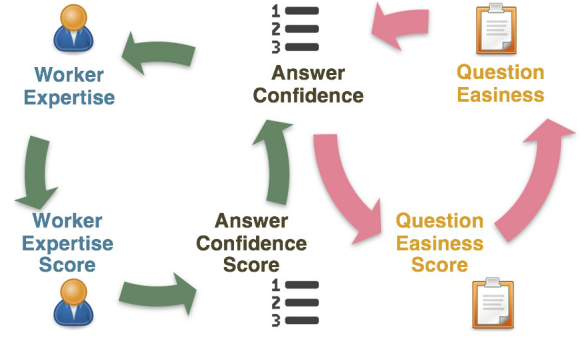


Fig. 1: The Dual-Cycle Parameter Estimation Model

quality of the returned answers; Worker Expertise denotes the probability of a worker that can provide correct answers for questions.

## 2.2 Iterative Parameter Estimation

The iterative parameter estimation model consists of two inference circles in Figure 1, namely the inference between worker expertise and answer confidence (the green cycle) and the inference between question easiness and answer confidence (the red cycle).

In the existing work [24], TRUTHFINDER is run on aggregated answers and performs well on conflicting web information that usually has a large number of votes (more than 5) regarding an identical object. However, in this work, we focus on paid crowdsourcing tasks where the number of needed repetitions for a question should be as less as possible to gain equally reliable estimated parameters, for example, answer confidence. And the main challenge is that when the votes are too few and diverse at the same time, we cannot directly utilize the power of the TRUTHFINDER prototype to resolve. Together with the requirement of tiny delay in the synchronized task assignment scenario, even with very few votes and they are quite diverse, the parameter estimation model should still have the ability to act efficiently and accurately to calculate the parameters.

### 2.2.1 Inference between Worker Expertise and Answer Confidence

As given in Definition 3, the *expertise of a worker  $w$*  can be expressed as:

$$e(w) = \frac{\sum_{a \in A(w)} c(a)}{|A(w)|}, \quad e(w) \in [0, 1) \quad (1)$$

where  $A(w)$  refers to the set of answers provided by  $w$ .

Given the question easiness  $d(q)$ , the confidence of the answer  $a$  is calculated as  $c(a) = d(q)(1 - \prod_{w \in W(a)} (1 - e(w)))$ , where  $W(a)$  refers to the set of workers offering the same answer  $a$  to question  $q$ . Then,  $\prod_{w \in W(a)} (1 - e(w))$  is the probability that no workers assigned to  $q$  chooses  $a$ , so  $c' = 1 - \prod_{w \in W(a)} (1 - e(w))$  represents the probability that  $a$  is selected. However, in this work, we think that answer confidence does not merely rely on the expertise of the workers but also is determined by the question nature itself, namely, the question easiness. Apparently, if a question is easier, the answer obtained from the same worker will have a higher possibility to be the truth, and this is the reason why we yield  $c$  by multiplying  $c'$  by a factor of  $d(q)$ .

However, to avoid the underflow situation in  $c(a)$  resulted from the repetitive multiplication of the term  $(1 - e(w))$ , we further define the *expertise score of a worker*  $w$ , denoted as  $sc.e(w)$ , as followings:

$$sc.e(w) = -\ln(1 - e(w)) \quad (2)$$

Similarly, for the ease of calculating answer confidence from the expertise score, *confidence score of an answer*  $a$ , denoted as  $sc.c(a)$ , is defined as:

$$sc.c(a) = -\ln(1 - \frac{c(a)}{d(q)}) \quad (3)$$

We derive the relationship between expertise score and confidence score as  $sc.c(a) = \sum_{w \in W(a)} sc.e(w)$  according to Equation 1 - 3. And the deduction process is shown in the following:

$$\begin{aligned} c(a) &= d(q)(1 - \prod_{w \in W(a)} (1 - e(w))) \\ \Leftrightarrow 1 - \frac{c(a)}{d(q)} &= \prod_{w \in W(a)} (1 - e(w)) \\ \Leftrightarrow \ln(1 - \frac{c(a)}{d(q)}) &= \sum_{w \in W(a)} \ln(1 - e(w)) \\ \Leftrightarrow sc.c(a) &= \sum_{w \in W(a)} sc.e(w) \end{aligned} \quad (4)$$

Next, we update the expression of *answer confidence* with the help of Equation 3:

$$\begin{aligned} e^{sc.c(a)} &= e^{-\ln(1 - \frac{c(a)}{d(q)})} \\ \Leftrightarrow c(a) &= \frac{d(q)}{1 - e^{-sc.c(a)}} \end{aligned} \quad (5)$$

In Equation 5, answer confidence  $c(a)$  can be negative, which is not reasonable according to our definition. In order to guarantee  $c(a)$  has a positive value, we apply a logistic function and Equation 5 is adjusted to  $c(a) = \frac{d(q)}{1 + e^{-sc.c(a)}}$ , where  $c(a)$  is mapped to a smaller range  $(0, 1)$  [24]. This adjustment can be intuitively interpreted: as the easiness of the question increases (higher  $d(q)$ ),  $c(a)$  is larger, meaning that the answer  $a$  has a higher probability to be the truth; when the overall expertise score of workers who vote  $a$  is higher ( $sc.c(a)$  becomes higher since  $sc.c(a) = \sum_{w \in W(a)} sc.e(w)$ ), then  $a$  also has a higher probability of being correct.

All  $c(a)$ ,  $a \in A(q)$  is normalized after the calculation is finished, which ensures the positive values of worker expertise. The inference between worker expertise and answer confidence is depicted as the green cycle in Figure 1.

### 2.2.2 Inference between Question Easiness and Answer Confidence

As suggested above, the answer confidence  $c(a)$  is adjusted to  $\frac{d(q)}{1 + e^{-sc.c(a)}}$ . If we set  $d(q)$  to 1, the inference between worker expertise and answer confidence is exactly the TRUTHFINDER framework [24]. However, TRUTHFINDER performs well simply when the question has a substantial number of votes. Based on its framework, we propose the question easiness concept in Definition 2 and develop another important inference cycle between answer confidence and question easiness, which is depicted as the red cycle in Figure 1.

Let  $A(q)$  be the set of all possible answers of  $q$ . Then the number of pairs within  $A(q)$  is  $p = \frac{|A(q)|(|A(q)|-1)}{2}$ . In addition, we define the *easiness score of a question*  $q$  (denoted by  $sc.d(q)$ ) as follows:

$$sc.d(q) = \frac{\sum_{a_i, a_j \in A(q), a_i \neq a_j} |c(a_i) - c(a_j)|}{p} \quad (6)$$

The confidence differences over all answer pairs are summed up and averaged with the motive to measure the certainty of the voted answers. For example, given a binary-choice question with two answers  $a_1$  and  $a_2$  and there are two cases of the confidence distribution: (1)  $c(a_1) = c(a_2) = 0.5$ ; (2)  $c(a_1) = 0.8, c(a_2) = 0.2$ . Obviously, case 2 is certain enough to return  $a_1$  while case 1 is far from certainty. In other words, a large confidence difference indicates high certainty.

However, it is noticed that  $sc.d(q)$  can vary from 0 to 1, if we pass the value of  $sc.d(q)$  to  $d(q)$  in the formula of  $c(a)$ ,  $c(a)$  is quite sensitive to the variations of  $d(q)$ . To smooth the fluctuation of  $sc.d(q)$ , a new concept *easiness* ( $d(q)$  in Definition 2) is introduced and we set  $d(q) = \frac{1}{1 + k \cdot e^{-sc.d(q)}}$ , where  $k \in (0, 1]$  and  $d(q)$  is thus mapped to  $[\frac{1}{1+k}, 1]$ .

The advantages of introducing question easiness are summarized into two aspects: a) For those difficult questions with uncertain answers, the confidence of their answers will be diminished because of the effect of  $d(q)$  in the expression of  $c(a)$ . Therefore, worker expertise or expertise scores that originally dominated by these questions are declined. Consequently, the difficult questions caused by few and uncertain answers tend to become easier since the confidence of the answers voted by high-expertise workers starts to increase and exceed that of others. Due to this reason, the parameter estimation can be accelerated; b) Since question easiness is defined to measure the certainty of the tasks, it is beneficial for us to know which questions are demanding additional reliable workers. Therefore, our parameter estimation model is applicable to crowdsourcing circumstances where the worker answers are very few and diverse. Besides, it acts as a helpful sign to dynamically guide the task assignment in the latter section.

