Contents lists available at ScienceDirect
Information Sciences
journal homepage: www.elsevier.com/locate/ins

Privacy-preserving task recommendation with win-win incentives for mobile crowdsourcing[☆]

Wenjuan Tang[a], Kuan Zhang[b], Ju Ren[a,*], Yaoxue Zhang[a], Xuemin (Sherman) Shen[c]

[a] *School of Computer Science and Engineering, Central South University, Changsha, China*
[b] *Department of Electrical and Computer Engineering, University Of Nebraska-Lincoln, Lincoln, NE, USA*
[c] *Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada*

A R T I C L E I N F O

Article history:
Received 30 May 2018
Revised 28 January 2019
Accepted 7 February 2019
Available online xxx

Keywords:
Crowdsourcing
Attribute-based encryption
Privacy-preservation
Win-win incentives
Outsourced decryption

A B S T R A C T

Mobile crowdsourcing enables mobile requesters to publish tasks, which can be accomplished by workers with awards. However, existing task allocation schemes face tradeoff between effectiveness and privacy preservation, and most of them lack consideration of win-win incentives for both requesters and workers participation. In this paper, we propose a privacy-preserving task recommendation scheme with win-win incentives in crowdsourcing through developing advanced attribute-based encryption with preparation/online encryption and outsourced decryption technologies. Specifically, we design bipartite matching between published tasks and participant workers, to recommend tasks for eligible workers with interests and provide valuable task accomplishment for requesters in a win-win manner. Furthermore, our scheme reduces encryption cost for requesters by splitting encryption into preparation and online phases, as well as shifts most of the decryption overhead from the worker side to the service platform. Privacy analysis demonstrates requester and worker privacy preservation under chosen-keyword attack and chosen-plaintext attack. Performance evaluation shows cost-efficient computation overhead for requesters and workers.

1. Introduction

With rapid development of information technology, mobile crowdsourcing becomes a promising paradigm for requesters to publish tasks beyond their abilities, which can be accomplished by workers with their resource contribution to obtain awards [21]. Since mobile crowdsourcing demonstrates several outstanding advantages: large coverage, low cost, mobile capability, and flexible interactive capability, it attracts amount of requesters and workers to participate in the system for resource reconciliation and exchange. Requesters publish their tasks to the service platform, and utilize the crowd intelligence to complete tasks that they lack sufficient resources to accomplish. Workers learn the task content, and contribute their capabilities of data sensing, computation and communication for task accomplishment to earn rewards. However, the complex task requests (e.g., knowledge base construction, image labeling and language translation) also bring critical task

☆ special issue name edited by Prof. W. Pedrycz.
* Corresponding author.
E-mail address: renju@csu.edu.cn (J. Ren).

https://doi.org/10.1016/j.ins.2019.02.011
0020-0255/© 2019 Elsevier Inc. All rights reserved.

allocation challenges for the flourish of mobile crowdsourcing as follows [40]. First, for workers, facing huge amount of heterogenous tasks, it is time-consuming to select suitable tasks that they are interested in and within their capabilities to accomplish. Especially, if the worker's contribution fails to be received and awarded by requesters, not only his nontrivial intelligence and time, but also significant resources (e.g., computing power and battery) may be wasted [6]. Second, for requesters, the task accomplishment quality can not be guaranteed since workers may reveal different computing behaviors towards different tasks [3]. If their published tasks cannot be received and accomplished by eligible workers efficiently, they cannot obtain expected valuable task accomplishment and may be unwilling to participate in the crowdsourcing system. As a result, achieving task allocation is essential for both attracting the user's further participation and improving the crowdsourcing system's effectiveness [1,2].

A straightforward method to realize effective task allocation is to learn the characteristics of tasks and workers, and allocate corresponding tasks to suitable workers according to the collected information, such as the historical behaviors of the worker and the correlations between different tasks [30]. Through exploiting devices of the worker to analyze the description of tasks and the personal historical behaviors, unsuitable tasks can be filtered out [9]. However, this approach faces huge computation and communication overhead for resource-constrained mobile devices since complex computation is required for task analysis, and workers' devices should receive all of the tasks' descriptions at least before learning whether these tasks match workers' interests. Benefiting from the powerful computation capability of the crowdsourcing server, the effective task allocation can be performed on the service platform [23,39]. For effective task allocation, incentives (aims) of requesters and workers are different [12,20]. From the aspect of a requester, his goal is not to outsource the task to anyone in the system, but to obtain data contributions responding to this specific task. The data contribution relies on two parts: the capability of the worker (if his capability satisfies the task requirement objectively) and the behaviors of the worker (if he has interests to perform the task subjectively) [22,37]. From the aspect of worker, his goal is to obtain awards, i.e., his data contribution can satisfy the task requirement, by performing his interested tasks (e.g., within his interested topics or location). Through collecting workers' information (such as profile, location and interests) [14,34], as well as requirement descriptions of tasks, the service platform can actively recommend suitable tasks to eligible workers. In this way, the requester can receive valuable data contributions with an efficient cost, and the worker can efficiently accomplish suitable tasks to obtain deserved benefits.

However, since the service platform is generally managed by a third-party, the task recommendation in crowdsourcing systems faces privacy challenges through collecting the sensitive information from workers and tasks [18,19,26,33]. Firstly, the privacy of a requester may be endangered because the task he publishes may reveal some sensitive information, as well as potential privacy leakage from their task descriptions [36,38]. For example, if a requester publishes crowdsourcing tasks that can only be fulfilled by psychologists, the service platform can infer this requester may suffer from some psychological diseases. Secondly, the privacy of a worker may be disclosed because his interests in tasks may reveal his sensitive information. For example, the worker sets his interested tasks be accomplished within a specific university, the service platform may infer that the worker may be nearby the specific location [33]. To preserve privacy for both requesters and workers, the task information and the identity related information of requesters and workers should be hidden from the service platform.

To address the privacy concerns for task recommendation, a promising approach is to employ cryptographic techniques. Anonymizing data or employing pseudonyms with hash function and signature techniques are widely proposed in statistical queries to protect the identity or the location of the worker [27,28]. However, workers may be de-anonymized with auxiliary information [15], as well as the requester personal information (e.g., preferences) can be linked to deduce the privacy. Exploiting secure search method that originally applied on encrypted outsourced data is the other kind scheme for privacy preserving task recommendation [16]. With this scheme, the worker can find his interested tasks; meanwhile, the privacy of worker interests as well as the descriptions of the task can be preserved. However, this scheme can hardly guarantee valuable task accomplishment for the requester, hindering the flourish of the crowdsourcing system. Above all, the task recommendation that can benefit both requesters and the workers as well as preserve participants' privacy is required.

In this paper, we propose a privacy-preserving task recommendation scheme with win-win incentives for crowdsourcing systems inspired by the dual policy attribute-based encryption [4,17]. In our scheme, the requester can encrypt the task with an access policy that indicate the task requirements, as well as the worker can define an access policy to demonstrate his interested tasks. The service platform analyzes both their access policies and recommend suitable tasks to workers through bipartite matching. Specifically, we list the contributions of our work as follows:

- We propose an effective task recommendation framework with win-win incentives for crowdsourcing systems. Through bipartite matching between the task and worker, both the requester and the worker benefit from the task recommendation. For the requester, he can obtain valuable data contribution from the eligible worker with efficient time and cost. For the worker, he can receive his interested tasks to perform, and obtain awards without wasting his contribution resources.
- We develop advanced attribute-based encryption to preserve the privacy for the requester and the worker. Although the interests of the worker and the task information are collected for matching analysis and task recommendation, we develop advanced attribute-based encryption to keep them confidential from the service platform. The capability, interest and task tag information is analyzed to be privacy preserving under Decisional Bilinear Diffie-Hellman assumption. Meanwhile, the task content can only be accessed by the authorized worker, whose capability attributes should satisfy the task requirement policy, and interest policy should match the task tag keywords.

- We shift most portions of the task encryption to the preparation process, which can be operated once and utilized for all the task encryption, thus the task encryption overhead is reduced for the requester. We utilize key capsulation method for the task decryption key, which can offload most of the task decryption cost to the service platform, such that the decryption overhead on the worker is cost-efficient.

The remainder of this paper is organized as follows. Section 2 reviews the related works on task recommendation and privacy preservation for crowdsourcing, and Section 3 introduces preliminaries. Then, we present models and goals in Section 3. Our scheme details can be seen in Section 4. The privacy discussions and performance evaluation are presented in Sections 5 and 6 respectively, followed by a conclusion in Sections 7.

2. Related works

In this section, we survey the related works from two aspects: task recommendation and privacy preservation for crowdsourcing systems.

2.1. Task recommendation

In crowdsourcing systems, task recommendation can help workers to find suitable tasks faster as well as help requesters to receive valuable task accomplishments. Yuen et al. [35] proposed a task recommendation framework based on a unified probabilistic matrix factorization, aiming to recommend tasks to workers in dynamic scenarios. Yang et al. [25] proposed a personalized task recommender system for mobile crowdsensing, which recommended tasks to users based on a recommendation score that jointly takes each user's preference and reliability into consideration. Merkourios et al. [10] adopted logistic regression techniques to estimate a user's probability of accepting a task, and tended to match tasks to users based on the information. Ari et al. [11] optimized the task recommendation by jointly maximizing the user longevity and user participation, as well as analyzing interests of contributing participants. Wang et al. [32] consider time-sensitive and location-dependent crowdsensing systems with random arrivals in crowdsourcing systems, and propose a two-level heterogeneous pricing mechanism to motivate the participation of participants. To consider online scenarios, some works are proposed to allocate the tasks to suitable workers in real-time dynamic environments. Zhang et al. [37] consider users' interest to tasks, and design online incentive mechanisms, to peruse platform utility, and crucial property of truthfulness. Ho et al. [8] learn the sequence of workers and their skills in advance, to propose an exploration-exploitation algorithm by applying online primal-dual techniques for accurate task assignment. Tong et al. [29] online define a micro-task allocation problem in spatial crowdsourcing, conducts a greedy strategy to assign each new arrival task to the corresponding worker with the highest utility.

