Trust and Privacy Considerations in Participant Selection for Social Participatory Sensing

Haleh Amintoosi Salil S. Kanhere Mohammad Allahbakhsh

University of New South Wales, Australia
{haleha,salilk,mallahbakhsh}@cse.unsw.edu.au

Technical Report
UNSW-CSE-TR-201409
March 2014

THE UNIVERSITY OF
NEW SOUTH WALES

School of Computer Science and Engineering
The University of New South Wales
Sydney 2052, Australia

Abstract

The main idea behind social participatory sensing is to leverage social friends to participate in mobile sensing tasks. A main challenge, however, is the identification and recruitment of sufficient number of well-suited participants. This becomes especially more challenging for large-scale online social networks where the network topology and friendship relations are not known to the applications. Moreover, the potential sparseness of the friendship network may result in insufficient participation, thus reducing the validity of the obtained information. In this paper, we propose a participant selection framework which aims to address the aforementioned limitations. The framework has two main modules. The nomination module makes use of a customized random surfer to crawl the social graph and identify suitable nominees among the requester's friends and friends-of-friends. The nominee selection is determined as a function of the suitability score of members and pairwise trust perception among members. The selection module is responsible for selecting the required participants from the set of nominees. The selection is done based on the nominee's timeliness, the number of participants selected so far and the task's remaining time. Moreover, to prevent any possible collusion, a further check is performed to determine whether the selection of a new participant may result in the formation of a colluding group among the selected participants. Simulation results demonstrate the efficacy of our proposed participant selection framework in terms of selecting a large number of reputable participants with high suitability scores, in comparison with state-of-the-art methods.

1 Introduction

Web 2.0-enabled applications harness the wisdom of crowds and human intelligence to collaborate and accomplish a wide variety of tasks such as creating content, performing tasks, etc. [40, 17]. The recent improvements in mobile phone technology and sensing capabilities (such as microphone, camera, accelerometer, GPS, etc) in particular, has led to the emergence of a new exciting paradigm known as participatory sensing [13]. In participatory sensing, the key idea is to recruit ordinary people to crowdsource data from their mobile phones [25].

The integration of participatory sensing systems with online social networks has resulted in the emergence of social participatory sensing [28]. In a typical social participatory sensing system, social network members can act as service requesters and utilize social friends and friends-of-friends as participants to contribute to their tasks. A pertinent example of such a system is Jelly [1] which is built on top of existing social networks like Facebook [2] and Twitter [3]. When the users encounter something unusual, they can take a picture of the object, formulate a query and submit it to their social network. Another instantiation of the concept of social participatory sensing is found in [16] where Twitter is used as the underlying social network substrate. The authors proposed two mobile applications: (i) a weather radar application in which, Twitter members send tweets indicating the weather condition and (ii) a noise-mapping application where members gather sound samples via their mobile phones and contribute the noise level via Twitter.

In order to obtain trustworthy contributions in a social participatory sensing system, well-suited participants should be identified and selected via trustable paths. Moreover, proper quality control techniques should be put in place [4] to determine the trustworthiness of contributions. In this paper we focus on participant selection issue, which entails evaluating the participant's suitability and selecting the well-suited participants.

One of the main challenges in the success of social participatory networks is identifying and recruiting sufficient number of well-suited participants. Typically, there is no explicit incentive for participation and people contribute altruistically. In the absence of adequate contributors, there is a danger that the application will fail to gather meaningful data. Another challenge, particularly for tasks which require domain-specific knowledge (e.g., taking photos of rare plant species), is the suitability of the participants to collect appropriate data [43]. A well-suited participant might be a reputable expert user, who possesses extensive knowledge on a range of topics or has perhaps garnered reputation for contributing successfully to prior tasks. It is logical that contributions produced by suitable participants should be trusted more than those prepared by others. On the other hand, some participants might be biased or malicious. They may even build collusive groups in order to maximize their unfair benefits by supporting the requesters of their own interest. So, it is desirable to identify the colluders in order to prevent their recruitment. To sum up, identifying and recruiting sufficient reputable and well-suited participants is essential for the reliability of the task outcome.

Existing solutions for identifying and recruitment of participants in social participatory networks such as [20, 3, 53] mostly suffer from several limitations. First, they are based on the assumption that the social graph topology and the social links between users are known to the system. This may be true in cases where the requester is utilizing an organizational social network or a small social network (such as groups of

[1] http://blog.jelly.co/post/72563498393/introducing-jelly
[2] http://facebook.com
[3] http://twitter.com

acquaintances, co-workers, alumnae, etc.). In such instances, it is possible to have a global view of the system to have access to the network statistics and utilize this knowledge to identify and recruit the suitable participants. This assumption, however, is not realistic in large-scale social networks such as Facebook or Twitter with hundreds of millions of users. As such, it may not be possible for third-parties and applications to have access to the social network information such as users' profiles, histories and relations.

The second limitation is the potential sparseness of a requester's friendship graph. The requester may have few friends or may lack close friends who may be willing to contribute to tasks initiated by the requester. It has been shown that online social networks can be considered as sparse [49]. For example in Yahoo! Pulse [4] which is an online social network involving hundreds of millions of users, almost half of the users only have one friend connection [52]. In such cases, identifying and recruiting sufficient number of participants is even more challenging.

