Secure and Efficient Probabilistic Skyline Computation for Worker Selection in MCS

Xichen Zhang, Rongxing Lu, *Senior Member, IEEE*, Jun Shao, Hui Zhu, *Senior Member, IEEE*, and Ali A. Ghorbani, *Senior Member, IEEE*

***Abstract*—The rapid advance of the Internet of Things (IoT) has enabled a new paradigm of the sensing network, i.e., mobile crowdsensing (MCS). Primarily, in MCS systems, a crowd of participating mobile users, namely, workers, are allocated by the MCS platforms to outsource their sensory data for specific tasks. Obviously, the reliability of workers and the trustability of their sensing data play significant roles in the service quality, thus the worker selection becomes crucial for the success of MCS applications. However, due to either a large number of candidates or their dynamic natures, selecting reliable workers poses big challenges to the MCS platform. Evidently, workers' reputation-based characteristics, such as trustability and credibility, are also pivotal for the worker selection in MCS, but they were often neglected in previous literature. In this article, aiming at addressing the above challenges, we propose a new privacy-preserving worker selection scheme based on the probabilistic skyline computation technique. Specifically, our proposed scheme is characterized by: 1) assigning a trustability score to each worker based on his/her past performance without revealing his/her sensitive information and 2) efficiently selecting a subset of reliable workers for a particular task. Detailed security analysis shows that our proposed scheme can preserve workers' privacy. In addition, performance evaluations via extensive simulations are conducted, and the results also demonstrate its effectiveness and efficiency for reliable worker selection in MCS applications.**

***Index Terms*—Encrypted integer comparison, mobile crowdsensing (MCS), probabilistic skyline computation, worker selection.**

I. Introduction

NOWADAYS, the Internet of Things (IoT) has been deemed one of the most significant drivers for the fourth industrial revolution [1], [2]. It is predicted that the number of smart devices will be 30 billion by 2020 and 500 billion by 2030 [3]. The fast proliferation of smart devices enables a new paradigm of sensing network, known as mobile crowdsensing (MCS), which promotes a large variety of applications, such as environmental monitoring [4], traffic management [5], [6], healthcare provisioning [7], and location-based social recommendation [8], [9]. In MCS applications, a crowd of mobile users, namely, workers, are recruited by the MCS platform to outsource their sensory data for certain tasks. By utilizing the dynamics and mobility of numerous workers, MCS services support large-scale sensing applications, which can hardly be accomplished by traditional sensing networks [10].

In MCS services, worker selection can allocate the sensing tasks to the proper workers. It has been regarded as one of the fundamental issues in MCS since the reliability of the workers and the trustability of their sensing data play significant roles in the service quality [1]. Recently, there has been extensive research on studying worker selection in MCS services [11]–[16]. However, most of them evaluate workers based on the information provided by the workers. Such studies are infeasible for determining workers' real qualifications if there are strategic and selfish workers who try to maximize their profits by providing inaccurate information. In real-world MCS applications, workers' sensing performance is always task-by-task due to the limitations of sensing equipment and the dynamics of sensing conditions. Therefore, it is significant to measure workers' trustability and reliability based on their previous unstable behaviors. In addition, workers' personal information is inevitable to be shared; thus their privacy could be exposed during the process of worker selection. As a result, an ideal MCS platform should take workers' trustability into account, and preserve workers' privacy as well.

Motivated by the above-mentioned issues, in this article, we propose an effective and privacy-preserving worker selection scheme for MCS services. Our proposed scheme is inspired by probabilistic skyline computation [17], which is useful and practical in multicriteria decision analysis. Specifically, we design a privacy-preserving approach for securely calculating workers' probabilistic skyline value based on their historical reviews, and this value is considered as workers' trustability and will be used for worker selection. Specifically, the main contributions of this article are threefold as follows.

1) We propose a skyline-based scheme for worker selection in MCS applications. To the best of our knowledge, this is the first skyline-based approach for MCS studies. Based on the nature of skyline computation, the selected workers are not dominated by other workers

Manuscript received March 3, 2020; revised July 16, 2020; accepted August 19, 2020. Date of publication August 25, 2020; date of current version December 11, 2020. The work of Rongxing Lu was supported by NSERC under Grant Rgpin 04009. The work of Ali A. Ghorbani was supported by the National Science and Engineering Research Council of Canada (NSERC) through the Discovery Grant and Canada Research Chair. This work was supported in part by NSF of Zhejiang Province under Grant LZ18F020003, and in part by NSFC under Grant U1709217 and Grant 61672411. *(Corresponding author: Rongxing Lu.)*

Xichen Zhang, Rongxing Lu, and Ali A. Ghorbani are with the Canadian Institute for Cybersecurity, Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada (e-mail: xichen.zhang@unb.ca; rlu1@unb.ca; ghorbani@unb.ca).

Jun Shao is with the School of Computer and Information Engineering, Zhejiang Gongshang University, Hangzhou 310018, China (e-mail: chn.junshao@gmail.com).

Hui Zhu is with the School of Cyber Engineering, Xidian University, Xi'an 710126, China (e-mail: zhuhui@xidian.edu.cn).

Digital Object Identifier 10.1109/JIOT.2020.3019326

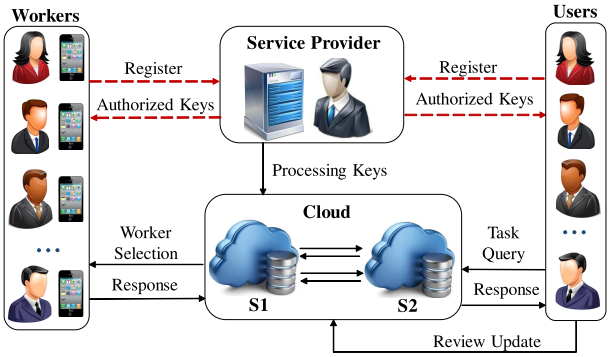


Fig. 1. Proposed system model contains a service provider SP, two cloud servers S_1 and S_2, authorized users $\mathcal{U}$, and participant workers $\mathcal{W}$.

on all the considered attributes. This property has not been achieved in the previous worker selection schemes in MCS applications.

2) We design a privacy-preserving probabilistic skyline technique for calculating workers' trustability based on their past performance. The method can aggregate the fluctuated historical reviews and output a reliable trustability value for the workers. To protect workers' personal information from being disclosed during the calculation, we design a noninteractive encrypted integer comparison protocol (NIEC), namely, *NIEC*, which can compare the relation between two integers in a certain range without revealing their real values and even the difference of the values.
3) We analyze the security of the proposed scheme and show its privacy-preserving effectiveness for workers' personal information. Moreover, we conduct extensive performance evaluations in the experiment, which validates the efficiency and reliability of the process of worker selection.

The remainder of this article is organized as follows. In Section II, we introduce our system model, security model, and design goals. In Section III, we describe some preliminaries. In Section IV, we propose our scheme in details. Then, in Sections V and VI, we present the security analysis and performance evaluation, followed by the related work in Section VII. Finally, we recap the conclusions in Section VIII.

II. Models and Design Goals

In this section, we formalize our system model, security model, and identify our design goals.

A. System Model

Our system model mainly consists of four entities, namely, a service provider (SP), a cloud platform with two servers $\mathcal{CS} = \{S_1, S_2\}$, some users who request the task $\mathcal{U} = \{u_1, u_2, \ldots\}$, and some workers $\mathcal{W} = \{w_1, w_2, \ldots\}$, as shown in Fig. 1.

1) *Service Provider (SP):* SP is the service organizer who is responsible for bootstrapping the entire system. SP generates and distributes keys to different authorized entities so that a certain task can be completed cooperatively. After that, SP just stays offline.
2) *Cloud Servers $\mathcal{CS} = \{S_1, S_2\}$:* There are two cloud servers (S_1, S_2) in our system model. After receiving a user's MCS task query Q, S_1 and S_2 will work together to select a subset of reliable workers based on workers' asking price ρ, task similarity μ, and probabilistic skyline value *PS-score*. After each sensing task is completed, S_1 and S_2 will work together to update each worker's *PS-score* based on users' reviews in an offline manner.
3) *Users $\mathcal{U} = \{u_1, u_2, \ldots\}$:* In MCS services, authorized users are the task initiators. At first, a user sends a task query Q to the cloud servers. After receiving the sensing data, the user is required to provide reviews for evaluating workers' performance. The metrics in the reviews may include the accuracy of the sensing data and the reasonability of the asking price.
4) *Workers $\mathcal{W} = \{w_1, w_2, \ldots\}$:* In MCS services, workers are the participants who wish to conduct a certain MCS task. They need to periodically send their approximate locations Δ_w to S_1. In order to be selected, workers are required to submit their sensing information to $\mathcal{CS}$, e.g., their asking price for the task ρ and the task similarity μ.

B. Security Model

In our security model, we consider the SP is trustable. However, S_1 and S_2 are semihonest, which means both of them strictly follow the protocol procedure, yet may be curious to learn additional information, i.e., workers' historical sensing performance. In addition, there is no collusion between S_1 and S_2. We assume that the users in our model are honest and their reviews for assessing workers' performance are correct and unbiased. For the workers, we assume that they are strategic and selfish for maximizing their profits. However, in order to be selected by the platform again in the future, they still need to provide the correct information and the reasonable asking cost as much as possible. It is worth noting that there may exist outside attackers who can exploit the vulnerabilities of the MCS system, but they are beyond the scope of this article and will be exploited in our future work.

