Privacy preserving computations for viral marketing: the case of rational players

Rica Gonen
Department of Management and Economics
The Open University
Ra'anana, Israel
ricagonen@gmail.com

Tamir Tassa
Department of Mathematics and Computer Science
The Open University
Ra'anana, Israel
tamirta@openu.ac.il

***Abstract*—Viral marketing is a methodology which is based on exploiting a pre-existing social network in order to increase brand awareness or product sales through self-replicating viral processes. An essential computational task towards setting up an effective viral marketing campaign is to estimate social influence. Such estimates are usually done by analyzing user activity data. The data analysis and sharing that is needed to estimate social influence raises important privacy issues that may jeopardize the legal, ethical and societal acceptability of such practice, and in turn, the concrete applicability of viral marketing in the real world. Tassa and Bonchi (EDBT 2014) devised secure multi-party protocols that allow a group of service providers and a social networking platform to jointly compute social influence in a privacy preserving manner. They assumed that the players are semi-honest, i.e., that they follow the protocol correctly, but at the same time they examine their view of the protocol in order to extract information on inputs provided by their peers. In this paper we discuss the case of selfish rational players; such players participate in the protocol and follow it correctly only if it is in their best interest and maximizes their utility. We enhance the protocol of Tassa and Bonchi by incorporating into it mechanisms that incentivize the players to participate in the protocol truthfully.**

***Keywords*-viral marketing; privacy-preserving data mining; game theory**

I. Introduction

Viral marketing is a methodology which is based on exploiting a pre-existing social network in order to increase brand awareness or product sales through self-replicating viral processes. A basic idea in the practice of viral marketing is to target a small subset of seed users in the network who, hopefully, will trigger a word-of-mouth driven cascade which will deliver the marketing message to a large portion of the network, with a small initial marketing cost.

The key computational problem in this context was formalized by Kempe *et al.* [4] and is known as *influence maximization*. The input to that problem consists of two ingredients. The first one is a directed social graph where each arc is labeled with a probability representing the strength of social influence along the arc; in other words, the label of an arc (u, v) is an estimate to the probability that an action which was carried out by u will influence v to do that action too. The second ingredient is an integer k, which represents the "budget" that is allocated for some viral marketing campaign. The goal is then to find k nodes in the graph which, if activated, will maximize the expected number of nodes that eventually get activated. This fundamental problem has received a substantial attention by the data mining research community ever since its formalization. However, despite being under the spotlights of academic research for more than ten years, the problem of influence maximization remains a theoretical one, which is still far from real-world applicability.

One of the unrealistic assumptions made by most of the studies in this field is that the input social graph is already labeled with the above described probabilities representing the influence strengths of the social links. However, real-world social networks do not come with such additional information. Consequently, several studies [11], [9], [2] focused on the more preliminary problem of learning social influence strength. That problem takes, as input, the social graph together with a log of past propagation traces (for instance, the history of sales of some products to the users in the social network). Assuming a *propagation model* which governs the manner in which influence-driven propagations occur, the goal is then to learn the parameters of the model, i.e., the influence probability associated to each arc.

But even though such studies helped bridging between theory and practice, the gap remained substantial. Another manifestation of that gap is the fact that all of the above mentioned studies assumed that all inputs to the problem of learning social influence strength are owned by one party. In real life, however, social networking platforms are owned by a third party such as Facebook or Twitter (we refer to the network owner as the *host*, denoted H), while the log of past propagation traces is distributed among several service providers, $P_1, \ldots, P_m$ who are different entities (e.g. e-book stores or travel websites). Viral marketing is offered by H as a service to $P_1, \ldots, P_m$ [1], [27]: this might be in the form of advertising space, paid by $P_1, \ldots, P_m$ to H on a pay-per-impression basis. H attempts to optimize the placement of advertisements in order to maximize influence diffusion. This optimization is provided as a service to $P_1, \ldots, P_m$, with the primary goal of making the social network more attractive as a marketing platform. However, in order to solve the influence maximization problem and to set up the viral marketing campaign, it is first needed to learn the influence strength.

Since each of the involved parties wish to keep their proprietary data secret (in order to maintain commercial

benefits and to respect privacy legislation[1][2], the problem becomes a problem of secure multi-party computation. The inputs to the computation are held by distinct and non-trusting parties (the host H and the service providers $P_1, \ldots, P_m$) – the former has the social graph, while the latter hold logs of past propagation traces; the output is the social strength of each link in the network. Tassa and Bonchi [22] addressed that problem and offered secure multi-party protocols that allow a group of service providers and a social network host to jointly compute the link influence strengths, while respecting their privacy. (The privacy analysis was carried out under the standard assumption that the participating players are semi-honest, i.e., they respect the protocol, but try to learn as much as they can from their own view of the protocol on the private information held by other players.[3])

Alas, even that seminal work which, to the best of our knowledge, initiated the study of privacy-preserving viral marketing, is not expected to make the industry embrace such innovative technological solutions with open arms. The problem now is that in the protocols that were presented in [22], only the host receives all information regarding the strength of the social links, while the service providers get nothing, even though they participate in the protocol by contributing their private data. However, rational players (service providers) might refuse to participate and contribute their private data or condition their participation in such protocols on receiving part of the computational result. Hence, it is needed to devise game-theoretic mechanisms that incorporate incentives for the service providers who participate in the protocol. By incorporating incentives and designing the game-theoretic mechanism accordingly, the service providers will know upfront that if they contribute their data to the computation, they too may learn part of the output which they can use for their own *direct marketing* purposes. In this paper we offer such game-theoretic mechanisms to enhance the protocols of [22] for estimating the social influence strength.

The paper is organized as follows. In Section II we review related work. In Section III we define the computational problem. In Section IV we provide a brief summary of the protocol that was proposed in [22] and is the basis of our rational protocol which we proceed to describe in Section V. We conclude in Section VI.

II. Background and related work

In this section we provide a brief overview of the problem of learning link influence strength and the field of secure multi-party computation (Sections II-A and II-B). Then, in Section II-C, we provide a somewhat more elaborated overview of the field of rational computations, as the main contribution of this study is the extension of an existing secure multi-party protocol to the realm of rational behavior.