The third limitation is the inefficiency of existing recruitment techniques and their vulnerability against malicious activities. Generally, the participant recruitment is done through following three methods: (i) the open call (ii) the auction and (iii) pre-selection [48, 50, 45]. The open call method broadcasts the tasks to all community members and allows all volunteers to contribute. This method is typically used when there is no ground truth available for data quality assessment and hence, the only quality indicator is the community consensus. So, open call method is not suitable for the tasks with pre-defined quality requirements. In the auction method, an open call is sent to all members without any participant pre-selection. After expiry of submission time, the requester assesses the quality of the submitted contributions and may choose one or a few number of contributions as the winner. The limitation of this method is that the responsibility of assessing the contributions' quality and fidelity is on the requester, which may prove to be an exhausting task [14]. The pre-selection method, as its name implies, broadcasts the task to the selected participants instead of all community members. The selection is based on choosing the participants who mostly satisfy the requirements of the tasks. The main problem with this recruitment method (which is also relevant for the other two methods) is its vulnerability to collusion. A group of malicious participants might form a colluding group so that they are recruited in preference to other potentially high-quality workers. The colluding group would then have the power to sway the outcome of the task in accordance with their agenda (More details about the existing recruitment methods are presented in Section 2).

In this paper, we propose a participant recruitment framework for social participatory sensing network. Contrary to our previous work [10, 11, 9, 8], we assume that the social graph topology is not known to our framework. The main implication of this assumption is that there is no prior access to the profile information of social network members, and hence, the identification and selection of suitable participants must be done on-the-fly while the social graph is being traversed. We also assume that the graph search is not uniform and not limited to a specified depth. Consequently, the graph search may progress deeper in some friendship chains and shallower in others, depending on the social trusts along the chain. We believe that making these assumptions will make our proposed framework more practical and closer to the reality.

The proposed framework consists of two main modules, which aim at addressing the key challenges raised above. The first module, known as the *nomination module*, is responsible for identifying well-suited members (known as *nominees*) in the

[4] pulse.yahoo.com

requester's social graph and inviting them to contribute. We define the nominees as those who (i) are suitable to contribute, and (ii) there exists a trustworthy path to them (starting from the requester). We argue that in order to have a comprehensive view of the participant's suitability, the following parameters should be taken into account: (i) the participant's expertise (in order to satisfy the task requirements), (ii) his reputation score (as an indication of being a highly trustable participant), (iii) the pairwise privacy score between the requester and participant (to minimize the privacy breach of requester's sensitive information), (iv) the requester's list of preferred participants (to give priority to those who are preferred by the requester to be recruited), and (v) the requester's blocked list (those with whom, the requester is reluctant to contribute (more details in Section 4.1). As mentioned above, we assume that the structure of the entire social network is not accessible. Hence, the nomination module relies on a customized random surfer to crawl the social network graph (originating at the requester) and identify well-suited nominees. The number of initiated random surfers for a requester is equal to the number of his friends. Each random surfer starts from one of the requester's friends and chooses its next visited nodes based on the suitability score of the nodes as well as the pairwise trusts along the path. Once the nominees are identified, they are invited to attend in the task.

The *selection module*, the second module of our recruitment framework, is responsible for selecting the final participants from the set of nominees who have accepted the invitation. In this module, we propose a time-aware and collusion-free recruitment method which is an extension to the pre-selection recruitment approach and aims at addressing the collusion issue. The selection module takes into account a set of parameters and decides whether to select the nominee. These parameters are (i) the selection score (i.e., the ratio of participants selected so far to the total number of required participants), (ii) the remaining time to the task deadline and (iii) the timeliness of the participant in previous tasks. The intuition behind considering these parameters is that when the task's remaining time is short, it is logical to select the nominee that has shown timely behaviour in his past contributions, as he is most likely to submit his contribution before the imminent deadline. If eligible to be selected, a final check is done to insure that the selection of this participant will not result in potential collusion. In particular, we aim to identify whether the addition of each new participant to the previously selected group will result in the formation of a group of colluders. Several indicators are used in literature to detect collusive behaviours [6, 39, 36]. We use the (i) group size (i.e., number of colluders), (ii) group support count (i.e., number of tasks in which colluders have collaborated in the past), and (iii) group time window (i.e., the time difference between the latest and earliest contribution of group members) as the indicators. The intuition behind considering these parameters is that the colluders usually make a group which is large enough to make a considerable impact. Moreover, group members usually target considerable number of tasks and collaborate together in contributing to these tasks. The colluders also desire to contribute during a short time window (groups working over a long time window are unlikely to have worked together). By considering all these indicators, we determine the collusion probability for each eligible participant by utilizing the well-known Frequent Itemset Mining technique [22]. More details about the collusion indicators will be presented in Section 4.2.

In summary, the main contributions of the paper are as follows:

- We propose a suitability assessment technique to compute the suitability score of a participant to contribute to a given task. In order to do so, the suitability assessment technique takes into account the participant's expertise, his reputa-

tion score, the pairwise privacy score between the requester and participant, the requester's list of preferred participants, and the requester's blocked list (more details in Section 4.1).

- To identify the suitable participants, we propose a customized random surfer inspired by the idea of random walks and Markov chain. Our proposed random surfer crawls the requester's social graph and identifies and invites the nominees via trustworthy paths. The proposed random surfer while provides the system with a set of suitable nominees, aims at addressing the bootstrapping problem for the newcomers (who have recently joined the network) by giving them the chance of being nominated in competition with more reputable participants (more details in Section 4.1).