### 2.2.3 Execution of the Dual-cycle Parameter Estimation Model

The whole parameter estimation model is demonstrated in Figure 1. As described above, we can estimate the worker expertise and question easiness if we know the answer confidence, and vice versa. We adopt an iterative method to run the model [24]. The execution of the model is as below.

The model is run on a batch basis. In the first batch, we initialize the expertise of workers to be equal, and we run the left cycle (parameters in green) for a fixed number of iterations. And starting from answer confidence, we run the right cycle (parameters in red) for the same fixed number of iterations. Finally, this dual-cycle estimation is repeated until answer confidence varies in a small range. Starting from the second batch of questions, the model starts to calculate the parameters according to the worker answers from last batch.

## 3 PROBLEM STATEMENT

In this section, we first introduce some preliminary definitions, and then formally propose the *synchronized task as-*

signment to maximize the number of completed questions (MCQ) problem. Finally, we prove that MCQ problem is NP-hard.

**Definition 4 (Easiness score Increase (EI)).** Given a question  $q$  with  $l$  answers ( $l \geq 2$ ), let  $W(a_i)$  denote the sets of workers who have voted  $a_i$ . As defined in Section 2, the number of answer pairs is  $p = \frac{l(l-1)}{2}$  and thus the easiness score of  $q$  is  $sc.d(q) = \frac{\sum_{a_i, a_j \in A(q), a_i \neq a_j} |c(a_i) - c(a_j)|}{p}$ . Pick a random answer  $a_k$ ,  $c(a_k) = \frac{d(q)}{1 + e^{-sc.c(a_k)}}$  where  $sc.c(a_k) = \sum_{w \in W(a_k)} sc.e(w)$ . Assume a coming worker  $w$  votes  $a_k$ , then  $EI(A(w) = a_k|q) = sc.d(q)' - sc.d(q)$ . Note that the calculation of  $sc.d(q)'$  is similar with  $sc.d(q)$ , except that in  $sc.d(q)'$ ,  $c'(a_k) = \frac{d(q)}{1 + e^{-sc.c(a_k)'}}$  where  $sc.c(a_k)' = sc.c(a_k) + sc.e(w)$ .

$EI$  can vary from negative to positive values. Specifically, when the answer  $a_k$  above reduces the average confidence distance among all answers, the easiness score will decrease, then  $EI(A(w) = a_k|q) = sc.d(q)' - sc.d(q) < 0$ , and vice versa. The definition of  $EI$  helps in measuring the benefit of a worker  $w$  to a question  $q$ , where the benefit is evaluated by the expected increase of the easiness score incurred by  $w$  regarding  $q$  as follows.

**Definition 5 (Expected Easiness score Increase ( $E^2I$ )).** The  $E^2I$  of a worker  $w$  regarding question  $q$  is:

$$E^2I(w|q) = \left| \sum_{k=1}^l P(A(w) = a_k|q) \cdot EI(A(w) = a_k|q) \right| \quad (7)$$

Equation 7 calculates the expectation of the easiness score increase ( $EI$ ) that a worker brings to a question.  $P(A(w) = a_k|q)$  means the probability of worker  $w$  giving the answer  $a_k$  to question  $q$ , and  $EI(A(w) = a_k|q)$  is the corresponding  $EI$  when  $w$  votes  $a_k$ . For each possible answer, these two terms are multiplied and accumulated, finally we get the absolute value of the result. The reason why we take the absolute value is that  $EI(A(w) = a_k|q)$  can be negative when the  $sc.d(q)$  decreases. Without knowing the correct answer of  $q$ , the answer that shows its advantage in determining the absolute sum, namely  $E^2I(w|q)$ , is regarded as the most possible answer given by  $w$ . Therefore,  $E^2I$  represents a fair expectation measurement for every worker since no default truth is assumed.

Next, we show how to calculate the term  $P(A(w) = a_k|q)$  in Equation 7. Let  $t$  denote the true answer of question  $q$ , then Equation 7 can be calculated as the following:

$$P(A(w) = a_k|q) = \sum_{r=1}^l P(A(w) = a_k|t=r) \cdot P(t=r), \quad (8)$$

where  $P(t=r)$  is the probability of answer  $r$  being the truth, which is exactly the answer confidence  $c(r)$ . And in order to find  $P(A(w) = a_k|t=r)$ , we assume the worker selection probability equals to  $p = \frac{1}{1 + e^{-c(w) \cdot d(q)}}$  [23], which means that worker expertise and question easiness can jointly affect worker  $w$  to select a correct answer to question  $q$ . Therefore, we have:  $P(A(w) = a_k|t=r) = p$ , if  $a_k = r$ , meaning the worker answer  $a_k$  happens to be the truth  $r$ ; otherwise,  $P(A(w) = a_k|t=r) = \frac{1-p}{l-1}$ , that is, worker has a equal probability  $\frac{1-p}{l-1}$  to choose a wrong answer.

As mentioned in Section 1, the **easiness score threshold**  $\delta$  is provided for the requester to control the accuracy of answers and reduce their cost. Questions with easiness score equal to or exceeding  $\delta$  will be returned from the question pool at the end of each batch. Therefore, if both the current easiness score  $sc.d(q)$  of question  $q$  and  $\delta$  are known, the remaining easiness score of  $q$ , denoted by  $c$ , is equal to  $\delta - sc.d(q)$ . Then we formally propose our optimization problem as the following.

**Definition 6 (the Maximizing Completions of Questions (MCQ) Problem).** Given a worker set  $W = \{w_1, w_2, \dots, w_n\}$  of size  $n$  and a question set  $Q = \{q_1, q_2, \dots, q_m\}$  of size  $m$  in the current batch. Let  $c_j$  denote the remaining easiness score of  $q_j$  and  $E^2I_{ij}$  denote the expected easiness score increase of worker  $w_i$  to question  $q_j$ . To maximize the number of completed questions and to make the most efficient utilization of the worker labor, our task assignment scheme  $I$  aims to assign each worker to one question in order that all workers can be packed into a question set  $Q_T$  and  $|Q_T|$  is minimized. Note that minimizing  $|Q_T|$  means to tightly assign all available workers to the least number of questions in order that the most number of questions can be finished in each batch.

**NP-hardness of the MCQ problem.** We prove that the MCQ problem is NP-hard by reducing it from the Bin Packing problem [19].

**Theorem 1.** The MCQ problem is NP-hard.

*Proof.* Specifically, we show the proof in detail by a reduction from the Bin Packing problem. A Bin Packing problem can be described as follows: given a set of bins  $s_1, s_2, \dots, s_m$  with the same size  $V$  and a list of  $n$  items with size  $a_1, a_2, \dots, a_n$ , the problem is to find the minimum number of bins  $B$  and a B-partition  $s_1 \cap s_2 \cap \dots \cap s_B$  s.t.  $\sum_{i \in S_k} a_i \leq V$ .

For a given Bin Packing problem instance, we can transform it to an instance of MCQ as follows: we set  $E^2I_{ij} = a_i$  and  $c_j = V$ , which means workers and questions are treated as items and bins accordingly, and workers have the same  $E^2I$  over all questions. According to the MCQ problem, our target is to assign all workers to questions in order that the number of assigned questions is minimized, which is equivalent to minimize the number of used bins of equal size under the constraint that all items should be packed into bins. Given this mapping, it is easy to show that Bin Packing Problem instance can be solved if and only if the transformed MCQ problem can be solved. Therefore, the optimal solution of the MCQ problem reveals the optimal solution of the Bin Packing Problem. Therefore, the MCQ problem is NP-hard.  $\square$

## 4 ASSIGNMENT ALGORITHMS

In this section, we propose two greedy algorithms to address the MCQ problem and study their approximation ratios. Before that, we first illustrate some preliminary concepts.

### 4.1 Preliminary Concepts

We now define the following terms which will be commonly used in the approximation algorithms.