A. Learning link influence strength

A basic computational problem in the area of viral marketing is that of selecting the set of users to be targeted by the campaign. The first algorithmic treatment of the problem was provided by Domingos and Richardson [17], [5], who modeled the diffusion process in terms of Markov random fields and proposed heuristic solutions. The problem was formalized by Kempe *et al.* [4] as a discrete optimization problem, named *influence maximization*: given a social network where each link is associated with an estimate of influence strength, and a budget k, find k nodes that provide the maximum *expected spread*, i.e., the expected number of active nodes at the end of the process. The activation of nodes is governed by a probabilistic propagation model. For instance, in the *Independent Cascade* (IC) model [4], when a node u first becomes active, say at time t, it is considered contagious. It has one chance of influencing each inactive neighbor v with probability $p_{u,v}$ (the strength of influence associated with the arc (u, v)), independently of the history thus far. If the activation tentative succeeds v becomes active at time $t + 1$, and consequently, has a one-time chance to influence its yet inactive neighbors. Part of the contribution of Kempe *et al.* [4] was to provide a simple (but computationally prohibitive) greedy algorithm with approximation guarantees. Following that seminal work, considerable effort was devoted to developing methods for improving the efficiency of influence maximization computations. Most of those studies assumed a weighted social graph as input and did not consider the problem of computing the link influence strength, e.g. [26], [25]. Other studies focused specifically on the latter problem [11], [9], [2].

Saito *et al.* [11] were the first to study how to learn the link influence probabilities from a set of past propagations. They neatly formalized the likelihood maximization problem and then applied Expectation Maximization (EM) to solve it. Although elegant, their formulation has some limitations when it comes to practice. First, real data needs to be heavily discretized to meet the assumed input format [16]. Second, the EM-based method is particularly prone to overfitting [3]. Finally, it is not very scalable as it needs to update the influence probability associated to each arc in each iteration.

For these reasons and for the sake of simplicity, we avoid the complexity of the EM-based approach when learning the influence probabilities for the influence maximization problem, and instead we follow a simpler definition by Goyal *et al.* [2], which also considered temporal decay of influence. We provide the details of this definition in Section III.

B. Secure multi-party computation

Consider a setting in which data is distributed among several parties who aim to jointly perform data mining on

[1] http://techcrunch.com/2013/01/24/my-precious-social-graph/

[2] http://mashable.com/2012/07/27/twitter-instagram-find-friends/

[3] See [18], [10] for a discussion and justification of that assumption.

the unified corpus of data that they hold, while protecting their privately owned data records. This is a problem of *secure multi-party computation* (SMC). In the general setting of SMC, there are several parties (or players), $P_1, \ldots, P_n$, where each party P_i holds a private value x_i. The goal is to compute the value $f(x_1, \ldots, x_n)$, where f is some publicly known function of n variables, so that each party does not learn anything about the private inputs of the other parties, except the information that is implied by his own input and the output result $f(x_1, \ldots, x_n)$. The problem of secure two-party computation was solved by Yao [29] for any function f that can be represented by a binary or an algebraic circuit. While generic protocols, such as Yao's and its extensions to any number of players, apply in theory to a wide class of functions, their applicability in practice is limited to functions that have a compact representation as a circuit, due to their high computational and communication complexities. The aim of further studies in this field is to find more efficient solutions for specific problems of SMC: e.g., decision trees [14], clustering [28], association rule mining [24], [12], [20], or anonymization [21], [23].

C. Rational computations

The fields of game theory and cryptographic protocol design are both concerned with the study of "interactions" among mutually distrusting parties. These two subjects have, historically, developed almost entirely independently within different research communities and, indeed, they tend to have a very different flavor. In the last decade, however, motivated by the desire to develop more realistic models of (and protocols for) such interactions, there has been significant interest in combining the techniques and approaches of both fields. Current research at the intersection of game theory and cryptography can be classified into two broad categories: applying cryptographic protocols to game-theoretic problems, and applying game-theoretic models and definitions to the general area of cryptographic protocol design. In our work we focus on the latter category, on which we proceed to elaborate.

Traditionally, cryptographic protocols are designed under the assumption that some parties are honest and faithfully follow the protocol, while some parties are malicious and behave in an arbitrary fashion. The game-theoretic perspective, however, is that all parties are simply rational and behave in their own best interests. This viewpoint is incomparable to the cryptographic one: although no one can be trusted to follow the protocol (unless it is in their own best interests), the protocol need not prevent "irrational" behavior. Early on Shoham and Tennenholtz [19] considered the following rational problem: Assume a set of parties $P_1, \ldots, P_n$, where party P_i begins by holding an input x_i. We assume that the vector of inputs $\mathbf{x} = (x_1, \ldots, x_n)$ is chosen according to some known distribution D. The parties want to compute a (possibly probabilistic) function f, where $f(\mathbf{x}) = \mathbf{y} = (y_1, \ldots, y_n)$ and P_i receives y_i. The parties run some protocol $\Pi = (\Pi_1, \ldots, \Pi_n)$, and we assume that this protocol is correct in the sense that it yields the correct output if it is run honestly (namely, if all players supply to the protocol their true inputs and then they execute the operations of the protocol correctly). However, we do not assume that the parties use their given true inputs. The utility function of P_i is now a polynomial-time function of its view during the execution of Π, the initial inputs $\mathbf{x}$, and the outputs $\mathbf{y}_{-i} := (y_1, \ldots, y_{i-1}, y_{i+1}, \ldots, y_n)$ of all other parties. In the computation of $f(\mathbf{x})$, a party P_i that has x_i as input can replace its input with some other value $x_i' = \delta_i(x_i)$, where δ_i denotes P_i's strategy. Shoham and Tennenholtz [19] define the class of Non-Cooperative Computation functions for which, roughly speaking, setting δ_i to the identity function is a Nash equilibrium for all distributions D. Focusing on Non-Cooperative Computation functions appears to be a mistake that unnecessarily limits the class of functions under study.

Halpern and Teague [7] were the first to suggest that Nash protocols do not suffice but, instead, stronger notions are needed. As a motivating example [7], consider t-out-of-n secret sharing ($t < n$) under the assumption that each party (1) prefers to learn the secret above all else; and (2) otherwise, prefers that other parties do not learn the secret. Consider the naïve protocol in which each party simply broadcasts their share. (We assume authenticated shares, so each party can choose either to broadcast the correct value or nothing.) It appears that a Nash equilibrium of this protocol is that each P_i would prefer not to broadcast if $t - 1$ parties broadcast. In this case only P_i recovers the secret. That is, following the protocol is weakly dominated by not following the protocol, and we might expect that no one follows the protocol, whence the protocol is not very useful. To address this, Halpern and Teague suggest looking for Nash protocols where players' strategies survive iterated deletion of weakly dominated strategies. Such protocols were constructed in [7], [8], [15], [6]. Kol and Naor [13] argue that the requirement of surviving iterated deletion does not suffice to rule out protocols that are, intuitively, irrational and they explicitly reject the idea of "weighting" the utility of strategies according to the probability with which a given history is reached (the basic idea of iterated deletion of weakly dominated strategies). They also claim that the notion of surviving iterated deletion is difficult to work with and does not seem to capture intuition very well. Moreover, it leads to other undesirable consequences such as the fact that, if we do not assume simultaneous channels, then protocols in which two parties are supposed to speak in the same round are inherently problematic, since each party will simply wait for the other to go first. Kol and Naor thus suggest another notion. Informally, they require that conditioned on reaching any history that occurs with positive probability, players' strategies should remain in equilibrium. In their definition of a computational game, they allow the action of a player P_i to run in time polynomial in $k + r$, where r is the number of rounds that have been played thus far and k is a security parameter provided to all parties at the beginning of the game.