- We propose a time-aware collusion-free selection module which is responsible for selecting the participants from the set of invited nominees who are willing to contribute to the task. The selection module considers the participant's timeliness, the remaining time to the task deadline and the selection score to decide whether to select the nominee as a final participant or not. It also calculates a collusion probability for each eligible participant to prevent any possible collusion on the task (more details in Section 4.2).

- The accuracy and usability of the proposed techniques has been tested using real world datasets from the Advogato social network and Wikipedia Adminship Election and simulated experiments. The evaluation results show superiority of our method over the other common recruitment methods.

The remainder of the paper is organized as follows. The related research is investigated in Section 2. In Section 3, we define the preliminaries and basic concepts that are used throughout the paper. Then, in Section 4, the proposed framework is explained in detail. Section 5 discusses the evaluation scenarios and results. Finally, Section 6 concludes the paper and presents future directions.

2 Related Work

Social participatory networks can be regarded as a subset of collective intelligence systems, which are defined broadly as groups of individuals doing things collectively that seem intelligent [38]. Due to the openness of such systems, the recruitment of sufficient well-suited participants has always been a great concern [25]. In the following, we will have a short review on the related works on this issue and will discuss the state-of-the-art.

2.1 Recruitment Issues in Online Communities

One of the important parameters in obtaining high quality contributions is the effectiveness of the methods utilized for participant selection. It is evident that the volume and the diversity of participants with different perspectives and knowledge can lead to accurate trustworthy contributions.

In most online recruitment systems such as CrowdFlower, Wikipedia, etc., the recruitment process includes a nomination step in which, nominees are selected amongst

the crowd based on a set of criteria. Sometimes, there is no explicit nomination. In such systems, the requester advertises the task to the crowd (similar to writing on Facebook wall) and everyone inside the system is able to contribute. Example is Amazon Mechanical Turk (Murk), which by default, outsources a task to everyone within the system in the form of an open call.

This open-call method, however, may lead to poor fidelity contributions, since it enables anyone, even those poorly equipped to fulfil the task, to contribute. Sometimes the requester may wish to specify that workers possess certain attributes in order to complete the task. For example, participant selection in MTurk can be restricted to residents of a specific country, or to workers who have completed more than a certain number of tasks with a specified rate of accuracy. Thus, smaller bespoke crowds can be assembled out of the workforce to complete highly specialized tasks [14].

On the other hand, sometimes, the task assignment is done on the basis of a publish/subscribe service, i.e., the participant (subscriber) shares his interests and preferences about a topic by subscribing to the server and a requester (publisher) posts and forwards messages to the interested users only [16, 15]. The main challenge in the above nomination methods is the lack of sufficient qualified participants to attend in the tasks that need specific knowledge or expertise [14, 18].

In social participatory sensing, where the social relationships are utilized to find and nominate the eligible participants, *random walk* [46] can be used for participant nomination. Given a social graph and a starting point as the requester, we select a friend of it at random, and move to this friend; then we select a friend of this point at random, and move to it etc. The (random) sequence of points selected this way is a random walk on the graph. The concept of random walk originates from graph theory [37], and has a wide range of applications. One of the important examples is the link prediction algorithms. Given a large network, say Facebook, at time t, for each user, link prediction algorithm is aimed at predicting what new edges (friendships) that user will create between t and some future time t_1 [12]. Similarly, link recommendation algorithms aim to suggest to each user a list of people that the user is likely to create new connections to [27]. Random walk has also been used for community detection in online communities. Authors in [31] leverage the idea that short length random walks on a graph tend to get trapped into densely connected parts corresponding to communities. In [44], authors leverage the idea of random walk for crowdsourcing and routing tasks that require people to collaborate and synchronize both in time and physical space. Graph sampling [29] and node ranking [2] are examples of other applications for random walk. In our framework, we leverage the concept of random walk for nominating the eligible participants. In particular, we propose a customized random surfer which is responsible for crawling the social graph and identifying the suitable candidates as nominees. Our proposed random surfer, however, is different from the typical random walk, as it does not select the next step purely random, but based on a probability matrix. Moreover, despite the typical random walk which continues until convergence happens, our proposed random surfer takes a limited number of steps to find well suited participants amongst the friendship network. This will reduce the complexity of our proposed nomination solution.

As mentioned earlier, if the open-call strategy is used for nomination, there is no need for selection as everyone can contribute. However, for some crowdsourcing systems such as oDesk [1], there exist restrictions in the number of participants. In such cases, a limited number of participants should be selected from the set of nominees

[1] http://odesk.com

who have been identified by the nomination method. This restriction may be due to the nature of the task itself (such as time-critical tasks) [41], limitation in the resources needed for incentivizing the participants and/or evaluating the contributions (cost-critical tasks) [41]. In order to select from the nominees, the pre-selection methods select a fixed number of participants to compete to contribute [50]. Our proposed selection module is different from the above mentioned methods as it restricts the number of nominees, and at the same time, gives priority to well-suited nominees to be selected as final participants.

