Joint Differentially Private Gale–Shapley Mechanisms for Location Privacy Protection in Mobile Traffic Offloading Systems

Yuan Zhang, Yunlong Mao, and Sheng Zhong

***Abstract*— Being an important application of spectrum sharing in cellular networks, mobile traffic offloading, which advocates third-party owners of network resource on unlicensed/licensed spectrum to share their spectrum and provide data offloading services, is considered a promising solution to severe spectrum shortage faced by cellular network service providers. In this paper, we consider a general mobile traffic offloading system that adopts the widely used Gale–Shapley algorithm to optimize its mobile phone users (MUs) to offloading stations allocation plan. We notice that without careful protection, such a system could cause serious threat to MUs' location privacy, and thus design effective countermeasures based on the powerful state-of-the-art differential privacy concept. Specifically, we have proposed two joint differentially private Gale–Shapley mechanisms with strong privacy protections for mobile traffic offloading systems. The first mechanism is able to protect each user's location privacy even when all other users collude against this user assuming the system administrator can be trusted. The second mechanism is able to achieve the same privacy guarantee against colluding users, and moreover against an untrusted semi-honest system administrator. We perform extensive experiments to evaluate our mechanisms, and the results show that our mechanisms have good efficiency, accuracy, and privacy protection.**

***Index Terms*— Mobile traffic offloading, spectrum sharing, security, location privacy, differential privacy, Gale-Shapley algorithm.**

I. Introduction

ENABLING flexible sharing and efficient utilization of licensed and unlicensed spectrum resource, spectrum sharing has been a catalyst for innovative solutions to the spectrum scarcity problem. Among a variety of successful applications of spectrum sharing, mobile traffic offloading which advocates third-party owners of unlicensed/licensed spectrum (e.g. owners of WIFI APs and small cells) to share their network resource and provide data offloading services to mobile phone users and cellular network service providers, has recently received increasing attention from both the academia and the industry.

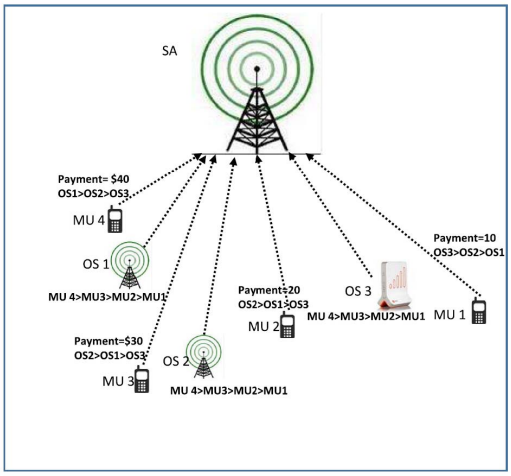


Fig. 1. Example of a mobile traffic offloading system running the Gale-Shapley algorithm.

As depicted in Figure 1, a mobile traffic offloading system generally consists of multiple mobile phone users, multiple *offloading stations* who possess spare network resources on unlicensed or licensed spectrum (e.g. WIFI APs or small cells), and a *system administrator* (e.g. a cellular service provider) who is responsible to provide network services to mobile phone users and interested in offloading these users' data traffic to offloading stations since its own licensed spectrum resource is intensively occupied. Due to the differences in mobile phone users' and offloading stations' physical locations, different mobile phone users generally experience different service quality at different offloading stations. To optimize the system's performance and mobile phone users' experience, an important mission for the administrator is to allocate each mobile phone user to a proper offloading station.

To help the system administrator to find an excellent allocation solution, mobile phone users are generally required to input their perceived connection quality regarding every offloading station. For example, in [1], each mobile phone user's preference over all offloading stations, which is determined by this user's perceived packet success rate (PSR) and delay regarding each offloading station's channel, is collected by the system administrator to compute a stable allocation.

Manuscript received April 15, 2016; revised August 2, 2016; accepted August 28, 2016. Date of publication September 2, 2016; date of current version October 13, 2016. This work was supported in part by the Jiangsu Province Double Innovation Talent Program and in part by the National Natural Science Foundation of China under Grant NSFC-61300235, Grant NSFC-61321491, Grant NSFC-61402223, and Grant NSFC-61425024. *(Corresponding author: Sheng Zhong.)*

The authors are with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China, and also with the Computer Science and Technology Department, Nanjing University, Nanjing 210023, China (e-mail: zhangyuan05@gmail.com; njucsmyl@163.com; zhongsheng@nju.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSAC.2016.2605798

In [2], perceived average signal to interference plus noise ratio (SINR) of each mobile phone user regarding every station's channel is collected to compute the optimal allocation. Note that all these channel quality data (e.g. PSR, SINR, and delay) of a mobile phone user is closely related to its physical location, or the distance between the user and the station to be more precise. Without carefully protecting these information, mobile phone users' location privacy could be severely endangered when participating the offloading system.

Same as [1], in this paper we consider the mobile traffic offloading system's administrator computes its allocation solution using the well-known Gale-Shapley (deferred acceptance) algorithm [3] (which is widely used in matching or allocation problems), and study how to thwart above location privacy threats. Rather than precise connection quality data regarding all stations, the Gale-Shapley algorithm takes only each user's *preference* over the stations, which is determined based on the connection quality experienced by the user, as its input. Although it's difficult to extract exact distances between an user and all stations from this preference, an adversary may still be able to infer the relative magnitude of these distances from it, and use this information to launch attacks that seriously break this user's location privacy. (Please see Section II-B for more details regarding this matter.) Therefore, it is vital to keep every user's preference well protected for preventing its location privacy from being compromised.

To solve our problem, we resort to the state-of-the-art, powerful *differential privacy* concept [4], [5], and seek to make the Gale-Shapley algorithm differentially private. Informally, making the allocating scheme differentially private in our problem would sanitize the scheme's output (i.e. the allocation solution) so that changing an arbitrary mobile phone user's input (i.e. its preference) would not make any noticeable difference on the output. Therefore, it would be extremely difficult for an adversary to infer any single user's private input from the output. There are two generic ways to make a scheme differentially private: the Laplace mechanism [4] for schemes with numeric outputs, and the exponential mechanism [5] for schemes with generic outputs. Given the allocating scheme's output is not numeric, theoretically we could apply the exponential mechanism to make it differentially private. However, since the allocating scheme's output space is exponentially large,[1] the complexity of computing the *score function* on all possible outputs, which is required by the exponential mechanism, is too high making it infeasible in practice.

To overcome above challenge, we construct novel joint differentially private Gale-Shapley mechanisms by exploring a key observation that each mobile phone user's move in the Gale-Shapley algorithm is determined by a series of application/applicant counters and this user's preference only. By adopting the *billboard model* [6], we let our privacy-preserving Gale-Shapley mechanism output differentially private counts only, rather than the final allocation solution, thus mobile phone users' preferences and their location privacy are protected well. Based on the "sanitized" counts and its private preference, each user simulates the Gale-Shapley algorithm on its own to figure out its allocated offloading station.

When designing our mechanisms following the ideas above, we consider two different cases. The first one is that the system administrator is *trusted*, so that mobile phone users can reveal their preferences to it. The second one is that the system administrator is *not trusted* and *semi-honest* in the sense that mobile phone users are not willing to reveal their preferences to it, and the administrator will try to infer mobile phone users' private preferences from its view in the mechanism, although it never deviates from the mechanism. Our major contributions can be summarized as follows.

- We propose DP-GS, a joint differentially private Gale-Shapley mechanism for mobile traffic offloading systems with trusted administrators. DP-GS can be used by all mobile phone users to reach an allocation solution that approximates the stable solution of Gale-Shapley deferred acceptance algorithm. In addition, the allocation solution reached in DP-GS is joint differentially private, and thus can protect the privacy of a mobile phone user's preference even when all other mobile phone users collude against this user.
- We propose EDP-GS, a joint differentially private Gale-Shapley mechanism for mobile traffic offloading systems whose administrators are not trusted and semi-honest. EDP-GS can be used by all mobile phone users to reach an allocation solution that approximates the stable solution of Gale-Shapley deferred acceptance algorithm. In addition, the allocation solution reached in DP-GS is joint differentially private. Moreover, the administrator's view is differentially private, thus the privacy of each user's preference is protected against the administrator also.
- We analyze the complexity of DP-GS and EDP-GS, and prove their privacy guarantees.
- We implement DP-GS and EDP-GS, and perform extensive experiments to evaluate the performance of our mechanisms. Evaluation results show that both two mechanisms achieve ϵ-*joint differential privacy*, (please see Section II-D for more details about this privacy guarantee.) and is able to generate allocation solutions that approximate the output of the Gale-Shapley algorithm well.