2.2. Privacy preservation

Miao et al. [16] proposed a basic outsourced ciphertext-policy attribute-based key encapsulation mechanism to preserve participants' privacy. Yang et al. [33] designed a privacy-preserving location matching mechanism to determine whether a worker is in geocast region of a spatial task or not without any knowledge about the task's geocast region and the worker's location based on Lagrange Interpolating Polynomials. Cheny and Co-authors [5,7] developed a privacy-aware optimization model of task selection that considers the intrinsic tradeoffs among utility, privacy and efficiency. Wang et al. [31] propose a personalized privacy-preserving task allocation framework for mobile crowdsensing. Through uploading the obfuscated distances and personal privacy level to the server instead of its true locations or distances, this scheme can allocate tasks effectively while providing personalized location privacy protection. Liu et al. [13] propose to reduce the unnecessary location privacy loss and maintain required quality of service (QoS), by employing economic theory to assist both the service provider and participants to decide their strategies. Zhuo et al. [41] preserve identity privacy and data privacy for participants during data aggregation and analysis in crowdsourcing systems, as well as verify the correctness of computation results.

However, existing works lacked to consider privacy preservation and win-win incentives in performing the task recommendation simultaneously, which is significant for the flourish of the mobile crowdsourcing system.

3. Models and goals

3.1. System model

Our system model is shown in Fig. 1. It consists of the following entities: Authority, Service Platform, Requester and Worker.

Authority. The authority initializes the system, provides registration services for both requesters and workers.

Service platform. The crowdsourcing service platform receives and stores the task ciphertext, as well as the capability and interest ciphertext from workers. It analyzes if the task matches the worker, and recommends suitable tasks for the worker. For efficiency of the system, the service platform also pre-decrypts the task ciphertext for the worker.

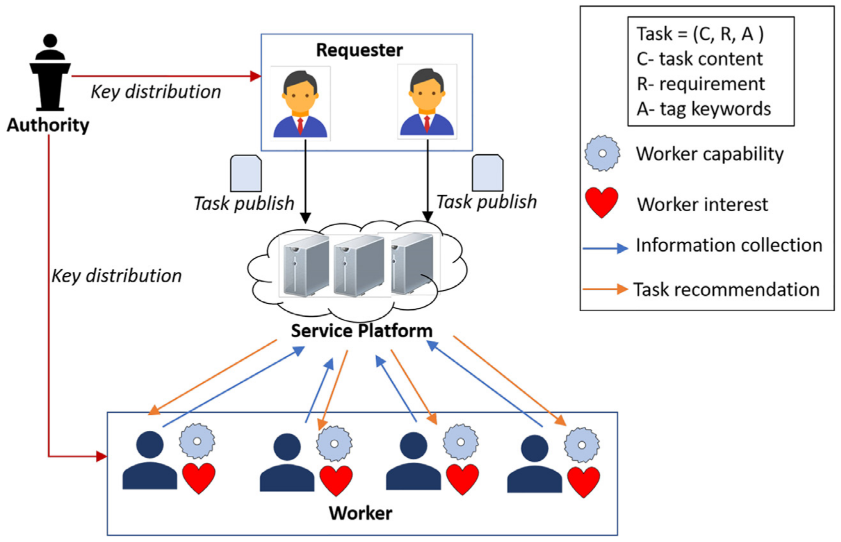


Fig. 1. System model.

Requester. The requester encrypts his task information under his requirement policy and sends his task ciphertext to the crowdsourcing service platform. The requester as well as encrypts corresponding tag attributes to his task, for enabling the task to be received by the interested worker.

Worker. The worker encrypts his interest policy with his interested attributes (e.g., preferred task category and expected payments), and sends the ciphertext to the service platform. When the worker receives suitable tasks that they are interested in and competent to perform, the worker decrypts the task ciphertext and perform the task.

3.2. Privacy model

In our task recommendation for mobile crowdsourcing, the authority is fully trusted in the system. The service platform is *honest-but-curious*. The service platform analyzes and matches the tasks to suitable workers, but it is curious about the task content and worker interest policy, intending to learn the privacy of the requester and worker. The requester is honest. The worker that perform the task is honest and does not disclose data privacy through a contract signed with the task requester.

For privacy-preserving task recommendation in our scheme, we preserve both the privacy for the worker and the requester.

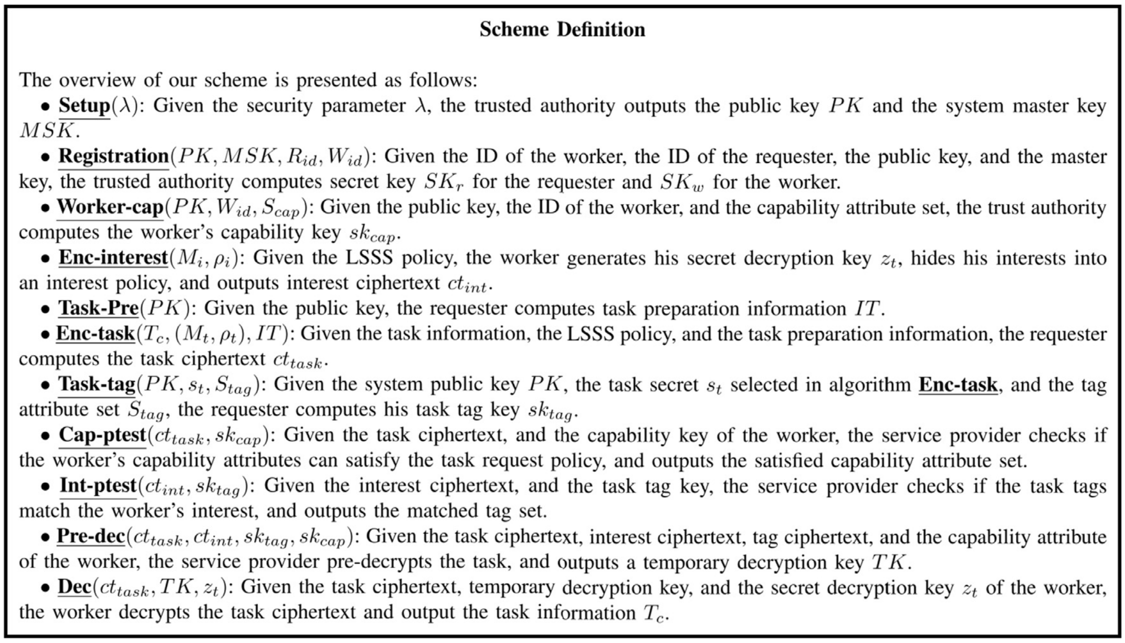
- *Worker privacy*. The information collected from the worker includes two parts: the capability keywords and the interest policy. (1) The capability keyword generation of the worker should not reveal any capability attribute information, and should be indistinguishable against chosen keyword attacks (IND-CKA). The capability keyword generation is IND-CKA secure if any adversary $\mathcal{A}$ cannot distinguish the capability keywords of two arbitrary capability attributes. (2) The interest policy of the worker should be IND-CKA and not reveal any interest attribute information, i.e., there is no adversary $\mathcal{A}$ can distinguish the interest policy of two arbitrary interest vectors.
- *Requester privacy*. The task information published by the requester that may reveal the requester privacy includes two parts: the task content ciphertext and the task tag keywords. (1) The task content ciphertext should only be decrypted by authorized workers. The task content encryption should be indistinguishable secure against chosen plaintext attacks (IND-CPA), i.e, there is no adversary $\mathcal{A}$ can distinguish the encryptions of two arbitrary data (chosen by $\mathcal{A}$), even if $\mathcal{A}$ can query secret keys. (2) The task tag keywords should be IND-CKA and not reveal any task tag attribute information, i.e., there is no adversary $\mathcal{A}$ can distinguish the task tag keywords of two arbitrary task tag attributes.

3.3. Design goals

(1) *Effective task recommendation*. Our scheme aims to match suitable tasks with requesters, and enable the tasks from requesters to be performed by capable and interested workers. In this way, the capable worker that perform the task can provide more qualified data contribution with a high probability, which can benefit the requester to obtain corresponding solutions with lower cost and time, as well as benefit the worker to receive rewards without wasting his contributed resources for task preformation. Meanwhile, the recommended task can fit the worker's interests, which can save much time and efforts in choosing suitable tasks from the amount of the heterogenous tasks for the worker.

Table 1
Notations.

Notation	Definition
R_{id}	ID number of requester
W_{id}	ID number of worker
sk_{cap}	Worker's capability key
(M_i, ρ_i)	Worker's interest policy
(M_t, ρ_t)	Task's requirement policy
ct_{int}	Worker's interest ciphertext
ct_{task}	Task content ciphertext
sk_{cap}	Task tag key
IT	Task preparation ciphertext
TK	Temporary decryption key



Scheme Definition

The overview of our scheme is presented as follows:

- **Setup**(λ): Given the security parameter λ, the trusted authority outputs the public key PK and the system master key MSK.
- **Registration**$(PK, MSK, R_{id}, W_{id})$: Given the ID of the worker, the ID of the requester, the public key, and the master key, the trusted authority computes secret key SK_r for the requester and SK_w for the worker.
- **Worker-cap**(PK, W_{id}, S_{cap}): Given the public key, the ID of the worker, and the capability attribute set, the trust authority computes the worker's capability key sk_{cap}.
- **Enc-interest**(M_i, ρ_i): Given the LSSS policy, the worker generates his secret decryption key z_t, hides his interests into an interest policy, and outputs interest ciphertext ct_{int}.
- **Task-Pre**(PK): Given the public key, the requester computes task preparation information IT.
- **Enc-task**$(T_c, (M_t, \rho_t), IT)$: Given the task information, the LSSS policy, and the task preparation information, the requester computes the task ciphertext ct_{task}.
- **Task-tag**(PK, s_t, S_{tag}): Given the system public key PK, the task secret s_t selected in algorithm **Enc-task**, and the tag attribute set S_{tag}, the requester computes his task tag key sk_{tag}.
- **Cap-ptest**(ct_{task}, sk_{cap}): Given the task ciphertext, and the capability key of the worker, the service provider checks if the worker's capability attributes can satisfy the task request policy, and outputs the satisfied capability attribute set.
- **Int-ptest**(ct_{int}, sk_{tag}): Given the interest ciphertext, and the task tag key, the service provider checks if the task tags match the worker's interest, and outputs the matched tag set.
- **Pre-dec**$(ct_{task}, ct_{int}, sk_{tag}, sk_{cap})$: Given the task ciphertext, interest ciphertext, tag ciphertext, and the capability attribute of the worker, the service provider pre-decrypts the task, and outputs a temporary decryption key TK.
- **Dec**(ct_{task}, TK, z_t): Given the task ciphertext, temporary decryption key, and the secret decryption key z_t of the worker, the worker decrypts the task ciphertext and output the task information T_c.