C. Design Goals

In this work, our goal is to select reliable and suitable workers for conducting a sensing task. In specific, the following two objectives should be satisfied.

1) *Privacy Preservation:* Our work needs to consider the leakage of workers' privacy since a lot of sensitive information could be disclosed and inferred from the designed model. In the process of worker selection, it is unavoidable to share workers' personal information. Consequently, there may exist adversaries who are highly motivated to deduce workers' historical sensing performance [12], and a hostile MCS platform can exploit workers' sensing capacity for unfair competition. Therefore, each worker's historical reviews should not be disclosed to S_1, S_2, and other workers.

TABLE I
SUMMARY OF IMPORTANT NOTATIONS USED IN THIS ARTICLE

Notation	Definition
System Notations	
SP	Service provider
$CS = \{S_1, S_2\}$	Two Cloud servers
Worker and User Notations	
$\mathcal{U} = \{u_1, u_2, \cdots\}$	Authorized users
$\mathcal{W} = \{w_1, w_2, \cdots\}$	Registered workers
$\gamma = \{\gamma_1, \gamma_2, ..., \gamma_d\}$	A review with d dimensions
$[\gamma] = \{[\gamma_1], [\gamma_2], ..., [\gamma_d]\}$	An encrypted d-dimension review
PS-score	Worker's probabilistic skyline score
ρ	Worker's asking price
μ	Worker's task similarity
Encryption Notations	
m_1, m_2	Two positive integers
p, q	Two large prime numbers, $n = pq$
K	Authorized key
H_a	A cryptographic hash function
$pk = (g, h, n, H_a)$	Public key
$sk = (p)$	Processing key
$pub = \{pk, H_a\}$	System key
Simulation Notations	
l	Number of total workers
δ	Number of candidate workers
N_s	Number of selected skyline workers
N_a	Number of all skyline workers
t	The task similarity threshold
R	Location filtering threshold
v	Number of reviews per worker
d	Number of questions per review

2) *Efficiency:* An efficient worker selection process is significant for supporting real-world MCS applications, and we need to keep the computational cost of the skyline query as low as possible. Given a set of workers with size δ, the time complexity for traditional skyline computation is $O(\delta^2)$ since each pair of workers should be compared. In this work, we aim to achieve a better efficiency such that the time complexity does only depend on N_s and the ratio $P_s = (N_s/\delta)$, where N_s is the number of skyline workers to be selected.

III. PRELIMINARIES

In this section, we introduce the background of the skyline and probabilistic skyline computation. Table I shows a summary of the notations used in this work.

A. Skyline Computation

Given a data set $D = \{\gamma_1, \gamma_2, \ldots, \gamma_D\}$. Each tuple is in d dimensional space, where $\gamma = (\gamma[1], \gamma[2], \gamma[3], \ldots, \gamma[d])$. Without loss of generality, we assume that for each dimension, smaller values are more preferable. Let γ_a and γ_b be two different tuples in D. We consider γ_a dominates γ_b, denoted by $\gamma_a \prec \gamma_b$, if for all i, $\gamma_a[i] \leq \gamma_b[i]$, and for at least one j, $\gamma_a[j] < \gamma_b[j]$. The skyline points are those tuples that are not dominated by others in D.

B. Probabilistic Skyline Computation

The concept of the probabilistic skyline is first introduced in [17] for uncertain data. It indicates the probability of an object to be selected in the skyline and can be used to calculate workers' trustability in this work.

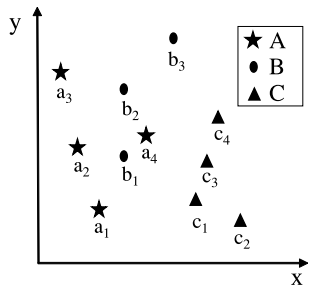


Fig. 2. Example of three objects with different numbers of tuples for determining dominance relation and computing probabilistic skyline.

TABLE II
DOMINANCE RELATION BETWEEN OBJECTS A AND B

	a_1	a_2	a_3	a_4
b_1	$a_1 \prec b_1$	NA	NA	$b_1 \prec a_4$
b_2	$a_1 \prec b_2$	$a_2 \prec b_2$	NA	NA
b_3	$a_1 \prec b_3$	$a_2 \prec b_3$	$a_3 \prec b_3$	$a_4 \prec b_3$

1) Dominance Relation Between Uncertain Objects: There are two objects A and B, where A contains u tuples (i.e., $A = \{a_1, a_2, \ldots, a_u\}$), and B contains v tuples (i.e., $B = \{b_1, b_2, \ldots, b_v\}$). For simplicity, we assume that each tuple in an object has the same probability to occur. For $a_i \in A$ and $b_j \in B$, the probability that $A \prec B$ can be calculated by the following [17]:

$$\Pr(A \prec B) = \frac{1}{u \times v} \sum_{i=1}^{u} \left| \left\{ b_j \in B | a_i \prec b_j \right\} \right|. \tag{1}$$

Example 1: In Fig. 2, A, B, and C are three workers, and each of them has 4, 3, and 4 historical reviews, respectively. In this example, we consider that each review has two dimensions (e.g., accuracy of the sensing data, and the reasonability of the ask price), and the smaller values are preferable. From Table II, we know that there are totally seven times that A dominates B, they are $\{a_1 \prec b_1, a_1 \prec b_2, a_1 \prec b_3, a_2 \prec b_2, a_2 \prec b_3, a_3 \prec b_3, a_4 \prec b_3\}$. Based on (1), we can simply get $\Pr(A \prec B) = (7/3 \times 4) = (7/12)$; B only dominates A with $\{b_1 \prec a_4\}$, so $\Pr(B \prec A) = (1/3 \times 4) = (1/12)$.

2) Probabilistic Skyline Computation: The probabilistic skyline value for object A can be calculated as

$$\text{PS-score} = \frac{1}{|A|} \sum_{a_i \in A} \Pr(a_i) \tag{2}$$

where for $a_i \in A$, $\Pr(a_i)$ is the skyline probability for tuple a_i and can be calculated by the following:

$$\Pr(a_i) = \prod_{A} \left(1 - \frac{\left| b_j \in B | b_j \prec a_i \right|}{|B|} \right). \tag{3}$$

Example 2: Let us consider the example in Fig. 2 again. For worker B, b_1 is dominated by a_1, b_2 is dominated by a_1 and a_2, and b_3 is dominated by a_1, a_2, a_3, a_4, and c_2. Then, $\Pr(b_1) = 1 - (1/4) = (3/4)$, $\Pr(b_2) = 1 - (2/4) = (1/2)$, and $\Pr(b_3) = (1 - (4/4)) \times (1 - (1/4)) = 0$. The probabilistic skyline value for worker B can be calculated as PS-score $= (1/|B|)(\Pr(b_1) + \Pr(b_2) + \Pr(b_3)) = (1/3) \times ((3/4) + (1/2) +$

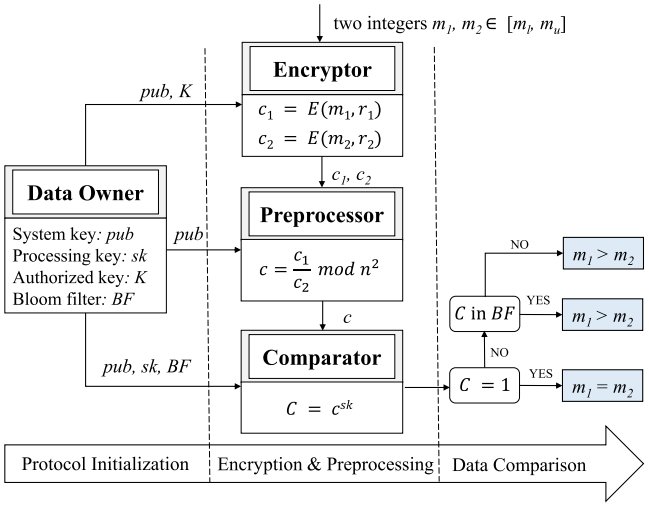


Fig. 3. Overview of the encrypted integer comparison protocol NIEC.

Algorithm 1: BF Generation for Comparing Two Integers

Input : A N-bit length array $A[N]$ where all bits are initialized to 0, k independent hash functions $\mathcal{H} = \{H_1, H_2, \ldots, H_k\}$, H_i:$\{0,1\}^* \to \{0, 1, \ldots, N-1\}$ for each $H_i \in \mathcal{H}$, a cryptographic hash function H_a, public key $pk = (g, h, n)$, processing key sk, and authorized key K

Output: A bloom filter BF that can indicate $m_1 > m_2$ for two $m_1, m_2 \in [m_l, m_u]$

for $m_1 = m_l$ *to* m_u **do**
  for $m_2 = m_l$ *to* m_u **do**
    if $m_1 > m_2$ **then**
      $e = g^{p(H_a(m_1\|K) - H_a(m_2\|K))} \bmod n^2$;
      for $j = 1$ *to* k **do**
        set $A[H_j(e)] = 1$;
      end
    end
  end
end
return $A[N]$;

0) = (5/12). Similarly, we can compute the probabilities that B and C are in the skyline.

IV. Proposed Scheme

In this section, we present our skyline-based privacy-preserving scheme for worker selection in MCS services, which mainly consists of four phases: 1) system initialization; 2) task request and response; 3) worker selection; and 4) review update. Before delving the details of the proposed scheme, we first describe an encrypted integer comparison protocol, called *NIEC*, as follows, which can be used to non-interactively compare two integers without privacy leakage.