2.2 Collusion Detection in Online Communities

Collusion detection has been widely studied in P2P systems [1, 35]. A comprehensive survey on collusion detection in P2P systems can be found in [1]. Reputation management systems are also targeted by collusion. Colluders in reputation management systems try to manipulate reputation scores by collusion. Many efforts are put into detecting collusion using majority rules, weight of the worker and temporal analysis of the behavior of the users [47] but none of these methods is strong enough to detect all sorts of collusion [47]. In [39], Mukherjee et al. have proposed a model for spotting fake review groups in online rating systems. The model analyzes textual feedback cast on products in Amazon's online market to find collusion groups. They use eight indicators to identify colluders and propose an algorithm for ranking collusion groups based on their degree of spamicity. However, their proposed method is still vulnerable to some attacks. For example, if the number of attackers is much higher than honest raters on a product the model cannot identify this as a potential case of collusion. In the domain of participatory sensing, Authors in [23] aim at detecting the collusion by leveraging a reputation management system and outlier detection algorithms. In [19], a trusted platform module (TPM) is provided with each sensor device to attest the integrity of sensor readings. This local integrity checking makes the system resistant to collusion. To the best of our knowledge, the collusion prevention has not been discussed in social participatory sensing, and the methods proposed for participatory sensing are not applicable to this domain.

3 Preliminaries and Basic Concepts

The participant selection framework proposed in this work is inspired by the concept of random surfer, well-known for its application in Google PageRank [42]. So in this section, we first provide a short overview of random surfer. Then, we formulate our problem, define some basic terms and list our main assumptions.

3.1 Random Surfer

Assume that we have a directed graph of nodes where some nodes have directed links to other nodes. One common approach to find the level of importance of each node in the set of all graph nodes is to use a random surfer [30]. The main idea of random surfing is as follows. One of the graph nodes is selected randomly as the staring node, from which, the surfer starts its journey. The random surfer, then, picks one of the neighbouring nodes randomly and moves to that node. This process is repeated for a fixed period of time or till there is no outgoing link to go further. The level of importance of each node in the graph is proportional to the number of times it has been visited by the

random surfer. In other words, the level of importance of node P is calculated using the following equation:

$$\text{Level of importance of node } p = \frac{\text{the number of times it has been visited}}{\text{total number of steps taken by the random surfer}}$$

The possibility of a particular node being visited by a surfer depends on the number of nodes that have outgoing links to this node. Recursively, for these neighbouring nodes, the possibility of being selected depends on the number of their incoming links. This implies that the importance of a node is greater if some other important nodes point to it. This is the main idea behind the PageRank algorithm for calculating ranks of Web pages.

From the mathematical point of view, the random surfer concept is based on the theory of markov chain. A markov chain is a memoryless stochastic process in which, in each step, selecting the next state only depends on the current state of the process, and not on its history (i.e. those states visited earlier). A markov chain is also called a '*random surfer*' or a '*random walk*'.

The random surfer concept is widely used for graph processing such as node ranking and clustering (to be further discussed in Section 2). Assume that for a node p_i, the number of outgoing links is denoted by $|p_i|$. Then, the stochastic matrix Π representing the random surfer is defined is as follows:

$$\Pi_{n\times n} = \begin{pmatrix} \pi_{11} & \pi_{12} & \cdots & \pi_{1n} \\ \pi_{21} & \pi_{22} & \cdots & \pi_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \pi_{n1} & \pi_{n2} & \cdots & \pi_{nn} \end{pmatrix}$$

in which, π_{ij} is the probability that the random surfer visits node p_j, assuming that it is currently visiting node p_i at the moment, and n is the number of nodes. In PageRank algorithm, these probabilities are the same in each row and calculated as follows:

$$\pi_{ij} = \frac{1}{|p_i|}, \text{ where } 1 \leq j \leq n$$

It means that in PageRank, the probability of moving from a page to each of its neighbors is the same. The matrix Π is the base for building the random surfer matrix in the PageRank algorithm. In this paper, we propose a modified version of random surfer to select the most suitable participants for a task. The details of our proposed random surfer are discussed in Section 4.1.

3.2 Network Abstraction

A social participatory network consists of a set of people who are related to each other by friendship relations. Such a network is best represented as an undirected graph where *M* is the set of nodes representing entities (social network members) and *F* is the set of relations between entities (friendship relations). Each member has a profile containing his attributes and related information. Some attributes represent the member's personal information such as name and address. Others include the settings and outcome of member's social behavior. Examples are the member's reputation score, the history of his previous transactions, the pairwise trust scores, etc. We represent members of the social network by $M = \{m_i\, 1 \leq i \leq n\}$.

A participatory task or simply a *task* is represented by θ_i, and Θ is the set of all the tasks to be solved ($\Theta = \{\theta_i\}$). The owner of the task is also called the *requester*. The platform may target and invite a set of eligible members to participate to the task. These members are called *nominees*. Nominees who accept the invitation and are selected to contribute to the task are called the *participants*.

A *pairwise trust* relationship is a directed weighted relation between two members of the social network and is denoted by τ_{ij}. The trust relationship τ_{ij} shows to what extend the member m_i trusts m_j for participation in his tasks.

4 Participant Selection Framework

The participant selection framework proposed in this paper consists of two main modules: *nomination* and *selection*. In this section, we first explain the nomination module and the proposed nomination algorithm in Section 4.1. Then, we will describe the selection module in Section 4.2.

4.1 Nomination Module

When defining the task, the requester may wish to define a set of requirements. For instance, he may require members who have a specified level of reputation (in terms of the history of previous contributions), or who live in a particular geographical region or who possess certain specific expertise [43]. In other words, these requirements act as criteria for the identification and selection of eligible participants. In order to identify eligible participants, the nomination module is responsible for crawling the social network graph and identifying potential candidates whose profile information match the task requirements. These candidates form the nominated group and will be invited to contribute to the task.