II. Preliminary

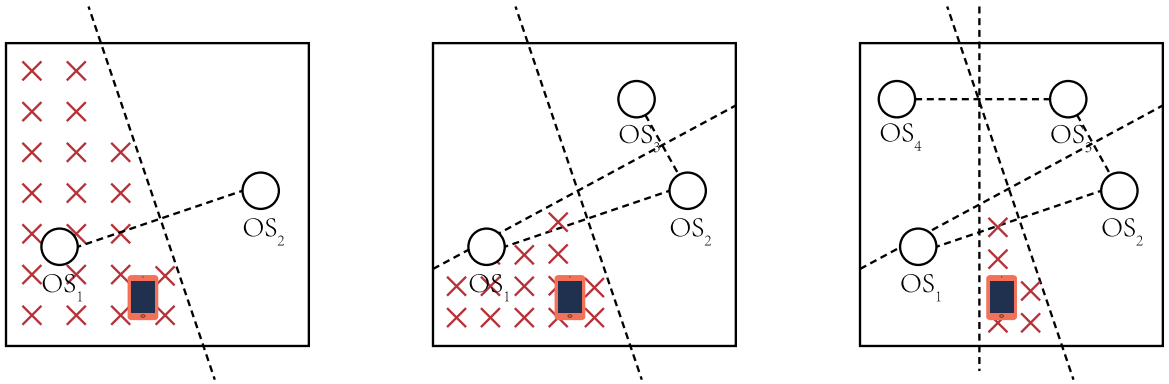
Before proceeding to detailed constructions, here we present our models, and give a short introduction to the concepts, tools that we use to build our mechanisms.

A. System Model

In this paper, we consider a general mobile traffic offloading system, which consists of one system administrator (SA), a set $M = \{1, \ldots, m\}$ of m offloading stations (OSs), and a set $N = \{1, \ldots, n\}$ of n mobile phone users (MUs).

Each MU $i \in N$ has an offloading task and a corresponding payment which are publicly known to the SA and OSs. Based on its perceived channel quality of the OSs, each MU i has

[1]For example, consider a small system with 10 offloading stations and 100 mobile users, the total number of possible outputs equals 10^{100} assuming stations' capacity is large enough.



(a) $d_1 \leq d_2$ (b) $d_1 \leq d_2 \leq d_3$ (c) $d_1 \leq d_2 \leq d_3 \leq d_4$

Fig. 2. An example showing how an adversary infers a mobile phone user's location given only the relative magnitude information of the distances between this user and offloading stations. Here d_i equals the user's distance to OS_i, and the red crosses in each sub-graph mark the possible area of this user's location that can be inferred from the information below. We can see the accuracy of adversary's inference increases rapidly as the offloading stations' number grows, which causes serious threats to the user's location privacy.

a **private** preference over the OSs, which is represented as a vector $\vec{x}_i = (x_{i,1}, \ldots, x_{i,m})$, where $x_{i,j} \in M$ is the index of MU's j-th preferred OS.

We consider each OS $j \in M$ has a **public** preference over all users' tasks based on their payments which is also represented as a vector $\vec{y}_j = (y_{j,1}, \ldots, y_{j,n})$ and a public known capacity z_j, where $y_{j,i} \in N$ is the index of OS j's i-th preferred MU and $z_j \in \mathbb{N}^+$ equals the maximum number of tasks that OS j can accommodate. Denote by $y_j^{-1}(i)$ MU i' rank in OS j's preference. We point out that OSs' preferences do not concern us even if they contain knowledge of OSs' locations. This is because their locations are generally static and publicly known in practice.

In our first mechanism, we assume there is a secure channel for each MU to send its preference to the SA. Each OS also sends its preference to the SA privately or publicly. After receiving the preferences of all MUs and OSs, SA runs our privacy-preserving mechanism $\mathcal{A}$ on these preferences to determine the allocation solution.

We remove above assumption about the private channels between MUs and the SA when we consider the SA cannot be trusted in our second mechanism.

Finally, we assume the SA maintains a *billboard* so that information put on this board can be publicly accessed. The billboard is used to help our mechanisms achieve our privacy goal.

B. Adversary Model and Threats to Mobile Phone User's Location Privacy Threats

In our paper, the adversary is modelled as an entity who is interested in gaining knowledge about a mobile phone user's private preference and uses this information to breach this user's location privacy.

The adversary could be an outsider of the mobile traffic offloading system, or the system/mechanism's participating mobile phone users or offloading stations, or even the system administrator (which is the case we study in our second mechanism). In either case, we assume the adversary is *semi-honest* [7], meaning it monitors the system passively and uses what it sees to extract knowledge about honest participants' private input, but never causes any deviation of the allocating mechanism.

In addition, we remark that we allow corrupted participants to collude with each other which encompasses the case in which an adversary compromises and controls multiple participants and the case in which multiple participants/adversaries launch an attack jointly.

The threats to a mobile phone user's location privacy in our problem derives from the fact that its preference over the offloading stations is closely related to the distances between this user and all stations. When information about these distances is compromised, this user's location privacy can be seriously broken.

For example, if the adversary knows an user's accurate distances to three offloading stations, it can easily compute the user's the accurate location via trilateration given offloading stations' locations are publicly known. Even if we assume the adversary only knows the magnitude relations of these distances, it can still infer the user's location with a very high precision by constructing a Voronoi graph, and refining the Voronoi cell of this user's nearest offloading station based on the magnitude relations of distances to the other offloading stations. A simple example which illustrates the devastating effects of this kind of attack is shown in Figure 2.

C. College Admission Game and Gale-Shapley Deferred Acceptance Algorithm

Without considering privacy, determining an allocation solution for all MUs and OSs to can be modeled as a *college admission game* which studies the strategic interactions among a group of students and a group of colleges.

In this game, each student has a preference over all colleges and each college also has a preference over all students. A feasible solution of the game allocates each student to one college (given the total capacity of all colleges is no less than the total number of students), and also guarantees every single college's capacity is not exceeded.

It has been shown that the following Gale-Shapley algorithm, first introduced in [3], guarantees that the allocation solution is *stable*, and also is *non-manipulable* for students of any coalition [8]. Specifically, given the total number of colleges and students being m and n respectively, the algorithm runs for at most mn rounds as follows:

- In each round, each "free" student applies to its most preferred college that it has never applied to yet. From all free students who apply to it in this round and all students that are currently on its waiting list, each college j creates a new waiting list by writing the top z_j preferred students on it, and rejects the other students. Here z_j equals college j's capacity.
- The algorithm stops until there are no more rejected student, or all rejected students have applied all colleges; otherwise, all rejected students go to the next round. When the algorithm stops, all colleges accept the students on their waiting lists.

D. Security Model

The security model we adopt in this paper is called *differential privacy* which was first defined by Dwork et al. in [4].

Definition 1: A mechanism/function $f : \mathcal{X}^n \to \mathcal{Y}$ or its output is ϵ-differentially private if for every $y \in \mathcal{Y}$, and for every possible input $\vec{x}$ and $\vec{x}'$ in $\mathcal{X}^n$ that differ at only one component,

$$|\ln(\frac{Pr[f(\vec{x}) = y]}{Pr[f(\vec{x}') = y]})| \leq \epsilon \tag{1}$$

always holds, where $\epsilon \in \{0\} \cup \mathbb{R}^+$ models the privacy requirement of the mechanism.

Informally speaking, differential privacy requires a multiple-input function's output be insensitive to the change of any one input. Based on the above definition, it is not difficult to prove a differentially private mechanism also protects privacy when facing multiple input components' changes.

Theorem 1 (Group Privacy Extension [9]): If a mechanism is ϵ-differentially private with respect to any one input component's change, it is $c\epsilon$-differentially private with respect to any c input components' changes ($c \in \mathbb{N}$).

Another excellent property of differential privacy is its composability. Specifically, we have the following theorem that allows one to conveniently construct new differentially private mechanisms from other differentially private mechanisms' outputs.

Theorem 2 (Sequential Composition Theorem [10]): If there are n independent mechanisms $f_1, \ldots, f_n$ whose privacy guarantee are $\epsilon_1, \ldots, \epsilon_n$ differential privacy, respectively, then any function g of them: $g(f_1, \ldots, f_n)$ is $(\sum_{i=1}^{n} \epsilon_i)$-differential private.

Due to the specificity of Gale-Shapley algorithm, we adopt a variation of differential privacy solution concept called *joint differential privacy* in this paper, which was first formalized by Kearns et al. [11] and also seen in [6], [12], and [13].