A. Noninteractive Encrypted Integer Comparison Protocol

For two positive integers $m_1, m_2 \in [m_l, m_u]$ (e.g., the lower bound $m_l = 1$ and the upper bound $m_u = 100$), the goal of *NIEC* is to check whether $m_1 = m_2, m_1 > m_2$, or $m_1 < m_2$, without revealing their actual values. The overall settings of *NIEC* are shown in Fig. 3, which consists of *data owner*, *encryptor*, *preprocessor*, and *comparator*. The details of *NIEC* are described as follows.

1) Protocol Initialization: Given a security parameter $\kappa \in \mathbb{Z}^+$, the *data owner* randomly selects two large prime numbers p and q, such that the bit length $|p| = |q| = \kappa$ and let $n = pq$. Then, the *data owner* randomly chooses $g \in \mathbb{Z}_{n^2}^*$ to guarantee the order of g is n, i.e., $g^n \equiv 1 \bmod n^2$, and computes $h = g^q \bmod n^2$. The public key is $pk = (g, h, n)$, and the processing key is $sk = (p)$. After that, the *data owner* creates an authorized key K and chooses a cryptographic hash function H_a, and uses Algorithm 1 to create a bloom filter BF. Finally, the *data owner* publishes the system key pub $= \{pk, H_a\}$, and, respectively, distributes the authorized key K to the *encryptor*, the processing key sk and BF to the *data comparator*.

2) Data Encryption: For two integers $m_1, m_2 \in [m_l, m_u]$, the *encryptor* first chooses two random numbers $r_1, r_2 \in \mathbb{Z}_n^*$ and then computes $c_1 = E(m_1, r_1) = g^{H_a(m_1\|K)} h^{r_1} \bmod n^2$ and $c_2 = E(m_2, r_2) = g^{H_a(m_2\|K)} h^{r_2} \bmod n^2$. Then, it sends (c_1, c_2) to the *data preprocessor*.

3) Data Preprocessing: Upon receiving (c_1, c_2) from the *encryptor*, the *data preprocessor* computes $c = (c_1/c_2) \bmod n^2$, and then sends c to the *data comparator*.

4) Data Comparison: Upon receiving c, the *data comparator* computes $C = c^p \bmod n^2$. If $C = 1 \bmod n^2$, then $g^{p(H_a(m_1\|K) - H_a(m_2\|K))} \bmod n^2 = 1$, which means that $H_a(m_1\|K) - H_a(m_2\|K) = 0$ and thus $m_1 = m_2$. Else if $C \neq 1 \bmod n^2$, the *data comparator* uses the bloom filter BF to check C, if C is in BF, then $m_1 > m_2$, otherwise, $m_1 < m_2$. Note that if there is no collusion between the *data preprocessor* and *data comparator*, the messages m_1 and m_2 cannot be recovered from c_1 and c_2.

B. Description of Our Proposed Scheme

1) System Initialization: As SP is the service organizer, it is reasonable to consider SP is responsible for bootstrapping the entire system. The details of system initialization are as follows.

1) First, SP plays the role of the *data owner* in the *NIEC* protocol, generates public key $pk = (g, h, n)$, processing key $sk = (p)$, a bloom filter BF, and an authorized key K, and chooses a hash function H_a by the approaches in Section IV-A. After that, SP will publish the system key pub $= \{pk, H_a\}$.
2) Then, both users and workers need to register for the MCS platform through SP. After their registrations, SP distributes K to the registered users and workers and sends (sk, BF) to the cloud server S_2 in a secure way.
3) SP maintains two global databases $\mathcal{W}$ and $\mathcal{U}$ for the registered workers and users, respectively, (see in Tables III and IV). After registering a new worker, SP assigns an ID to the worker and adds him/her into $\mathcal{W}$. In addition, each worker's *PS-score* is initialized as 0.5 and is used as the trustability of this worker. $[\gamma]$ denotes workers' encrypted reviews and is initialized as empty. After registering all the workers and users, SP distributes $\mathcal{U}$ and $\mathcal{W}$ to cloud server S_1.
4) For being selected, each registered worker is required to regularly send his/her approximate location Δ_w to S_1. Using the geo-indistinguishability technique introduced

TABLE III
GLOBAL WORKER TABLE $\mathcal{W}$ IN SYSTEM INITIALIZATION

W_{id}	$[\gamma]$	*PS-score*
w_1		0.5
w_2		0.5
w_3		0.5
...		...

TABLE IV
GLOBAL USER TABLE $\mathcal{U}$ IN SYSTEM INITIALIZATION

U_{id}
u_1
u_2
u_3
...

in [18], workers' approximate location Δ_w can be computed by adding some random differential privacy noise to their accurate positions. As a result, Δ_w can achieve the differential privacy property while still releasing approximate information.

After initializing the entire system, SP just stays offline, and will not participate in the MCS services.

2) Task Request and Response: This phase mainly contains three steps: 1) *users' task request*; 2) *location matching*; 3) *workers' task response*.

1) *Users' Task Request:* At the beginning of an MCS task, a user sends a task query Q to S_1, which contains the task location Δ_q and the task content Ω_q, i.e., $Q = \{\Delta_q, \Omega_q\}$.
2) *Location Matching:* After receiving the task query Q, S_1 finds the workers whose location information Δ_w matches the query's location requirement Δ_q. The matched workers are denoted as $\mathcal{W}_\Delta$, and $\mathcal{W}_\Delta \subseteq \mathcal{W}$.
3) *Workers' Task Response:* In this step, S_1 announces the query Q to $\mathcal{W}_\Delta$. Each active worker in $\mathcal{W}_\Delta$ who wishes to conduct the task is required to send the feedback information Λ to S_1. $\Lambda = (\rho, \mu)$, where $\rho \in (0, 1)$ is worker's task similarity and $\mu \in (0, 1)$ is the worker's asking price for the task. ρ is used to indicate the relevancy between the task content and the workers' activities. Usually, a task content Ω_q is announced in the form of a sequence of keywords. A worker can then decide his/her task similarity by information retrieval models, such as text relevance analysis [19]. We assume that smaller ρ indicates higher similarity, so for both ρ and μ, the smaller values are more preferred. Then, given a predefined threshold t, S_1 only selects the workers whose ρ is smaller than t.

After the similarity matching, S_1 joins the workers' *PS-score* with their Λ based on worker ID. The group of active workers is called $\mathcal{W}_\delta$ (see in Table V), such that $\mathcal{W}_\delta \subseteq \mathcal{W}_\Delta$.

3) Worker Selection: In this step, we propose a new skyline computation algorithm (see in Algorithm 2), which can be employed to find N_s skyline workers efficiently. The general idea of our algorithm is introduced as follows. All workers in $\mathcal{W}_\delta$ have three attributes, namely, asking price ρ, task similarity μ, and *PS-score*. First we preprocess and normalize *PS-score* with the equation: PS-score = (PS-score$_{\min}$/PS-score), where PS-score$_{\min}$ is the minimum *PS-score* among all the workers. After that, all the three attributes are ranged from 0 and 1 and the smaller values are preferred. We sort all the workers in a descending order by $S(w_i)$ where $S(w_i) = \rho_i + \mu_i +$PS-score$_i$. Then, the first worker [i.e., with minimum $S(w_i)$] is added in the skyline pool S_{skyline} and is deleted from $\mathcal{W}_\delta$. Next, the worker with minimum $S(w_i)$ is selected and compared with the workers in S_{skyline}. If he/she is not dominated by all the workers in S_{skyline}, then add him/her into S_{skyline} and delete him/her from $\mathcal{W}_\delta$. Otherwise, we can directly delete him/her from $\mathcal{W}_\delta$. This algorithm repeats the aforementioned steps for the remaining workers until either $\mathcal{W}_\delta$ is empty or there are N_s workers in S_{skyline}. Consequently, S_1 announces all the skyline workers to conduct the task. After the sensing task is completed, S_1 collects workers' data and reports it to the user.

Algorithm 2: Skyline-Based Worker Selection

Input : A worker database $\mathcal{W}_\delta$ such that $|\mathcal{W}_\delta| = \delta$, and each worker w_i in $\mathcal{W}_\delta$ can be represented as $w_i = (\rho_i, \mu_i, \textit{PS-score}_i)$; A number N_s which is defined by users
Output: The skyline workers of $\mathcal{W}_\delta$

```
Initialize PQ to an empty minHeap Priority Queue;
for i = 1 to δ do
    PS-score = PS-score_min / PS-score;
    S(w_i) = ρ_i + μ_i + PS-score_i;
    PQ.Insert((S(w_i), w_i));
end
Set S_skyline as an initially empty skyline set;
count = 0;
while PQ is not empty && count < N_s do
    w_min ← PQ.RemoveMin();
    if there is no worker s in S_skyline such that s ≺ w_min then
        S_skyline.Add(w_min);
        count = count + 1;
    end
end
return S_skyline;
```

TABLE V
SUBSET WORKER TABLE $\mathcal{W}_\delta$ AFTER WORKERS' RESPONSE

W_{id}	μ	ρ	*PS-score*
w_1	0.3	0.5	0.5
w_2	0.6	0.8	0.5
w_3	0.8	0.7	0.5
...	...	...	...

4) Review Update: This phase mainly contains two steps: 1) *review submission* and 2) *probabilistic skyline computation*.