Fig. 2. Scheme definition.

(2) *Efficient computation overhead*. Our scheme aims to guarantee acceptable encryption and decryption consumption for the requester and worker. For privacy preservation, the task encryption and decryption may introduce extra resource consumption, which might hinder the requester and the worker to utilize the mobile crowdsourcing system. In this way, we should guarantee efficient task encryption for the requester and efficient task decryption for the worker.

4. Privacy-preserving task recommendation with win-win incentives

In this section, we present the details of our scheme: privacy-preserving task recommendation with win-win incentives. For easier illustration, we list some important notations and parameters in Table 1.

4.1. Scheme overview

Our scheme consists of several algorithms, namely **Setup**, **Registration**, **Worker-cap**, **Enc-interest**, **Task-pre**, **Enc-task**, **Task-tag**, **Cap-ptest**, **Int-ptest**, **Pre-dec**, and **Dec**, which is shown in Fig. 2.

Besides, we present our scheme process in Fig. 3. In step (1), the authority initiates the system, generates the public key *PK* and system master key *MSK*. The authority registers the worker and the requester, and generates the capability attribute keys for the worker. In step (2), the service platform collects the worker interest ciphertext. In step (3), the requester encrypts and sends the task content ciphertext with his requirement policy, and the task tag key to the service platform. In step (4), the service platform analyzes the task ciphertext and the interest ciphertext collected from the worker, to test if the worker's capability attributes satisfy the task requirement policy, and to test if the task tag attributes match the worker's interest policy. If both of them match with each other, the service platform recommends the task to the worker. In step (5), the worker sends the acceptation symbol to the service platform to show that he intends to perform the task. The service

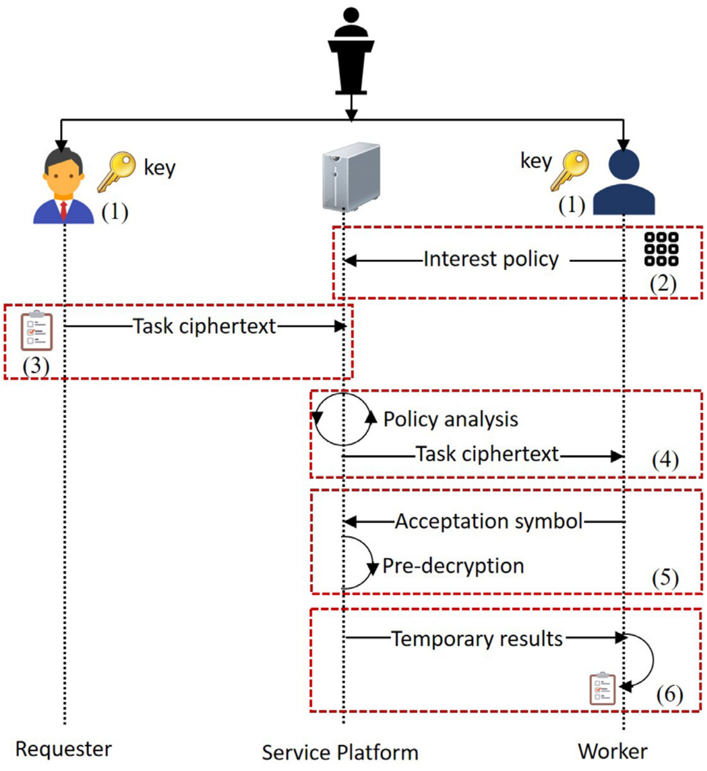


Fig. 3. Scheme process.

platform pre-decrypts the corresponding task with worker's capability attribute key and task tag key, and transmits the temporary results to the worker. In step (6), the worker decrypts the task ciphertext with his secret decryption key, and learns the task plaintext.

4.2. Scheme construction

We construct our scheme into following phases: system initialization, task publication, task recommendation, task pre-decryption and task decryption.

Phase 1 System initialization

The trusted authority initializes the system by running the **Setup** and **Registration** algorithm.

Setup $(\lambda)\to(PK, MSK)$: The trusted authority inputs universal attributes U and the security parameter λ, outputs the system public key PK and the system master key MSK. It chooses two multiplicative groups $\mathbb{G}$ with generator g and $\mathbb{G}_{\mathbb{T}}$ of prime order p, and a bilinear map $e:\mathbb{G}\times\mathbb{G}\to\mathbb{G}_{\mathbb{T}}$. The trusted authority picks the random numbers $g, u, h, w, v \in \mathbb{G}$ and a random exponent $\alpha \in \mathbb{Z}_p$. The trusted authority outputs the public key PK and the system master key MSK.

$$PK = g, u, h, w, v, g^{\alpha}, e(g, g)^{\alpha} \tag{1}$$

$$MSK = \alpha \tag{2}$$

Registration $(W_{id}, PK, MSK) \to (SK_W)$. The trusted authority provides registration services for workers and requesters.

When a worker joins in the crowdsourcing system, he provides W_{id} to the trusted authority. The trusted authority computes $K_{W0} = g^{\alpha} w^{W_{id}}$, $K_{W1} = g^{W_{id}}$, and sends $SK_W = (K_{W0}, K_{W1})$ to the worker.

Worker-cap $(PK, W_{id}, S_{cap} = (cap_1, cap_2, \ldots, cap_n)) \to sk_{cap}$. This algorithm takes the public key, worker's ID, and the capability attribute set as inputs. It picks n random exponents $r_1, r_2, \ldots, r_n \in Z_p$, then it computes for every $\iota \in [n]$

$$K_{W\iota,2} = g^{c_\iota} \textit{ and } K_{W\iota,3} = (u^{cap_\iota} h)^{r_\iota} v^{-W_{id}} \tag{3}$$

The worker capability key is $sk_{cap} = \{SK_W, K_{W\iota,2}, K_{W\iota,3}\}_{\iota\in[n]}$.

Phase 2 Worker interest collection

When a worker intends to receive his interested tasks, he encrypts his interest access policy by running the $Enc-interest$ algorithm and sends the outputs to the crowdsourcing platform.

Enc-interest $(M_i, \rho_i) \to ct_{int}$: This algorithm takes a LSSS policy, with $M_i \in \mathbb{Z}_p^{l\times n}$ and $\rho_i : [l] \to \mathbb{Z}_p$ as inputs. It first selects a decryption key $z_t \in \mathbb{Z}_p$ and an interest key $s_w \in \mathbb{Z}_p$, to compute a decryption key $K_0' = (g^{\alpha z_t}) w^{z_t s_w}$. It then selects

$\vec{y_w} = (y_2, \ldots, y_n)^\top \in \mathbb{Z}_p^{n-1}$. It computes vector $\vec{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_l)^\top = M_i \cdot (\frac{s_w}{z_t} + W_{id}, y_2, \ldots, y_n)^\top$. It then selects l random exponents $t_1, t_2, \ldots, t_l \in \mathbb{Z}_p$. It computes $C_{W0} = g^{s_w}, \tilde{C_{W0}} = g^{\alpha z_t} w^{s_w}$, for every $\tau \in [l]$, it encrypts the interest policy

$$C_{W\tau,1} = w^{\lambda_\tau} v^{t_\tau},\ C_{W\tau,2} = (u^{\rho(\tau)} h)^{-t_\tau}$$
$$C_{W\tau,3} = g^{t_\tau},\ C_{W\tau,4} = w^{\lambda_\tau} \tag{4}$$

The worker's interest ciphertext is:
$ct_{int} = (K'_0, C_{W0}, (C_{W\tau,1}, C_{W\tau,2}, C_{W\tau,3}, C_{W\tau,4})_{\tau\in[l]})$

Phase 3 Task publication

Since a requester may have amount of tasks to publish, the requester can generate a task encryption preparation ciphertext first by running the $Task - pre$ algorithm. To keep the task privacy as well as guarantee the task can be accomplished with high-quality data contribution from capable workers, the requester defines a task requirement policy to encrypt the task by running the $Enc - task$ and $Task - tag$ algorithm. The $Enc - task$ algorithm is used to encrypt the task content, and the $Task - tag$ is used to encrypt the task tags.

Task-Pre $(PK) \to IT$: Let P be the maximum bound rows in any LSSS access structure used in a ciphertext. For $i = 1$ to P, the requester chooses a random $\lambda'_j, x_j, t_j \in \mathbb{Z}_p$, and computes $\{C_{Ti,1}, C_{Ti,2}, C_{Ti,3}\}_{i\in[1,P]}$.

$$C_{Tj,1} = w^{\lambda'_j} v^{t_j},\ C_{Tj,2} = (u^{x_j} h)^{-t_j},\ and\ C_{Tj,3} = g^{t_j} \tag{5}$$

Let Q be the maximum task tags in the crowdsourcing system, and the task tags are $\{tag_j\}_{j\in[1,Q]}$. For $j = 1$ to Q, the requester chooses a random $r_j \in \mathbb{Z}_p$ to compute $\{K_{Tj,1}, K'_{Tj,2}\}$.

$$K_{Tj,1} = g^{r_j}\ and\ K'_{Tj,2} = (u^{tag_j} h)^{-r_j} \tag{6}$$

The requester computes and stores the task preparation information $IT = ((C_{Ti,1}, C_{Ti,2}, C_{Ti,3})_{i\in[1,P]}), (K_{Tj,1}, K'_{Tj,2})_{j\in[1,Q]})$.