- **Feasible task assignment.** A question assignment scheme  $I$  is called feasible if  $\forall q_j$ , the total  $E^2I$  of its assigned

---

**Algorithm 1: First Match Algorithm**


---

**Input:**  $E^2I, W, Q$  and  $C = c_1, c_2, \dots, c_n$   
**Output:**  $Q_T$ , question assignment scheme I

```

1 for  $w_i \in W$  do
2   for  $q_j \in Q$  do
3     Let  $U_{i,j} = c_j - E^2I_{i,j}$ , the remaining  $c_j$  after
        $w_i \rightarrow q_j$ ;
4 Find  $U_{i',j'} = \min U_{i,j}, \forall w_i \in W, \forall q_j \in Q$ ;
5 Let  $w_{i'}$  be  $w_1, I = \{w_1 \rightarrow q_{j'}\}$ ;
6 Let  $Q_o$  denote the set of open questions sorted by
  the insertion order;
7  $Q_o = \{q_{j'}\}$ ;
8 Randomly order the workers from  $w_2$  to  $w_n$ ;
9  $\forall i > 1$ , update  $U_{i,j'} = U_{i,j'} - E^2I_{1,j'}$ ;
10 for  $w_i \in W - \{w_1\}$  do
11   if  $\forall q_j \in Q_o, U_{i,j} < 0$  then
12     Let  $U_{i,k} = \min U_{i,j}, \forall q_j \in Q - Q_o$ ;
13     Open the closed question  $q_k$ , set
        $Q_o = Q_o \cup \{q_k\}$ ;
14   else
15     Find the first  $q_k \in Q_o$  s.t.  $U_{i,k} \geq 0$ ;
16    $I = I \cup \{w_i \rightarrow q_k\}$ ;
17    $\forall z > i$ , update  $U_{z,k} = U_{z,k} - E^2I_{i,k}$ ;
18  $Q_T = Q_o$ 

```

---

workers  $W(q_j)$  does not exceed its remaining easiness score, which means  $\sum_{w_i \in W(q_j)} E^2I_{i,j} \leq c_j$ .

- **Open/closed questions.** A question is called open when it is assigned to workers and is called closed if no workers are assigned to it.

Next, we introduce two algorithms, namely, the *First Match Algorithm* and *Best Match Algorithm*. And their time complexities are further analyzed.

## 4.2 First Match Algorithm

The whole procedure of First Match is illustrated in Algorithm 1. Note that our target is to assign each worker to one question s.t. the final question set  $|Q_T|$  is minimized. In order to satisfy this target, the main idea of the First Match Algorithm is that whenever we consider a new worker, we simply look at the open questions and assign him/her to the First Matching open question without exceeding its remaining easiness score. If the  $EI$  of this worker cannot fit into all open questions, at this time we open a new closed question. First match controls the number of open questions by adopting a lazy assignment, where it either assigns a worker to the First Matching open question or conservatively opens a new question when all open questions cannot hold the  $EI$  of this worker.

We use the symbol  $\rightarrow$  to denote the task assignment operation. In lines 1-3,  $U_{i,j}$  denotes the remaining easiness score of question  $q_j$  if answered by  $w_i$ . In lines 4-5, we open the first question by finding the smallest  $U_{i',j'}$ , then  $w_{i'}$  is set to  $w_1$  and the assignment  $w_1 \rightarrow q_{j'}$  is added to the assignment scheme  $I$ . In addition,  $q_{j'}$  is added to the set of open questions  $Q_o$ . For the remaining workers  $w_i \in W - \{w_1\}$  where  $i > 1$ , we randomly index them and update their  $U_{i,j'}$  (lines 8-9). For each worker  $w_i$ , the First

Match algorithm then opens a question only if the current worker  $w_i$  cannot fit in all previous open questions  $Q_o$  and it selects a new open question that has the least remaining easiness score if answered by  $w_i$  (lines 11-13). Otherwise, if  $w_i$  can fit into more than one open questions in  $Q_o$ , he/she will be assigned to the question with the lowest index, that is, the question opened the earliest (lines 14-15). Finally, the assignment is added to  $I$  and after the assignment,  $U$  is updated (lines 16-17).

**The Time Complexity.** We need  $O(mn)$  steps to initialize  $U_{i,j}$  and at most  $O(mn)$  steps to figure out the  $\min U_{i',j'}$  respectively (line 1-4). In lines 10-15, for each worker  $w_i$ , we need  $O(m)$  time to find either the  $\min U_{i,k}$  ( $q_k \in Q - Q_o$ , that is the set of closed questions) or the first open question  $q_k \in Q_o$ . And updating  $U_{z,k}, \forall z > i$  requires additional  $O(n)$  time cost in the worst case since the worker set  $W$  of size  $n$  is looped. Totally, the time complexity of the First Match Algorithm is  $O(mn + mn + n(m + n))$ , which is equal to  $O(n^2 + nm)$ .

## 4.3 Best Match Algorithm

As mentioned above, the First Match Algorithm adopts a lazy question assignment scheme where a worker is always assigned to the first feasible question  $q_j$  in the set of open questions  $Q_o$ . However, there exists a case that if  $w_i$  is assigned to some other question  $q_k$  in  $Q_o$ , the remaining easiness score of  $q_k$  is less than that of  $q_j$ , namely,  $0 \leq c_k - E^2I_{i,k} < c_j - E^2I_{i,j}$ , and  $k > j$ . It is highly possible that we can yield more completed questions at the end of this batch if  $w_i \rightarrow q_k$ . However, First Match does not consider this situation.

In order to select the questions more carefully for the workers and further increase the number of potentially completed questions, we propose another approximation algorithm, namely Best Match Algorithm, as shown in Algorithm 2.

The second algorithm, the Best Match algorithm, is similar to the First Match algorithm in lines 1-15. However, the main difference is that when  $w_i$  can fit in more than one open question in  $Q_o$ ,  $w_i$  will be assigned to the question with the smallest remaining score incurred by him/her. This is revealed in lines 16-19, where the best question is first found by performing a traversal of the remaining easiness scores over the set of open questions  $Q_o$  and then finding the minimum  $U_{i,z}$ . Therefore, worker  $w_i$  is assigned to question  $q_z$  in line 18. After that, all affected  $U$  is updated in line 19.

**The Time Complexity.** Similar with the First Match algorithm, we need  $O(mn)$  steps to initialize  $U_{i,j}$  and at most  $O(mn)$  steps to figure out the  $\min U_{i',j'}$  respectively (lines 1-4). In lines 10-19, for each worker  $w_i$ , we need  $O(m)$  time to find either the  $\min U_{i,k}$  ( $q_k \in Q - Q_o$ , which is the set of closed questions) (lines 12-14) or the  $\min U_{i,z}$  ( $q_z \in Q_o$ , which is the set of open questions) (lines 17-18). Furthermore, updating affected  $U$  requires additional  $O(n)$  time cost in the worst case since the worker set  $W$  of size  $n$  is looped. To sum up, the time complexity of the Best Match Algorithm is  $O(mn + mn + n(m + n))$ , which is equal to  $O(n^2 + nm)$ . The time complexities of the two algorithms are identical.

**Algorithm 2: Best Match Algorithm**


---

**Input:**  $E^2I, W, Q$  and  $C = c_1, c_2, \dots, c_n$   
**Output:**  $Q_T$ , question assignment scheme I

```

1 for  $w_i \in W$  do
2   for  $q_j \in Q$  do
3     Let  $U_{i,j} = c_j - E^2I_{i,j}$ , the remaining  $c_j$  after
        $w_i \rightarrow q_j$ ;
4  $\forall w_i \in W, \forall q_j \in Q$ , find  $U_{i',j'} = \min U_{i,j}$ ;
5 Let  $w_{i'}$  be  $w_1, I = \{w_1 \rightarrow q_{j'}\}$ ;
6 Let  $Q_o$  denote the set of open questions;
7  $Q_o = \{q_{j'}\}$ ;
8 Randomly order the workers from  $w_2$  to  $w_n$ ;
9  $\forall i > 1$ , update  $U_{i,j'} = U_{i,j'} - E^2I_{1,j'}$ ;
10 for  $w_i \in W - \{w_1\}$  do
11   if  $\forall q_j \in Q_o, U_{i,j} < 0$  then
12     Let  $U_{i,k} = \min U_{i,j}, \forall q_j \in Q - Q_o$ ;
13     Open the closed question  $q_k$ , set
        $Q_o = Q_o \cup \{q_k\}$ ;
14      $I = I \cup \{w_i \rightarrow q_k\}$ ;
15      $\forall z > i$ , update  $U_{z,k} = U_{z,k} - E^2I_{i,k}$ ;
16   else
17     Find the best question  $q_z$  s.t.  $U_{i,z} = \min U_{i,j}$ ,
        $\forall q_j \in Q_o$ ;
18      $I = I \cup \{w_i \rightarrow q_z\}$ ;
19      $\forall v > i$ , update  $U_{v,z} = U_{v,z} - E^2I_{v,i}$ ;
20  $Q_T = Q_o$ 

```

---

**4.4 The Approximation Ratios**

Next, we study the approximation ratios of the *First Match Algorithm* and *Best Match Algorithm* together since we prove that they have the same approximation ratios. We use XM to represent both of them for the convenience of description.