Definition 2: A mechanism or function $f : \mathcal{X}^n \to \mathcal{Y}^n$ is joint ϵ-differentially private if for every $i \in \lfloor n \rceil$, every $\vec{y}$ in $\mathcal{Y}^{n-1}$, and for every possible vector input $\vec{x}$ and $\vec{x}'$ in the domain that differ at only the i-th component,

$$|\ln(\frac{Pr[f(\vec{x})|_{-i} = \vec{y}]}{Pr[f(\vec{x}')|_{-i} = \vec{y}]})| \leq \epsilon \tag{2}$$

always holds, where $f(\vec{x})|_{-i}$ denotes the mechanism's output on $\vec{x}$ after removing the i-th component, and $\epsilon \in \{0\} \cup \mathbb{R}^+$ models the privacy requirement of the mechanism.

Remark: We note that joint differential privacy can be used to provide a very strong privacy guarantee in semi-honest model, namely *anti-collusion*. The joint differential privacy model specifically considers a computing scenario in which a group of data contributors jointly compute a multiple-input-multiple-output function that takes one input from each user and outputs one input to this user privately. Due to this setting, if a group of colluding data contributors want to infer one users private input, they can infer it based on their joint view of their own outputs only (since they cannot see that users output). However, according the definition of joint e-differential privacy, for each contributor, changing its private input makes little difference on the distribution of all other contributors outputs. Therefore, even all other data contributors collude, they can extract little knowledge about the only honest contributors private input from their joint view. Our mechanisms are able to protect privacy against collusion since they are joint differentially private.

E. The Billboard Model

One fundamental method that we use to construct our joint differentially private mechanisms is the *billboard model* which is introduced by Hsu et al. in [6]. Applying the billboard model or method properly on a differentially private mechanism, one can construct a new mechanism that is joint differentially private.

Specifically, in the billboard model, a multiple-input-multiple-output mechanism $f : \mathcal{X}^n \to \mathcal{Y}^n$ is considered. Hsu et al. has proved the *Billboard Lemma* [6] which says if there exists ϵ-differentially private mechanism $f' : \mathcal{X}^n \to \mathcal{Y}$, and if for each $i \in \lfloor n \rceil$, the i-th component of $f(\vec{x})$ is determined only by $f'(\vec{x})$ and x_i, we know mechanism f is ϵ-joint differentially private.

F. Differentially Private Streaming Counters

One fundamental tool that we use to construct our mechanisms is the *differentially private streaming counter* proposed by Chan et al. [14] and by Dwork et al. [15].

Let $\mathcal{BM}(\epsilon, T)$ denote Chen et al.'s *binary mechanism* initialized with privacy parameter ϵ and the maximum input length T, and let $Ctr := \mathcal{BM}(\epsilon, T)$ denote the differentially private counter outputted by $\mathcal{BM}(\epsilon, T)$.

Specifically, feeding a bit stream $\delta = (\delta_1, \delta_2, \ldots, \delta_T)$ as the inputs, Ctr returns a series of sanitized, approximate counts $\{Ctr(t)\}_{t \in \lfloor T \rceil}$, such that:

1) $\{Ctr(t)\}_{t \in \lfloor T \rceil}$ is ϵ-differentially private;
2) $Ctr(t)$ is (α, β)-useful for approximating $c_\delta(t) = \sum_{i=1}^{t} \delta_i$ for each time $t \in \lfloor T \rceil$, i.e., with probability

at least $(1-\beta)$, $|Ctr(t) - c_\delta(t)| \leq \alpha$ holds, where $\alpha = \frac{2\sqrt{2}}{\epsilon}\ln(\frac{2}{\beta})(\sqrt{\log T})^5$.

III. Differentially Private Gale-Shapley Mechanism With a Trusted SA

In this section, we propose DP-GS, a joint differentially private Gale-Shapley mechanism for scenarios where the SA is trusted.

A. Design Rationale

Making the Gale-Shapley algorithm differentially private usually involves sanitization to the its output (i.e. the allocation solution) so that the output becomes insensitive to any change on a single input (i.e. a MU's preference). Given a predefined privacy requirement, the amount of sanitization that requires to be added is closely related by the algorithm's *sensitivity* or how sensitively the algorithm's output changes when a single input changes. Unfortunately, in our problem, output of the Gale-Shapley algorithm contains the OS that is allocated to each MU, which usually can be quite sensitive to this MU's preference. Even we could apply the exponential mechanism to make the Gale-Shapley algorithm's output differentially private without worrying about its complexity issue, the sanitization would be too much and the allocation solution's accuracy would be greatly destroyed.

Due to above reasons, we resort to the billboard model and choose to make our mechanism joint differentially private. Note that we still need a differentially private ingredient of the Gale-Shapley algorithm as the cornerstone of our joint differentially private mechanism according to the billboard model. To obtain this important piece, we first make a key observation on the Gale-Shapley algorithm.

Specifically, in each round, Gale-Shapley algorithm allocates a free MU (student) to the waiting list of a OS (college) based on two pieces of information. One is this MU's preference, which determines the particular OS that this MU applies to in each round. The other is, among all MUs who are already on the waiting list of the selected OS, how many are more preferred to the OS compared to this MU. This piece of information determines whether a MU's application would be accepted in each round. Our key observation is, if any MU is provided with the second piece of information, it can easily figure out the correct MU that it is assigned to in the final allocation solution since it possesses the first piece of information naturally. Moreover, this piece of information is a series of counts which can be made differentially private effectively and used as the cornerstone of DP-GS.

Therefore, DP-GS lets the SA use several differentially private counters to record the second piece of information, and publish differentially private versions of this information to all MUs. Based on the differentially private public information, together with its own preference, each MU figures out the final OS to which it is allocated on its own and keeps the result private. Following the Billboard Lemma, we can prove DP-GS achieves joint differential privacy.

B. The Overview of DP-GS

DP-GS is performed by the SA to approximately simulate the Gale-Shapley algorithm.

Before DP-GS starts, all MUs send their preferences to the SA.

Following the Gale-Shapley algorithm, SA simulates each free MU to apply to its most preferred OS that it has not applied to yet in each round. Meanwhile, SA maintains several differentially private counters to record a sanitized applying history. Based on these sanitized history, the SA simulates each MU's application result.[2]

Finally, SA publishes the sanitized history. With this information, each MU locally simulates the mechanism and gets its final allocated OS.

In more detail, we explain DP-GS below.

1) Constructing the Private Counters: To construct DP-GS, first we construct several differentially private counters whose outputs can be used by MUs or the SA to simulate the application process and determine the final allocation solution as follows.

Recall that in each round, Gale-Shapley determines whether MU i's application to OS j is accepted by verifying whether MU i is one of the top z_j preferred MUs among all MUs that are currently on OS j's waiting list and itself. Although the above criteria is actually a set inclusion predicate, we point out it is equivalent to a numeric comparison. Therefore we can use a (differentially private) counter to provide the required information, and perform the verification.

Specifically, denote $Ctr_{<j,k>}$ $(j \in \lfloor m \rceil, k \in [1, n-1])$ the counter that records, among OS j's the 1st, the 2nd,…, and the k-th most preferred MUs, the total number of those who have applied to OS j. We know:

Proposition 3: In Gale-Shapley algorithm, to verify whether MU i is one of the top z_j preferred MUs among all MUs that are currently on OS j's waiting list and MU i, it is equivalent to verify whether the current value of $Ctr_{<j,y_j^{-1}(i)-1>}$ is less than z_j.

Proof: Recall that in OS j's preference, MU i is the $y_j^{-1}(i)$-th most preferred. According to Gale-Shapley algorithm, if MU i's application is rejected when it applies to OS j, there must be no less than z_j MUs in the current waiting list and these MUs are more preferred to OS j compared with MU i. Accordingly, we know if the total number of MUs who are on the waiting list and also are more preferred is less than z_j, MU i's application should be accepted. Since a MU has to apply the OS before it is put on this OS's waiting list, we know if the total number of MUs who have applied OS i and are more preferred compared with MU j is less than z_j (i.e. $Ctr_{<j,y^{-1}(i)-1>} < z_j$,), the total number of MUs who are on the waiting list and also are more preferred is no greater than $Ctr_{<j,y^{-1}(i)-1>}$, and MU i's application should be accepted.