1) *Review Submission:* After receiving the worker's data, the user is required to rate each worker's performance by filling in an evaluation form. Fig. 4 shows a simple example of a such form, which contains four evaluation questions. Users can use a bootstrap slider to select the satisfactory level of each question. The level is an integer $\in [m_l, m_u]$ (e.g., $m_l = 1$ and $m_u = 100$), where m_l means strongly agree and m_u means strongly disagree. So any review can be considered as a fixed-dimension vector $\gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_d\}$, such that d is the number of questions, and $\gamma_i \in [m_l, m_u]$ is the integer that indicates the satisfaction degree.

Algorithm 3: RateDominance(γ_1, γ_2)

Note: given two original reviews $\gamma_1 = \{x_1, x_2, \ldots, x_d\}$ and $\gamma_2 = \{y_1, y_2, \ldots, y_d\}$, the corresponding encrypted reviews $[\gamma_1] = \{[x_1], [x_2], \ldots, [x_d]\}$ and $[\gamma_2] = \{[y_1], [y_2], \ldots, [y_d]\}$ are computed with public key pk in NIEC protocol in Section IV-A;
Input : the processing key $sk = p$; the bloom filter BF; an encrypted vector $V_c = (c_1, c_2, \ldots, c_d)$, where each $c_i = [x_i]/[y_i] \bmod n^2$, and $[x_i]$ belongs to $[\gamma_1] = ([x_1], [x_2], \ldots, [x_d])$, $[y_i]$ belongs to $[\gamma_2] = ([y_1], [y_2], \ldots, [y_d])$
Output: 0, 1, 2

$token_{xy} = true$; // indicating $\gamma_1 \prec \gamma_2$
$token_{yx} = true$; // indicating $\gamma_2 \prec \gamma_1$
$count = 0$;
for $i = 1$ *to* d **do**
  $C_i = c_i^p \bmod n^2$;
  if C_i *is in BF* // indicating $x_i > y_i$, so $\gamma_1 \not\prec \gamma_2$
  then
    $token_{xy} = false$;
  if $C_i == 1 \bmod n^2$ // indicating $x_i = y_i$
  then
    $count++$;
  if C_i *is not in BF* && $C_i \neq 1 \bmod n^2$ // indicating $x_i < y_i$, so $\gamma_2 \not\prec \gamma_1$
  then
    $token_{yx} = false$;
if $token_{xy} == true$ && $count < d$ **then**
  return *1 indicating* $\gamma_1 \prec \gamma_2$
if $token_{yx} == true$ && $count < d$ **then**
  return *2 indicating* $\gamma_2 \prec \gamma_1$
return *0 indicating* $\gamma_1 \not\prec \gamma_2$ *and* $\gamma_2 \not\prec \gamma_1$

Then, users need to encrypt each answer in γ using pk based on the *NIEC* protocol. After encryption, each encrypted review $[\gamma]$ will be sent to S_1 and will be added in $\mathcal{W}$ by S_1. $[\gamma]$ can be used for computing and updating the *PS-score* for each worker. Table VI shows the global worker table $\mathcal{W}$ after updating each worker's review. $[\gamma]_{i,j} = \{[\gamma_1]_{i,j}, [\gamma_2]_{i,j}, \ldots, [\gamma_d]_{i,j}\}$, where i means the worker ID, j means the jth review for the worker. Different workers may have different numbers of reviews.

2) *Updating Probabilistic Skyline:* In our proposed scheme, the *PS-score* is updated in an offline manner. After receiving all the encrypted reviews from users, S_1 and S_2 work together to update each worker's *PS-score* in offline. More specifically, given $[\gamma]_{w_i} = \{[\gamma_1]_{w_i}, [\gamma_2]_{w_i}, \ldots, [\gamma_d]_{w_i}\}$ from worker w_i and $[\gamma]_{w_j} = \{[\gamma_1]_{w_j}, [\gamma_2]_{w_j}, \ldots, [\gamma_d]_{w_j}\}$ from worker w_j, S_1 will first compute $V_c = (c_1, c_2, \ldots, c_d)$ where $c_s = [([\gamma_s]_{w_i})/([\gamma_s]_{w_j})]$ for $s \in (1, d)$. After computing for all pairs of workers from $\mathcal{W}$, S_1 sends V_c to S_2. Then, S_2 can determine the dominance relation between all pairs of workers using Algorithm 3, update each worker's *PS-score* and fill in the 2-D array $\mathcal{W}_s[l][l+1]$ in Algorithm 4. Finally, S_2 returns $\mathcal{W}_s[l][l+1]$ to S_1.

V. Security Analysis

This section evaluates the security properties and shows how workers' reviews can be protected from being disclosed in the proposed scheme.

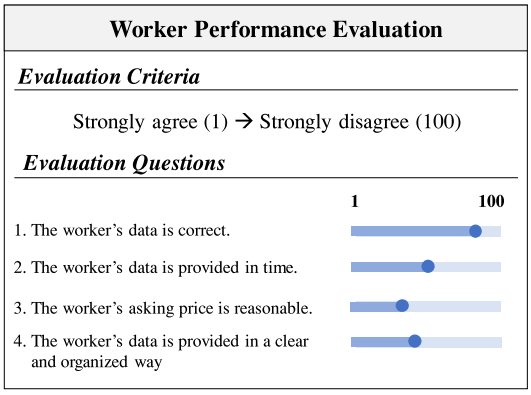


Fig. 4. Simple example of the worker evaluation form with four evaluation questions. For each question, a bootstrap slider can be used to answer users' satisfaction level (from 1 to 100) for a task in terms of a certain aspect.

TABLE VI
Global Worker Table $\mathcal{W}$ After Updating Reviews

W_{id}	$[\gamma]$	*PS-score*
w_1	$[\gamma]_{1,1}$ ...	0.5
w_2	$[\gamma]_{2,1}$ ...	0.5
w_3	$[\gamma]_{3,1}$ ...	0.5
...	...	...

Algorithm 4: WorkerDominance(w_1, w_2)

Input : Two worker such that $w_1 = \{[\gamma]_{1,1}, [\gamma]_{1,2}, \ldots, [\gamma]_{1,u}\}$, $w_2 = \{[\gamma]_{2,1}, [\gamma]_{2,2}, \ldots, [\gamma]_{2,v}\}$, and $|w_1| = u$, $|w_2| = v$
Output: $Pr(w_1, w_2)$ indicating the probability that $w_1 \prec w_2$, and $Pr(w_2, w_1)$ indicating the probability that $w_2 \prec w_1$

$count_1, count_2 = 0$;
for $i = 1$ *to* u **do**
  for $j = 1$ *to* v **do**
    if *RateDominance*($[\gamma]_{1,i}, [\gamma]_{2,j}$) = *1* // indicating $[\gamma]_{1,i} \prec [\gamma]_{2,j}$
    then
      $count_1++$;
    if *RateDominance*($[\gamma]_{1,i}, [\gamma]_{2,j}$) = *2* // indicating $[\gamma]_{2,j} \prec [\gamma]_{1,i}$
    then
      $count_2++$;
$Pr(w_1, w_2) = \frac{count_1}{u \times v}$;
$Pr(w_2, w_1) = \frac{count_2}{u \times v}$;
return $Pr(w_1, w_2)$ *and* $Pr(w_2, w_1)$;

1) *Privacy Preservation of Users' Reviews:* Our goal is to enable S_1 and S_2 to compute workers' *PS-score* without disclosing the exact answers of their reviews. On the one hand, according to the *NIEC* protocol, S_1 plays the role of the *data preprocessor*, and has access to $pk = (g, h, n)$. For one original review answers m_1 and m_2, S_1 knows their corresponding ciphertexts $c_1 = g^{H_a(m_1 \| K)} h^{r_1} \bmod n^2$ and $c_2 = g^{H_a(m_2 \| K)} h^{r_2} \bmod n^2$. However, without knowing $sk = (p)$ and the random numbers (r_1, r_2), S_1 cannot remove the random factors $h^{r_1} \bmod n^2$ and $h^{r_2} \bmod n^2$ in the ciphertexts. Furthermore, S_1 has no information about H_a and K. As a result, S_1 cannot recover any plaintexts of the original reviews answers in the proposed scheme.

Algorithm 5: *PS-score* Calculation

Input : The global worker table $\mathcal{W}$, $|\mathcal{W}| = l$
Output: A 2-D array $\mathcal{W}_s[l][l+1]$ that contains the updated *PS-score* for each worker in $\mathcal{W}$

```
Initialize W_s[l][l+1] to an initially empty 2-D array;
for i = 1 to l do
    Initialize PS-score = 1;
    for j = 1 to l && j ≠ i do
        if W_s[i][j] is empty then
            p_1, p_2 = WorkerDominance(w_i, w_j);
            W_s[j][i] = p_1;
            W_s[i][j] = p_2;
        PS-score = PS-score × (1 − W_s[i][j]);
    W_s[i][l+1] = PS-score ;
return W_s[l][l+1]
```

On the other hand, S_2 plays the role of the *data comparator* in the *NIEC* protocol, and has access to $sk = (p)$. Once receiving $c = (c_1/c_2)$ from S_1, S_2 can calculate $C = c^p = g^{p(H_a(m_1\|K)-H_a(m_2\|K))} \bmod n^2$. However, S_2 cannot obtain the original values of m_1 and m_2 for the following two reasons: a) only knowing $pk = (g, h, n)$ and $sk = (p)$, due to the discrete logarithm problem, it is hard for S_2 to compute $p(H_a(m_1\|K) - H_a(m_2\|K))$ and b) we know each review is in a certain range, e.g., $m_1, m_2 \in [m_l, m_u]$. Even though S_2 might know the lower boundary m_l and the upper boundary m_u of the reviews answers m_1 and m_2, without knowing H_a and K, S_2 cannot calculate $H_a(m_1\|K)$ and $H_a(m_2\|K)$. So S_2 cannot brute force the plaintexts of m_1 and m_2 by checking all their possible values either. Therefore, the reviews are privacy preserving for S_2 as well.