As explained in Section 1, constraining the graph crawling only to friends may lead to insufficient nominees, due to the potential sparseness of friendship graph or lack of enough experts among friends. Therefore, in our proposed method, we extend the social graph crawling deeper in the social graph in order to maximize the possibility of finding well-suited participants. In our framework, the nomination module relies on a customized random surfer to find well-suited participants. Our implementation of the random surfer is different from that employed in web page ranking and link recommendation systems such as [2, 12]. The typical random walk [30] is an iterative process wherein, in each iteration the next node to be visited is selected randomly and uniformly. The random walker may traverse each node multiple times based on the number of incoming links of a node. The importance of the node (i.e., the node's rank) is the number of times it has been visited by the random walker. It has been shown that in some instances convergence can take several iterations [30]. The proposed surfer is different since the probability of selecting a node as the next step is not the same for all available candidates. Some nodes have a higher chance than others to be visited by the random surfer. In other words, the selection of the next node is based on a probability matrix, which is directly related to the node's suitability score (details in Section 4.1). Moreover, the surfer takes a limited number of steps and the graph traversal concludes when a sufficient number of workers are recruited, or if it is not possible to go deeper. This significantly reduces the time complexity of the algorithm.

In the following, we first explain the evaluation of member's suitability score in Section 4.1 and then, describe the customized random surfer in detail in Section 4.1.

Suitability Assessment

In order to evaluate the suitability of a member, a set of parameters should be considered and evaluated. In the following, we first explain the evaluation of each parameter in detail and then discuss the calculation of suitability score.

- **Reputation**

  The requester may specify a minimum level of the reputation as a requirement for participation, in order to obtain high quality contributions. We assume that a reputation management system such as [11, 24] is already in place, which calculates a reputation score ρ_i for each member m_i (a survey on reputation management systems can be found in [5]). We also assume that the required reputation score of the task is denoted by ρ_{req}. In that case, the required level of reputation score for member m_i to participate in task θ_j, denoted by $\Delta\rho_{ij}$ is as follows:

$$\Delta\rho_{ij} = \begin{cases} \rho_i & \text{if } \rho_i \geq \rho_{req} \\ 0 & \text{otherwise} \end{cases}$$

  The higher the value of $\Delta\rho_{ij}$, the higher the eligibility of the participant to contribute to the task. We assume that $\Delta\rho_{ij}$ is a number in the range of [0,1].

- **Expertise**

  Expertise is defined as the measure of a participant's knowledge and is particularly important in tasks that require specific knowledge about a particular domain. In other words, participants may be asked to have specific expertise such as programming skills, familiarity to a geographical area, proficiency with a particular language or so on. Greater credence is placed in contributions made by a participant who has expertise in the task. Expert finding systems such as [20, 3] may be employed for evaluating expertise. These systems employ social networks analysis and natural language processing (text mining, text classification, and semantic text similarity methods) to analyse explicit information such as public profile data and group memberships as well as implicit information such as textual posts to extract user interests and fields of expertise [3]. Expertise evaluation is done by incorporating text similarity analysis to find a match between the task keywords and participant's expertise. We denote the level of match between the i^{th} member's expertise and the j^{th} task requirements by ΔE_{ij}. Assume that the E_j^t is the set of skills required by the task θ_j and E_i^m is the set of skills of the member m_i, then

$$\Delta E_{ij} = \frac{|E_j^t \bigcap E_i^m|}{|E_j^t|}$$

  ΔE_{ij} is a number in the range of [0,1]. The higher the value of the ΔE_{ij}, the higher the match between the member's profile and the task requirement.

- **Privacy Requirements**

Privacy preservation has always been a great concern in social networks. When discussing about privacy in social networks, it is important to specify what defines failure to preserve privacy. One type of privacy breach occurs when a piece of sensitive information about an individual is disclosed to an adversary [54]. In social participatory sensing, privacy leakage may implicitly occur during the recruitment process through the nature of the query. In particular, it may lead to the disclosure of the requester's geographical location, his personal interests, political or religious views and so on. For example, if the requester asks for the vegetarian restaurants in a specific geographical area, it is probable that he is vegetarian and lives in that place. So, it is desirable to maximize the privacy preservation of the requester in the recruitment process.

There are different solutions for evaluating the pairwise privacy scores (some have been discussed in our previous work [8]). In this paper, we assume that the probability of a privacy breach in one-hop neighbourhood of the requestor is zero. This is reasonable since friends are assumed to be trustworthy. For other non-friend nodes, the probability of privacy breach is greater in nodes who have been involved in greater number of tasks initiated by that requester. The intuition behind this assumption is that the more m_i attends in tasks initiated by m_j, the more of m_j's sensitive information will be revealed to him. So, the pairwise privacy score will decrease as the number of mutual tasks increases. With this intuition in mind, the pairwise privacy score of giving (the non-friend) m_i the permission to contribute to the task θ_j initiated by m_j, denoted by ΔPr_{ij} is calculated via the following function:

$$\Delta Pr_{ij} = \begin{cases} 1 - (\frac{t_{ij}}{T})^2 & \text{if } t_{ij} \leq T \\ \\ 0 & \text{otherwise} \end{cases}$$

where, t_{ij} is the number of tasks m_i has done for m_j so far, and T is a system defined parameter which denotes the maximum number of the tasks initiated by m_j that m_i can participate in, following which m_i's privacy score reduces to zero.