Regardless of the equilibrium definition the literature mentioned above, e.g. [7], [8] point to the fact that the only hope of getting a practical mechanism for secret sharing lies in using uncertainty about when the game will end to induce cooperation. In order to induce the uncertainty, the literature composes mechanisms with an addition entity that is in charge of initiating multiple rounds of the protocol such that the timing of the last round is unknown. Such a mechanism structure allows for a punishment strategy where the player might gain if he chooses not to cooperate in the last round, however, he will be "punished" with much reduced utility if he is caught being uncooperative and it is not the last round.

III. Problem definition

We consider a social network, which is a graph $G = (V, E)$ where the nodes are users and the links denote some relation between the users. When the relation is symmetric (such as the "friendship" relation in Facebook) the graph is undirected; when the relation is asymmetric (such as the "following" relation in Twitter) the graph is directed. We shall assume hereinafter that the graph is directed[4] and a link $(u, v) \in E$ indicates the fact that v is a follower of u, i.e., v is notified about u's activities, or in other terms, u can influence v.

We are also given a relation $\mathbb{L}(User; Action; Time)$, that we call *action log*. Each record in that log is a tuple of the form (v, α, t) which indicates the fact that user v performed action α at time t. We assume that: (a) the projection of $\mathbb{L}$ on the first column is contained in the set V of nodes of the social graph $G = (V, E)$;[5] (b) the set of all possible actions is denoted $\mathcal{A}$; and (c) time is represented by positive integers. Moreover, we assume that any given user performs any given action at most once. (Hence, if a user bought, for example, the same book twice, we consider only the first purchase.)

There are m service providers, $P_1, \ldots, P_m$, and each P_i, $1 \leq i \leq m$, owns an action log $\mathbb{L}_i$ that consists of the actions performed at the site of that service provider. The unified log is $\mathbb{L} = \bigcup_{i=1}^{m} \mathbb{L}_i$. These m service providers, together with the host, wish to compute the influence strength of each link in the social graph in a privacy-preserving manner.

For each arc $(u, v) \in E$, we define the influence probability $p_{u,v}$, as an estimate to the probability that user v will perform some action if u performs that action. The estimate is based on the past activity of those users as reported in the action logs. To define $p_{u,v}$ (following [2]), we introduce the following notations:

- a_u is the number of tuples in $\mathbb{L}$ in which the first component is u. It equals the number of actions that u did.
- $b_{u,v} = b^h_{u,v}$ is the number of actions α such that $\mathbb{L}$ includes a record (u, α, t) as well as a record (v, α, t'), where $t < t' \leq t + h$, for some integer $h \geq 1$. It represents the number of times in which v followed u in doing some action, assuming a memory window of width h.
- $c^h_{u,v}$ is the number of actions α such that $\mathbb{L}$ includes a record (u, α, t) as well as a record $(v, \alpha, t+h)$. It is the number of times in which v followed u in doing some action exactly h time steps after u performed that action.

One way of defining $p_{u,v}$ is by picking some value of h and then set

$$p_{u,v} = \frac{b_{u,v}}{a_u}\,. \tag{1}$$

Namely, it is the fraction of times in which u succeeded in influencing v (to follow him within h time steps). A more generalized definition would be

$$p_{u,v} = \frac{\sum_{\ell=1}^{h} w_\ell c^\ell_{u,v}}{a_u}\,, \tag{2}$$

where $0 < w_\ell$ and $\sum_{\ell=1}^{h} w_\ell = h$. The definition in equation (1) is a special case of the definition in equation (2) when $w_\ell = 1$ for all $1 \leq \ell \leq h$. By taking $w_1 > \cdots > w_\ell$, one may achieve a temporal decay effect; namely, the faster v follows u in doing an action, the more we consider v's action to be a result of u's influence on him. In either of these definitions, $p_{u,v}$ is set arbitrarily to zero if the denominator a_u is zero (since if u performed no action, we have no way of determining his influence on others). Here we concentrate on the first definition, Eq. (1). The extension of the protocols to the more general definition, Eq. (2), is straightforward and thus omitted.

IV. A secure multi-party protocols for computing the link influence strengths

In this Section we describe the secure multi-party protocol that was offered in [22] (see Protocol 4 there). That protocol is the basis for the rational version which we introduce here in the next section.

The players H (the host) and $P_1, \ldots, P_m$ (the service providers) wish to compute jointly influence probabilities $p_{u,v}$ for all $(u, v) \in E$, as defined in Section III. Their goal is to perform those computations while preserving their private information. The private information of P_i, $1 \leq i \leq m$, is his private log $\mathbb{L}_i$. The private information of H is the arc structure E.

Tassa and Bonchi [22] distinguished between two cases: (1) the exclusive case, where each action is supported exclusively by one of the service providers, and (2) the non-exclusive case, where each action can be supported by more than one service provider. In the first case, all evidence of information propagation that relate to some specific action $\alpha \in \mathcal{A}$ are contained within the action of log $\mathbb{L}_i$ of the service provider P_i that supports that action. For example, if α is the action of purchasing a specific book and that book is offered solely by P_1, then

[4]Undirected graphs will be thought of as directed graph where each undirected edge $\{u, v\}$ is replaced by the two directed arcs (u, v) and (v, u).

[5]In practice, the service providers may have users that are not members of the social network. Such users may be ignored since questions of influence are relevant only in a social framework.

all episodes of influence that relate to that action will be recorded within $\mathbb{L}_1$. In the non-exclusive case, however, if the same action is supported by say P_1 and P_2 (since both are, say, e-book stores that sell the book that the action α represents) then it is possible that Alice bought that book through P_1 while Bob, her Facebook friend, saw her post about it and bought it through P_2. In order to deal with the non-exclusive case, Tassa and Bonchi suggested a secure multi-party protocol (see Protocol 5 there) that can be executed solely by $P_1, \ldots, P_m$, and reduces the non-exclusive case to the exclusive one by modifying the private action logs accordingly. Hence, we concentrate here on the exclusive case; the discussion of the non-exclusive case is deferred to the full version.