2) *Privacy Preservation of the Difference Between Users' Reviews:* Given any two plaintexts m_1 and m_2, neither S_1 nor S_2 knows the difference of them, i.e., $m_1 - m_2$. More specifically, on the one hand, S_1 can only know $c = (c_1/c_2) \bmod n^2 = g^{(H_a(m_1\|K)-H_a(m_2\|K))}h^{r_1-r_2} \bmod n^2$. Without knowing H_a, K, h, and random numbers (r_1, r_2), S_1 cannot determine $m_1 - m_2$ even though S_1 might know (m_l, m_u). In addition, given another pair of review answers m'_1 and m'_2, we assume that S_1 knows (m'_1, m'_2) and the corresponding ciphertexts $(c'_1, c'_2, c', c_1, c_2)$ such that $c'_1 = g^{H_a(m'_1\|K)}h^{r'_1} \bmod n^2$, $c'_2 = g^{H_a(m'_2\|K)}h^{r'_2} \bmod n^2$ and $c' = (c'_1/c'_2) = g^{(H_a(m'_1\|K)-H_a(m'_2\|K))}h^{r'_1-r'_2} \bmod n^2$. However, S_1 cannot obtain whether $m'_1 - m'_2 = m_1 - m_2$ by simply comparing c and c' due to the existence of random factors $(h^{r_1} \bmod n^2, h^{r_2} \bmod n^2, h^{r'_1} \bmod n^2, h^{r'_1} \bmod n^2)$. On the other hand, as we mentioned before, S_2 cannot recover $p(H_a(m_1\|K) - H_a(m_2\|K))$ from $C = g^{p(H_a(m_1\|K)-H_a(m_2\|K))} \bmod n^2$, which means S_2 cannot determine $m_1 - m_2$ either. Next, let us consider the following scenario. We assume for two pairs of plaintexts (m_1, m_2) and (m'_1, m'_2), S_2 knows (m'_1, m'_2, C, C') such that $C' = g^{p(H_a(m'_1\|K)-H_a(m'_2\|K))} \bmod n^2$. Still, S_2 cannot decide whether $m'_1 - m'_2 = m_1 - m_2$ or not, because even though $C = C'$ might hold [i.e., $H_a(m'_1\|K) - H_a(m'_2\|K) = H_a(m_1\|K) - H_a(m_2\|K)$ holds], the differences of their original values are not the same, i.e., $m'_1 - m'_2 \neq m_1 - m_2$. Therefore, S_2 cannot obtain whether $m_1 - m_2 = m'_1 - m'_2$ by only checking whether $C = C'$ or not. Furthermore, based on C and C', S_2 can calculate $(C/C') = [(g^{p(H_a(m_1\|K)-H_a(m_2\|K))})/(g^{p(H_a(m'_1\|K)-H_a(m'_2\|K))})] \bmod n^2 = g^{p(H_a(m_1\|K)-H_a(m_2\|K)-H_a(m'_1\|K)+H_a(m'_2\|K))} \bmod n^2$. Even though $H_a(m_1\|K) - H_a(m_2\|K) - H_a(m'_1\|K) + H_a(m'_2\|K)$ might happen to be 0 and (C/C') might happen to be 1, S_2 cannot determine whether $m'_1 - m'_2 = m_1 - m_2$. This is because after using the hash function H_a, $m'_1 - m'_2 = m_1 - m_2$ does not mean $H_a(m_1\|K) - H_a(m_2\|K) = H_a(m'_1\|K) - H_a(m'_2\|K)$, and *vice versa*.

In summary, both the real values of the review answers and the difference of any pair of review answers are privacy preserving for both S_1 and S_2 as long as there is no collusion between them. In addition, by introducing the hash function H_a and the authorized key K, our proposed scheme can effectively resist the known-plaintext attacks.

VI. Performance Evaluation

In this section, we study the effectiveness of our proposed scheme using a custom simulator built-in Java programming language. First, we assess the performance of the proposed scheme theoretically in terms of storage and computational overhead. Second, a series of experiments are being conducted to investigate how our scheme's performance varies across different experimental settings. The performance metrics are: 1) the number of candidate workers; 2) the running time for selecting skyline workers and updating the *PS-score* for all the workers; and 3) the *PS-score* for the selected skyline workers. The running time can be used to examine the efficiency of the proposed scheme, the number of candidate workers and the *PS-score* for the selected workers are good indicators for the effectiveness of the scheme, i.e., how many workers can be selected from, and how reliable and trustable the selected workers are.

A. Theoretical-Based Analysis

1) Storage Overhead: Assume that there are totally l registered workers in $\mathcal{W}$. Each worker w_i contains v reviews, and each review $\gamma = \{\gamma[1], \gamma[2], \ldots, \gamma[d]\}$ has d dimensions. Each dimension $\gamma[i]$ for $i \in (1, d)$ needs to be encrypted before outsourcing to $\mathcal{CS}$. So the overall storage overhead for all the encrypted reviews is $\sum_{i=1}^{l}\sum_{j=1}^{v}(d \cdot \text{Len})$, where Len = 2048 is the length of the ciphertext for each dimension.

2) Computational Cost: Next, we discuss the computation cost for the online skyline computation and the offline probabilistic skyline computation in our proposed scheme.

1) *Computational Cost for Skyline Computation:* The time complexity for building a minHeap queue in Algorithm 2 for all the candidate workers is $O(\delta)$, where δ is the size of W_δ. Then, the time complexity for skyline computation depends on N_s and P_s, where $P_s = (N_a/\delta)$ for N_a is the number of all the skyline workers in W_δ.

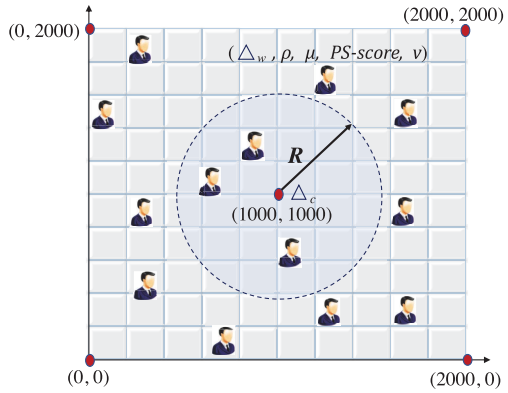


Fig. 5. Overall illustration of the experimental simulation and parameter settings.

TABLE VII
SUMMARY OF SIMULATION SETTINGS

Parameters	Settings
Simulation area	$\mathcal{A}$ = $2000m \times 2000m$
Worker's parameters	
The number of workers	l = 40, 60, 80, 100
Workers' locations	$x_w, y_w \in [0, 2000m]$
Task similarity	$\rho \in [0, 1]$
Asking price	$\mu = Dis(w, \Delta_c)/Dis_{max}$
Probabilistic skyline value	The ranking of *PS-score*
Process parameters	
Location filtering threshold	$R \in [300m, 800m]$
Task similarity filtering threshold	$t = 0.40, 0.42, 0.44, 0.46$
The number of skyline workers	$N_s = 1, 2, 3, 4$
The number of reviews for workers	v froms normal distribution
The number of questions in reviews	d = 5, 10, 15, 20

a) When $N_s = 1$, we can just return the first worker in the queue, and the cost is $O(1)$.
b) When $N_s = 2$, the expected computational cost is $2 \cdot P_s + 3 \cdot (1 - P_s) \cdot P_s + \cdots + l \cdot (1 - P_s)^{\delta-2} \cdot P_s = \sum_{i=0}^{\delta-2} (i + 2) \cdot P_s \cdot (1 - P_s)^i$, which is equal to $1 + (1/P_s)$ when δ is quite large.

Obviously, given a fixed δ, the cost is only inversely proportional to P_s. So our proposed algorithm is more efficient when N_s is small and P_s is large, and this conclusion can also be applied to the scenarios where $N_s > 2$.

2) *Computational Cost for Probabilistic Skyline Computation:* To update all the workers' *PS-score*, the dominance relationship between every two workers needs to be determined. Assume every worker has v reviews and each review has d dimensions, in total, *NIEC* needs to be run $[(l \times (l - 1))/2] \times v^2 \times d$ times. So the overall cost for probabilistic skyline updating depends on both l, v and d.

B. Simulation-Based Analysis

1) Experimental Setup: We simulate a sensing area $\mathcal{A}$ with the size of 2000 m×2000 m and a collection of l workers with varied key parameters, which are demonstrated in Fig. 5. For each worker, the simulating parameters include his/her approximate location $\Delta_w = (x_w, y_w)$, task similarity ρ, asking price μ, *PS-score*, and the number of reviews v.

We assume that all the MCS tasks occurred at Δ_c, the center location of $\mathcal{A}$ with the coordinate of (x_c = 1000 m, y_c = 1000 m). After a user requests a sensing task, S_1 selects the matching workers based on a location filtering threshold R and task similarity filtering threshold t. More specifically, S_1 only selects a subset of workers (denoted as $\mathcal{W}_\delta$) whose distance to Δ_c is less than R, and task similarity ρ is less than t. Then, S_1 finds N_s skyline workers from $\mathcal{W}_\delta$ to execute the sensing task. After the task is finished, the users are required to submit an evaluation form with d reviewing questions to S_1, which contains their opinions for evaluating workers' performance. At the end of the day, after all tasks are finished, S_1 and S_2 update every worker's *PS-score* by the reviews in an offline manner.