- **Requester's Preferred and Blocked Lists**

  The requester may be provided with a list of preferred participants, whom the requester prefers to recruit in his future tasks. This list may be automatically generated by the application and would typically contain the requester's friends who have demonstrated trustworthy behaviour in tasks originated by the requester. The requester may also add some participants to the list manually, based on his trust upon them. It is clear that those who appear in this list should be assigned a higher suitability score. So, for the member m_i who is being considered for nomination for the task initiated by m_j, we define the parameter Pf_{ij} to be 1 if m_i belongs to the m_j's preferred list, and zero otherwise. Similar to the requester's preferred list, a blocked list may also be available for the requester, which contains the list of those whom the requester desires to exclude form the list of contributors. This may be because of their poor behaviour in previous tasks, or due to privacy issues. It is obvious that the members belonging to this list should not be nominated. So, for the member m_i who is being considered

for nomination for the task initiated by m_j, we define the parameter B_{ij} to be 1 if m_i belongs to the m_j's blocked list, and zero otherwise.

Computing the Suitability Score

Once the above parameters are evaluated, they should be combined to arrive at a single value for the member's suitability score. To do so, the suitability score for a member m_i to attend in task $theta_j$ initiated by m_j, referred to as σ_i, is calculated as a weighted sum of parameters as:

$$\sigma_i = \begin{cases} 0 & \text{if } B_{ij}\text{=1} \\ w_1 * \Delta\rho_{ij} + w_2 * \Delta E_{ij} + w_3 * \Delta Pr_{ij} + w_4 * Pf_{ij} & \text{otherwise} \end{cases}$$

where w_i is the weight of each parameter, and $\sum_{i=1}^{4}(w_i)$ equals to 1. The adjustment of the weights is application-dependant. For example, for privacy-aware applications, w_3 is set to be considerably high to give more importance to privacy parameter. Similarly, for tasks where expertise requirements are important, a higher weight may be associated with expertise (w_2). The suitability score is in the range of [0,1].

Customized Random Surfer

In the following, we first define the proposed customized version of random surfer. Recall that a typical random surfer traverses graph nodes based on the concept of markov chain. In other words, it starts from a particular node and randomly selects the next to-be-visited node in a memoryless manner. This selection is done randomly and uniformly from the set of all possible nodes. In this paper, we propose a customized random surfer which does not act in a purely random manner, but is biased in a way that it considers the suitability score of the nodes for the selection. The intuition behind this strategy is to give better suited members a greater chance to be selected. Corresponding to a random surfer, we have a stochastic matrix Π which contains the probabilities of transitioning from one node to each of its direct neighbours. This matrix is called *transition probability matrix*. Each row π_i, called the *probability distribution row*, denotes the i^{th} row of the Π and contains the probability distribution row corresponding to the i^{th} member of the network.

Assume that in a social network, member m_i serves as the requester and intends to publish a task. Let $\varphi_i = \{m_j | m_j \text{ is friend with } m_i\}$ be the set of m_i's friends. In order to find suitable participants, we initiate K random surfers where $K = |\varphi_i|$. Each random surfer, denoted by ω_j, starts from the friend m_j and walks through the graph to find and nominate suitable participants. Assume that the current state of a random surfer ω_j is m_{cur}. The random surfer first checks the suitability of m_{cur}. If the suitability score is greater than a predefined threshold, he will be invited to contribute. The surfer then continues its journey to find other nominees from the list of m_{cur}'s friends and φ_i is updated accordingly. The next step will be selected from φ_{cur} based on the suitability scores. In other words, the probability of selecting m^j_{cur} (the j^{th} friend of m_{cur}) as the next step, denoted by $\pi_{cur,j}$, is:

$$\pi_{cur,j} = \frac{\sigma_j * \tau_{cur,j}}{\sum_{k:k\in\varphi_{cur}} \sigma_k * \tau_{cur,k}}$$

where σ_j is the m_{cur}^j's suitability score and $\tau_{cur,j}$ is the pairwise trust of m_{cur} upon his j^{th} friend. It is evident that for each member m_i, the sum of probabilities of moving to his friends is equal to 1. In other words, $\sum_{j:1}^{K} \pi_{i,j} = 1$. Based on this, the stochastic matrix Π (described in Section 3.1) can be filled as $\Pi_{n \times n} = \{\pi_{i,j}\}$ in which, each element $\pi_{i,j}$ is the probability of selecting m_j as the next step for a random surfer that currently is in m_i. Π can be used by random surfers to determine the next steps.

The random surfer continues walking through the graph and inviting nominees to contribute. In order to control how far a random surfer can move from the requester, we define a parameter called the *propagation factor* and denote it by λ. The selection of an appropriate value for λ is challenging. A greater value of λ allows the random surfer to crawl deeper in the social graph, and thus increases the chance of finding more suitable workers. On the other hand, it may increase the risk of privacy leakage due to getting far from the requester's friendship network.

In the following, we present the practical implementation of the process discussed above, in the form of an algorithm. Algorithm 1 presents our proposed nomination algorithm. In this algorithm, W is a shared list which is accessible to all random surfers initiated for task θ_j, and includes the ID of all members who are invited by random surfers. Therefore, in each step, W contains the list of participants which have been nominated so far. This list is used as a shared memory among random surfers to prevent them from nominating a member twice. The algorithm first initialises an empty list W. It also extracts the list of all m_i's friends. Upon each friend m_i^j (j^{th} friend of m_i), a separate random surfer ω_j is initiated with the current state set to be m_i^j (lines 1 to 8 in the algorithm).