Enc-task$(T_c, (M_t, \rho_t), IT) \to ct_{task}$: The $Enc - task$ algorithm takes the task content T_c and an LSSS task requirement policy as inputs, where $M_t \in \mathbb{Z}_p^{l\times n}$ and $\rho_t : [l] \to \mathbb{Z}_p$. It first selects $\vec{y_T} = (s_t, y_2, \ldots, y_n)^\top \in \mathbb{Z}_p^{n_1}$, where s_t is the task secret to be shared. The vector of shares is $\vec{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_l)^\top = M_t \vec{y_t}$. It then selects l random exponents $t_1, t_2, \ldots, t_l \in \mathbb{Z}_p$. It computes $C_T = T_c \cdot e(g, g)^{\alpha \cdot s_t}$, $C_{T0} = g^{s_t}$. For every $\tau \in [l]$, the requester computes

$$C_{T\tau,4} = \lambda_\tau - \lambda'_\tau,\ C_{T\tau,5} = t_\tau \cdot (\rho(\tau) - x_\tau)$$
$$C_{T\tau,6} = w^{\lambda_\tau} \tag{7}$$

The task content encrypted with a requirement policy is $ct_{task} = (C_T, C_{T0}, C_{T\tau,1}, C_{T\tau,2}, C_{T\tau,3}, C_{T\tau,4}, C_{T\tau,5}, C_{T\tau,6\,\tau\in[l]})$. For privacy issue, the requester encrypts the task tag attributes by running $Task_{tag}$ algorithm.

Task-tag $(PK, s_t, S_{tag} = (tag_1, tag_2, \ldots, tag_n), IT) \to sk_{tag}$. This algorithm is run by the requester, it takes the system public key and the tag sets as inputs. First, to bind the task content ciphertext with the tag attributes, the requester computes $K_{T3} = g^{s_t}$. Then the requester selects the corresponding task tags in $\{K_{Tj,1}, K'_{Tj,2}\}$ with the same task tags in S_{tag}, then he computes for every $\iota \in [n]$

$$K_{T\iota,2} = K'_{T\iota,2} \cdot v^{s_t} \tag{8}$$

The encrypted tag ciphertext is $sk_{tag} = (K_{T\iota 1}, K_{T\iota 2}, K_{T3})_{\iota\in[1,n]}$.

The requester sends his task ciphertext that consist of the task content ciphertext with the task tag ciphertext $C_{tt} = (ct_{task}, sk_{tag})$ to the service provider.

Phase 4 Task recommendation

The task recommendation is processed by the service platform. It analyzes and judges whether the capability attributes of the worker can satisfy the task requirement policy from the requester, and whether the task tags match the interest policy of the worker. If both of them match, the task matching is successful, and then the service platform recommends the matching tasks to the worker. For every registered worker that is online in the crowdsourcing system, when there is a new task published on this platform, the platform analyzes if the task with ciphertext C_{tt}, the worker with sk_{cap} and ct_{int}, can match each other by running the $Cap - ptest$ algorithm and $Int - ptest$ algorithm.

First, the crowdsourcing platform checks if the capability attributes from the worker can satisfy the task requirement policy.

Cap-ptest $(ct_{task}, sk_{cap}) \to I_t$. For every capability attribute cap_i with keys $\{k_{Wi,2}, k_{Wi,3}\}$, the service platform checks if $cap_i = \rho_t(\tau)$ with the following equation

$$\frac{e(C_{T\tau,6}, K_{W1})}{e(C_{T\tau,1}, K_{W1}) \cdot e(w^{C_{T\tau,4}}, K_{W1})} \cdot \frac{1}{e(C_{T\tau,2} u^{C_{T\tau,5}}, K_{Wi,2}) \cdot e(C_{T\tau,3}, K_{Wi,3})} \overset{?}{=} 1 \tag{9}$$

If the above equation equals 1, it means the corresponding $cap_i = \rho_t(\tau)$, i.e., this capability attribute cap_i is in the requirement policy. For each matched keyword $cap_i = \rho_t(\tau)$, we add the matching paring (i, τ) into set I_t. When finishing search for all the tags, it outputs an index set I_t, with which the task requirement policy can be easily verified whether satisfied or not.

If the task requirement policy can be satisfied, the service platformchecks if the tag attributes of the task can match the worker interest policy by running $Int - ptest$.

Int-ptest $(ct_{int}, sk_{tag}) \rightarrow I_i$, the service platform checks if $tag_i = \rho_i(\tau)$ with the following equation

$$\frac{e(C_{W\tau,4}, K_{T1})}{e(C_{W\tau,1}, K_{T1}) \cdot e(C_{W\tau,2}, K_{Ti,2}) \cdot e(C_{\tau,3}, K_{Ti,3})} \overset{?}{=} 1 \tag{10}$$

If the above equation equals 1, it means the corresponding $tag_i = \rho_i(\tau)$, i.e., this tag attribute tag_i is in the interest policy. For each matched keyword $tag_i = \rho_i(\tau)$, we add the (i, τ) into set I_i. When finishing search for all the tags, it outputs an index set I_i, with which the task requirement policy can be easily verified whether satisfied or not.

If both the task requirement policy and worker interest policy can be satisfied, the service platform recommends the task to the worker.

Phase 5 Task pre-decryption

After the task recommendation, the suitable worker can receive the recommending tasks. If the worker wants to perform the task, he sends a message to the service platform (the message can be an acceptation button in the system). When the service platform receives the message from the worker, the service platform pre-decrypts the task content by running $Pre - dec$ algorithm.

Pre-dec $(PK, SK_W, SK_{cap}, CT_W, C_{tt}) \rightarrow TK$. The service platform takes the system public key PK, the secret keys of the worker, the capability key SK_{cap}, interest ciphertext CT_{int} from the worker and the ciphertext C_{tt} of the task as inputs, and pre-decrypts the task ciphertext with TK as the output. Here, we term the corresponding capbility attribute index as $j1$ and the corresponding task ciphertext index as $j2$ for the $j - th$ element in set I_t, which means $cap_{j1} = \rho_{t(j2)}$. If the worker has capability attributes to satisfy the task requirement policy, the service platform can find a set of constants $\{c_{t,j}\}$ that $\sum_{j \in I_t} \lambda_{j2} \cdot c_{t,j} = s_t$. The service platform computes TK_1 as

$$\begin{aligned} TK_{11} &= e(C_{T_{j2},1}, K_{W1}) \cdot e(w^{C_{T_{j2},4}}, K_{W1}) \\ TK_{12} &= e(C_{T_{j2},2} u^{C_{T_{j2},5}}, K_{W_{j1},2}) \\ TK_{13} &= e(C_{T_{j2},3}, K_{W_{j1},3}) \\ TK_1 &= \prod_{j \in I_t} (TK_{11} \cdot TK_{12} \cdot TK_{13})^{c_{t,j}} \\ &= e(g, w)^{W_{id} \cdot s_t} \end{aligned} \tag{11}$$

Similarly, we term the corresponding tag attribute index as $i1$ and the corresponding interest ciphertext index as $i2$ for the $i - th$ element in set I_i, which means $tag_{i1} = \rho_{i(i2)}$. If the task with tag attributes to match the interest policy of worker, the service platform can find a set of constants $\{c_{i,i}\}$ that $\sum_{i \in I_i} \lambda_{i2} \cdot c_{i,i} = s_w \cdot z_t + W_{id}$. The service platform computes TK_2 as

$$\begin{aligned} TK_{21} &= e(C_{W_{i2},1}, K_{T1}) \\ TK_{22} &= e(C_{W_{i2},2}, K_{T_{i1},2}) \\ TK_{23} &= e(C_{W_{i2},3}, K_{T_{i1},3}) \\ TK_2 &= \prod_{i \in I_i} (TK_{21} \cdot TK_{22} \cdot TK_{23})^{c_{i,i}} \\ &= e(g, w)^{s_t \cdot (s_w \cdot z_t + W_{id})} \end{aligned} \tag{12}$$

With TK_1 and TK_2, the service platform can compute TK as

$$\begin{aligned} TK &= \frac{e(C_{T0}, K_0') \cdot TK_1}{TK_2} \\ &= e(g, g)^{\alpha s_t z_t} \end{aligned} \tag{13}$$

After the task pre-decryption, the service platform sends $\{C_T, TK\}$ to the worker.

Phase 6 Task decryption

Upon receiving the pre-decrypted data, the worker can efficiently decrypt the data by running the decryption algorithm.

Dec $(C_T, TK, z_t) \rightarrow T$. The task can be easily decrypted as

$$\begin{aligned} T &= \frac{C_T}{TK^{1/z_t}} \\ &= \frac{T \cdot e(g, g)^{\alpha s_t}}{e(g, g)^{\alpha s_t z_t (1/z_t)}} \end{aligned} \tag{14}$$

4.3. Scheme correctness

Task Recommendation Correctness

We analyze the service platform can recommend suitable tasks to the worker, where the worker's capability attributes can satisfy the task requirement policy, and the task tag keywords can match the interest policy of the worker.

The correctness about the worker's capability attributes can satisfy the task requirement policy is shown as follows.

$$\begin{aligned}&\frac{e(C_{T\tau,6},K_{W1})}{e(C_{T\tau,1},K_{W1})\cdot e(w^{C_{T\tau,4}},K_{W1})}\cdot\frac{1}{e(C_{T\tau,2}u^{C_{T\tau,5}},K_{Wi,2})\cdot e(C_{T\tau,3},K_{Wi,3})}\\ &=\frac{e(w^{\lambda_\tau},g^{W_{id}})}{e(w^{\lambda'_\tau}v^{t_\tau},g^{W_{id}})\cdot e(w^{\lambda_\tau-\lambda'_\tau},g^{W_{id}})}\cdot\frac{1}{e(u^{\rho_{t\tau}}h^{-t_\tau},g^{c_i})\cdot e(g^{t_\tau},u^{cap_1}h^{c_1}v^{W_{id}})}\\ &=\frac{1}{e(u^{\rho_t(\tau)}h,g)^{-t_\tau c_i}\cdot e(u^{cap_i}h,g)^{-t_\tau c_i}}\end{aligned}\tag{15}$$

If $cap_i=\rho_t(\tau)$, the above equation equals 1. This capability attribute cap_i is in the requirement policy. Once all the capability attributes have located their positions (row number) in M_t, it is easy to evaluate whether the worker's capability attributes can satisfy the task requirement policy.