Worker's Location Information: In the experiment, each worker is randomly assigned a pair of coordinates (x_w, y_w) such that $x_w, y_w \in$(0, 2000 m).

Worker's Task Similarity: For each MCS task, a worker's task similarity ρ is set to be a random value $\in$ (0, 1). We assume that the smaller the ρ, the higher similarity between the worker's activity and the announced sensing task.

Worker's Asking Price: We assume that a worker's asking price is always proportional to the distance between his/her location Δ_w and the task's location Δ_c. The following equation is used to normalize each worker's asking price: $\mu = \text{Dis}(w, \Delta_c)/\text{Dis}_{\max}$, where $\text{Dis}(w, \Delta_c) = \sqrt{(x_w - x_c)^2 + (y_w - y_c)^2}$, and $\text{Dis}_{\max}$ is the possible maximum distance between Δ_w and Δ_c.

Worker's PS-Score: According to the definition mentioned in Section III-B, each worker's *PS-score* is a value between 0 and 1. However, for simplicity, we use the ranking score of each worker's *PS-score* instead of their real values in our experiment. For instance, if there are 100 workers, then each worker's *PS-score* is assigned to be a random value between 0 and 1. After that, all the workers are sorted decreasingly by *PS-score*, and each worker's *PS-score* is reassigned by his/her own ranking, i.e., a worker's *PS-score* is set to be 1 (100) if this worker has the largest (smallest) *PS-score* value. A smaller *PS-score* means higher probabilistic skyline values.

The detailed parameter settings in the simulations are summarized in Table VII. We perform the experiments with Java programming language and conduct experiments on an Intel Core i7-6700 CPU @3.40-GHz Windows System with 32-GB RAM. In order to more accurately evaluate the running time, the average results are reported. Specifically, the simulations are repeated ten times for updating all the workers' *PS-score*; the rest of the experiment runs 10 000 times.

2) Simulation Results: Fig. 6 shows the number of candidate workers in different experimental settings, i.e., the numbers of workers l = 40, 60, 80, 100, location filtering threshold R = 500, 600, 700, 800 m, and similarity task filtering threshold t = 0.40, 0.42, 0.44, 0.46. From Fig. 6, we can see that the number of candidate workers is proportional to both l, t and R. The total number of active workers in a specific targeted sensing area during a certain period of time is always fixed, so it is significant to set proper thresholds to get a reasonable number of matching candidates. The size of

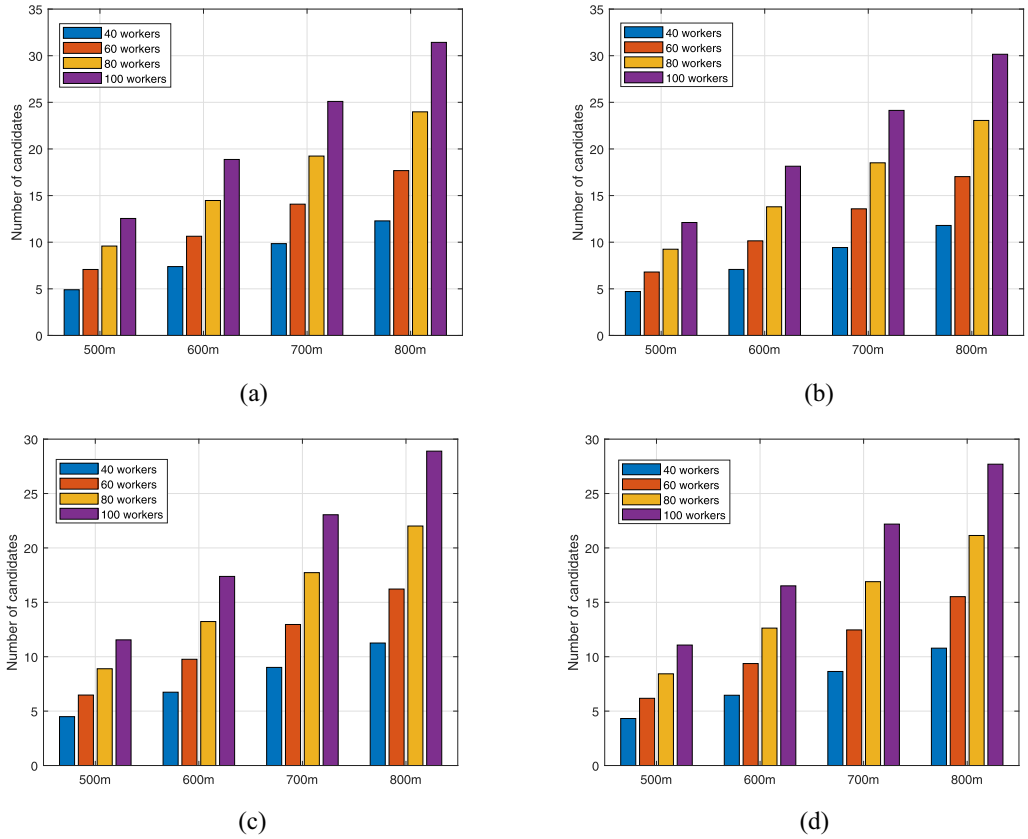


Fig. 6. Number of candidate workers under with different experimental settings. (a) Task similarity threshold t is set to 0.46, the total number of workers l varies from 40 to 100, and the location filtering threshold R varies from 500 to 800 m. (b) Task similarity threshold t is set to 0.44, the total number of workers l varies from 40 to 100, and the location filtering threshold R varies from 500 to 800 m. (c) Task similarity threshold t is set to 0.42, the total number of workers l varies from 40 to 100, and the location filtering threshold R varies from 500 to 800 m. (d) Task similarity threshold t is set to 0.40, the total number of workers l varies from 40 to 100, and the location filtering threshold R varies from 500 to 800 m.

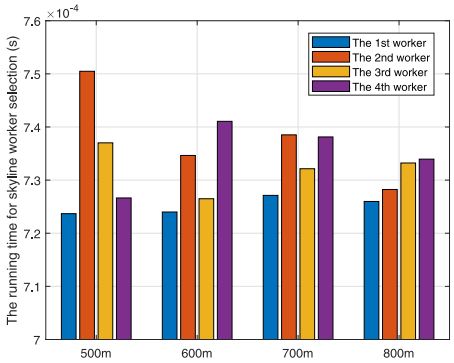


Fig. 7. Running time for selecting skyline workers where the total number of workers l is 100, the number of selected skyline workers N_s is 4, the task similarity threshold t is 0.46, and location filtering threshold R varies from 500 to 800 m.

candidate workers can bring a positive effect on the reliability of the selected skyline workers. The MCS platform can avoid randomness and uncertainty if more candidates are taken into consideration. Especially, when the number of total workers in $\mathcal{A}$ is small, it is better to set relatively larger filtering thresholds to get sufficient candidates.

Fig. 7 compares the running time for selecting different numbers of skyline workers with different parameter settings. In this experiment, N_s = 1, 2, 3, or 4, l is fixed to 100, t is set to 0.46, R varies from 500 to 800 m.

1) We observe that for each scenario, worker selection can be finished in a very short period of time (in the order of 1×10^{-4} s), which validates the efficiency of our proposed scheme.
2) When $N_s = 1$, the running time is the lowest for all the cases. In our proposed scheme, the sum of attributes $S(w_i)$ for all the workers in $\mathcal{W}_\delta$ are calculated at first. Then, the worker with the minimum $S(w_i)$ is added in the skyline pool as the first selected worker (see in Algorithm 2). This algorithm ensures that the worker with the lowest $S(w_i)$ value is the skyline worker without comparing the dominance relationship. As expected, Fig. 7 shows that the running time is the lowest among different experimental settings when N_s = 1.
3) We can see that when R is small, the running time for selecting skyline workers tends to be longer. According to Algorithm 2, the running time for skyline worker selection highly depends on the number of dominance-relationship comparisons between two workers. After all the candidate workers are being sorted by their $S(w_i)$, the 1st worker with minimum $S(w_i)$ will be chosen as the skyline worker. When N_s is larger than 1, the rest of the candidate workers need to be compared with the skyline

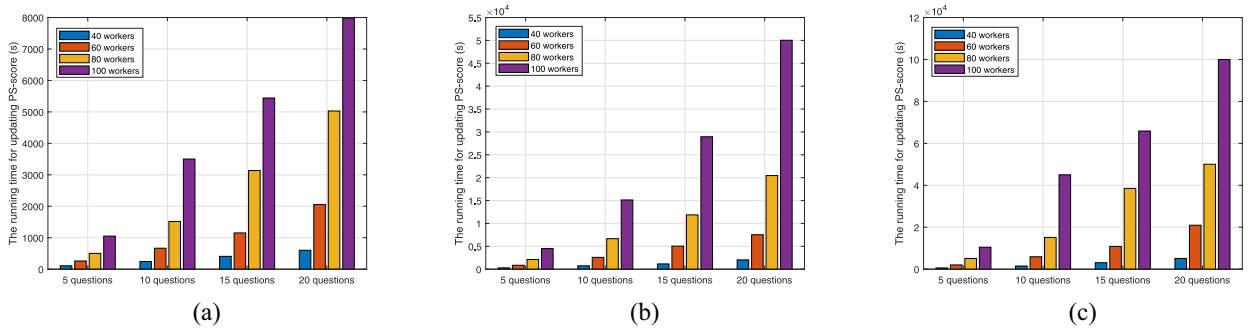


Fig. 8. Running time for updating all workers' *PS-score* varies with different experiment settings. (a) Number of reviews v is set to 4, the total number of workers l varies from 40 to 100, and the number of questions in each review d varies from 5 to 20. (b) Number of reviews v is set to 6, the total number of workers l varies from 40 to 100, and the number of questions in each review d varies from 5 to 20. (c) Number of reviews v is set to 8, the total number of workers l varies from 40 to 100, and the number of questions in each review d varies from 5 to 20.