The lines 9 to 28 are the steps that each random surfer takes independently. Each random surfer, ω, checks to see if its current state is suitable to contribute to the task, using the equations proposed in the Section 4.1. If the member is suitable to do the task, he will be nominated. Then, the random surfer ω loads the row of Π corresponding to the current state of ω and then updates the transition probabilities. In order to do so, the pairwise trust between the current node and the nominee is investigated. If less than a specific threshold, m_j's suitability score will be set to zero. Otherwise, it will be updated with the value $\tau_{cur,j}$ stated in Section 4.1. As can be seen, the probability of a particular member being visited by a surfer is in direct relationship with his suitability score.

4.2 Selection Module

As a result of the nomination process, a finite set of eligible participants are nominated and invited to participate in the task. The selection module is responsible for selecting the participants from the set of nominees who have accepted the invitation. As mentioned in Section 1, our proposed selection module is aimed at overcoming the relevant issues in existing selection methods (i.e., open-call, auction, and pre-selection) by utilizing the best of these recruitment strategies. In this module, we propose a time-aware and collusion-free recruitment method which is an extension to the pre-selection recruitment approach and aims at preventing the collusion.

Eligibility Assessment

Whenever a nominee accepts the invitation, the selection module takes into account a set of time-aware parameters and decides whether to select the participant. These parameters are (i) the selection score (i.e., the ratio of participants selected so far to the

Algorithm 1 Nomination Algorithm

Input: Π as the transition probability matrix, m_i as the requester, NoP is the required number of participants, and θ_j as the advertised task
Output: W as the list of nominees.

```
1: φ_i = list of m_i's friends
2: Initialize W as an empty list.
3: for all f ∈ φ_i do
4:    Initiate a random surfer ω_j from f
5:    //current state of ω_j is f
6: end for
7: Ω = set of all initiated random surfers
8: for all ω ∈ Ω do
9:    L=λ
10:   while true do
11:      // m_cur denotes the current state of the random surfer ω
12:      if m_cur is suitable for θ_j then
13:         if |W| <= 2 * NoP then
14:            Nominate m_cur
15:            Add m_cur to W
16:         end if
17:      else
18:         Stop random surfer ω
19:         Exit
20:      end if
21:      Load π = Π_cur //the row of Π corresponding to the current node
22:      Update π // see the algorithm description for details
23:      if π is empty then
24:         // there are no choices for next step
25:         Stop random surfer ω
26:         Exit
27:      end if
28:      Select a member of π as m_cur // see the algorithm description for details
29:      L = L+1
30:      if L ≥ λ then
31:         Stop random surfer ω
32:      end if
33:   end while
34: end for
35: return W
```

total number of required participants), (ii) the remaining time to the task deadline and (iii) the timeliness of the participant in previous tasks. The intuition behind considering these parameters is that when the task's remaining time is short, it is rational to select the nominee that has shown timely behaviour in his past contributions, to maximise the chance of receiving a contribution before the deadline. The first two parameters are combined via a geometric mean function to form a time suitability score. Geometric mean is often used for comparing different items and finding a single "figure of merit" for these items, when each item has multiple properties. A geometric mean, unlike an arithmetic mean, tends to dampen the effect of very high or low values, which might bias the mean if a straight average (arithmetic mean) were calculated. So, for the member m_i to be selected to attend in task θ_j, the time suitability will be as follows:

$$\text{Time suitability}(m_i) = \sqrt{\text{Timeliness}(m_i) * \text{Remaining time}(\theta_j)}$$

The time suitability is then combined with the selection score via a fuzzy inference engine. The result is an eligibility score for the nominee as follows.

$$\text{Eligibility Score}(m_i) = Fuzzy(\text{Time Suitability}(m_i), \text{Selection Score}(\theta_j))$$

If greater than a predefined threshold, the nominee will be considered to be eligible to participate.

Fuzzy inference system

Our proposed framework employs fuzzy logic to calculate the Eligibility Score (ES) for each nominee. The use of fuzzy logic allows us to achieve a meaningful balance between the time suitability and the selection score. We cover all possible combinations of Time Suitability (TS) and Selection Score (SS) and address them by leveraging fuzzy logic in mimicking the human decision-making process. The inputs to the fuzzy inference system are the crisp values of TS and SS. In the following, we describe the fuzzy inference system components.

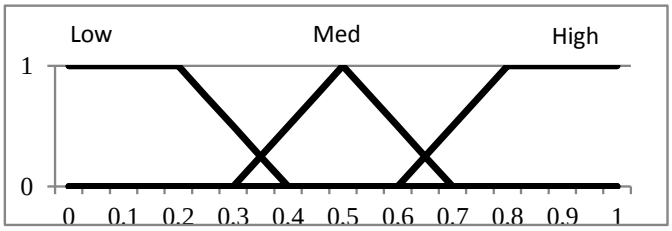
- Fuzzifier: The fuzzifier converts the crisp values of input parameters into a linguistic variable according to their membership functions. In other words, it determines the degree to which these inputs belong to each of the corresponding fuzzy sets. The fuzzy sets for TS, SS and ES are defined as:
  T(TS)=T(SS)={Low, Med, High}, T(ES)= { VL, L, M, H, VH}.