Note that in Definition 6, our MCQ problem minimizes  $|Q_T|$ .

**Lemma 4.1.** *Index the questions of  $Q_T$  in the order opened by XM. Consider the  $j$ th ( $j \geq 2$ ) question, any worker that was assigned to it cannot fit into questions opened prior to  $q_j$ .*

*Proof.* We show the proof of Lemma 4.1 by contradiction. Suppose there is a question  $q_k$ , where  $q_k \in Q_T$  and  $k < j$ . Assume worker  $w$  can be fit into  $q_k$  and is assigned to  $q_j$  for First Match Algorithm. However, if there is more than one question in  $Q_T$  that can hold  $w$ , the First Match will pick the first satisfied question. Since  $k < j$ ,  $q_k$  should be selected to  $w$ , which forms a contradiction. Thus, Lemma 4.1 is proved under the First Match. The proof of Best Match algorithm is similar to the First Match and is omitted here.  $\square$

In order to continue finding the approximation ratios, we divide workers and questions of  $Q_T$  into two types respectively as:

- **Experts/normal workers.** A worker  $w_i$  is called an expert if his/her  $E^2I$  is greater than the half remaining easiness score of all open questions, which means  $\forall q_j \in Q_T, E^2I_{ij} \geq \frac{c_j}{2}$ , where  $Q_T$  refers to the set of open questions. Otherwise, he/she is a normal worker.

**Algorithm 3: The Helper Procedure**


---

```

1 for  $i = 1$  to  $v - 1$  do
2   Let  $X_{j'}$  be the nonempty set with the highest
     index;
3   if  $j' = i$  then
4     Stop.
5   else
6     Let  $w$  be the worker with smallest  $E^2I$  to  $q_{j'}$ ;
7     Set  $X_i \leftarrow X_i \cup \{w\}$ ; and  $X_{j'} \leftarrow X_{j'} \setminus \{w\}$ ;

```

---

- **Type N/E questions.** A question is defined to be Type N if it is only assigned to normal workers and is of Type E if it is not Type N, namely, it has at least one assigned expert.

We know that XM can ultimately get  $|Q_T|$  questions assigned. For a given integer  $v, 2 \leq v \leq |Q_T|$ , select  $v$  questions from  $Q_T$  and index them in the order that opened by XM. Let  $X_j$  be the set of workers assigned by XM to the  $j$ th question,  $j = 1, 2, \dots, v$ .

Then we manually partition  $Q_T$  produced by XM into two sets. The first set includes only Type N questions, and the second set includes the remaining questions produced by XM. Assume there are  $c$  experts among the workers. Since every worker is assigned to one question, then  $|Q_T| - c$  is the number of Type N questions and  $c$  is the number of Type E questions. Index the questions in the first set in the order they are opened, from 1 to  $|Q_T| - c$ .

Based on Lemma 4.1, we construct a lower bound on the optimal  $|Q_T|$  with the help of Algorithm 3: the Helper Procedure. In the Helper Procedure, workers assigned to latter questions in  $Q_T$  are picked and re-assigned to former questions (line 6-7). However, according to Lemma 4.1, workers are not able to fit into previous questions in First Match or Best Match. Therefore, every assignment in the Helper Procedure will cause the overused situation of some question  $q_j$ , namely, the joint  $E^2I$  of workers assigned to  $q_j$  is larger than its remaining easiness score  $c_j$ .

So we let  $v = |Q_T| - c$  and apply the Helper Procedure to the set of Type N questions. Assume the Helper Procedure can produce  $m$  questions, there are at least  $m - 1$  questions are overused. This can be easily derived. If the number of overused questions is less than  $m - 1$ , say, equal to  $k$ , the Helper Procedure tends to pick another worker who was assigned to the  $m^{th}$  question and re-assign him/her to  $(k + 1)^{th}$  question which will be overused. And this process will continue until no former questions before the  $m^{th}$  question are available, which means the number of overused questions is at least  $m - 1$ .

With the help of the Helper Procedure, Lemma 4.2 is proposed as follows:

**Lemma 4.2.**  $\frac{c}{2} + m^* - 1 < |Q_T^*|$ , where  $Q_T^*$  is the optimal solution and  $m$  is produced by the Helper Procedure on the Type N questions of  $Q_T^*$ .

*Proof.* Assume the optimal solution of the MCQ problem finally achieves  $Q_T^*$  as the final set of assigned questions.  $Q_T^*$  is divided into two sets: the first set contains the Type N questions while the second set contains the Type E question. According to the above, the size of Type E questions is



exactly  $c$ . And assume there are  $y$  questions of Type N. We apply the Helper Procedure to the first set and obtain  $m^*$  questions with at least  $m^* - 1$  questions are overused, where  $m^* - 1 < y$ . Furthermore, from the definition of *experts*, we know that there exist no two experts that can be assigned to one identical question. Therefore, since  $|Q_T^*| = c + y$ , and  $m^* - 1 < y$ , we then yield  $|Q_T^*| > c + m^* - 1 > \frac{c}{2} + m^* - 1$ .  $\square$

**Theorem 2.** *XM Algorithms have approximation ratios of  $2 + \frac{2}{|Q_T^*|}$ .*

*Proof.* As mentioned previously, the Helper Procedure produces  $m^*$  questions with at least  $m^* - 1$  overused, which means at least  $m^* - 1$  questions contain 2 workers. Therefore, we have  $m^* \geq \frac{|Q_T^*| - c}{2}$  or  $|Q_T^*| - m^* - c \leq m^*$ . Combining Lemma 4.2, we have the following:

$$\begin{aligned} & |Q_T| - m^* - c + \left(\frac{c}{2} - 1\right) \leq m^* + \left(\frac{c}{2} - 1\right) < |Q_T^*| \\ \Leftrightarrow & |Q_T| < |Q_T^*| + m^* + \frac{c}{2} + 1 < |Q_T^*| + (m^* + \frac{c}{2} - 1) + 2 \\ \Leftrightarrow & |Q_T| < 2|Q_T^*| + 2 \\ \Leftrightarrow & \frac{|Q_T|}{|Q_T^*|} < 2 + \frac{2}{|Q_T^*|}, \end{aligned}$$

where the first line is obtained by adding  $\frac{c}{2} - 1$  to both sides of formula  $|Q_T| - m^* - c \leq m^*$ . This approximation ratios mean the XM algorithms can reach solutions (measured in the number of assigned questions in one batch) that is less than  $2 + \frac{2}{|Q_T^*|}$  times of the optimal solution, which gives a bounded guarantee of the minimum questions that we have to assigned in each batch.  $\square$

## 5 EXPERIMENTS

We extensively evaluate our MCQ approaches on both synthetic and real datasets. The results are presented and analyzed in this section. Before presenting the results, we first introduce the datasets we use.

### 5.1 Data Sets

We use both real and synthetic data to study our proposed MCQ approaches. Specifically, for real data, we use one of the Figure Eight datasets [1] which contains worker arrival times and the Duck Identification dataset from [2] that contains worker labels.

**Figure Eight Dataset.** The dataset we use is called *Relevance of terms to disaster relief topics*, where workers are asked to label the relevance of pairwise terms related to disaster relief. Most importantly, the dataset contains the concrete dates and times of all worker submissions. In total, there are 1,400 workers and 18,062 worker submissions. On the average, each worker has 12.9 submissions. The time span is 20 days.