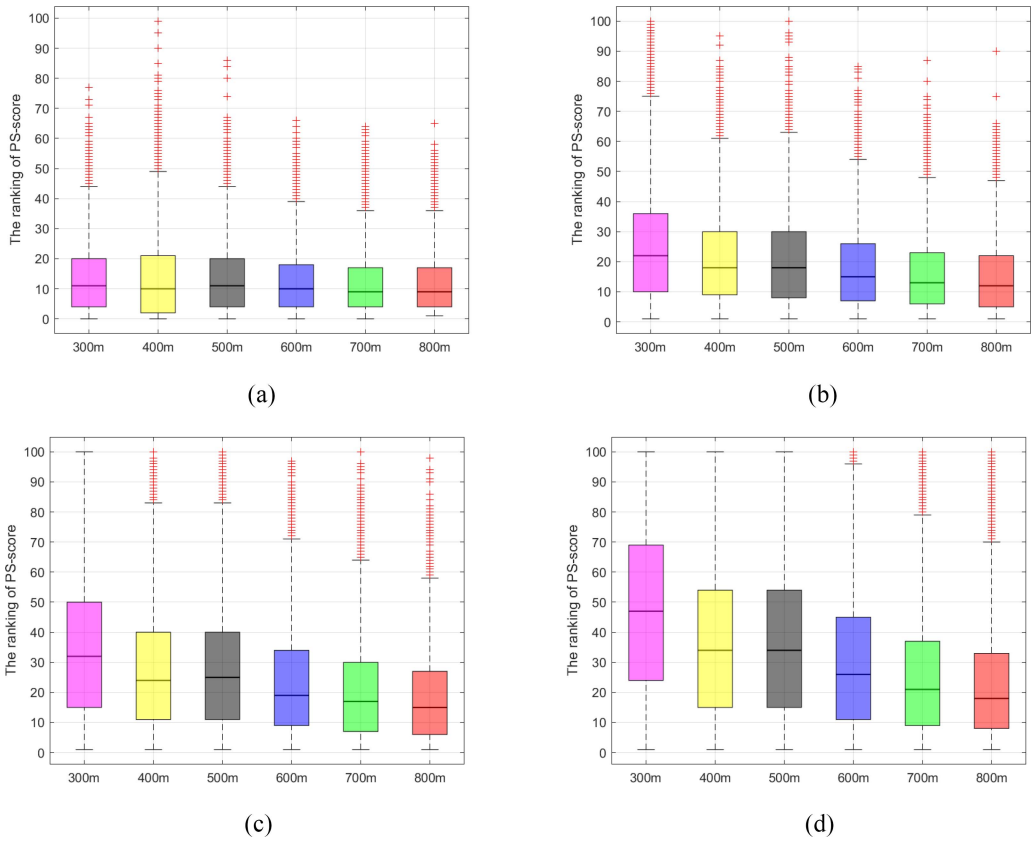


Fig. 9. Ranking of *PS-score* of selected skyline workers varies with different experimental settings. (a) Ranking of *PS-score* for the 1st skyline worker when the total number of workers l is set to 100, the task similarity threshold t is set to 0.46, the number of selected skyline workers N_s is set to 4, and the location filtering threshold R varies from 300 to 800 m. (b) Ranking of *PS-score* for the 2nd skyline worker when the total number of workers l is set to 100, the task similarity threshold t is set to 0.46, the number of selected skyline workers N_s is set to 4, and the location filtering threshold R varies from 300 to 800 m. (c) Ranking of *PS-score* for the 3rd skyline worker when the total number of workers l is set to 100, the task similarity threshold t is set to 0.46, the number of selected skyline workers N_s is set to 4, and the location filtering threshold R varies from 300 to 800 m. (d) Ranking of *PS-score* for the 4th skyline worker when the total number of workers l is set to 100, the task similarity threshold t is set to 0.46, the number of selected skyline workers N_s is set to 4, and the location filtering threshold R varies from 300 to 800 m.

workers, one worker will be selected as a new skyline worker if he/she is not dominated by any existing skyline worker. Otherwise, the next worker in the ranking list will be compared; this process will be repeated until all the N_s skyline workers are found. When R is large, the number of candidate workers is large (see in Fig. 6), and the workers who are sorted in the top positions by $S(w_i)$ are more likely to be the skyline workers. This explains why the running time for skyline worker selection tends to be longer when R is small.

Fig. 8 depicts the running time for updating *PS-score* for all the workers with different parameters. In the experiment, $l = 40, 60, 80,$ or 100, the average number of reviews for each worker v is from a normal distribution with mean $= 4$, 6, or 8 and the standard deviation $= 1$, and the number of questions $d = 5, 10, 15,$ or 20. From Fig. 8, we can see that

the running time for updating *PS-score* is proportional to both l, v, and d. In Fig. 8, when $l = 100, v = 8$, and $d = 20$, the maximum running time is about 10×10^4 s (around 27 h), which is quite large. However, the result is obtained by a local computer with a single processor. The fast evolution of parallel and distributed computing can provide more efficient solutions to address running time issues. In a real-world MCS platform, this computation can be distributed over a cluster of online computing machines, and the running time might be decreased to a reasonable level.

In Fig. 9, using the ranking of *PS-score* as the evaluation metric, we compare the reliability of selected skyline workers. In the experiment, we set $l = 100$, $N_s = 4$, $t = 0.46$, and R varying from 300 to 800 m. It is worth to note that the lower ranking score indicates the higher trustability of a worker. From Fig. 9, we can see that the rankings of the first selected skyline workers are the lowest (around 10) and are similar among different location filtering thresholds. In addition, under the same R, the trustability of the selected four skyline workers can be sorted as: the 1st worker > the 2nd worker > the 3rd worker > the 4th worker. This result validates the discovery in Fig. 7 that the workers who are sorted in the top positions of the ranking list show higher trustability than the latter ones. Finally, we can see that the skyline workers from a larger size of candidates (i.e., larger R) are more reliable than those from a smaller size of candidates (i.e., smaller R). In real-world applications, in order to select reliable and trustable workers, it is better to generate sufficient candidate workers.

VII. Related Work

In this section, we briefly review some related works that target worker selection in MCS. Ren *et al.* [20] introduced a reputation management scheme for selecting the well-suited workers under a fixed task budget; however, they do not consider the potential privacy leakage for the workers. Kazemi and Shahabi [21] studied the problem of a spatial task assignment for spatial crowdsourcing. To maximize the overall number of assigned tasks, their scheme highly depends on the locations of mobile workers, in which workers' essential personal information might be disclosed. Xiong *et al.* [22] investigated the problem of energy-efficient task allocation. Their research aims to guarantee a minimum number of anonymous workers who return their sensing results within a specified time frame. Kazemi *et al.* [23] proposed a framework to evaluate the validity of the results provided by workers with different trust levels in MCS. They assume that every worker has a reputation score that states the trustability of this worker. However, in their study, they do not describe how the reputation score is calculated. Recently, more studies focus on the problem of privacy-enhanced worker selection. Ni *et al.* [24] proposed a privacy-preserving MCS worker selection framework based on the points of interest and the location of users. Jin *et al.* [12] presented an incentive mechanism for worker selection in MCS services, which can choose workers that are more likely to provide reliable data, while protecting workers' privacy as well. Guo *et al.* [11] designed an MCS scheme for multitask worker selection, which considers both workers' intentional movement for time-sensitive tasks and unintentional movement for delay-tolerant tasks. Wang *et al.* [14] provided a personalized privacy-preserving task allocation framework for MCS applications.

Different from the above, our work applies the (probabilistic) skyline computation to select suitable workers securely. The proposed scheme can solve the fundamental problem of how workers' trustability scores are calculated while guaranteeing the privacy of their personal information.

VIII. Conclusion

In this article, we have proposed a privacy-preserving worker selection scheme for MCS applications based on (probabilistic) skyline computation. Our proposed scheme can efficiently select suitable and reliable workers without revealing workers' personal information. Specifically, the privacy-preserving probabilistic skyline was used to calculate workers' *PS-score* based on historical reviews. *PS-score* reveals the fluctuation of workers' past performance and can be used as their trustability. Then, skyline computation was used for selecting workers in terms of ask price, task similarity, and trustability. An NIEC was designed for supporting our scheme. Security analysis demonstrated that our scheme was privacy preserving, and theoretical performance evaluation verified the efficiency of the proposed scheme in terms of storage and computational costs. Besides, extensive simulations have been conducted to demonstrate the effectiveness of the proposed scheme. For future work, the probabilistic skyline over sliding windows will be studied to enhance the performance of the current scheme.

References

[1] X. Zhang, R. Lu, J. Shao, H. Zhu, and A. A. Ghorbani, "Achieve secure and efficient skyline computation for worker selection in mobile crowdsensing," in *Proc. 11th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Xi'an, China, 2019, pp. 1–6.