In addition, suppose $Ctr_{<j,y^{-1}(i)-1>} \geq z_j$. This means at least z_j MUs have applied OS j and all these MUs are more preferred compared with MU i. If all these MUs are still on the waiting list, the waiting list is full and MU i's application should be rejected. If any one of these MUs is

[2]Since SA also knows all MUs' preferences, an alternative is that the SA computes the allocation solution, and informs each MU the OS that it is allocated to via a secure channel. In this case, the secure channel between the SA and each MU that we have assumed needs to be bi-directional.

not on the waiting list, this means: 1) the waiting list is full since replacements only happen after the list is full; and 2) all MUs on the waiting list are more preferred compared with this replaced MU, thus are more preferred compared with MU i. MU i's application should be rejected. ■

Therefore, one can use $m(n-1)$ counters and all MUs' preferences to simulate Gale-Shapley algorithm. Since we want to make the allocation solution joint differentially private, we let the SA generate $m(n-1)$ differentially private counters, denoted by $\{DCtr_{<j,k>}\}_{j\in\lfloor m\rceil,k\in[1,n-1]}$, using the binary mechanism [14] in the beginning of our DP-GS mechanism:

$$DCtr_{<j,k>} := \mathcal{BM}(\epsilon/2mn, mn^2) \tag{3}$$

for each $j \in \lfloor 1, m\rceil, k \in \lfloor 1, n-1\rceil$, where the first parameter of $\mathcal{BM}$ is set to guarantee that our mechanism achieves joint ϵ-differential privacy (please see the proof of Theorem 4 for more details.), and the second parameter is set based on the maximum number of counts we need to record in our mechanism.

2) Simulating MUs' Moves: In each round of DP-GS, SA simulates all MUs one by one to make their moves or applications as follows.

Specifically, when MU i's turn arrives, let $j^* \in \lfloor 1, m\rceil$ denote the index of its most preferred OS that it has not applied yet.

If MU i is free (or not on any waiting list) at the end of previous round, SA lets it apply to OS j^*, and determines the result of this application by verifying whether $DCtr_{<j^*,y^{-1}(i)-1>} < z_{j^*}$ holds. If the verification is passed, SA puts MU i on OS j^*'s waiting list. Otherwise, SA rejects MU i's application.

If MU i is not free (supposing it is on MU j's waiting list) at the end of previous round, SA re-examines if it has been replaced by other MUs whose turns arrive before MU i's in this round. To do this, SA verifies whether $DCtr_{<j,y^{-1}(i)-1>} < z_j$ holds. If the verification is passed, SA lets MU i not make any move in this round. Otherwise, SA lets MU i apply to OS j^*, and determines the result same as it does when MU i is free at the end of previous round.

3) Updating the Counter: At the end of each MU's turn in each round, SA updates the private counters based on this MU's move in SA's simulation as follows.

If MU i applied to OS j^*, SA updates the following counters regarding OS j^*:

$$DCtr_{<j^*,y_{j^*}^{-1}(i)>}, DCtr_{<j^*,y_{j^*}^{-1}(i)+1>}, \ldots, DCtr_{<j^*,n-1>} \tag{4}$$

by feeding 1 to these counters. Recall that $DCtr_{<j,k>}$ records, among OS j's the 1st, the 2nd,…, and the k-th most preferred MUs, the total number of those who have applied to OS j. Therefore, MU i's application changes the values of all above counters. In addition, SA updates the following counters regarding each OS $j \neq j^*$:

$$DCtr_{<j,y_j^{-1}(i)>}, DCtr_{<j,y_j^{-1}(i)+1>}, \ldots, DCtr_{<j,n-1>} \tag{5}$$

by feeding 0 to these counters.

Algorithm 1 DP-GS Mechanism (With a Trusted SA)

Require:
All MUs' preferences, $\{\vec{x}_i\}_{i\in\lfloor m\rceil}$;
All OSs' preferences and capacities, $\{\vec{y}_j, z_j\}_{j\in\lfloor n\rceil}$;
Ensure:
Differentially private counters $\{DCtr_{<j,i>}\}_{j\in\lfloor m\rceil,i\in\lfloor n-1\rceil}$;

1: SA initializes $m(n-1)$ differentially private counters: $DCtr_{<j,i>} := \mathcal{BM}(\epsilon/2mn, mn^2)$, for every $j \in \lfloor m\rceil$, $i \in \lfloor n-1\rceil$, and generates m empty waiting lists $\{W_j\}_{j\in\lfloor m\rceil}$;
2: SA simulates the following application rounds until no MU is free, or all free MUs have applied to all OSs:
- In each round, for each MU i and its most preferred OS j^* that MU i has not applied to yet, MU i simulates its own move by updating the counters and its allocated OS $\mu(i)$ as follows:
  - If MU i is free, feed 1 to $DCtr_{<j^*,k>}$ for each $k \in [y_{j^*}^{-1}(i), n-1]$, and feed 0 to $DCtr_{<j,k>}$ for each $j \in \lfloor m\rceil \setminus \{j^*\}$ and $k \in [y_j^{-1}(i), n-1]$; In addition, if $DCtr_{<j,y_{j^*}^{-1}(i')-1>}(t) < z_j^*$ where t is the counter's current time point, add i to W_j^*;
  - If MU i is on the waiting list of OS j at the end of last round, and $DCtr_{<j,y_j^{-1}(i)-1>}(t) \geq z_j$, remove i' from W_j, feed 1 to $DCtr_{<j^*,k>}$ for each $k \in [y_{j^*}^{-1}(i), n-1]$, and feed 0 to $DCtr_{<j,k>}$ for each $j \in \lfloor m\rceil\setminus\{j^*\}$ and $k \in [y_j^{-1}(i), n-1]$; In addition, if $DCtr_{<j,y_{j^*}^{-1}(i')-1>}(t) < z_j^*$ where t is the counter's current time point, add i to W_j^*;
  - If MU i is on the waiting list of OS j at the end of last round, and $DCtr_{<j,y_j^{-1}(i)-1>}(t) < z_j$, feed 0 to $DCtr_{<j,k>}$ for each $j \in \lfloor m\rceil$ and $k \in \lfloor y_j^{-1}(i), n-1\rceil$. $Ctr_{<j,y_j^{-1}(i')-1>}(t) \geq z_j$, remove i' from W_j, and set MU i''s status to free.
3: **return** SA publishes $\{DCtr_{<j,i>}\}_{j\in\lfloor m\rceil,i\in\lfloor n-1\rceil}$;

If MU i makes no move, SA updates the following counters regarding OS j for $j \in \lfloor m\rceil$:

$$DCtr_{<j,y_j^{-1}(i)>}, DCtr_{<j,y_j^{-1}(i)+1>}, \ldots, DCtr_{<j,n-1>} \tag{6}$$

by feeding 0 to these counters.

Note that we intentionally perform feeding operations to OSs besides OS j^* when MU i only applies to j^*, and to every OS when MU i is not free. This is because we want to prevent adversaries from inferring MU i's move by exploring the difference on all counters' current lengths.

4) Termination: DP-GS mechanism adopts the same terminating condition as Gale-Shapley algorithm does. Specifically, when no MU are free, or all MUs have applied all OSs, DP-GS terminates the simulation.

Below, we summarize our DP-GS mechanism in Algorithm 1.

C. Performance Analysis

In this section, we analyze the efficiency, privacy guarantees and the accuracy of DP-GS.

Efficiency: In DP-GS, the SA simulates the Gale-Shapley algorithm via maintaining differentially private counters. It is known that Gale-Shapley would terminate after mn rounds at most. Therefore, DP-GS would also run for mn rounds at most. In each round, SA simulates n MUs and needs to maintain $m(n-1)$ counters.

In total, the computation complexity of DP-GS is $O(m^2n^3)$.

Privacy: Following the Billboard Lemma, we have the follow theorem regarding DP-GS privacy guarantee.

Theorem 4: DP-GS is ϵ-joint differentially private.

Proof: Note that DP-GS complies with the billboard model in the following sense: given the (differentially private) counters, and the private preference of any MU i, the final allocation to that MU can be locally computed by simulating the sequence of this MU's application.

According to the Billboard Lemma, to prove our theorem, it suffices to prove these counters' output sequences are ϵ-differentially private regarding the change on one MU's preference.

Note that for every counter regarding MU i, SA always feeds 1 in one round at most, and feeds 0 in the rest rounds in the simulation (since MU i would apply to each OS for once at most, and each counter records the applying history regarding one OS only). Accordingly, SA's input can only be one of the following two types: (1) a bit sequence with all bits being 0 (corresponding to the case where no application has been made ever) and (2) a bit sequence with all bits being 0 except for the x-th bit being 1 (corresponding to the case where application has been made in the x-th round). If SA input is changed from type (1) to type (2) or from type (2) to type (1), we can see the editing distance between the two inputs is only 1. If SAs input is changed from a type (2) sequence to a different type (2) sequence, the editing distance between the two inputs is 2. (Note that SA's input cannot change from a type (1) sequence to another type (1) sequence since there is only one type (1) sequence.) Thus, the greatest change to these counters is a two-bit change in the feeding stream when MU i changes its private preference.