As defined in Section III, a_u is the number of actions that user u performed, while $b_{u,v}$ is the number of actions that v performed following u. For each $1 \leq i \leq m$, let a_u^i denote the number of records in $\mathbb{L}_i$ that involve u. Since we assumed that each user may perform any given action at most once, then $a_u = \sum_{i=1}^m a_u^i$. Let $b_{u,v}^i$ denote the number of actions α such that $\mathbb{L}_i$ includes a record (u, α, t) as well as a record (v, α, t'), where $t < t' \leq t+h$. Then, since each action α can appear in only one of the logs (as we focus here on the exclusive case), we have

$$b_{u,v} = \sum_{i=1}^{m} b_{u,v}^i \,. \tag{3}$$

We proceed to describe Protocol 1 (which is Protocol 4 in [22]) that H and $P_1, \ldots, P_m$ run towards learning the link influence probabilities. That protocol is based on two secure arithmetic protocols:

• Protocol Π_S is a secure protocol for computing integer additive shares in the sum of private inputs (see [22, Protocol 2]). Specifically, assume that each of the players $P_1, \ldots, P_m$ holds a private integer x_i where $x_i \in [0, A]$ for some integer A and also $x = \sum_{i=1}^m x_i \in [0, A]$, for some publicly known upper bound A. Let N be an integer where $N \gg A$. Then Π_S ends with P_1 getting a random $s_1 \in [0, N-1]$ and P_2 getting $s_2 := x - s_1$.

• Protocol Π_Q is a secure protocol for computing the quotient of private integers (see [22, Protocol 3]). Assume that P_i has an integer $a_i \in [0, A]$, $i = 1, 2$. Then the protocol, which is executed by P_1, P_2 and H, ends with H getting $q := a_1/a_2$, if $a_2 > 0$, or $q = 0$ if $a_2 = 0$, while P_1 keeps a_1 private (from P_2 and H) and P_2 keeps a_2 private (from P_1 and H). In a nutshell, P_1 and P_2 jointly generate a random real number $M \sim Z$ where Z is the distribution on $[1, \infty)$ with probability density function $f_Z(\mu) = \mu^{-2}$; then they jointly generate a random $r \sim U(0, M)$, and they send to H the values ra_i, $i = 1, 2$, which H proceeds to divide in order to recover $q = a_1/a_2$. (The selection of the probability distribution from which the masking multiplier r is drawn is made in order to minimize the information that H can extract from ra_i on a_i.)

We now turn to describe Protocol 1, which involves the service providers and the host, and enables the latter

Protocol 1 - Secure computation of link influence probability.

Input: H owns the social graph, $G = (V, E)$. P_i, $1 \leq i \leq m$, owns an action log $\mathbb{L}_i$.

Output: H gets $p_{u,v}$ for all $(u, v) \in E$.

1: H generates a set $E' \subset V \times V$ such that $E' \supset E$ and $|E'| \geq c|E|$ for some given constant $c > 1$.
2: H sends E' to P_i, $1 \leq i \leq m$.
3: $P_1, \ldots, P_m$ perform Protocol Π_S *in parallel* for $a_u = \sum_{i=1}^m a_u^i$, for all $u \in V$. At the end of that protocol, P_i, $i = 1, 2$, has s_u^i such that $s_u^1 + s_u^2 = a_u$.
4: $P_1, \ldots, P_m$ perform Protocol Π_S *in parallel* for $b_{u,v} = \sum_{i=1}^m b_{u,v}^i$, for all $(u, v) \in E'$. At the end of that protocol, P_i, $i = 1, 2$, has $s_{u,v}^i$ such that $s_{u,v}^1 + s_{u,v}^2 = b_{u,v}$.
5: For each $u \in V$, P_1 and P_2 jointly generate independent random real numbers $M_u \sim Z$, where Z is as in Protocol Π_Q.
6: For each $u \in V$, P_1 and P_2 jointly generate independent random real numbers $r_u \sim U(0, M_u)$.
7: P_i, $i = 1, 2$, sends $r_u s_u^i$ for all $u \in V$ and $r_u s_{u,v}^i$ for all $(u, v) \in E'$ to H.
8: For each $(u, v) \in E$, H computes $p_{u,v} = \sum_{i=1}^2 r_u s_{u,v}^i / \sum_{i=1}^2 r_u s_u^i$ (where the quotient is set to zero if the denominator is).

to compute the influence probabilities $p_{u,v}$, for all arcs $(u, v) \in E$, according to the definition in Eq. (1). First, H hides his arc set within a larger set of arcs, E', which he sends to $P_1, \ldots, P_m$ (Steps 1-2); the arcs in $E' \setminus E$ are selected uniformly at random from the set of all pairs $(u, v) \notin E$, where $u \neq v$.

Then, the service providers perform Protocol Π_S for a_u and $b_{u,v}$ for all pairs of nodes that appear in the augmented arc set E' (Steps 3-4). Note that Protocol Π_S is designed to compute integer additive shares for a single sum. Here, we perform that protocol in parallel for $n + |\Omega_{E'}|$ sums, namely, all counters a_u and $b_{u,v}$. Note that each of those counters is bounded by $A = |\mathcal{A}|$; the value of $N \gg A$ that is used in Protocol Π_S is chosen jointly by the service providers.

At this stage, P_1 and P_2 hold random additive shares in a_u (the number of actions that u performed in total) and $b_{u,v}$ (the number of times where v followed u) for all users $u \in V$ and all arcs in E'. They now wish to let H compute the quotient $b_{u,v}/a_u$, being a measure of the influence that u has on v, without leaking to him information on a_u and $b_{u,v}$ beyond what is implied by the quotient. To that end, P_1, P_2 and H engage in a variant of Protocol Π_Q, where the multipliers r_u are chosen independently for each user u (Steps 5-8). (It is a variant of Protocol Π_Q since here the masking random value r_u multiplies the shares of the numerator $b_{u,v}$ and the denominator a_u, rather than directly $b_{u,v}$ and a_u).

It is clear that Protocol 1 is correct and complete in the sense that it ends with H having $p_{u,v} = b_{u,v}/a_u$ for all

arcs in E. Indeed,

$$\frac{\sum_{i=1}^{2} r_u s_{u,v}^i}{\sum_{i=1}^{2} r_u s_u^i} = \frac{\sum_{i=1}^{2} s_{u,v}^i}{\sum_{i=1}^{2} s_u^i} = \frac{b_{u,v}}{a_u}\,.$$

A protocol for computing the link influences by Eq. (2) goes along the same lines, since also in that definition the numerator is a sum of private values which each P_i has, $1 \leq i \leq m$. Hence, the only modification in Protocol 1 is in Step 4, where the players invoke Protocol Π_S to compute additive shares in $\sum_{\ell=1}^{h} w_\ell c_{u,v}^\ell$; all other steps remain the same.