[2] X. Zhang and A. A. Ghorbani, "Human factors in cybersecurity: Issues and challenges in big data," in *Security, Privacy, and Forensics Issues in Big Data*. Hershey, PA, USA: IGI Global, 2020, pp. 66–96.

[3] *Living Up to the Potential of IoT*, Cengn, Kanata, ON, Canada, 2020. Accessed: Feb. 16, 2020. [Online]. Available: http://www.cengn.ca/iot-potential/

[4] C. Leonardi, A. Cappellotto, M. Caraviello, B. Lepri, and F. Antonelli, "SecondNose: An air quality mobile crowdsensing system," in *Proc. 8th Nordic Conf. Hum. Comput. Interaction Fun Fast Found.*, 2014, pp. 1051–1054.

[5] V. Coric and M. Gruteser, "Crowdsensing maps of on-street parking spaces," in *Proc. IEEE Int. Conf. Distrib. Comput. Sens. Syst.*, Cambridge, MA, USA, 2013, pp. 115–122.

[6] X. Wang, J. Zhang, X. Tian, X. Gan, Y. Guan, and X. Wang, "Crowdsensing-based consensus incident report for road traffic acquisition," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 8, pp. 2536–2547, Aug. 2018.

[7] R. Pryss, M. Reichert, B. Langguth, and W. Schlee, "Mobile crowd sensing services for tinnitus assessment, therapy, and research," in *Proc. IEEE Int. Conf. Mobile Serv.*, New York, NY, USA, 2015, pp. 352–359.

[8] M. A. Rahman and M. S. Hossain, "A location-based mobile crowdsensing framework supporting a massive ad hoc social network environment," *IEEE Commun. Mag.*, vol. 55, no. 3, pp. 76–85, Mar. 2017.

[9] J. Liu, H. Shen, H. S. Narman, W. Chung, and Z. Lin, "A survey of mobile crowdsensing techniques: A critical component for the Internet of Things," *ACM Trans. CSP*, vol. 2, no. 3, p. 18, 2018.

[10] Y. Xu, J. Tao, Y. Gao, and L. Zeng, "Location-aware worker selection for mobile opportunistic crowdsensing in VANETs," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Singapore, 2017, pp. 1–6.
[11] B. Guo, Y. Liu, W. Wu, Z. Yu, and Q. Han, "ActiveCrowd: A framework for optimized multitask allocation in mobile crowdsensing systems," *IEEE Trans. Hum. Mach. Syst.*, vol. 47, no. 3, pp. 392–403, Jun. 2017.
[12] H. Jin, L. Su, H. Xiao, and K. Nahrstedt, "INCEPTION: Incentivizing privacy-preserving data aggregation for mobile crowd sensing systems," in *Proc. 17th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2016, pp. 341–350.
[13] L. Wang, D. Yang, X. Han, T. Wang, D. Zhang, and X. Ma, "Location privacy-preserving task allocation for mobile crowdsensing with differential geo-obfuscation," in *Proc. Int. World Wide Web Conf. Steering Committee (WWW'17)*, 2017, pp. 627–636.
[14] Z. Wang *et al.*, "Personalized privacy-preserving task allocation for mobile crowdsensing," *IEEE Trans. Mobile Comput.*, vol. 18, no. 6, pp. 1330–1341, Jun. 2019.
[15] H. Jin, L. Su, B. Ding, K. Nahrstedt, and N. Borisov, "Enabling privacy-preserving incentives for mobile crowd sensing systems," in *Proc. IEEE 36th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Nara, Japan, 2016, pp. 344–353.
[16] B. Guo *et al.*, "TaskMe: Toward a dynamic and quality-enhanced incentive mechanism for mobile crowd sensing," *Int. J. Hum. Comput. Stud.*, vol. 102, pp. 14–26, Jun. 2017.
[17] J. Pei, B. Jiang, X. Lin, and Y. Yuan, "Probabilistic skylines on uncertain data," in *Proc. 33rd Int. Conf. Very Large Data Bases*, 2007, pp. 15–26.
[18] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: Differential privacy for location-based systems," 2012. [Online]. Available: arXiv:1212.1984.
[19] G. Cong, C. S. Jensen, and D. Wu, "Efficient retrieval of the top-k most relevant spatial Web objects," *Proc. VLDB Endowm.*, vol. 2, no. 1, pp. 337–348, 2009.
[20] J. Ren, Y. Zhang, K. Zhang, and X. S. Shen, "SACRM: Social aware crowdsourcing with reputation management in mobile sensing," *Comput. Commun.*, vol. 65, pp. 55–65, Jul. 2015.
[21] L. Kazemi and C. Shahabi, "GeoCrowd: Enabling query answering with spatial crowdsourcing," in *Proc. 20th Int. Conf. Adv. Geograph. Inf. Syst.*, 2012, pp. 189–198.
[22] H. Xiong, D. Zhang, L. Wang, J. P. Gibson, and J. Zhu, "EEMC: Enabling energy-efficient mobile crowdsensing with anonymous participants," *ACM Trans. Intell. Syst. Technol. (TIST)*, vol. 6, no. 3, p. 39, 2015.
[23] L. Kazemi, C. Shahabi, and L. Chen, "GeoTrucrowd: Trustworthy query answering with spatial crowdsourcing," in *Proc. 21st ACM SIGSPATIAL Int. Conf. Adv. Geograph. Inf. Syst.*, 2013, pp. 314–323.
[24] J. Ni, K. Zhang, X. Lin, Q. Xia, and X. S. Shen, "Privacy-preserving mobile crowdsensing for located-based applications," in *Proc. IEEE Int. Conf. Commun. (ICC'17)*, Paris, France, 2017, pp. 1–6.

Xichen Zhang received the B.E. degree from Changsha University of Science and Technology, Changsha, China, in 2010, and the M.S. degree in computer science from the Canadian Institute for Cybersecurity (CIC), Faculty of Computer Science, University of New Brunswick, Fredericton, NB, Canada, in 2018, where he is currently pursuing the Ph.D. degree.

He worked as a Research Assistant in CIC. His research interests are data mining in cybersecurity, privacy-enhancing technologies, and IoT-big data security and privacy.

Rongxing Lu (Senior Member, IEEE) received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in 2012.

He worked as an Assistant Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, from April 2013 to August 2016. He is currently an Associate Professor with the Faculty of Computer Science (FCS), University of New Brunswick (UNB), Fredericton, NB, Canada. He worked as a Postdoctoral Fellow with the University of Waterloo from May 2012 to April 2013. His research interests include applied cryptography, privacy-enhancing technologies, and IoT-big data security and privacy. He has published extensively in his areas of expertise.

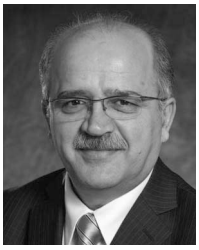
Dr. Lu was a recipient of eight best (student) paper awards from some reputable journals and conferences. He was awarded the most prestigious "Governor General's Gold Medal" and won the 8th IEEE Communications Society (ComSoc) Asia–Pacific Outstanding Young Researcher Award, in 2013. He is the Winner of 2016–2017 Excellence in Teaching Award, FCS, UNB. He currently serves as the Vice-Chair (Conferences) of IEEE ComSoc Communications and Information Security Technical Committee. He is currently a Senior Member of IEEE Communications Society.

Jun Shao received the Ph.D. degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2008.

He was a Postdoctoral Researcher with the School of Information Sciences and Technology, Pennsylvania State University, State College, PA, USA, from 2008 to 2010. He is currently a Professor with the School of Computer Science and Information Engineering, Zhejiang Gongshang University, Hangzhou, China. His research interests include network security and applied cryptography.

Hui Zhu (Senior Member, IEEE) received the B.Sc. degree from Xidian University, Xi'an, China, in 2003, the M.Sc. degree from Wuhan University, Wuhan, China, in 2005, and the Ph.D. degree from Xidian University in 2009.

He was a Research Fellow with the School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore, in 2013. Since 2016, he has been a Professor with the School of Cyber Engineering, Xidian University. His current research interests include applied cryptography, data security, and privacy.



Ali A. Ghorbani (Senior Member, IEEE) has held a variety of academic positions for the past 39 years and is currently a Professor of computer science, Tier 1 Canada Research Chair in Cybersecurity, and the Director of the Canadian Institute for Cybersecurity, which he established in 2016. He served as the Dean of the Faculty of Computer Science, University of New Brunswick, Fredericton, NB, Canada, from 2008 to 2017. He is also the Founding Director of the Laboratory for Intelligence and Adaptive Systems Research. He has spent over 29 years of his 39-year academic career, carrying out fundamental and applied research in machine learning, cybersecurity, and critical infrastructure protection. He is the co-inventor on three awarded and one filed patent in the fields of Cybersecurity and Web Intelligence and has published over 280 peer-reviewed articles during his career. He has supervised over 190 Research Associates, Postdoctoral Fellows, and students during his career. He has authored the book *Intrusion Detection and Prevention Systems: Concepts and Techniques* (Springer, October 2010). He developed several technologies adopted by high-tech companies and co-founded three startups, Sentrant Security, EyesOver Technologies, and Cydarien Security in 2013, 2015, and 2019. He is the co-founder of the Privacy, Security, Trust (PST) Network in Canada and its annual international conference.

Dr. Ghorbani was a recipient of the 2017 Startup Canada Senior Entrepreneur Award, and the Canadian Immigrant Magazine's RBC top 25 Canadian immigrants of 2019. He served as the Co-Editor-in-Chief for *Computational Intelligence: An International Journal* from 2007 to 2017.