Since the differentially private counter's output is $\epsilon/(2mn)$-differentially private with respect to a single-bit change in the feeding stream, we know each counter's output is $\epsilon/(mn)$-differentially private with respect to one MU's change of its preference according to the Group Privacy Extension Theorem. In addition, note that the change would be fed into at most mn counters at the same time, therefore the joint output of these counters is ϵ-differentially private following the Sequential Composition Theorem. ■

IV. Enhanced Differentially Private Gale-Shapley Mechanism With an Untrusted SA

In this section, we propose EDP-GS, an enhanced differentially private Gale-Shapley mechanism that can be used to protect MUs' private preferences in scenarios where the SA cannot be trusted. We assume the SA is semi-honest. In other words, the SA would follows the predefined mechanism without any deviation. However, it is allowed to infer MUs' private preferences based on its view (e.g. the message it receives, the intermediate output of the mechanism it computes, etc.) during the mechanism's execution.

A. Design Rationale and Mechanism Overview

Note that DP-GS mechanism directly reveals MUs' private preferences to the SA, which immediately breaks the privacy of MUs against a semi-honest SA. To prevent this from happening, EDP-GS lets all MUs maintain a group of differentially private counters locally. Instead of sending its private preference, each MU sends sanitized outputs of these counters to the SA. Thus, each MU's private preference is protected against the SA and other MUs. With the sanitized outputs, every MU and the SA could approximately simulate the Gale-Shapley algorithm.

The main idea of EDP-GS is intuitive, however, we note that simply making the DP-GS mechanism distributed by letting each MU maintain part of the differentially private counters used in DP-GS and performing the simulation does not work. Furthermore, the termination criteria used by DP-GS cannot be applied by EDP-GS. (We will explain the reasons shortly in the Subsections IV-A.1 and IV-A.4.) Belows, we describe how EDP-GS mechanism works in more detail.

1) Constructing Private Counters: Notice that most of the counters used in DP-GS record more than one MU's application. Although updating these counters is easily achieved when the SA knows all MUs' preferences and thus is able to simulate all MUs to apply, it is impossible in our scenario. This is because each MU does not know other MUs' preferences, therefore cannot simulate other MUs' moves to update the counters.

To solve our problem, we come up with new counters in EDP-GS such that the counters maintained by each MU can be updated by this MU without knowing other MUs' preferences, and also the Gale-Shapley algorithm can be simulated based on these counters.

Specifically, EDP-GS lets each MU i to maintain the following m differentially private counters:

$$DCtr_{<1,i>}, DCtr_{<2,i>}, \ldots, DCtr_{<m,i>}, \tag{7}$$

where

$$DCtr_{<j,i>} := \mathcal{BM}(\epsilon/(2m), mn) \tag{8}$$

for every $j \in \lfloor m \rceil$. Here $DCtr_{<j,i>}$ is used to record whether MU i has applied to OS j at every round with the count one meaning the application has been made and zero meaning not. The first parameter of $\mathcal{BM}$ is due to our privacy requirement on EDP-GS, and the second one equals the maximum number of counts we need to record in EDP-GS. It is straightforward to see MU i is capable of updating these counters since they record MU i's own application history in the simulation.

We will see these counters also provide the required information to (approximately) simulate the Gale-Shapley algorithm in the next subsection.

2) Determining MUs' Moves: In each round of EDP-GS, MUs determine their own moves one by one as follows.

Specifically, when MU i's turn arrives, let j^* be the index of its most preferred OS that it has not yet applied.

If MU i is free (or not on any waiting list) at the end of the previous round, MU i simulates itself to apply to OS j^*, and determines the result of its own application by verifying whether

$$\sum_{i'<y_{j^*}^{-1}(i)} DCtr_{<j^*,y_{j^*,i'}>}(t) < z_{j^*}, \tag{9}$$

where t is the current time point, and $DCtr_{<j,i>}(t)$ denotes the sanitized output of the private counter $DCtr_{<j,i>}$ at the current time point. MU i acquires these outputs from the SA' billboard. If the verification is passed, MU i simulates the SA to put itself on OS j^*'s waiting list. Otherwise, MU i simulates the SA to reject its own application.

If MU i is not free (supposing it is on the MU j's waiting list according to its own record) at the end of previous round, EDP-GS re-examines whether it has been replaced by other MUs whose turns arrive before MU i's in this round. To do this, EDP-GS verifies

$$\sum_{i'<y_j^{-1}(i)} DCtr_{<j,y_{j,i'}>}(t) < z_j, \tag{10}$$

If the verification is passed, MU i does not make any move. Otherwise, MU i simulates itself to apply to OS j^*, and determines the result same as it does when MU i is free at the end of previous round.

Proposition 5: The above processes correctly simulate the Gale-Shapley algorithm (in an approximate manner).

Proof: Recall that $y_j^{-1}(i)$ denotes MU i's rank in OS j's preference, accordingly the sets $\{y_{j^*,i'}\}_{i'<y_{j^*}^{-1}(i)}$ and $\{y_{j,i'}\}_{i'<y_j^{-1}(i)}$ contain the index of all MUs who are more preferred to OS j^* and to OS j respectively compared with MU i.

Therefore, neglecting the approximating matter of differentially private counters, (9) (and (10) resp.) actually verifies whether the total number of MUs, who have applied to the same OS that MU i is currently applying to (and the OS that MU i whose waiting list MU i is currently on resp.) and are also more preferred this OS compared with MU i, is greater than the OS j^*'s capacity z_j. According to the proof of Proposition 3, we know this is equivalent to verify whether MU i is one of the top z_j preferred MUs among all MUs that are currently on this OS's waiting list and MU i, which is the very criteria used by Gale-Shapley. ■

3) Updating the Counters and Sending Out the Output: If MU i applies to OS j^* in the current round, it updates its counter regarding OS j^*:

$$DCtr_{<j^*,i>} \tag{11}$$

by feeding 1 to it, and updates its other counters:

$$\{DCtr_{<j,i>}\}_{j\neq j^*} \tag{12}$$

by feeding 0 to these counters.

If MU i makes no move, SA updates all m counters:

$$\{DCtr_{<j,i>}\}_{j\in\lfloor m\rceil} \tag{13}$$

by feeding 0 to them.

Algorithm 2 EDP-GS Mechanism (With an Untrusted, Semi-Honest SA)

Require:
Each MU i knows its private preference $\{\vec{x}_i\}_{i\in\lfloor m\rceil}$;
All OSs' preferences and capacities $\{\vec{y}_j, z_j\}_{j\in\lfloor n\rceil}$ are publicly known;

Ensure:
$\{DCtr_{<j,i>}\}_{i\in\lfloor n\rceil, j\in\lfloor m\rceil}$

1: Each MU i initializes m differentially private counters: $DCtr_{<j,i>} := \mathcal{BM}(\epsilon/(2m), mn)$, for every $j \in \lfloor m\rceil$, and sets the index of its allocated OS $\mu(i)$ to null;
2: All MUs jointly simulate Gale-Shapley for mn rounds as follows:
3: • In each round, for each MU i and its most preferred OS j^* that MU i has not applied to yet, SA simulates MU i's move by updating the counters and the waiting lists as follows:
– If $\mu(i)$ equals null, it feeds 1 to $DCtr_{<j^*,i>}$, and feeds 0 to $DCtr_{<j,i>}$ for each $j \in \lfloor m\rceil \setminus \{j^*\}$; In addition, MU i gets $\{DCtr_{<j^*,y_{j^*,i'}>}\}_{i'<y_{j^*}^{-1}(i)}$ from the SA's billboard, and sets $\mu(i)$ to j^* if $\sum_{i'<y_{j^*}^{-1}(i)} DCtr_{<j^*,y_{j^*,i'}>}(t) < z_{j^*}$, where t is the counter's current time point;
– If $\mu(i) = j$ at the end of previous round, and $\sum_{i'<y_j^{-1}(i)} DCtr_{<j,y_{j,i'}>}(t) \geq z_j$, it sets $\mu(i)$ to null, feeds 1 to $Ctr_{<j^*,i>}$, and feeds 0 to $Ctr_{<j,i>}$ for each $j \in \lfloor m\rceil \setminus \{j^*\}$; In addition, if $\sum_{i'<y_{j^*}^{-1}(i)} DCtr_{<j^*,y_{j^*,i'}>}(t) < z_{j^*}$, sets $\mu(i)$ to j^*;
– If $\mu(i) = j$ at the end of previous round, and $\sum_{i'<y_j^{-1}(i)} DCtr_{<j,y_{j,i'}>}(t) < z_j$, it feeds 0 to $DCtr_{<j,i>}$ for each $j \in \lfloor m\rceil$;
– MU i sends the sanitized outputs of its private counters $\{DCtr_{<j,i>}(t)\}_{j\in\lfloor m\rceil}\}$ to AS. AS publishes them on its billboard.
4: OS $\mu(i)$ is allocated to MU i for each $i \in \lfloor n\rceil$
5: **return** $\{DCtr_{<j,i>}\}_{i\in\lfloor n\rceil, j\in\lfloor m\rceil}$

In either case, MU i sends the sanitized outputs of all m counters at the current time point to the SA. And SA publishes these outputs on its billboard.