The correctness about the task tag keywords can match the interest policy is shown as follows.

$$\begin{aligned}&\frac{e(C_{W\tau,4},K_{T1})}{e(C_{W\tau,1},K_{T1})\cdot e(C_{W\tau,2},K_{Ti,2})}\cdot\frac{1}{e(C_{\tau,3},K_{Ti,3})}\\ &=\frac{e(w^{\lambda_\tau},g^{s_t})}{e(w^{\lambda_\tau}v^{t_\tau},g^{s_t})\cdot e((u^{\rho_{i\tau}}h)^{-t_\tau},g^{c_i})}\cdot\frac{1}{e(g^{t_\tau},(u^{tag_i}h)^{c_i}v^{s_t})}\\ &=\frac{1}{e(u^{\rho_i(\tau)}h,g)^{-t_\tau c_i}\cdot e(u^{tag_i}h,g)^{-t_\tau c_i}}\end{aligned}\tag{16}$$

If $tag_i=\rho_i(\tau)$, the above equation equals 1. This task tag attribute tag_i is in the worker's interest policy. Once all the task tag attributes have located their positions (row number) in M_i, it is easy to evaluate whether the task tag keywords can match the worker's interest policy.

Decryption Correctness

To analyze the correctness of the decryption, we should prove the correctness of the pre-decryption at first.

The service platform can compute TK_1 as

$$\begin{aligned}TK_{11}&=e(C_{T_{j2},1},K_{W1})\cdot e(w^{C_{T_{j2},4}},K_{W1})\\ &=e(w^{\lambda'_{j2}}v^{t_{j2}},g^{W_{id}})\cdot e(w^{\lambda_{j2}-\lambda'_{j2}},g^{W_{id}})\\ &=e(g,w)^{W_{id}\lambda_{j2}}e(g,v)^{W_{id}t_{j2}}\end{aligned}\tag{17}$$

$$\begin{aligned}TK_{12}&=e(C_{T_{j2},2}u^{C_{T_{j2},5}},K_{W_{j1},2})\\ &=e((u^{x_{j2}}h)^{-t_{j2}}u^{t_{j2}\cdot(\rho_t(j2)-x_{j2})},g^{r_{j1}})\\ &=e((u^{\rho_t(j2)}h)^{-t_{j2}},g^{r_{j1}})\\ &=e(g,u^{\rho_{t(j2)}}h)^{-r_{j1}t_{j2}}\end{aligned}\tag{18}$$

$$\begin{aligned}TK_{13}&=e(C_{T_{j2},3},K_{W_{j1},3})\\ &=e(g^{t_{j2}},(u^{cap_{j1}}h)^{r_{j1}}v^{-W_{id}})\\ &=e(g,u^{cap_{j1}}h)^{r_{j1}t_{j2}}\cdot e(g,v)^{-W_{id}t_{j2}}\end{aligned}\tag{19}$$

$$\begin{aligned}TK_1&=\prod_{j\in I_t}(TK_{11}\cdot TK_{12}\cdot TK_{13})^{c_{t,j}}\\ &=e(g,w)^{\sum_{j\in I_t}W_{id}\cdot\lambda_{j2}}\\ &=e(g,w)^{W_{id}\cdot s_t}\end{aligned}\tag{20}$$

Similarly, the service platform computes TK_2 as

$$\begin{aligned}TK_{21}&=e(C_{W_{i2},1},K_{T3})\\ &=e(w^{\lambda_{i2}}v^{t_{i2}},g^{s_t})\\ &=e(g,w)^{\lambda_{i2}s_t}e(g,v)^{t_{i2}s_t}\end{aligned}\tag{21}$$

$$\begin{aligned} TK_{22} &= e(C_{W_{i2},2}, K_{T_{i1},1}) \\ &= e((u^{\rho_i(i2)}h)^{-t_{i2}}, g^{r_{i1}}) \\ &= e(g, u^{\rho_{i(i2)}}h)^{-r_{i1}t_{i2}} \end{aligned} \tag{22}$$

$$\begin{aligned} TK_{23} &= e(C_{W_{i2},3}, K_{T_{i1},2}) \\ &= e(g^{t_{i2}}, (u^{tag_{i1}}h)^{r_{i1}}v^{s_t}) \\ &= e(g, u^{tag_{i1}}h)^{r_{i1}t_{i2}} e(g, v)^{-t_{i2}s_t} \end{aligned} \tag{23}$$

$$\begin{aligned} TK_2 &= \prod_{i\in I_i}(TK_{21}\cdot TK_{22}\cdot TK_{23})^{c_{i,i}} \\ &= e(g, w)^{\sum_{i\in I_i} s_t\cdot(\lambda_{j2}\cdot c_{i,i})} \\ &= e(g, w)^{s_t\cdot(s_w\cdot z_t + W_{id})} \end{aligned} \tag{24}$$

$$\begin{aligned} TK &= \frac{e(C_{T0}, K_0')\cdot TK_1}{TK_2} \\ &= \frac{e(g^{s_t}, g^{\alpha z_t}w^{z_t s_w})\cdot e(g, w)^{W_{id}\cdot s_t}}{e(g, w)^{s_t\cdot(s_w\cdot z_t + W_{id})}} \\ &= e(g, g)^{\alpha s_t z_t} \end{aligned} \tag{25}$$

5. Privacy analysis

In this section, we analyze our scheme can preserve privacy for both workers and requesters.

5.1. Worker privacy

We analyze that our scheme can preserve capability attribute privacy and interest policy privacy for the worker.

(1) Capability attribute privacy. The capability attributes of the worker should be preserved, although the service platform utilizes the capability attributes of the worker to analyze if the worker is capable to perform the task with requirement policy. We analyze that if there is an adversary $\mathcal{A}$ can obtain worker capability attributes in our scheme with a non-negligible advantage, it can solve decisional DBDH problem in a polynomial time as follows.

Setup: The challenger $\mathcal{C}$ runs the $Setup(\lambda)$ algorithm and set $Mask = \alpha$, and public key $PK = (g, u, h, w, v, g^{\alpha}, e(g, g)^{\alpha})$. Then sends the public key to $\mathcal{A}$.

Phase 1: $\mathcal{A}$ queries the secret key with attribute w_i. To simulate the secret keys and other parameters, $\mathcal{C}$ first computes $u^{w_i}h$. Then, it maintains a table to record $(w_i, u^{w_i}h)$. If an existing w_i is queried before, the $u^{w_i}h$ is sent to $\mathcal{A}$. Otherwise, it creates a new tuple accordingly, and stores the tuple into the table. Then, it randomly chooses $w_i \in \mathbb{Z}_p$ and $r_i \in \mathbb{Z}_p$ for each capability attribute w_i, and returns $K = K_0, K_1, K_{i,2}, K_{i,3}$ to $\mathcal{A}$, where

$$\begin{aligned} K_0 &= g^{\alpha}w^{w_i},\ K_1 = g^{\alpha w_i} \\ K_{i,2} &= g^{\alpha r_i}, K_{i,3} = (u^{w_i}h)^{r_i}v^{-w_i} \end{aligned} \tag{26}$$

Challenge: $\mathcal{A}$ submits two capability attribute vectors $w_0* = \{w_{0,1}, \ldots, w_{0,n}\}$ and $w_1* = \{w_{1,1}, \ldots, w_{1,n}\}$. For any keyword in these vectors, it has not been queried in the previous query phase. $\mathcal{A}$ also provides a challenge policy (M_t, r_t) which can be satisfied by both w_0* and w_1*. $\mathcal{C}$ first flips a random coin b, and chooses a random number $c_j \in \mathbb{Z}_b$ for every attribute $w_{b,j}$, then simulates the capability keys as $Td = M_t, K_{1j}, K_{2j}$. where $\{K_{1j} = g^{\alpha c_j}, K_{2j} = (u^{w_{b,j}}h)^{c_j}v^{w_{id}}\}$.

Phase 2: Same as Phase 1.

Guess: $\mathcal{A}$ outputs a guess b' of b.

Then we show that if the adversary has a non-negligible advantage in the above game, i.e., $b' = b$, the adversary $\mathcal{A}$ can solve DBDH problem with a non-negligible advantage. Given $A = g^{\alpha}, B = v = g^{v'}, C = g^{w_{id}}$, the adversary $\mathcal{A}$ can compute

$$\begin{aligned} e(g, g)^{\alpha v' w_{id}} &= \frac{e(K_{2j}, A)}{e(K_{1j}, u^{w_{b',j}}h))} \\ &= \frac{e((u^{w_{b,j}}h)^{c_j}v^{w_{id}}, g^{\alpha})}{e(g^{\alpha c_j}, u^{w_{b',j}}h)} \\ &= \frac{e(u^{w_{b,j}}h, g)^{\alpha c_j}e(g^{v' w_{id}}, g^{\alpha})}{e(u^{w_{b',j}}h, g)^{\alpha c_j}} \end{aligned} \tag{27}$$

If the adversary can guess $b' = b$ with non-negligible advantage, then it can compute $e(g,g)^{\alpha v' w_{id}}$ with non-negligible advantage.

(2) Interest policy privacy. The interest policy of the worker should be preserved, although the service platform utilizes the interest policy of the worker to analyze if the worker has interests to perform the task associate with task tags. We analyze that if there is an adversary $\mathcal{A}$ can obtain worker interest policy in our scheme with a non-negligible advantage, it can solve DBDH problem in a polynomial time as follows.