V. Extending the secure protocol for computing link influence strengths to the realm of rational players

The problem with Protocol 1 is that all of its output goes to H, while $P_1, \ldots, P_m$ get nothing, even though they participate in the protocol by contributing their private data. While Protocol 1 is secure and does not reveal the private data of the service providers, rational players might condition their participation in such a protocol on receiving part of the computational result. Hence, in order to motivate $P_1, \ldots, P_m$ to participate in the protocol we devise herein a version of it that incentivizes H to share with $P_1, \ldots, P_m$ part of the output that he recovers. The design of this variant incentivizes H by integrating a utility for H of which the expected value is maximized when H participates in the protocol truthfully.

This section is organized as follows. In Section V-A we describe the benefits that the service providers request for themselves if they participate in the protocol. In Section V-B we formalize the utility functions for each of the interacting players. We then describe in Section V-C the modified protocol, and analyze its properties in Section V-D.

A. The requested benefits for the service providers

The output of Protocol 1 is $\mathcal{O} := \{p_{u,v} : (u,v) \in E'\}$ where E' is a superset of the exact arc set E in the social network. (Recall that $|E'| = c|E|$; higher values of c yield better protection of the arc set E, but at the same time they entail higher computational and communication costs.) Assume that each service provider P_i, $1 \leq i \leq m$, is interested in a subset of the nodes (users) $V_i \subset V$ in the graph that H owns. Then the benefit which P_i requests, as an incentive to participate in the protocol, is $\mathcal{O}_i := \{p_{u,v} : (u,v) \in E_i'\}$ where $E_i' := \{(u,v) \in E' : u \in V_i \text{ or } v \in V_i\}$. Namely, P_i is interested in the strength of all links in E' that involve at least one user in V_i. The sets V_i, $1 \leq i \leq m$, need not be disjoint. However, $|V_i|$ should be proportional to the publicly known size of the business that P_i is running; namely, large businesses that contribute much larger activity logs to the process of learning the link influence probabilities should be rewarded with larger portions of the output.

B. Utility functions of the interacting players

We assume that the host H is a rational selfish player; hence, as we would like to incentivize him to share with $P_1, \ldots, P_m$ their requested part of the input, we define H's utility in the following way: Let $c_{u,v}^j$, $nc_{u,v}^j$ be parameters denoting whether player $j \in \{H, 1, \ldots, m\}$ learned the link strength estimate $p_{u,v}$ of some true arc $(u,v) \in E$ correctly or incorrectly, respectively. Let

$$U_H : \{c_{u,v}^H, nc_{u,v}^H, c_{u,v}^1, nc_{u,v}^1, \ldots c_{u,v}^m, nc_{u,v}^m\} \to \mathbb{R}$$

be a utility function for H (not necessarily positive). Let Γ denote an execution of the protocol. The outcome of Γ can be described by the following set of binary variables:

$$\gamma_{u,v}^H\,, \quad (u,v) \in E\,,$$

and

$$\gamma_{u,v}^i\,, \quad (u,v) \in E_i\,, \quad 1 \leq i \leq m\,,$$

where $\gamma_{u,v}^H \in \{c_{u,v}^H, nc_{u,v}^H\}$ indicates whether H learned the strength $p_{u,v}$ of the arc $(u,v) \in E$ or not, while $\gamma_{u,v}^i \in \{c_{u,v}^i, nc_{u,v}^i\}$ indicates whether P_i learned the strength $p_{u,v}$ of the arc $(u,v) \in E_i := E_i' \cap E$ correctly or not, $1 \leq i \leq m$. Then the overall utility for H from Γ is

$$U_H^\Gamma := \sum_{(u,v)\in E} U_H(\gamma_{u,v}^H) + \sum_{i=1}^{m} \sum_{(u,v)\in E_i} U_H(\gamma_{u,v}^i)\,. \quad (4)$$

On the one hand, if H participates truthfully in the protocol then all players learn all arcs' strengths that they were expected to learn, and so the resulting utility is

$$U_H^c := \sum_{(u,v)\in E} U_H(c_{u,v}^H) + \sum_{i=1}^{m} \sum_{(u,v)\in E_i} U_H(c_{u,v}^i)\,. \quad (5)$$

On the other hand, if H dose not participate truthfully in the protocol and cheats on a subset E^d of d arcs in E then there are two possible scenarios:
(1) H is caught cheating and, consequently, the protocol aborts prematurely and then no player learns anything; in this scenario the resulting utility is

$$U_H^{nc} := \sum_{(u,v)\in E} U_H(nc_{u,v}^H) + \sum_{i=1}^{m} \sum_{(u,v)\in E_i} U_H(nc_{u,v}^i)\,. \quad (6)$$

(2) H is not caught cheating and, consequently, H learns all arcs' strengths while the service providers learn their designated output except for the strength of arcs in E^d; the resulting utility in this case is

$$U_H^o(E^d) := \sum_{(u,v)\in E} U_H(c_{u,v}^H) + \sum_{i=1}^{m} U_{H:P_i} \quad (7)$$

where

$$U_{H:P_i} = \sum_{(u,v)\in E_i \setminus E^d} U_H(c_{u,v}^i) + \sum_{(u,v)\in E_i \cap E^d} U_H(nc_{u,v}^i)\,.$$

Note that H's utility from a successful cheat may depend on the service provider who was mislead and on the arcs in the subset E^d.