**Duck Identification Dataset.** This dataset is released by the project in [31]. It contains 108 questions and 39 workers, and every worker provides answers to all questions, where workers are asked to label 0 or 1 to denote whether the task contains a duck. All the questions are gold standard questions with true answers.

**Synthetic Dataset.** For Synthetic dataset, we randomly generate the truths of *binary-choice questions*. The worker quality

TABLE 3: Parameter Settings for Synthetic Experiment

Parameter	Description	Value
$m$	# questions	250, 500, 750, 1,000
$\delta$	easiness score threshold	0.2, 0.3, 0.4, 0.5, 0.6
$rep$	# repetitions in qasca	1, 3, 4
$k$	top- $k$ tasks in qasca	1
$\lambda$	the worker arriving rate	1

and question easiness are drawn from Gaussian distributions, which are simply utilized to simulate the worker selection probability. Note that our parameter estimation model does not initialize the parameters to the above distributions.

### 5.2 Baseline and Evaluation Metrics

Our *synchronized task assignment* scheme executes in batches. At the end of each batch, when all worker answers are recorded, the iterative parameter estimation model starts to run, parameters like worker expertise, question easiness, and answer confidence are adaptively inferred and leveraged to perform task assignment in next batch. Therefore, the synchronized task assignment requires no historical performances or profiles of workers. With the above assumption, we adopt the online top- $k$  task assignment policy (using F-score) as the baseline (denoted as Qasca) since the way we achieve and update the worker quality is similar [32]. Furthermore, the online scheme is enforced to run in batches in order to compare with ours. We use FM and BM to denote First Match Greedy and Best Match Greedy respectively.

We compare our solutions with Qasca in two evaluation metrics: 1) the accuracy of the returned results; 2) the number of required batches to complete all questions. For all solutions, we repeat the question assignment for 100 times and record the average values of each evaluation metric.

### 5.3 Experiments on Synthetic Data

#### 5.3.1 Experimental Settings

According to the synthetic dataset mentioned above, the worker expertise  $e$  and question easiness  $d$  are drawn from Gaussian distributions. Specifically,  $e \sim \mathcal{N}(0.7, 0.1)$  and  $d \sim \mathcal{N}(0.9, 0.03)$ . With such distributions, the worker expertise is mostly set to be larger than 0.5 since we think the worst worker is the one who chooses randomly and the largest value of worker expertise is set to be 1. We adjust  $d$  to a relatively high value to guarantee that the correct answers are more than erroneous ones. Note that we control the value of  $d$  without exceeding 1 due to its definition. With the above settings, a worker is simulated to select a correct answer with the probability  $p = \frac{1}{1+e^{-e \cdot d}}$  [23]. For the parameter estimation model, we initialize the worker expertise, question easiness and answer confidence to 0.5 for every worker, question, and answer respectively. In order to accelerate the inference, we consider the influences between answers [24]. In binary-choice questions, the high confidence score of one answer indicates the low confidence score of the other. Therefore, we update their confidence scores by using  $sc.c(a_1)' = sc.c(a_1) - sc.c(a_2)$  and  $sc.c(a_2)' = sc.c(a_2) - sc.c(a_1)$ , where  $a_1$  and  $a_2$  are the two answers for each question. In addition, we replace



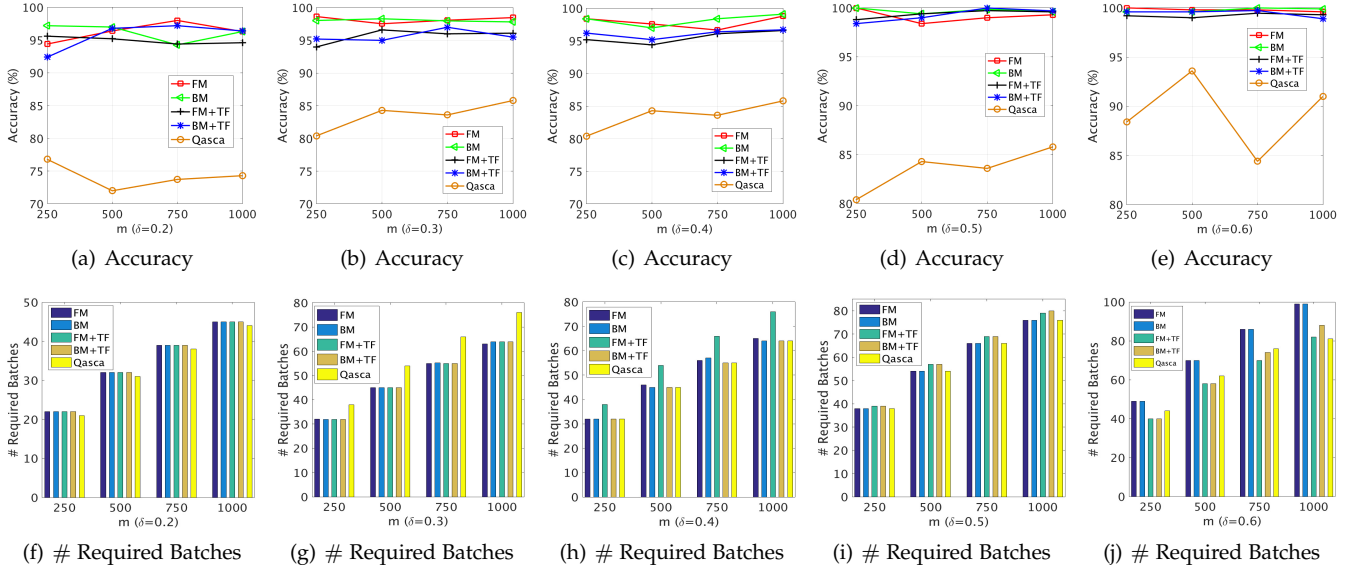


Fig. 2: The Effect of  $m$  on Synthetic Data

our parameter estimation model in both FM and BM with TRUTHFINDER (TF) [24] to evaluate the improvement of our proposed model, which are represented as FM+TF and BM+TF, respectively. In FM+TF and BM+TF, the parameter estimation model TF simply contains the inference between worker expertise and answer confidence (the left cycle), the inference between answer confidence and question easiness is removed and treated as an external unit that is calculated for one iteration after TF reaches a steady state. The above settings of FM+TF and BM+TF are identical with those in FM and BM. The settings of other important parameters are revealed in Table 3. For FM, BM, FM+TF and BM+TF, we vary  $\delta$  from 0.2 to 0.6 ( $\delta = 0.3$  means that the answer confidences have to reach 0.65 and 0.35). Answers with higher confidence will be returned as the truths. For Qasca, the number of repetitions of every question has to be fixed [32]. And we set  $rep = 1$  for  $\delta = 0.2$ ,  $rep = 3$  for  $\delta = 0.3, 0.4, 0.5$  and  $rep = 4$  for  $\delta = 0.6$ . The reason is that when we set  $\delta = 0.3$  (for example), a question in FM, BM, FM+TF, and BM+TF is assigned to 3 workers on average. Therefore,  $rep = 3$  makes sure that the total number of repetitions (or assignment) is identical for all solutions, thus the comparisons of the overall batches and accuracy are fair (the same reason for other cases of  $\delta$  and  $rep$ ). In addition,  $k = 1$  ensures the case that a worker in Qasca is assigned to 1 question, which is the same as FM, BM, FM+TF, and BM+TF. The truths of Qasca are inferred using EM algorithm. For the above five methods, the worker processing time is set to one batch per question, and the worker arriving rate ( $\lambda$ ) is set to one worker per batch.

### 5.3.2 Experimental Results

**Accuracy.** To compare the accuracy of the five methods, the number of questions  $m$  varies from 250 to 1000,  $\delta$  varies from 0.2 to 0.6, and  $\lambda = 1$ . The results are shown in Figure 2(a)-2(e). From the above five figures, we can observe that for each possible value of  $\delta$ , our methods FM and BM generally maintain a high accuracy within 95%-100% even when the quantity of questions is small ( $m = 250$ ); Furthermore,

our approaches have outperformed Qasca whose highest accuracy is less than 95%. Although FM+TF, BM+TF have high accuracy above 90%, their highest accuracy is lower than ours for most cases, especially when  $\delta = 0.3$  and 0.4. There is small increase of accuracy when  $m$  becomes larger.