The adversary $\mathcal{A}$ submits (M_{i0}, ρ_{i0}) and (M_{i1}, ρ_{i1}), where $M_{i\{0,1\}} \in \mathbb{Z}_p$ to the challenger $\mathcal{C}$. The challenger randomly selects a secret a vector $\vec{y} = \{s, y_1, \ldots, y_n\}^T$, and randomly selects $b \in \{0, 1\}$. It computes $\overrightarrow{\lambda_b} = \{\lambda_{b1}, \ldots, \lambda_{bl}\}^T = M_{ib} \cdot \vec{y}$, and selects l random exponents $t_1, \ldots, t_l \in \mathbb{Z}_p$. Then $\mathcal{C}$ computes the following ciphertext.

$$\begin{aligned} C_{W_{b\tau,1}} &= w^{\lambda_{b\tau}}, C_{W_{b\tau,2}} = (u^{\rho_{ib}} h)^{-t_\tau} \\ C_{W_{b\tau,3}} &= g^{t_\tau},\ C_{W_{b\tau,4}} = w^{\lambda_{b\tau}} \end{aligned} \tag{28}$$

Similar to the capability attribute privacy, we can prove that if adversary $\mathcal{A}$ can distinguish each pair of (M_{i0}, ρ_{i0}) with (M_{i1}, ρ_{i1}) with non-negligible advantage, when given $g, w = g^{w'}, v = g^{v'}, T = g^{t_\tau}$, $\mathcal{A}$ can compute $e(g,g)^{w'v't_\tau}$ with non-negligible advantage.

$$\begin{aligned} e(g,g)^{w'v't_\tau} &= \frac{e(C_{W_{b\tau,1}}, w) \cdot e(C_{W_{b\tau,2}}, g)}{e(w^{\lambda_{b'\tau}}, w) \cdot e(u^{\rho_{ib'}} h, T^{-1})} \\ &= \frac{e(g,g)^{v'w't_\tau} \cdot e(g^{w'\lambda_{b\tau}}, w) \cdot e((u^{\rho_{ib}} h)^{-t_\tau}, g)}{e(g^{w'\lambda_{b'\tau}}, w) \cdot e(u^{\rho_{ib'}} h, g^{-t_\tau})} \end{aligned} \tag{29}$$

5.2. Requester privacy

We analyze that our scheme can preserve task tag privacy and task content privacy for the requester.

(1) Task tag privacy. The task tag attributes of the requester should be preserved, although the service platform utilizes the task tag attributes of the requester to analyze if the task can match the interest policy of the worker. We analyze that if there is an adversary $\mathcal{A}$ can obtain task tag attributes in our scheme with a non-negligible advantage, it can solve DBDH problem in a polynomial time as follows.

$\mathcal{A}$ submits two task tag attribute vectors $t_0* = \{t_{0,1}, \ldots, t_{0,n}\}$ and $t_1* = \{t_{1,1}, \ldots, t_{1,n}\}$ to the challenger $\mathcal{C}$. $\mathcal{C}$ first flips a random coin b, and chooses a random number $c_j \in \mathbb{Z}_p$ and $r_{id} \in \mathbb{Z}_p$ for every attribute $w_{b,j}$, then simulates the capability keys as $T = \{T_{1j}, T_{2j}\}$. where $\{T_{1j} = g^{\alpha c_j}, T_{2j} = (u^{t_{b,j}} h)^{c_j} v^{r_{id}}\}$.

Then we show that if the adversary has a non-negligible advantage to guess $b' = b$, the adversary $\mathcal{A}$ can solve BDH problem with a non-negligible advantage. Given $g, g^\alpha, v = g^{v'}, C = g^{r_{id}}$, the adversary $\mathcal{A}$ can compute

$$\begin{aligned} e(g,g)^{\alpha v' r_{id}} &= \frac{e(T_{2j}, A)}{e(T_{1j}, u^{t_{b',j}} h))} \\ &= \frac{e((u^{t_{b,j}} h)^{c_j} v^{r_{id}}, g^\alpha)}{e(g^{\alpha c_j}, u^{t_{b',j}} h)} \\ &= \frac{e(u^{t_{b,j}} h, g)^{\alpha c_j} e(g^{v' r_{id}}, g^\alpha)}{e(u^{t_{b',j}} h, g)^{\alpha c_j}} \end{aligned} \tag{30}$$

If the adversary can guess $b' = b$ with a non-negligible advantage, then it can compute $e(g,g)^{\alpha v' r_{id}}$ with a non-negligible advantage.

(2) Task content privacy. Our scheme can preserve task content privacy through keeping the task information confidential from unauthorized entities.

The encryption algorithm is constructed based on the CP-ABE proposed in [24], which is proved to be IND-CPA secure in standard model. Meanwhile, the pre-decryption scheme is proven to be semantic security against chosen plain text attacks. Similarly, our scheme can be proven to be IND-CPA secure.

6. Performance evaluation

In this section, we analyze the performance of our scheme in terms of computation cost of the requester, the worker and the service platform in Table 2. E_i (respectively, M_i) represents an exponentiation (respectively, multiplication) in the group $\mathbb{G}_i$, and e represents the paring time. The bilinear operations are the dominate cost, such that we ignore minor factors such as arithmetic in $\mathbb{Z}_p$. P represents the maximum rows of the task encryption LSSS policy; Q represents the unified task tag numbers; P_c represents the capability attribute numbers of the worker; P_i represents the number of attributes in the worker's interest policy; P_t represents the task tag attribute numbers of the requester; P_r represents the number of attributes in the requester's task requirement policy. We ignore small numbers of operations, such as arithmetic in $\mathbb{Z}_p$.

We compare our scheme with PAB-MKS [16] on a 256-bit Bareto-Naehrig curve using version 0.3.1 of the RELIC library. Since the requester and the worker are mobile users, we evaluate their performance with ARM Cortex-A9 CPU, and 1 GB

Table 2
Computation overhead.

Task encryption preparation	$(5E_1 + 2M_1) \cdot P + (3E_1 + M_1) \cdot Q$
Task content encryption	$E1 \cdot P_r + E_1 + E_2 + M_1$
Task tag encryption	$M_1 \cdot P_t + E_1$
Interest encryption	$3E_1 + M_1 + (6E_1 + 2M_1) \cdot P_i$
Task recommendation	$4e \cdot (P_c \cdot P_r + P_i \cdot P_t)$
Task pre-decryption	$(3e + 2M_1 + E_1) \cdot P_r + (3e + 2MT + ET) \cdot P_i + e + 2M_1$
Task decryption	$2M_1$

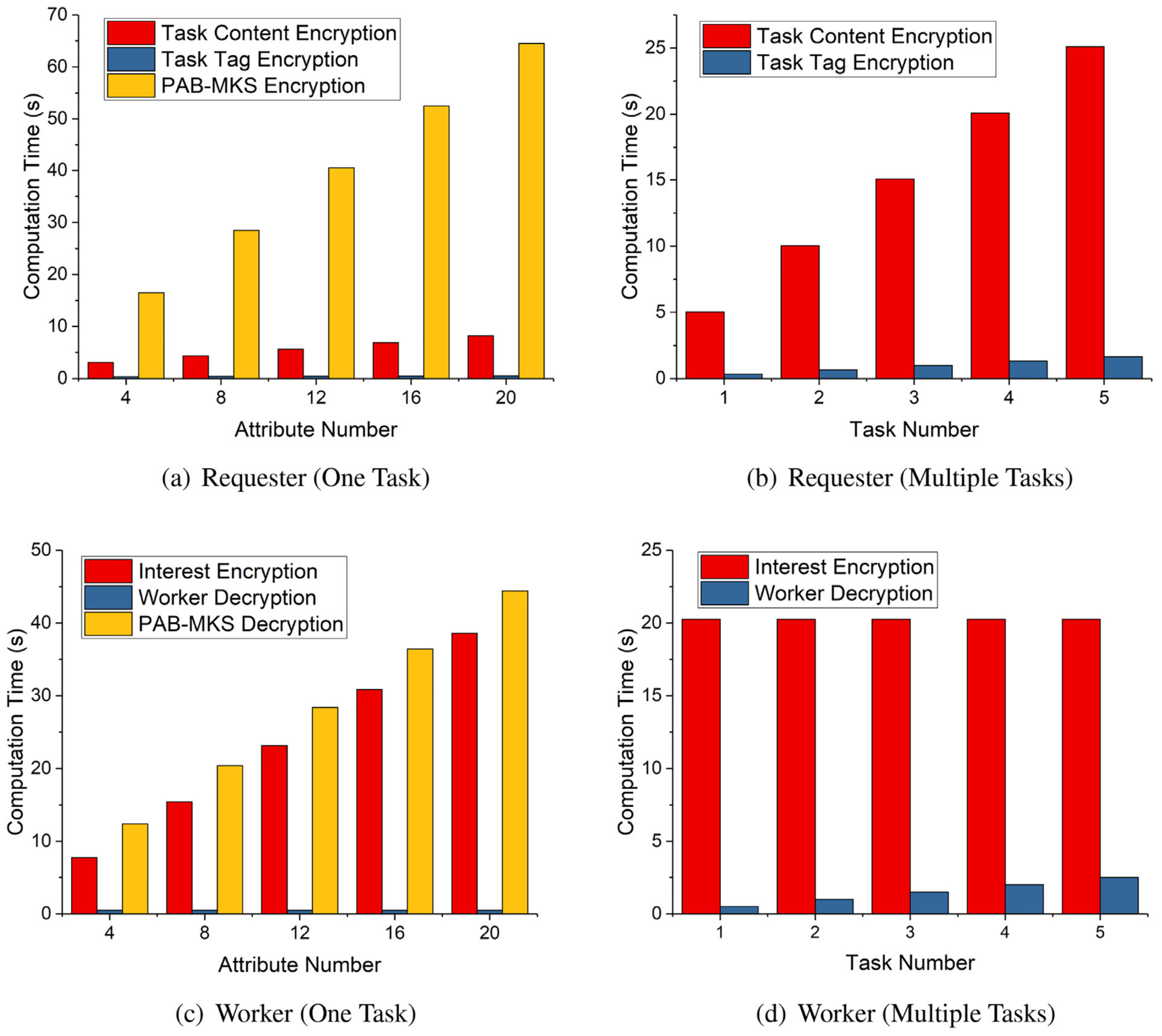
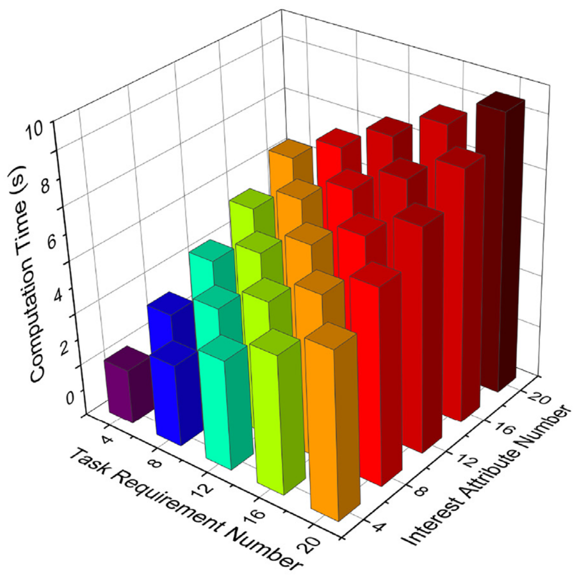


Fig. 4. Computation overhead on requester and worker.