4) Termination: Recall that DP-GS terminates whenever all MUs have applied all OSs or no MU is free. However, EDP-GS cannot use the same terminating condition since the real applying status of each MU is known to itself only.

Therefore, EDP-GS terminates after mn rounds. This is due to the fact that Gale-Shapley can always terminate within mn rounds.

Below, we summarize our EDP-GS mechanism in Algorithm 2.

B. Performance Analysis

In this section, we analyze the efficiency, privacy guarantees and the accuracy of EDP-GS.

Efficiency: Different from DP-GS, EDP-GS is performed jointly by all MUs. Specifically, each MU updates its m private

counters, and sends the output of these counters to AS for at most mn rounds. Therefore, EDP-GS has a round complexity of $O(mn)$, and the computation complexity for each MU is $O(m^2n)$ since it only needs to feed m counters in each round.

Privacy: Following the Billboard Lemma, we have:

Theorem 6: EDP-GS is ϵ-joint differentially private.

Proof: Similar to the proof of Theorem 4, we prove this theorem by proving all counters' output sequences, which are also EDP-GS's output, are ϵ-differentially private. Note that for every counter that MU i is supposed to feed in the simulation, MU i always feeds 1 in one round at most, and feeds 0 in the rest rounds (since it would apply to each OS for once at most, and each counter records the applying history regarding one OS only.). Thus, same to the counters in DP-GS, the greatest change to each counter in EDP-GS is a two-bit change in the feeding stream when MU i changes its private preference. Since the differentially private counter's output is $\epsilon/(2m)$-differentially private with respect to a single-bit change in the feeding stream, we know each counter's output is $\epsilon/(m)$-differentially private with respect to one MU's change of its preference according to the Group Privacy Extension Theorem. In addition, note that the change would be fed into m counters at the same time in EDP-GS, therefore the joint output of these counters is ϵ-differentially private following the Sequential Composition Theorem. ■

V. Evaluation

We have implemented both DP-GS and EDP-GS mechanisms and performed a series of experiments to evaluate their performance. Specifically, we design experiments to test our mechanisms on three aspects: efficiency, accuracy and privacy. To have a better idea on our mechanisms' performance, we use the original Gale-Shapley (GS) algorithm as the baseline in efficiency and accuracy. As for privacy, we adopt a popular measure called *privacy leakage*:

Definition 3 (Privacy Leakage [16]): Given a mechanism M, let $\vec{\phi}$ and $\vec{\phi}'$ be two input vectors which only differ in a single component. Let O be the outcome space. The privacy leakage is the maximum of absolute differences between the logarithmic probabilities of any outcome, i.e.,

$$\max |\ln \pi_o - \ln \pi'_o|, \tag{14}$$

where π and π' is the probability distribution over the outcome space with respect to $\vec{\phi}$ and $\vec{\phi}'$, respectively.

All results are averaged over 1000 runs, except that the privacy-testing data is generated by repeating the mechanisms for enough times till the outcome space is fully covered, and then the results are averaged over 100 runs.

A. Evaluating DP-GS

Efficiency: There are four factors that affect DP-GS's efficiency: input size of MUs, input size of OSs, OS's capacity and privacy parameter ϵ. In order to understand these factors' effects on the efficiency, we let one factor vary and fix the others in each time. The default system setting we use is: 100 MUs, 10 OSs, each OS's capacity is 10 and $\epsilon = 0.3$. When we let one factor vary, others would be set according to the default setting. Results are shown in Figure 3a to Figure 3c.

From Figure 3a, we can see that input sizes of MUs and OSs affect the efficiency in different ways. It is easy to understand the running time is higher when there are more users. But when number of users is small, OSs' numbers' effection on the efficiency is not so obvious. When number of users is big, it takes a long time for all users to become stable due to the lack of OSs. The capacity of OS has a similar effect as the number of OSs, only is relatively more obvious. The smaller OS's capacity is, the harder finding a stable matching will be. This can be seen in Figure 3b. Figure 3c shows how ϵ affects the efficiency. The result is reasonable because when we use a smaller ϵ, we sacrifice more efficiency and accuracy for privacy.

Accuracy: First, we set a rating method to evaluate the outputs of both GS and DP-GS: For each OS, if it has a MU which is its j-th preferred MU on its waiting list, the output will score $m - j$ points. After traversing all OSs' waiting list and sum up points, the output will have a final score. Then, we run GS and DP-GS on the same preference profiles of MUs and OSs, and rate two mechanisms' outputs. And we use the ratio of DP-GS's score to GS's score to evaluate the accuracy of our DP-GS mechanism. Results are shown in Figure 3d to Figure 3f. We can see the input size such as number of MUs has an obvious impact on accuracy. In addition, we find that number of OSs and OS's capacity have very similar impacts on accuracy. The trade between accuracy and privacy is obvious too. According to the figure, $\epsilon = 0.6$ seems to be the best choice considering both accuracy and privacy in our test.

Privacy: We test the privacy leakage from different angles. First, we are interested in the distribution of the privacy leakage, which is affected mainly by ϵ and output space of the mechanism. Figure 3g shows the cumulative distribution of privacy leakage with $\epsilon = 0.5$, and Figure 3h shows the cumulative distribution with $\epsilon = 1.0$. It is obvious that bigger outcome space preserves better privacy. Our DP-GS mechanism can guarantee that most outputs are within leakage 0.2 even with $\epsilon = 1.0$. Figure 3i shows the direct relation between ϵ and privacy leakage. It is harder to achieve less leakage when ϵ is smaller than 0.3. Thus we suggest to choose ϵ near 0.5.

B. Evaluating EDP-GS

Efficiency: Our EDP-GS has better efficiency than DP-GS, because less counters are maintained in each round. Another advantage of EDP-GS is that number of OSs has little impact on its efficiency which is shown in Figure 4a. Because number of MUs determines the running time while number of OSs mainly affects the running space. Capacity of OS affects the efficiency greatly. It can be seen from Figure 4b that smaller capacity makes the mechanism harder to become stable. Figure 4c shows how ϵ affects the running time. But this time ϵ has a weaker impact on the mechanism's efficiency since counters in EDP-GS are much less than in DP-GS.

Accuracy: An interesting finding in both Figure 4d and Figure 4e is that accuracy curves rise