Besides, as  $\delta$  increases, the accuracy of all methods regarding the same  $m$  (number of questions) is improved, which exactly proves that  $\delta$  controls the quality of the returned answers. A higher value of  $\delta$  indicates a higher accuracy of the returned answers.

**Number of required batches.** Next, we study the latency of each method depicted in Figure 2(f)-2(j) that measured in number of batches. As  $\delta$  increases, the number of required batches of all methods regarding the same  $m$  (the number of questions) is increased. From the analysis above, we know that  $\delta$  controls the quality of the returned answers and a higher value of  $\delta$  indicates that the returned answers have higher accuracy. In order to maintain an overall higher accuracy of questions, all methods choose to assign more repetitions of questions to workers and more certain answers can be obtained, which results in a longer latency (the number of required batches).

More importantly, when  $\delta$  is ranging from 0.2 to 0.5 (Figure 2(f)-2(i)), the latency of FM and BM is smaller than that of Qasca, FM+TF and BM+TF for different numbers of questions for most cases (sometimes they have similar values). Combining Figure 2(a)-2(d) where the accuracy of our methods is better than those of others, we can conclude that when  $\delta$  is ranging from 0.2 to 0.5, FM and BM are able to balance the latency and accuracy and they perform better than Qasca, FM+TF and BM+TF. In other words, the accuracy is improved without sacrificing the latency, namely, assigning more repetitions to workers to increase the quality. When  $\delta = 0.6$ , FM and BM have longer latency than that of other methods (Figure 2(j)), but the accuracy is still the highest (Figure 2(e)). This also reveals that when  $\delta$  has a high value like 0.6, the increase of accuracy for FM and BM tends to be smaller while the latency may be larger than that of others. Therefore, in latter experiments, we set  $\delta = 0.3$  as the default value (since the accuracy is

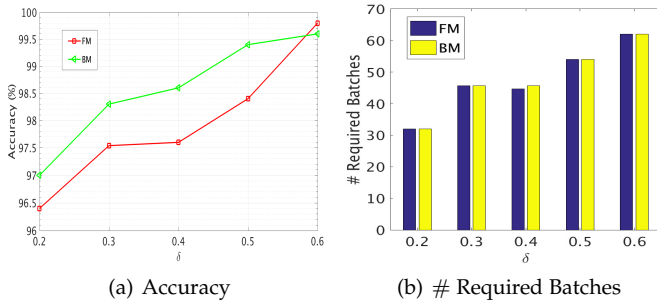


Fig. 3: The Effect of  $\delta$  on Synthetic Data

already high, and starting from it, the increase of accuracy becomes smaller but the overall latency will become larger), and  $rep = 3$  accordingly. Besides, with the increase of the number  $m$  of questions, the growth of required batches becomes slower, which shows that our methods are effective to apply to question pools of large quantities.

**The effect of value  $\delta$ .** Note that some observations related to  $\delta$  are discussed above. In this experiment, to better compare the performance of FM and BM, we set  $m = 500$  and vary  $\delta$  from 0.2 to 0.6. The results are shown in Figure 3(a) and 3(b). From Figure 3(a), as we increase the standard of returned questions, the easiness score threshold  $\delta$  is enlarged, BM dominates FM in terms of the accuracy in most cases. The reason is that BM always chooses the best worker for a question, whereas FM simply assigns a worker to the first open question. By checking Figure 3(b), we find that BM maintains a similar latency to FM while achieving higher accuracy than FM. Most importantly, this experiment proves that the setting of  $\delta$  is useful and beneficial for the requesters to control the accuracy of returned answers.

**Steady state of the parameter estimation model.** In this experiment, we let  $m = 500$  and  $\delta = 0.3$ , and we aim at studying the speed of the parameter estimation model to reach the steady state (convergence). The left and right cycles of the parameter estimation model are executed for 5 iterations respectively, and then this process is repeated for 20 iterations. The confidence of all the answers are tracked and the average variation in each iteration are drawn in Figure 4. From the figure, we can observe that the changes of the answer confidence decline at a fast speed for the first 5 iterations and extremely approach 0 starting from the 7th iteration. This discovery shows the effectiveness of the model and guides us to set an appropriate value for the # iterations. We simply fix the number of external iterations to 10 which is enough for us to get a stable answer confidence and control the latency caused by transitions between batches.

## 5.4 Experiments on Real Data

### 5.4.1 Experimental Settings

First of all, we extract the tuples with each containing one worker and a list of his/her question submission times from the Figure Eight dataset. Since the time span of the whole dataset lasts for 20 days and the questions are simple, the worker processing times are neglected. Therefore, the question submission times are approximated as the worker arrival times. For 39 workers in the Duck Identification

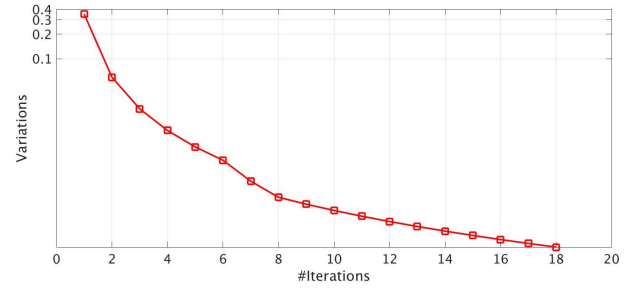


Fig. 4: Variations of Answer Confidence.

dataset, we randomly match each of them with one worker in the Figure Eight dataset to monitor their arrival times. This mapping method is also adopted in [4]. In order to yield persuasive results, for FM, BM and Qasca, we repeat the random matching process and conduct question assignment for 100 times, and then the accuracy and the number of required batches are averaged. After the mapping is finished, the whole time span of the workers is evenly divided into 20 batches by default.

The parameter settings are shown in Table 4. In addition, settings of the parameter estimation model are exactly the same as those in the synthetic experiments. Furthermore,  $\delta$  (easiness score threshold) is from 0.25 to 0.45, and 0.3 is picked as the default value (the reason is indicated in the first experiment on synthetic dataset);  $rep = 3$  and  $k = 1$  for Qasca;  $m$  (number of questions) varies from 20 to 100 and the default value is 40.

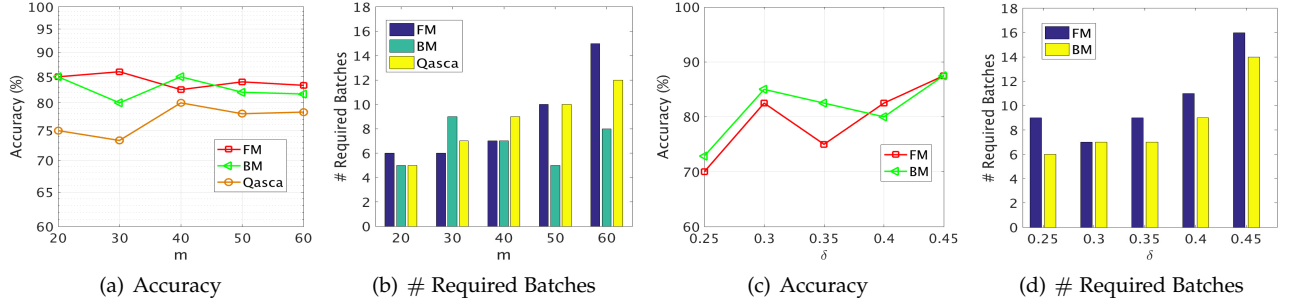
TABLE 4: Parameter Settings for Experiments on Real Data

Parameter	Description	Value
$m$	# questions	20, 30, 40, 50, 60
$\delta$	easiness score threshold	0.25, 0.3, 0.35, 0.4, 0.45
$b$	# total batches	20, 200, 2K, 20K
$rep$	# repetitions in qasca	3
$k$	top- $k$ tasks in qasca	1

### 5.4.2 Experimental Results

**Accuracy and number of required batches.** To evaluate the accuracy and the number of required batches for different numbers of questions, the numbers of questions are set from 20 to 60 and the easiness score threshold  $\delta$  is 0.3. From the results in Figure 5(a), FM and BM have higher accuracy than that of Qasca. Furthermore, the number of required batches (latency) are recorded in Figure 5(b). It is noticed that the latency increases as more questions are released for the workers. And FM and BM achieve smaller overall latency than Qasca for most of the time. These two experiments (combining the experiment in Figure 5(a)) together prove that our MCQ approaches have the ability to balance the latency and the accuracy on real data, namely, the resulted high accuracy is not caused by assigning more repetitions to workers.