We assume herein that the utility function U_H satisfies the following condition for all $(u,v) \in E$:

$$\textbf{[CH1]:} \quad U_H(c^H_{u,v}) + \sum_{i:(u,v)\in E_i} U_H(c^i_{u,v}) > U_H(nc^H_{u,v}) + \sum_{i:(u,v)\in E_i} U_H(nc^i_{u,v})\,.$$

Namely, for every arc in E, H prefers learning its arc strength, and at the same time having all interested service providers learn it too, over the opposite option in which no player (neither him, nor any service provider) learns that value. Summing up inequality **[CH1]** over all $(u,v) \in E$ we infer that

$$U^c_H > U^{nc}_H\,; \tag{8}$$

i.e., H prefers to act truthfully over being caught cheating (a case that ends with aborting the protocol). We also assume herein that the utility function U_H satisfies the following condition for all $1 \le i \le m$ and all $(u,v) \in E_i$:

$$\textbf{[CH2]:} \quad U_H(nc^i_{u,v}) > U_H(c^i_{u,v})\,.$$

Namely, for every arc in E H prefers P_i not learning the arc's strength over learning it. (Note that if there are arcs in E for which the opposite of **[CH2]** holds then it is an uninteresting case in the current context since H would simply not attempt cheating regarding those arcs' strengths.) Finally, inequality **[CH2]** implies that

$$U^o_H(E^d) > U^c_H\,. \tag{9}$$

As for the service providers, each P_j, $1 \le j \le m$, can either participate in the protocol or not. If he does, and the protocol ends successfully, then all involved players H and $P_1,\ldots,P_m$ will learn their designated output. However, if P_j stays out, then he will not learn $\mathcal{O}_j$ while all the others will learn (again, in the case of a successful termination of the protocol) an approximate output $\tilde{\mathcal{O}}_i$, for $i \in \{H,1,\ldots,m\}\setminus\{j\}$, which is the result of computing the arcs' strengths without P_j providing his inputs. If P_j is a large business then the errors in $\tilde{\mathcal{O}}_i$ in comparison to $\mathcal{O}_i$ are more significant than in the case where P_j is a smaller business. Our assumption on P_j's utility is as follows:

$$\textbf{[CP]} \sum_{(u,v)\in E'} U_{P_j}(c^H_{u,v}) + \sum_{i=1}^{m} \sum_{(u,v)\in E'_i} U_{P_j}(c^i_{u,v}) > \sum_{(u,v)\in E'} U_{P_j}(\tilde{c}^H_{u,v}) + \sum_{(u,v)\in E'_j} U_{P_j}(nc^j_{u,v}) + \sum_{i\in\{1,\ldots,m\}\setminus\{j\}} \sum_{(u,v)\in E'_i} U_{P_j}(\tilde{c}^i_{u,v})\,,$$

where $c^i_{u,v}$ (resp. $\tilde{c}^i_{u,v}$) indicates that player $i \in \{H,1,\ldots,m\}$ learned (u,v)'s strength correctly (resp. approximately). Hence, condition **[CP]** states that P_j prefers collaborating over not collaborating.

We note that conditions **[CH1]** as well as **[CP]** depend on the size of the sets V_i. On one hand, the host H may decide to refrain from participating in the protocol if the sets V_i are too large (say, if $|\bigcup_{i=1}^m V_i| > |V|/2$); in such cases condition **[CH1]** will no longer be true, since in such cases H would prefer that no one (including himself) learns anything. On the other hand, the service providers may choose not to participate if the sets V_i are too small (say, if $V_i = \emptyset$). H and $P_1,\ldots,P_m$ are therefore expected to engage in a negotiation for determining the size of the sets V_i which will make condition **[CH1]** hold and will also make condition **[CP]** hold for a sufficiently large subset of service providers.

C. The modified protocol

In order to turn Protocol 1 into a protocol that incentivizes H to share with P_i, $1 \le i \le m$, his requested part of the output, we suggest to perform Protocol 1 several times (so called "rounds") with an additional player T that acts as a coordinator. In each round, the coordinator will toss a coin that issues the result "fake" with probability $1-\varphi$ and "true" with probability φ, for some preset parameter $\varphi \in (0,1)$. In fake rounds, all computations will yield meaningless random results for the link influence probabilities. Only in the true round, which will be the final one, the results will be correct.

The preliminary stage.

Before running the sequence of rounds, the parties perform the following steps:

1) H and $P_1,\ldots,P_m$ decide on a random labeling of the users in V which they keep secret from T.
2) H generates a set $E' \subset V \times V$ such that $E' \supset E$ and $|E'| = c|E|$ for some given constant c.
3) H sends to $P_1,\ldots,P_m$ the set E'.
4) P_i computes $E'_i := \{(u,v) \in E' : u \in V_i \text{ or } v \in V_i\}$, $1 \le i \le m$.

The procedure in each round: Part 1. In each round of the modified protocol (MP) the players perform the following steps:

- Step MP1: The players perform Steps 3-4 of Protocol 1. After doing that, the two service providers P_i, $i = 1,2$, hold additive shares in all counters a_u and $b_{u,v}$ (denoted, respectively, by s^i_u and $s^i_{u,v}$).
- Step MP2: The coordinator T tosses a $(\varphi, 1-\varphi)$-coin to determine whether the current round is true or fake. (We defer to a later stage the discussion of how to choose φ.)
- Step MP3: For each $u \in V$, T generates independent random real numbers $M_u \sim Z$, where Z is as in Protocol Π_Q.
- Step MP4: For each $u \in V$, T generates independent random real numbers $r^{i,\mathrm{n}}_u, r^{i,\mathrm{d}}_u \sim U(0,M_u)$, $i = 1,2$, and uniformly random signs $\sigma^{\mathrm{n}}_u, \sigma^{\mathrm{d}}_u \in \{-1,1\}$.
- Step MP5: T sends $\{r^{i,\mathrm{n}}_u, r^{i,\mathrm{d}}_u, \sigma^{\mathrm{n}}_u, \sigma^{\mathrm{d}}_u : u \in V\}$ to P_i, $i = 1,2$.
- Step MP6: P_i, $i = 1,2$, sends to H the two sets of values as follows: $\{\sigma^{\mathrm{d}}_u r^{i,\mathrm{d}}_u s^i_u : u \in V\}$ and $\{\sigma^{\mathrm{n}}_u r^{i,\mathrm{n}}_u s^i_{u,v} : (u,v) \in E'\}$.

- Step MP7: For each $(u,v) \in E$, H computes

$$p_{u,v} = \frac{\sum_{i=1}^{2} \sigma_u^{\mathrm{n}} r_u^{i,\mathrm{n}} s_{u,v}^i}{\sum_{i=1}^{2} \sigma_u^{\mathrm{d}} r_u^{i,\mathrm{d}} s_u^i} \tag{10}$$

  (where the quotient is set to zero if the denominator is zero).