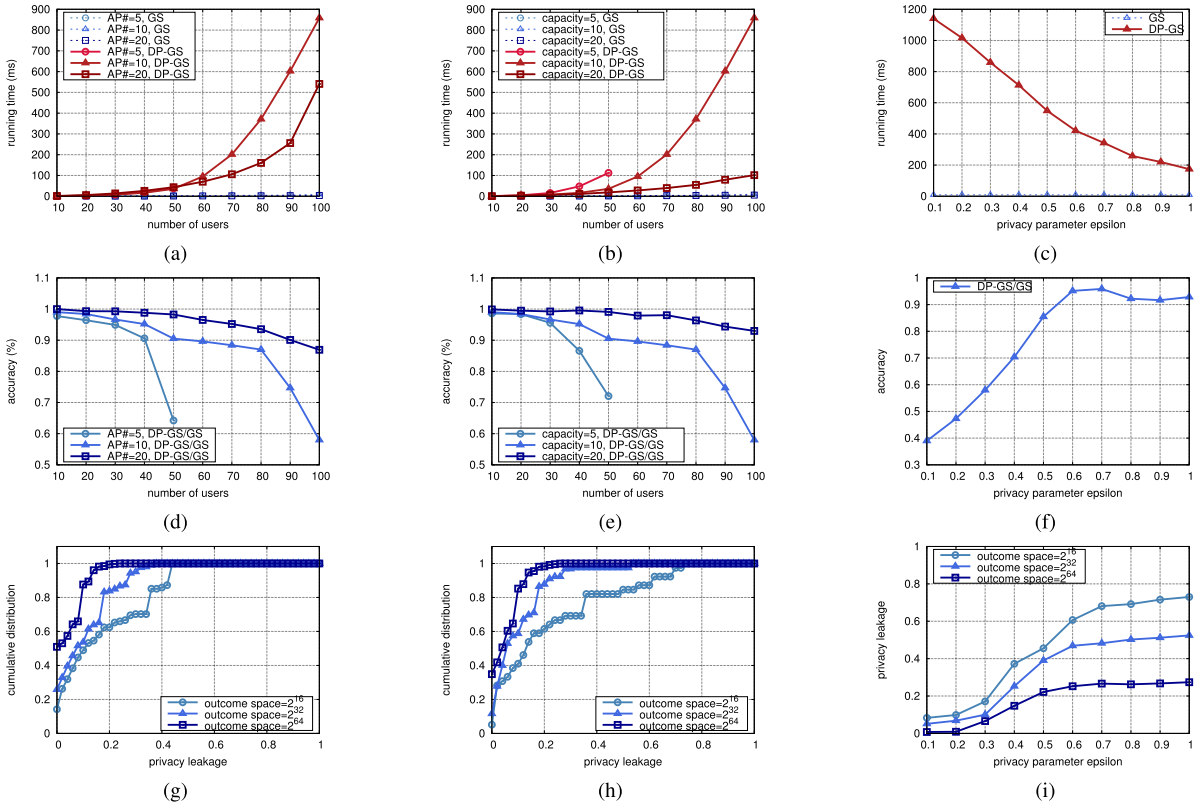


Fig. 3. Evaluating results of DP-GS. (a-c) efficiency results; (d-f) accuracy results; (g-i) privacy results.

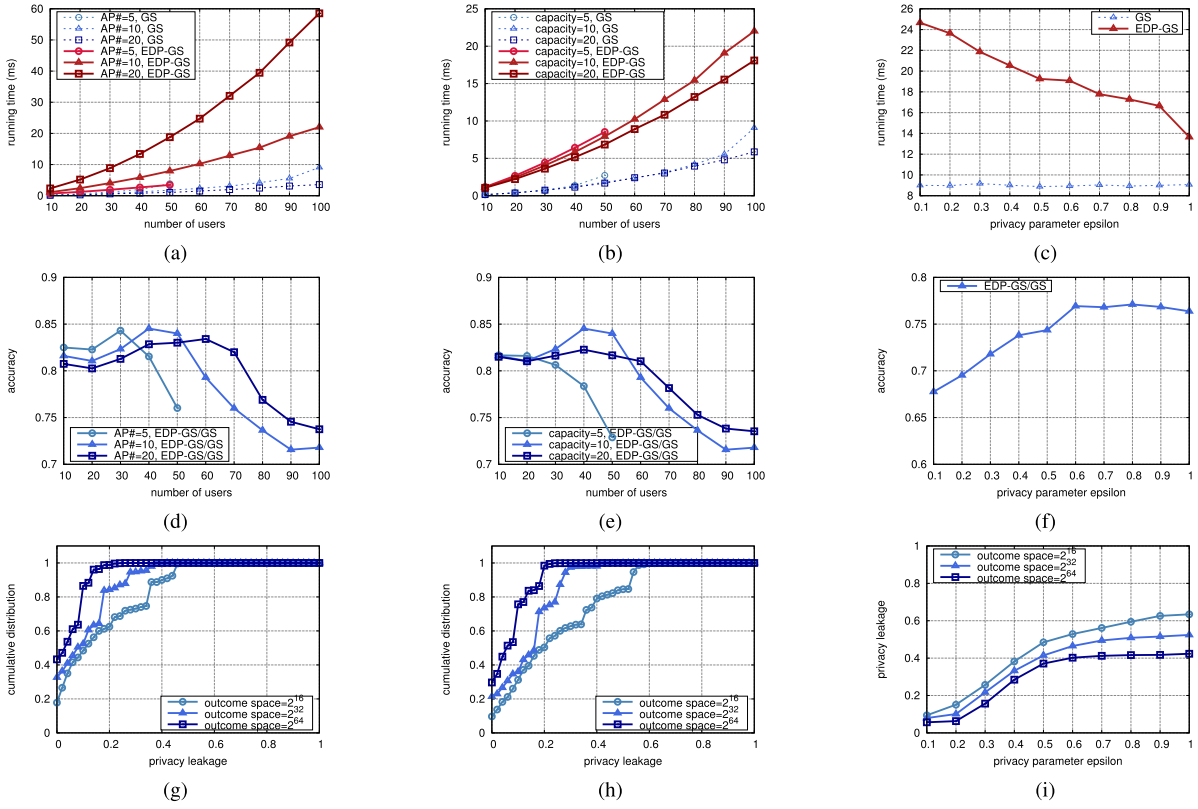


Fig. 4. Evaluating results of EDP-GS. (a-c) efficiency results; (d-f) accuracy results; (g-i) privacy results.

first and then descend quickly. Because number of counters is small, privacy parameter's impact on accuracy is small when the number of MUs is small. But ϵ's cumulative impact makes its effect increase sharply with the increasing of number of MUs. Accuracy's descending speed caused by privacy parameter is much higher than accuracy's increasing

speed caused by number of MUs. Also, accuracy's increasing cause by number of MUs is not linear. Thus, once descending speed is higher than increasing speed, accuracy will decrease quickly. Capacity's impact is more obvious when there are enough MUs. In Figure 4f we can see that ϵ does not affect accuracy as effectively as it does to DP-GS.

Privacy: Results are shown in Figure 4g to Figure 4i. Our EDP-GS can guarantee that most outputs have privacy leakage less than 0.5 even when outcome space is very small and ϵ is 1.0. Privacy parameter is still the main factor that affects privacy leakage. Tendencies of leakage caused by ϵ are more gentle in EDP-GS. This leads to a good phenomenon where leakage grows slowly with the increase of ϵ.

VI. Related Works

Spectrum sharing has been studied in several different areas [17]–[22] such as cognitive raido networks, D2D networks and cellular networks, to mitigate the spectrum scarcity problems in corresponding networks. In this paper, we study an important security problem in the mobile traffic offloading system, which is one of spectrum sharing's applications in cellular networks.

Huang et al. [23] are among the first to study protecting bidders privacy in spectrum auctions. Their solution aims to enable the computation of the auctions outcome without revealing too much information about bidders private evaluation, thus does not protect auctions outcome. Noticing adversaries may exploit auctions outcome to infer bidders private evaluations, researchers start resorting to differential privacy to design spectrum auction mechanisms. For example, Zhu et al. [24] first propose a differentially private auction mechanism that achieves approximate revenue maximization and approximate truthfulness. Later, Zhu and Shin [16] adopts a new payment computation method and design a differential private auction mechanism that achieves strict truthfulness. In a recent work by Wu et al. [25], authors further take fairness into consideration and design an auction mechanism that achieve approximate truthfulness, approximate revenue maximization and also fairness. The above three works on differentially private spectrum auction have one common goal which is to maximize the auctioneers revenue. This allows they to use the revenue as a utility function and adopt the Exponential mechanism to achieve differential privacy. However, in our problem, we have no such a goal and thus we cannot apply Exponential mechanism to achieve differential privacy as they do.

User allocation problems in mobile traffic offloading systems have been considered in a few recent works [1], [2], [26]–[30] under different optimizing objectives such as throughout maximization, load balance, fairness, stableness, and etc. However, none of these works considers the possible threats to mobile phone users' location privacy.

Location privacy protection has been mostly studied in location-based service (LBS) areas. There are mainly two types of works to protect location privacy in LBS. One is by obfuscating user's location (e.g. [31]). The other one is by anonymizing location data (e.g. [32]). Our problem requires to protect the data privacy in a multi-party computation scenario, which is more complicated compared with the single-user-single-server scenario that is most studied in LBS area.

Differential privacy was first defined by Dwork et al. [4], and is mostly used for answering numeric queries on private datasets. So far, there is only a handful of works that study private optimization mechanisms. A similar work to ours is [6]. In [6], Hsu et al. design a joint differentially private goods allocation mechanism called PMatch to approximately maximize the social welfare. PMatch aims to protect each user's real valuation of a good which is a numeric value, and use a differentially private counter for each good to record the intermediate bidding status of several parallel ascending price auctions. Despite that our work adopts the same joint differential privacy concept, and the differentially private counters as our fundamental building block, our mechanisms are totally different due to the underlying allocation problems or algorithms in two works are entirely different.