**The effect of value  $\delta$ .** In this experiment, we set  $m = 40$  and vary  $\delta$  from 0.25 to 0.45. From Figure 5(c), as  $\delta$  becomes larger, the accuracy of FM and BM are generally increasing. The reason is that as  $\delta$  increases, an answer can only be returned when reaching a higher confidence, which provides a guarantee of quality for the returned answers. Combining the observation in Figure 5(d) where BM completes all questions in a smaller quantity of batches for most cases,

Fig. 5: The Effects of  $m$  and  $\delta$  on Real Data

BM performs better than FM. Therefore, we conclude that as  $\delta$  increases, BM becomes more advantageous to provide higher accuracy and better latency than FM.

**The effect of value  $b$ .** Next, we investigate the effect of value  $b$ , namely, the number of total batches.  $b$  is set to 20 by default but in this experiment, it is set to 4 different values from 20 to 20K. The results are shown in Figure 6. Note that  $b$  is the number of overall batches configured by the requesters while # required batches refers to the overall batches needed to complete all questions. In Figure 6(a) and Figure 6(b), compared to FM and Qasca, BM achieves the highest accuracy and shortest latency for most of the cases. Furthermore, as  $b$  increases, the required batches of FM and BM are increased and their accuracy also drops. This problem is not caused by our approaches. It can be explained by the fact that the average number of workers in a batch will be extremely small or even equal to 0 if we divide the time span into substantial batches. In this case, our approaches are similar to the online scenarios since the joint benefits of workers cannot be considered, which results in a less satisfying accuracy. What's more, since the knowledge of workers is hard to obtain for the initial batches, the answer confidence will take a longer time to reach a steady state, thus the delay is enlarged. Therefore, how to choose a proper batch interval is very important.

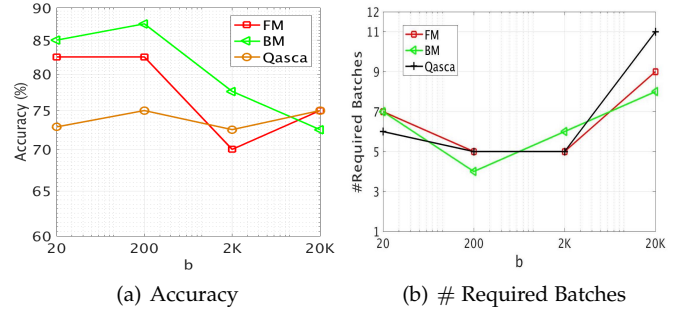
## 5.5 Summary

We finally summarize our experimental findings as follows.

- In most experiments on synthetic and real datasets, our proposed MCQ solutions FM and BM have better performance than the baseline Qasca in terms of the accuracy and overall latency. We conjecture that our solutions in the synchronized task assignment can improve the accuracy of answers while maintaining an acceptable latency.
- The BM solution assigns workers to questions with the largest increase of easiness score and it outperforms FM in most cases.
- The iterative parameter estimation model has a fast speed to reach a steady state. Furthermore, it performs better than TRUTHFINDER framework in the synchronized task assignment scenarios.
- The introduction of easiness score threshold  $\delta$  effectively controls the accuracy of the returned answers.

## 6 RELATED WORK

**Parameter Estimation.** In crowdsourcing, parameter estimation is widely used to infer latent variables such as worker

Fig. 6: The Effect of  $b$  on Real Data

quality and task truths. For example, [24] proposes a so-called TRUTHFINDER framework to discover truths over conflicting Internet information, and parameters like worker expertise is iteratively computed until convergence; based on [24], Dong et al. [7] leverage Bayesian analysis to estimate the accuracy of a data source by taking the influence of source dependence into consideration. [22], [14], [8], [13], [20], [29], [30] devise their truth inference model based on an Expectation-Maximization (EM) method [6]. Specifically, Peter Welinder et al. [22] utilize the characteristics of images and the multidimensional entities with variables representing the skills/knowledge of annotators, then propose a model to accurately predict ground truth labels of images. David R. Karger et al. [14] focus on minimizing the total price to achieve a target overall reliability. Ju Fan et al. [8] exploit the relations between tasks to propose a graph-based estimation model to estimate the accuracies of tasks. Panagiotis et al. [13] incorporate cost-sensitive classification errors in the overall framework and seamlessly integrate unsupervised and supervised techniques for inferring the quality of the workers. Caihua Shan et al. [20] integrate each worker's answers on different attributes to effectively learn his/her trustworthiness and the true data values to fill missing values for tabular data. Yudian Zheng et al. [29] propose methods to solve Jury Selection Problem (JSP) based on Bayesian Voting as it can reach the optimality of JSP. Yudian Zheng et al. [30] propose a DOCS framework to estimate the domains of tasks to guide the task assignment such that the truth inference can be accurate and efficient.

**Task Assignment** [5]. Recent work regarding online task assignment in general-purpose crowdsourcing mainly aims to provide a quality control on the achieved results [8], [15], [18], [32], [12]. For example, [12] investigates the accuracy related to the spatial distances between workers and loca-

tion of POIs to maximize the overall accuracy; [15] studies when to insert a gold standard question that helps estimate the worker accuracy and supports blocking of poor workers; [18] estimates the workers' accuracy according to their previous performance and the core quality-sensitive model is able to control the processing latency; [32] is the state-of-the-art work and it optimizes the evaluation metrics such as F-score on task selection and improves the previous work [18]. In addition, there are other online approaches. Yuan et al. [25] devise a task recommendation framework based on the unified probabilistic matrix factorization; [11] targets at a set of heterogeneous tasks and skill sets of workers are inspected over the tasks they have done.

Except for online task assignment, offline assignment strategies are also fully investigated in [28], [14], but they are less relevant to our work. To be specific, the parameters such as worker quality in our work are adaptively updated in batches without assuming prior distributions, whereas offline work like [28] retrieve them from the historical performance of workers in crowdsourcing platforms like AMT. Furthermore, the offline assignment is one shot, which cannot handle the case that worker arrives or leaves halfway after the questions are assigned.

Although TRUTHFINDER [24] has a good performance to infer website quality and truth of web information, it is run on aggregated facts of large quantity, whereas a crowdsourcing task has a few votes. Applying TRUTHFINDER to crowdsourcing will result in slow convergence and inaccurate answer confidence. In order to address this issue, question easiness is integrated into the framework that accelerates the inference and assists in the task assignment optimization. With the insertion of question easiness, the whole model is adapted to better fit in crowdsourcing circumstances. Furthermore, synchronized task assignment is popular nowadays [27]. Therefore, unlike most of the online task assignment that selects the top-k beneficial questions [32] to an individual worker, this work considers the synchronized task assignment scenario in which collaborative worker labor is efficiently utilized and the number of completed questions within each batch is maximized. In addition, compared to some offline task assignment work [28], our model requires no prior knowledge of worker behaviors and is practical to apply on open crowdsourcing platforms where workers are quite dynamic.

There are many excellent surveys/tutorials about crowdsourcing [16], [17], [3], which can give a more comprehensive picture of crowdsourcing. In [16], [17], the authors identify three important problems in crowdsourced data management: (1) Quality Control; (2) Cost Control; (3) Latency Control. They survey a wide spectrum of existing studies that address three factors through designing crowdsourced tasks, develop crowdsourced manipulation operators, and optimize plans consisting of multiple operators. In [3], the authors reviewed crowd-powered data mining operations, including classification, clustering, pattern mining, machine learning using the crowd and knowledge discovery, and they pointed out the emerging challenges in crowdsourced data mining.

## 7 CONCLUSION

In this paper, we formally propose the MCQ problem in synchronized task assignment scenario, which assigns questions to workers to make the most efficient utilization of worker labor and maximize the number of completed questions in each batch. The problem is proved to be NP-hard and two greedy approximation solutions are proposed to address the problem. Furthermore, we develop an efficient parameter estimation model to assist in the task assignment optimization. Extensive experiments on synthetic and real datasets are conducted to evaluate our approaches. The experimental results show that our approaches outperform the baseline and achieve a high accuracy while maintaining an acceptable latency.