The main idea is that in fake rounds, for any given $u \in V$, T will generate in Step MP4 the four multipliers $\{r_u^{i,\mathrm{n}}, r_u^{i,\mathrm{d}} : 1 \leq i \leq 2\}$ independently. However, in the true round he will select only $r_u^{1,\mathrm{n}}, r_u^{1,\mathrm{d}} \sim U(0, M_u)$ and then will set $r_u^{2,\mathrm{n}} = r_u^{1,\mathrm{n}}$ and $r_u^{2,\mathrm{d}} = r_u^{1,\mathrm{d}}$. As a result, in fake rounds, the values $p_{u,v}$ which H computes in Step MP7 will be totally unrelated to the true link influence probabilities

$$\hat{p}_{u,v} := b_{u,v}/a_u \,. \tag{11}$$

On the other hand, in the true round we shall have

$$p_{u,v} = \alpha_u \hat{p}_{u,v} \,, \quad \alpha_u := \frac{\sigma_u^{\mathrm{n}} r_u^{1,\mathrm{n}}}{\sigma_u^{\mathrm{d}} r_u^{1,\mathrm{d}}} \,. \tag{12}$$

Two comments are in order:

(1) The usage of the sign multipliers $\sigma_u^{\mathrm{n}}, \sigma_u^{\mathrm{d}}$ is necessary since, without them, the fraction that H computes in the true round, (12), would be always nonnegative, while the fractions in fake rounds could be negative. (Recall that some of the additive shares s_u^i and $s_{u,v}^i$, $i = 1, 2$, could be negative.) Hence, without using the sign multipliers, H could have inferred that a round in which $p_{u,v} \geq 0$ for all $(u,v) \in E$ is the true round with almost certainty.

(2) In fake rounds, the numbers that H holds at this stage are totally meaningless, since $r_u^{2,\mathrm{n}}$ is independent of $r_u^{1,\mathrm{n}}$ and, likewise, $r_u^{2,\mathrm{d}}$ is independent of $r_u^{1,\mathrm{d}}$. In the true round, however, those numbers reveal the relations between the probabilities over arcs that emerge from the same node u, since they are all multiplied by the same factor α_u. But they do not reveal such relations globally, since the factors α_u are chosen randomly and independently for each node. Moreover, H is being told that the current round is the true round only when it was verified that he respected his part in the agreement (as we discuss below) and it is possible to allow him to compute the correct results.

The procedure in each round: Part 2.

We now proceed to describe the next stage of the protocol in which H exchanges with $P_1, \ldots, P_m$ the required information for completing the computation. To this end, we first elaborate on three ingredients in that part of the protocol:

Encryption. In the revised and enhanced protocol that we describe below, we assume that each service provider P_i, $1 \leq i \leq m$, has a probabilistic[6] public key encryption function $\mathcal{E}_i(\cdot)$ for which the domain and the range are some large finite field F_i. Everyone can compute $\mathcal{E}_i(\cdot)$, since the encryption key is publicly known. The decryption function on the other hand, $\mathcal{E}_i^{-1}(\cdot)$, depends on a private key and, hence, can be computed only by P_i.

Encoding real numbers as finite field elements. We shall need hereinafter to encode real numbers as finite field elements in order to apply on them discrete circuit computations. Let us assume that all real numbers are taken from the range $[-M, M]$. Let us also assume that our tolerance for rounding effects is, say, 2^{-d}. Then if q is a prime larger than $2^{d+1}M$, we can encode such real numbers as follows:

$$y \in [0, M] \mapsto \langle y \rangle = \lfloor 2^d \cdot y \rfloor \,,$$

and

$$y \in [-M, 0) \mapsto \langle y \rangle = \lfloor q + 2^d \cdot y \rfloor \,.$$

y can be recovered from $\langle y \rangle$, up to a rounding error. Specifically, if $\langle y \rangle < q/2$ then it is clear that $y \geq 0$ and then $z := 2^{-d}\langle y \rangle$ is an approximate recovery of the original real value y. Specifically, $z = y + \varepsilon$ where $-2^{-d} < \varepsilon \leq 0$. (The recovery of a negative y is similar.)

Encoding the subgraphs. The host H, upon computing the resulting graph with the corresponding values of $p_{u,v}$, can construct for each P_i a vector over the encryption field F_i with the strength of all arcs in E_i'. Such a vector will consist of $|E_i'|$ triplets of the form $(u, v, \ell_{u,v})$ where (u,v) is an arc in E_i' and $\ell_{u,v} := \langle p_{u,v} \rangle$ is a representation in F_i of the real fraction $p_{u,v}$ that H computed for that arc. Consequently, we can encrypt such a vector representation of the graph by applying $\mathcal{E}_i$ on each of its components, independently. We denote hereinafter that vector of information for P_i by g_i and its encryption by $\mathcal{E}_i(g_i)$.

Now we are ready to describe the final steps in the protocol:

- Step MP8: H sends to T the vectors $\mathcal{E}_i(g_i)$, $1 \leq i \leq m$; we assume that the arcs appear in g_i in a random order.
- Step MP9: T and $P_1, \ldots, P_m$ perform a statistical check on the received vectors to see that H did not cheat. (See details below). If he did, the protocol aborts.
- Step MP10: T sends to P_i the vector $\mathcal{E}_i(g_i)$, $1 \leq i \leq m$.
- Step MP11: P_i, $1 \leq i \leq m$, decrypts the received vectors and recovers g_i. He checks that all arcs in E_i' appear in g_i. If not, the protocol aborts.
- Step MP12-f: In fake rounds T stops at this stage and starts a new round.
- Step MP12-t: In the true round, T sends to both H and the service providers $P_1, \ldots, P_m$ all the multipliers $\{\alpha_u : u \in V\}$. Consequently, H recovers $\mathcal{O} := \{p_{u,v} : (u,v) \in E\}$ while P_i recovers $\mathcal{O}_i := \{p_{u,v} : (u,v) \in E_i'\}$, $1 \leq i \leq m$.

It remains to discuss the verification in Step MP9:

- T selects a small and random subset of triplets from $\mathcal{E}_i(g_i)$ and sends them to P_i, $1 \leq i \leq m$.

[6] A probabilistic encryption function is a randomized encryption function; in such ciphers, if the same plaintext is encrypted several times, the resulting ciphertexts are typically different.

- P_i decrypts the obtained triplets; as a result, he gets a list of triplets of the form $(u, v, \ell_{u,v})$.
- P_i sends the obtained triplets to T and he also informs P_1 and P_2 on the pairs (u, v) in those triplets. (Note that P_i can be one of P_1, P_2.)
- P_i, $i = 1, 2$, sends to T the shares $s^i_{u,v}$ and s^i_u for all those pairs.
- T computes $p_{u,v}$ as in Step MP7 above and sees if it agrees with the corresponding $\ell_{u,v}$.
- If all equality verification passed successfully, the overall verification procedure ends successfully. Otherwise, it fails.

Note that in the above verification procedure T learns a_u and $b_{u,v}$ for some pairs (u, v). However, due to the secret random labeling of the users that H and $P_1, \ldots, P_m$ generated upfront, T does not know who are the users behind the labels u, v.