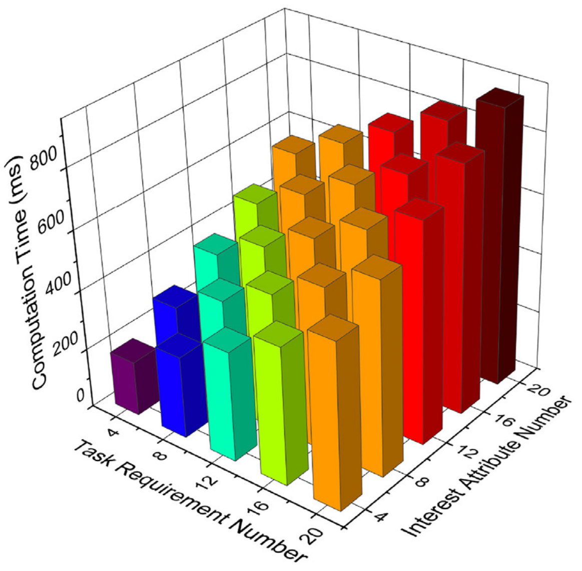
RAM. The service platform can be a powerful server, we evaluate its performance with Intel Core i5 CPU, and 4 GB RAM. Times are measured in milliseconds (averaged over 10,000 iterations).

6.1. Overhead on requester

The computation on the requester of our scheme mainly includes two parts: the task content encryption with the task requirement policy and the task tag encryption with the task tag attributes. In Fig. 4(a), we illustrate the computation overhead on the requester under the condition that only one task is published in our crowdsourcing system. We take the computation time as y-axis, and the task requirement policy attribute number and task tag attribute number as the x-axis.



(a) Recommendation



(b) Pre-decryption

Fig. 5. Computation overhead on service platform.

As shown in Fig. 4(a), we can demonstrate that the task content encryption time increases linearly with the task requirement policy attribute number. The task tag encryption time increases linearly with the task tag attribute number. Both of the task content encryption and the task tag encryption are much less than the encryption of PAB-MKS [16], which demonstrates the efficiency of our scheme on the requester side.

In Fig. 4(b), we illustrate the computation overhead on the requester under the condition that multiple tasks are published in our crowdsourcing system. We take the computation time as y-axis, and the task number as x-axis. For simplicity, we set average task requirement attribute number and task tag attribute number as 10. From this figure, we can demonstrate that the task content encryption and tag encryption increases linearly with the task number, since more tasks are published, more task encryption should be performed by the requester.

6.2. Overhead on worker

The main computation overhead on the worker also includes two parts: the worker interest encryption with the interest policy and the task decryption. In Fig. 4(c), we illustrate the computation overhead on the worker on the condition that there is only one task. We take the computation time as the y-axis, and the worker interest policy attribute number and the task requirement policy attribute number as the x-axis. As shown in Fig. 4(c), we can demonstrate that the worker interest encryption time increases linearly with the worker interest policy attributes since if the worker has more interest attributes to be hidden in the policy, the worker should provide more computation time to encrypt them. The decryption time is constant nearly to be zero, because most portion of the decryption operations (especially the time-consuming paring operations) are offloaded to the service platform. Compared with [16], the worker decryption in our scheme is much less than PAB-MKS; the worker interest encryption is less but approximate to PAB-MKS. In fact, if the worker interests remain stable in a period of time, the worker interest encryption can be performed once when the worker is joined into the crowdsourcing system, such that our scheme is efficient on the worker side.

In Fig. 4(d), we illustrate the computation overhead on the worker on the condition that there are multiple tasks recommended to the worker. As shown in Fig. 4(d), we can demonstrate that the worker interest encryption time is constant no matter how many tasks are recommended to the worker. If the worker does not need to update his interest information in tasks, only one time of worker interest encryption is required in our scheme. The task decryption increases linearly with the task number, but is efficient since the time consumption for one task is very small.

6.3. Overhead on service platform

In Fig. 5, we illustrate the computation overhead on the service platform. We take the computation time as the z-axis, the task requirement policy attribute number as the x-axis, and the interest attribute number as the y-axis. The recommendation time increases linearly with the worker interest policy attribute number, worker capability attribute number, task requirement policy attribute number, and the task tag attribute number. For simplicity, in Fig. 5(a), we set the worker capability attribute number and the task tag attribute number as 12. We can demonstrate that the recommendation time increases with the task requirement number and the interest attribute number. As shown in Fig. 5(b), we can demonstrate that the pre-decryption time increases with the task requirement number and the interest attribute number.

In summary, since the task encryption on the worker is split into preparation phase (preparation information can be used for every task encryption) and online phase. For every task encryption, the requester only employs the online encryption, such that the computation on the requester is efficient. For the worker, the interest policy encryption only needs to be performed once, and the decryption time is nearly zero, such that the computation on the worker is efficient.

7. Conclusion

In this paper, we have proposed privacy-preserving task recommendation with win-win incentives for mobile crowdsourcing, which can achieve effective task recommendation to benefit both requesters and workers, as well as guarantee participants' privacy. Firstly, our scheme design bipartite matching to support effective task recommendation to suitable workers, leading the valuable task accomplishments for requesters and efficient task selection for workers. Secondly, our scheme preserves both privacy for requesters and workers through encrypting the collected interest information for the worker, and the task tag keywords for the requester; as well as keeping the tasks to be accessed by capable and interested workers. Thirdly, our scheme reduces the computation cost for the worker by offloading most portion of the task decryption to the service platform, and for the requester through splitting the task encryption to preparation phase and online encryption phase. In our future work, we will consider the variability of the worker interests, and design privacy-preserving task recommendation with flexible and dynamic interest policy update for workers in mobile crowdsourcing.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No. 61702562), 111 Project (No. B18059), China Scholarship Council, the Innovation Foundation for Postgraduate of Central South University under Grant Number 2016zzts057, Innovation-Driven Project of Central South University (No. 2016CXS013), International Science & Technology Cooperation Program of China (No. 2013DFB10070), China Hunan Provincial Science & Technology Program (No. 2012GK4106), and NSERC of Canada.

Appendix A. Linear Secret Sharing System (LSSS)

A secret-sharing scheme Π over a set of parties $\mathcal{P}$ is called linear (over $\mathbb{Z}_p$) if

1. The shares for each party form a vector over $\mathbb{Z}_p$.
2. There exists a matrix $\mathbb{M}$ with l rows and n columns called the share-generating matrix for Π. For all $i = 1, \ldots l$, the ith row $\mathbb{M}_i$ of $\mathbb{M}$ is labeled by a party $\rho(i)$ where ρ is a function from $\{1, 2, \ldots, m\}$ for parties $\mathcal{P}$. When we consider a

column vector $v = (s, r_2, \dots, r_n)$, where $s \in \mathbb{Z}_p$ is the secret to be shared, and $r_2, \dots, r_n \in \mathbb{Z}_p$ are randomly chosen. The $\mathbb{M}\vec{v}$ is the vector of l shares of the secret s. The share $\lambda_i = (\mathbb{M}\vec{v})_i$, i.e., the inner product $\mathbb{M}_i\vec{v}$ belongs to party $\rho(i)$.

Suppose 2 is a LSSS denoted by $(\mathbb{M}, \rho)$ for attribute access structure $\mathcal{A}$. Let $S \in \mathcal{A}$ be an authorized attributes set, and $I = \{i : \rho(i) \in S\}$, $I \subset \{1, 2, \dots, l\}$. Then there exist constants $\{w_i \in \mathbb{Z}_p\}_{i \in I}$ satisfying the $\sum_{i \in I} w_i \mathbb{M}_i = (1, 0, \dots, 0)$, so that if λ_i are valid shares of any secret s according to 2, the $\sum_{i \in I} w_i \lambda_i = s$. Furthermore, these constants w_i can be found in polynomial time in the size of the share-generating matrix $\mathbb{M}$. For any unauthorized set, no such constants exists.

Appendix B. Bilinear Groups

The bilinear pairings namely Weil pairing and Tate paring of algebraic curves are defined as a map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_1$, where $\mathbb{G}$ is a cyclic additive group generated by g, whose order is a prime p, and $\mathbb{G}_1$ is a cyclic multiplicative group of the same order q. Discrete logarithm problems (DLP) in both $\mathbb{G}$ and $\mathbb{G}_1$ are hard. Bilinear pairings have the following properties:

- Bilinearity: for any $u, v \in \mathbb{G}$, and $a, b \in \mathbb{Z}_p$, it has $e(u^a, v^b) = e(u, v)^{ab}$.
- Non-degeneracy: $e(g, g) \neq 1$, 1 is the unit parameter in $\mathbb{G}_1$.
- Computability: for all $u, v \in \mathbb{G}$, there is an efficient algorithm to compute $e(u, v)$.