## 8 ACKNOWLEDGMENT

The work is partially supported by the Hong Kong RGC GRF Project 16207617, the National Science Foundation of China (NSFC) under Grant No. 61729201, Science and Technology Planning Project of Guangdong Province, China, No. 2015B010110006, Hong Kong ITC ITF grants ITS/470/18FX and ITS/212/16FP, Didi-HKUST joint research lab project, Microsoft Research Asia Collaborative Research Grant and Wechat Research Grant.

## REFERENCES

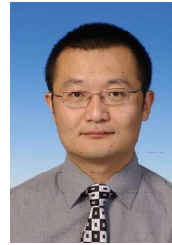
- [1] [online] data for everyone library — crowdflower. <https://www.figure-eight.com/data-for-everyone/>.
- [2] [online] truth inference in crowdsourcing. [http://i.cs.hku.hk/~ydzheng2/crowd\\_truth\\_inference/index.html#Downloads](http://i.cs.hku.hk/~ydzheng2/crowd_truth_inference/index.html#Downloads).
- [3] C. Chai, J. Fan, G. Li, J. Wang, and Y. Zheng. Crowd-powered data mining (tutorial). *KDD*, 2018.
- [4] P. Cheng, H. Xin, and L. Chen. Utility-aware ridesharing on road networks. In *Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD '17*, pages 1197–1210, New York, NY, USA, 2017. ACM.
- [5] A. I. Chittilappilly, L. Chen, and S. Amer-Yahia. A survey of general-purpose crowdsourcing techniques. *IEEE Transactions on Knowledge and Data Engineering*, 28(9):2246–2266, 2016.
- [6] A. P. Dawid and A. M. Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Applied statistics*, pages 20–28, 1979.
- [7] X. L. Dong, L. Berti-Equille, and D. Srivastava. Integrating conflicting data: the role of source dependence. *Proceedings of the VLDB Endowment*, 2(1):550–561, 2009.
- [8] J. Fan, G. Li, B. C. Ooi, K.-I. Tan, and J. Feng. icrowd: An adaptive crowdsourcing framework. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1015–1030. ACM, 2015.
- [9] D. Haas, J. Wang, E. Wu, and M. J. Franklin. Clamshell: Speeding up crowds for low-latency data labeling. *Proceedings of the VLDB Endowment*, 9(4):372–383, 2015.
- [10] M. Hirth, T. Hoffeld, and P. Tran-Gia. Analyzing costs and accuracy of validation mechanisms for crowdsourcing platforms. *Mathematical and Computer Modelling*, 57(11):2918–2932, 2013.
- [11] C.-J. Ho and J. W. Vaughan. Online task assignment in crowdsourcing markets. In *AAAI*, volume 12, pages 45–51, 2012.
- [12] H. Hu, Y. Zheng, Z. Bao, G. Li, J. Feng, and R. Cheng. Crowd-sourced poi labelling: Location-aware result inference and task assignment. In *Data Engineering (ICDE), 2016 IEEE 32nd International Conference on*, pages 61–72. IEEE, 2016.
- [13] P. G. Ipeirotis, F. Provost, and J. Wang. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 64–67. ACM, 2010.
- [14] D. R. Karger, S. Oh, and D. Shah. Iterative learning for reliable crowdsourcing systems. In *Advances in neural information processing systems*, pages 1953–1961, 2011.



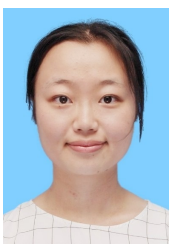
- [15] A. R. Khan and H. Garcia-Molina. Crowddqs: Dynamic question selection in crowdsourcing systems. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1447–1462. ACM, 2017.
- [16] G. Li, J. Wang, Y. Zheng, and M. J. Franklin. Crowdsourced data management: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 28(9):2296–2319, 2016.
- [17] G. Li, Y. Zheng, J. Fan, J. Wang, and R. Cheng. Crowdsourced data management: Overview and challenges. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1711–1716. ACM, 2017.
- [18] X. Liu, M. Lu, B. C. Ooi, Y. Shen, S. Wu, and M. Zhang. Cdas: a crowdsourcing data analytics system. *Proceedings of the VLDB Endowment*, 5(10):1040–1051, 2012.
- [19] R. G. Michael and S. J. David. Computers and intractability: a guide to the theory of np-completeness. *WH Free. Co., San Fr*, pages 90–91, 1979.
- [20] C. Shan, N. Mamoulis, G. Li, R. Cheng, Z. Huang, and Y. Zheng. T-crowd: Effective crowdsourcing for tabular data. *ICDE*, 2018.
- [21] J. Wang, G. Li, T. Kraska, M. J. Franklin, and J. Feng. Leveraging transitive relations for crowdsourced joins. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, pages 229–240. ACM, 2013.
- [22] P. Welinder, S. Branson, P. Perona, and S. J. Belongie. The multi-dimensional wisdom of crowds. In *Advances in neural information processing systems*, pages 2424–2432, 2010.
- [23] J. Whitehill, T.-f. Wu, J. Bergsma, J. R. Movellan, and P. L. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in neural information processing systems*, pages 2035–2043, 2009.
- [24] X. Yin, J. Han, and S. Y. Philip. Truth discovery with multiple conflicting information providers on the web. *IEEE Transactions on Knowledge and Data Engineering*, 20(6):796–808, 2008.
- [25] M.-C. Yuen, I. King, and K.-S. Leung. Taskrec: A task recommendation framework in crowdsourcing systems. *Neural Processing Letters*, 41(2):223–238, 2015.
- [26] C. J. Zhang, L. Chen, H. Jagadish, and C. C. Cao. Reducing uncertainty of schema matching via crowdsourcing. *Proceedings of the VLDB Endowment*, 6(9):757–768, 2013.
- [27] L. Zhang, T. Hu, Y. Min, G. Wu, J. Zhang, P. Feng, P. Gong, and J. Ye. A taxi order dispatch model based on combinatorial optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '17*, pages 2151–2159, New York, NY, USA, 2017. ACM.
- [28] Z. Zhao, F. Wei, M. Zhou, W. Chen, and W. Ng. Crowd-selection query processing in crowdsourcing databases: A task-driven approach. In *EDBT*, pages 397–408, 2015.
- [29] Y. Zheng, R. Cheng, S. Maniu, and L. Mo. On optimality of jury selection in crowdsourcing. In *Proceedings of the 18th International Conference on Extending Database Technology, EDBT 2015*. OpenProceedings.org., 2015.
- [30] Y. Zheng, G. Li, and R. Cheng. Docs: a domain-aware crowdsourcing system using knowledge bases. *Proceedings of the VLDB Endowment*, 10(4):361–372, 2016.
- [31] Y. Zheng, G. Li, Y. Li, C. Shan, and R. Cheng. Truth inference in crowdsourcing: Is the problem solved? *Proc. VLDB Endow.*, 10(5):541–552, Jan. 2017.
- [32] Y. Zheng, J. Wang, G. Li, R. Cheng, and J. Feng. Qasca: A quality-aware task assignment system for crowdsourcing applications. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1031–1046. ACM, 2015.



**Peng Cheng** received his BS degree and MA degree in Software Engineering in 2012 and 2014, from Xi'an Jiaotong University, China, and the PhD degree in Computer Science and Engineering from the Hong Kong University of Science and Technology (HKUST) in 2017. He is now a researcher in the School of Computer Science and Software Engineering at East China Normal University. His research interests include crowdsourcing, spatial crowdsourcing, ridesharing and graph databases.



**Lei Chen** received his BS degree in Computer Science and Engineering from Tianjin University, China, in 1994, the MA degree from Asian Institute of Technology, Thailand, in 1997, and the PhD degree in computer science from University of Waterloo, Canada, in 2005. He is now a professor in the Department of Computer Science and Engineering at Hong Kong University of Science and Technology. His research interests include crowdsourcing, uncertain and probabilistic databases, multimedia and time series databases, and privacy. He is a member of the IEEE.



**Jiayang Tu** received both her BS and MPhil degrees in Computer Engineering, from Hong Kong University of Science and Technology, China. Her research interests include crowdsourcing and knowledge graph. She is now a research engineer in Lenovo Machine Intelligence Center, Hong Kong, working on data fusion, pattern mining and knowledge reasoning in knowledge graph.