D. Properties of the protocol

We conclude the subsection by claiming the properties of our modified protocol. Let $w := \sum_{i=1}^{m} |E'_i|$ denote the number of triplet-entries in the vectors g_i, $1 \leq i \leq m$. In view of assumption **[CH2]**, H may wish to prevent some of the service providers from learning the strength of some true arcs. Assume that, in a given round, H cheats by reporting wrong values for d of those entries, while in the verification stage T randomly selects t of those entries for verification. Then the probability of H to successfully pass the verification stage in that round is

$$\psi(d) := \frac{\binom{w-d}{t}}{\binom{w}{t}} . \tag{13}$$

Next, we proceed to determine an upper bound on φ, the probability of selecting a true round, that will guarantee that a rational host H would behave truthfully. To that end, recall that U^c_H and U^{nc}_H, Eqs. (5) and (6), denote the utilities for H in case of a truthful behavior or in case of an untruthful behavior that ended with aborting the protocol, respectively, while $U^o_H(E^d)$, Eq. (7), denotes the utility for H in case of an untruthful behavior regarding the d entries associated with the arcs in the subset $E^d \subset E$.

Claim 5.1: If

$$\varphi \leq \min_{1 \leq d \leq w, E^d \subset E} \frac{U^c_H - (1 - \psi(d))U^{nc}_H}{\psi(d) \cdot U^o_H(E^d)} , \tag{14}$$

then H is truthful in our modified protocol and the service providers are incentivized to participate.

A proof outline is given in the appendix. The expression to be minimized on the right hand side of Eq. (14) equals

$$F(t; d) := \frac{U^{nc}_H + (U^c_H - U^{nc}_H)/\psi(d)}{U^o_H(E^d)} . \tag{15}$$

$F(t; d)$ is a quotient between two values that increase with d; indeed, the numerator increases with d as implied by Eqs. (8) and (13), while the denominator increases with d as implied by assumption **[CH2]**. In order to achieve a minimal expected number of rounds, the coordinator T should compute, for any fixed value of t, the minimum $\min_{1 \leq d \leq w, E^d \subset E} F(t; d)$, and then set t so that this minimum is maximized.

VI. Conclusions

In this paper we extended a viral marketing model and protocol that were presented in [22] to include a host and service providers that are rational players. Namely, they will cooperate and engage in a computational protocol only if it is in their best interest; otherwise they will not. We integrated game theoretic principles – punishment strategy and uncertainty regarding the number of the protocol's rounds, in order to establish truthful behavior among the protocol participants – the host as well as the service providers.

The current modified protocol (as well as the original protocols in [22]) is not immune against coalitions; indeed, P_1 and P_2 may collude in order to recover the values of all counters. While our assumption of rationality in the modified protocol as well as the semi-honest assumption in the original protocol excludes the possibility of coalitions, in future work we intend to enhance our protocols to be immune against coalitions of service providers of bounded sizes.

References

[1] B. Luicer A. Borodin, M. Braverman and J. Oren. Strategyproof mechanisms for competitive influence in networks. In *WWW*, 2013.

[2] F. Bonchi A. Goyal and L. V. S. Lakshmanan. Learning influence probabilities in social networks. In *WSDM*, 2010.

[3] F. Bonchi A. Goyal and L. V. S. Lakshmanan. A data-based approach to social influence maximization. *PVLDB*, 5:73–84, 2011.

[4] J. M. Kleinberg D. Kempe and É. Tardos. Maximizing the spread of influence through a social network. In *KDD*, 2003.

[5] P. Domingos and M. Richardson. Mining the network value of customers. In *KDD*, 2011.

[6] S.D. Gordon and J. Katz. Rational secret sharing, revisited. In *Security and Cryptography for Networks (SCN)*, 2006.

[7] J. Halpern and V. Teague. Rational secret sharing and multiparty computation. In *STOC*, 2004.

[8] R. Gonen I. Abraham, D. Dolev and J. Halpern. Distributed computing meets game theory: Robust mechanisms for rational secret sharing and multiparty computation. In *PODC*, 2006.

[9] C. Wang J. Tang, J. Sun and Z. Yang. Social influence analysis in large-scale networks. In *KDD*, 2009.

[10] W. Jiang and C. Clifton. A secure distributed framework for achieving k-anonymity. *The VLDB Journal*, 15:316–333, 2006.

[11] R. Nakano K. Saito and M. Kimura. Prediction of information diffusion probabilities for independent cascade model. In *KES*, 2008.

[12] M. Kantarcioglu and C. Clifton. Privacy-preserving distributed mining of association rules on horizontally partitioned data. *TKDE*, 16:1026–1037, 2004.

[13] G. Kol and M. Naor. Cryptography and game theory: Designing protocols for exchanging information. In *TCC*, 2008.

[14] Y. Lindell and B. Pinkas. Privacy preserving data mining. In *Crypto*, 2000.

[15] A. Lysyanskaya and N. Triandopoulos. Rationality and adversarial behaviour in multi-party computation. In *Crypto*, 2006.

[16] C. Castillo A. Gionis M. Mathioudakis, F. Bonchi and A. Ukkonen. In *KDD*, 2011.

[17] M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *KDD*, 2002.

[18] Z. Yang S. Zhong and R. Wright. Privacy-enhancing k-anonymization of customer data. In *PODS*, 2005.

[19] Y. Shoham and M. Tennenholtz. Non-cooperative computation: Boolean functions with correctness and exclusivity. *Theoretical Comp. Sci.*, 343:97–113, 2005.

[20] T. Tassa. Secure mining of association rules in horizontally distributed databases. *TKDE*, 26:970–983, 2014.

[21] T. Tassa and E. Gudes. Secure distributed computation of anonymized views of shared databases. *Transactions on Database Systems*, 37, Article 11, 2012.

[22] Tamir Tassa and Francesco Bonchi. Privacy preserving estimation of social influence. In *EDBT*, 2014.

[23] Tamir Tassa and Dror J. Cohen. Anonymization of centralized and distributed social networks by sequential clustering. *IEEE Trans. Knowl. Data Eng.*, 25:311–324, 2013.

[24] J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *KDD*, 2002.

[25] C. Wang W. Chen and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *KDD*, 2010.

[26] Y. Wang W. Chen and S. Yang. Efficient influence maximization in social networks. In *KDD*, 2009.

[27] A. Goyal W. Lu, F. Bonchi and L. V. S. Lakshmanan. The bang for the buck: Fair competitive viral marketing from the host perspective. In *KDD*, 2013.

[28] C. Clifton X. Lin and M. Zhu. Privacy-preserving clustering with distributed EM mixture modeling. *Knowl. Inf. Syst.*, 8:68–81, 2005.

[29] A. Yao. Protocols for secure computation. In *FOCS*, 1982.