Appendix C. Decisional Bilinear Diffie-Hellman (DBDH) Assumption

A challenger chooses a group $\mathbb{G}$ of prime order p based on the security parameter of system. Let $a, b, c, z \in \mathbb{Z}_p$ be selected randomly and g be a generator of $\mathbb{G}$. With $(g,\ A = g^a,\ B = g^b,\ C = g^c)$, the adversary must distinguish a valid tuple $e(g, g)^{abc}$ from $e(g, g)^z$.

An algorithm B that outputs a guess $\mu \in \{0,\ 1\}$ has the advantage ε in solving DBDH if the following formula was satisfied.

$$\left| \begin{array}{l} Pr[B(g, A, B, C, e(g, g)^{abc}) = 0] \\ -Pr[B(g, A, B, C, e(g, g)^z) = 0] \end{array} \right| \geq \varepsilon$$

References

[1] A. Alamer, J. Ni, X. Lin, X. Shen, Location privacy-aware task recommendation for spatial crowdsourcing, in: Proc. of IEEE WCSP, 2017, pp. 1–6.
[2] E. Aldhahri, V. Shandilya, S. Shiva, Towards an effective crowdsourcing recommendation system: a survey of the state-of-the-art, in: Proc. of IEEE SOSE, 2015, pp. 372–377.
[3] M. Allahbakhsh, B. Benatallah, A. Ignjatovic, H. Motahari-Nezhad, E. Bertino, S. Dustdar, Quality control in crowdsourcing systems: issues and directions, IEEE Internet Comput. 17 (2) (2013) 76–81.
[4] N. Attrapadung, H. Imai, Dual-policy attribute based encryption, in: Proc. of ACNS, 2009, pp. 168–185.
[5] J. Cheny, D. Zhao, A quality-aware attribute-based filtering scheme for participatory sensing, in: Proc. of IEEE MSN, 2016, pp. 179–186.
[6] D. Geiger, M. Schader, Personalized task recommendation in crowdsourcing information systems current state of the art, Decis Support Syst 65 (2014) 3–16.
[7] Y. Gong, L. Wei, Y. Guo, C. Zhang, Y. Fang, Optimal task recommendation for mobile crowdsourcing with privacy control, IEEE Internet Things J. 3 (5) (2016) 745–756.
[8] C. Ho, S. Jabbari, J.W. Vaughan, Adaptive task assignment for crowdsourced classification, in: Proc. of ICML, 2013, pp. 534–542.
[9] W. Jiang, J. Wu, G. Wang, On selecting recommenders for trust evaluation in online social networks., ACM Trans. Internet Technol. 15 (4) (2015). 14–1.
[10] K. Merkourios, K. Iordanis, T. Michalis, First learn then earn: optimizing mobile crowdsensing campaigns through data-driven user profiling, in: Proc. of ACM MobiHoc, 2016, pp. 271–280.
[11] K. Ari, H.T. Chun, I. Panagiotis, G. Evgeniy, Getting more for less: optimized crowdsourcing with dynamic tasks and goals, in: Proc. of WWW, 2015, pp. 592–602.
[12] J. Lin, S. Amini, J. Hong, N. Sadeh, J. Lindqvist, J. Zhang, Expectation and purpose: understanding users' mental models of mobile app privacy through crowdsourcing, in: Proc. of ACM UbiComp, 2012, pp. 501–510.
[13] B. Liu, W. Zhou, T. Zhu, H. Zhou, X. Lin, Invisible hand: a privacy preserving mobile crowd sensing framework based on economic models, IEEE Trans. Veh. Technol. 66 (5) (2017) 4410–4423.
[14] H. Liu, H. Ning, Q. Xiong, L.T. Yang, Shared authority based privacy-preserving authentication protocol in cloud computing, IEEE Trans. Parallel Distrib. Syst. 26 (1) (2015) 241–251.
[15] C. Ma, D. Yau, N. Yip, N. Rao, Privacy vulnerability of published anonymous mobility traces, IEEE/ACM Trans. Netw. 21 (3) (2013) 720–733.
[16] Y. Miao, J. Ma, X. Liu, X. Li, Z. Liu, H. Li, Practical attribute-based multi-keyword search scheme in mobile crowdsourcing, IEEE Internet Things J.
[17] J. Ning, Z. Cao, X. Dong, K. Liang, H. Ma, L. Wei, Auditable σ-time outsourced attribute-based encryption for access control in cloud computing, IEEE Trans. Inf. Forensics Secur. 13 (1) (2018) 94–105.
[18] C. Niu, Z. Zheng, F. Wu, X. Gao, G. Chen, Achieving data truthfulness and privacy preservation in data markets, IEEE Trans Knowl Data Eng (2018).
[19] X. Peng, J. Ren, L. She, D. Zhang, J. Li, Y. Zhang, BOAT: a block-streaming app execution scheme for lightweight IoT devices, IEEE Internet Things J. 5 (3) (2018) 1816–1829.
[20] J. Ren, H. Guo, C. Xu, Y. Zhang, Serving at the edge: a scalable IoT architecture based on transparent computing, IEEE Netw. 31 (5) (2017) 96–105.
[21] J. Ren, Y. Zhang, K. Zhang, X. Shen, Exploiting mobile crowdsourcing for pervasive cloud services: challenges and solutions, IEEE Commun. Mag. 53 (3) (2015) 98–105.
[22] J. Ren, Y. Zhang, K. Zhang, X.S. Shen, SACRM: social aware crowdsourcing with reputation management in mobile sensing, Comput. Commun. 65 (2015) 55–65.
[23] J. Ren, Y. Zhang, N. Zhang, D. Zhang, X. Shen, Dynamic channel access to improve energy efficiency in cognitive radio sensor networks, IEEE Trans. Wireless Commun. 15 (5) (2016) 3143–3156.
[24] Y. Rouselakis, B. Waters, Practical constructions and new proof methods for large universe attribute-based encryption, in: Proc. of ACM CCS, 2013, pp. 463–474.
[25] S. Yang, K. Han, Z. Zheng, S. Tang, F. Wu, Towards personalized task matching in mobile crowdsensing via fine-grained user profiling, in: Proc. of IEEE INFOCOM, 2016.

[26] W. Tang, K. Zhang, J. Ren, Y. Zhang, X.S. Shen, Flexible and efficient authenticated key agreement scheme for bans based on physiological features, IEEE Trans. Mob. Comput. (2018). doi: 10.1109/TMC.2018.2848644.
[27] H. To, G. Ghinita, L. Fan, C. Shahabi, Differentially private location protection for worker datasets in spatial crowdsourcing, IEEE Trans. Mob. Comput. 16 (4) (2017) 934–949.
[28] H. To, G. Ghinita, C. Shahabi, A framework for protecting worker location privacy in spatial crowdsourcing, in: Proc. of PVLDB, 2014, pp. 919–930.
[29] Y. Tong, J. She, B. Ding, L. Wang, L. Chen, Online mobile micro-task allocation in spatial crowdsourcing, in: Proc. of IEEE ICDE, 2016, pp. 49–60.
[30] L. Wang, D. Zhang, D. Yang, A. Pathak, C. Chen, X. Han, H. Xiong, Y. Wang, SPACE-TA: cost-effective task allocation exploiting intradata and interdata correlations in sparse crowdsensing, ACM Trans. Intell. Syst. Technol. 9 (2) (2017) 20.
[31] Z. Wang, J. Hu, R. Lv, J. Wei, Q. Wang, D. Yang, H. Qi, Personalized privacy-preserving task allocation for mobile crowdsensing, IEEE Trans. Mob. Comput. (2018).
[32] Z. Wang, R. Tan, J. Hu, J. Zhao, Q. Wang, F. Xia, X. Niu, Heterogeneous incentive mechanism for time-sensitive and location-dependent crowdsensing networks with random arrivals, Comput. Netw. 131 (2018) 96–109.
[33] K. Yang, K. Zhang, J. Ren, X. Shen, Security and privacy in mobile crowdsourcing networks: challenges and opportunities, IEEE Commun. Mag. 53 (8) (2015) 75–81.
[34] Z. Yu, H. Xu, Z. Yang, B. Guo, Personalized travel package with multi-point-of-interest recommendation based on crowdsourced user footprints, IEEE Trans. Hum. Mach. Syst. 46 (1) (2016) 151–158.
[35] M. Yuen, I. King, K. Leung, TaskRec: a task recommendation framework in crowdsourcing systems, Neural Process. Lett. 41 (2) (2015) 223–238.
[36] Q. Zhang, L.T. Yang, Z. Chen, P. Li, M.J. Deen, Privacy-preserving double-projection deep computation model with crowdsourcing on cloud for big data feature learning, IEEE Internet Things J. 5 (4) (2018) 2896–2903.
[37] X. Zhang, Z. Yang, Z. Zhou, H. Cai, L. Chen, X. Li, Free market of crowdsourcing: incentive mechanism design for mobile sensing, IEEE Trans. Parallel Distrib. Syst. 25 (12) (2014) 3190–3200.
[38] Y. Zhang, J. Ren, J. Liu, C. Xu, H. Guo, Y. Liu, A survey on emerging computing paradigms for big data, Chin. J. Electron. 26 (1) (2017) 1–12.
[39] D. Zhao, X. Li, H. Ma, Budget-feasible online incentive mechanisms for crowdsourcing tasks truthfully, IEEE/ACM Trans. Netw. 24 (2) (2016) 647–661.
[40] Y. Zhao, Q. Han, Spatial crowdsourcing: current state and future directions, IEEE Commun. Mag. 54 (7) (2016) 102–107.
[41] G. Zhuo, Q. Jia, L. Guo, M. Li, P. Li, Privacy-preserving verifiable data aggregation and analysis for cloud-assisted mobile crowdsourcing, in: Proc. of IEEE INFOCOM, 2016, pp. 1–9.