Finally, we note that secure implementation of Gale-Shapley algorithm is also studied in [33] and [34]. These works focus on generating correct allocation without revealing the preferences to the SA or MUs. Our second mechanism EDP-GS also achieves this goal only in a differentially private manner. However, the protocols proposed in these works do not protect the allocation solution, and cannot deal with colluding users.

VII. Conclusion and Future Works

Following the idea of spectrum sharing, mobile traffic offloading systems are proposed to mitigate the severe spectrum scarcity faced by cellular service providers nowadays. In this paper, we study the problem of location privacy protection in the mobile traffic offloading system, and propose two joint differentially private Gale-Shapley mechanisms for it. Our mechanisms provide strong privacy guarantees regarding each mobile phone user's private preference even when all other mobile phone users collude against this user. There are a few interesting issues which deserve further study. The first one is how to further optimize the stableness of the mechanisms' outputs. The second one is to study the case that offloading stations' preferences are not publicly known.

References

[1] W. Saad, Z. Han, R. Zheng, M. Debbah, and H. V. Poor, "A college admissions game for uplink user association in wireless small cell networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Toronto, ON, Canada, Apr./May 2014, pp. 1096–1104.

[2] W. Wang, X. Wu, L. Xie, and S. Lu, "Femto-matching: Efficient traffic offloading in heterogeneous cellular networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Hong Kong, Apr./May 2015, pp. 325–333.

[3] D. Gale and L. S. Shapley, "College admissions and the stability of marriage," *Amer. Math. Monthly*, vol. 69, no. 1, pp. 9–15, 1962.

[4] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography*. Berlin, Germany: Springer, 2006, pp. 265–284.

[5] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *Proc. 48th Annu. IEEE Symp. Found. Comput. Sci. (FOCS)*, Oct. 2007, pp. 94–103.

[6] J. Hsu, Z. Huang, A. Roth, T. Roughgarden, and Z. S. Wu, "Private matchings and allocations," in *Proc. 46th Annu. ACM Symp. Theory Comput.*, 2014, pp. 21–30.

[7] O. Goldreich, *Foundations of Cryptography: Basic Applications*. New York, NY, USA: Cambridge Univ. Press, vol. 2. 2009, pp. 619–625.

[8] L. E. Dubins and D. A. Freedman, "Machiavelli and the Gale–Shapley algorithm," *Amer. Math. Monthly*, vol. 88, no. 7, pp. 485–494, 1981.
[9] C. Dwork, "Differential privacy," in *Proc. 33rd Int. Colloq. Automata, Lang. Program. II (ICALP)*, vol. 4052. Venice, Italy, Jul. 2006, pp. 1–12.
[10] F. D. McSherry, "Privacy integrated queries: An extensible platform for privacy-preserving data analysis," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2009, pp. 19–30.
[11] M. Kearns, M. Pai, A. Roth, and J. Ullman, "Mechanism design in large games: Incentives and privacy," in *Proc. 5th Conf. Innov. Theor. Comput. Sci.*, 2014, pp. 403–410.
[12] F. McSherry and I. Mironov, "Differentially private recommender systems: Building privacy into the net," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 627–636.
[13] C. Dwork, M. Naor, and S. Vadhan, "The privacy of the analyst and the power of the state," in *Proc. IEEE 53rd Annu. Symp. Found. Comput. Sci. (FOCS)*, Oct. 2012, pp. 400–409.
[14] T.-H. H. Chan, E. Shi, and D. Song, "Private and continual release of statistics," *ACM Trans. Inf. Syst. Secur.*, vol. 14, no. 3, 2011, Art. no. 26.
[15] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum, "Differential privacy under continual observation," in *Proc. 42nd ACM Symp. Theory Comput.*, 2010, pp. 715–724.
[16] R. Zhu and K. G. Shin, "Differentially private and strategy-proof spectrum auction with approximate revenue maximization," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr./May 2015, pp. 918–926.
[17] I. F. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty, "A survey on spectrum management in cognitive radio networks," *IEEE Commun. Mag.*, vol. 46, no. 4, pp. 40–48, Apr. 2008.
[18] Y. Zou, Y.-D. Yao, and B. Zheng, "Cooperative relay techniques for cognitive radio systems: Spectrum sensing and secondary user transmissions," *IEEE Commun. Mag.*, vol. 50, no. 4, pp. 98–103, Apr. 2012.
[19] G. Ding, J. Wang, Q. Wu, Y.-D. Yao, F. Song, and T. A. Tsiftsis, "Cellular-base-station-assisted device-to-device communications in TV white space," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 1, pp. 107–121, Jan. 2016.
[20] B. Han, P. Hui, V. S. A. Kumar, M. V. Marathe, J. Shao, and A. Srinivasan, "Mobile data offloading through opportunistic communications and social participation," *IEEE Trans. Mobile Comput.*, vol. 11, no. 5, pp. 821–834, May 2012.
[21] D. Xu and Q. Li, "Effective capacity region and power allocation for two-way spectrum sharing cognitive radio networks," *Sci. China Inf. Sci.*, vol. 58, no. 6, pp. 1–10, 2015.
[22] G. Zhang, P. Liu, K. Yang, Y. Du, and Y. Hu, "Orthogonal resource sharing scheme for device-to-device communication overlaying cellular networks: A cooperative relay based approach," *Sci. China Inf. Sci.*, vol. 58, no. 10, pp. 1–9, 2015.
[23] Q. Huang, Y. Tao, and F. Wu, "Spring: A strategy-proof and privacy preserving spectrum auction mechanism," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 827–835.
[24] R. Zhu, Z. Li, F. Wu, K. Shin, and G. Chen, "Differentially private spectrum auction with approximate revenue maximization," in *Proc. 15th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2014, pp. 185–194.
[25] C. Wu, Z. Wei, F. Wu, G. Chen, and S. Tang, "Designing differentially private spectrum auction mechanisms," *Wireless Netw.*, vol. 22, no. 1, pp. 105–117, 2016.
[26] W. Zhao and S. Wang, "Cell planning for heterogeneous cellular networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2013, pp. 1032–1037.
[27] E. Aryafar, A. Keshavarz-Haddad, M. Wang, and M. Chiang, "Rat selection games in HetNets," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 998–1006.
[28] S. Singh and J. G. Andrews, "Joint resource partitioning and offloading in heterogeneous cellular networks," *IEEE Trans. Wireless Commun.*, vol. 13, no. 2, pp. 888–901, Feb. 2014.
[29] Y. Zhao, J. Wu, F. Li, and S. Lu, "On maximizing the lifetime of wireless sensor networks using virtual backbone scheduling," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 8, pp. 1528–1535, Aug. 2012.
[30] X. Ma, M. Sheng, J. Li, and J. Xin, "Interference migration using concurrent transmission for energy-efficient HetNets," *Sci. China Inf. Sci.*, vol. 59, no. 2, pp. 1–10, 2016.
[31] B. Gedik and L. Liu, "Protecting location privacy with personalized k-anonymity: Architecture and algorithms," *IEEE Trans. Mobile Comput.*, vol. 7, no. 1, pp. 1–18, Jan. 2008.
[32] H. Kido, Y. Yanagisawa, and T. Satoh, "Protection of location privacy using dummies for location-based services," in *Proc. 21st Int. Conf. Data Eng. Workshops*, Apr. 2005, p. 1248.
[33] P. Golle, "A private stable matching algorithm," in *Financial Cryptography and Data Security*. Berlin, Germany: Springer, 2006, pp. 65–80.
[34] M. Franklin, M. Gondree, and P. Mohassel, "Improved efficiency for private stable matching," in *Topics in Cryptology*. Berlin, Germany: Springer, 2006, pp. 163–177.

Yuan Zhang received the B.S. degree in automation from Tianjin University in 2005, the M.S. degree in software engineering from Tsinghua University in 2009, and the Ph.D. degree in computer science from the State University of New York at Buffalo in 2013. He is interested in security, privacy, and economic incentives.

Yunlong Mao is currently pursuing the Ph.D. degree in computer science and technology with Nanjing University. He is interested in wireless communication, privacy and security.

Sheng Zhong received the B.S. and M.S. degrees from Nanjing University in 1996 and 1999, respectively, and the Ph.D. degree from Yale University in 2004, all in computer science. He is interested in security, privacy, and economic incentives.