  For any set X, a membership function on X is any function from X to the real unit interval [0,1]. The membership function which represents a fuzzy set A is usually denoted by μ_A. The membership degree $\mu_A(x)$ quantifies the grade of membership of the element x to the fuzzy set A. The value 0 means that x is not a member of the fuzzy set; the value 1 means that x is fully a member of the fuzzy set. The values between 0 and 1 characterize fuzzy members, which belong to the fuzzy set only partially.
  Fig.4.1(a) represents the membership function of TS and SS and Fig.4.1(b) depicts the ES membership function. We used trapezoidal shaped membership functions since they provide adequate representation of the expert knowledge, and at the same time, significantly simplify the process of computation.

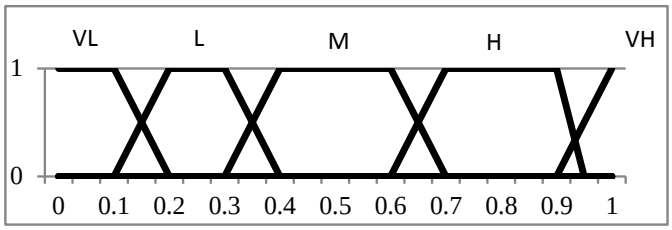
- Inference Engine: The role of inference engine is to convert fuzzy inputs (TS and SS) to the fuzzy output (ES) by leveraging If-Then type fuzzy rules. The combination of the above mentioned fuzzy sets create 3*3=9 different states which

Table 4.1: Fuzzy rule base for defining ES according to TS and SS

Rule no.	if TS	and SS	Then ES
1	Low	Low	M
2	Low	Med	L
3	Low	High	VL
4	Med	Low	H
5	Med	Med	M
6	Med	High	L
7	High	Low	VH
8	High	Med	H
9	High	High	M



(a) Membership function for TS and SS



(b) Membership function for ES

Figure 4.1: Membership functions of input and output linguistic variables

have been addressed by 9 fuzzy rules as shown in Table 4.1. Fuzzy rules help in describing how we balance the various eligibility aspects. The rule base design has been done *manually*, based on the experience and beliefs on how the system should work [51]. To define the output zone, we used *max-min* composition method as: $\mu_{T(ES)}(ES) = max[\min_{\substack{X \in T(TS), \\ Y \in T(SS)}} (\mu_X(TS), \mu_Y(SS))]$. The result of the inference engine is ES which is a linguistic fuzzy value.

- Defuzzifier: A defuzzifier converts the ES fuzzy value to a crisp value in the range of [0, 1]. We employ the Centre of Gravity (COG) [32] defuzzification method, which computes the center of gravity of the area under ES membership function. COG is perhaps the most commonly used and popular defuzzification technique with the advantage of quick and highly accurate computations.

Once the crisp value for the eligibility score is computed, it is compared to a predefined threshold (has been set to 0.5 in the implementation). If greater than the threshold, the nominee is considered as eligible.

Collusion Prevention

Once the member m_i is considered to be eligible for being selected, a final check is done to ensure that the selection of m_i will not result in potential collusion. In particular, we aim to identify whether the addition of m_i to the set of previously selected participants will result in the formation of a group of colluders.

Collaborative attacks which are also called collusion attacks are those in which, a group of people collaborate on changing the results of a task [39]. For example, they may collaborate to be selected as the final participants and then, produce poor quality contributions that severely impact the goal of the task. Most existing collusion detection techniques rely on behavioural indicators to identify colluding groups [6, 39, 36]. These indicators reflect suspicious behaviour from a group of members which indicates the possibility of collusion. For example, the collaboration of a group of participants may be considered as collusion if the following suspicious behaviors are observed: i) R members of the group have collaborated (ii) these members attend in the same k tasks; (iii) they submitted their contributions within a small time window. All these factors occurring together strongly suggest suspicious activities.

In our selection module, for each new participant to be selected, we consider a set of indicators that suggest the likelihood of the formation of a colluding group among the selected participants. Note that these indicators reflect the likelihood of collusion only when they all occur together. The first indicator is the Group Size (GS) which is the number of colluders who collaborate in similar tasks. The larger the group, the more damaging it is. The second indicator is the Group Support Count (GSC) which is the number of tasks in which the group members have collaborated in the past. Groups with high support counts are more likely to be colluding as the probability of a group of random people to have attended the same tasks together is rather small. The third indicator is the Group Time Window (GTW) which indicates the time window of the group contributing to a task. A group of participants contributing to a task within a short burst of time is more prone to be colluding. These indicators happening together indicate the collusion probability. So, for each eligible participant m_i to be selected, these indicators are investigated. If all greater than certain related thresholds, it implies that the selection of m_i may lead to potential collusion, and hence, the participant will not be selected.

In order to compute the collusion probability, the first step is to identify all existing subgroups in the group of selected participants (including m_i), who have collaborated on multiple tasks in the past. To do so, we use the well-known technique called the Frequent Itemset Mining [22] which has performed well for collusion detection in previous literature [39, 7]. In our context, a set of items are the set of all selected participants for the current task. The set of transactions are the set of all tasks that m_i has been involved in the past. By mining frequent itemsets, we find groups of participants who have contributed to multiple tasks together. We consider a group as a tuple in the form of ($\gamma_i^\theta(P)$, $\gamma_i^\theta(T)$, $\gamma_i^\theta(C)$). $\gamma_i^\theta(P)$ is the set of group members' IDs, $\gamma_i^\theta(T)$ is the list of all tasks these members has collaborated in, and $\gamma_i^\theta(C)$ is the set of all contributions made by members to these tasks. Each contribution has a timestamp indicating its submission time. The difference between the latest and earliest timestamp of contributions submitted by the group members indicates the group time window, denoted by $\gamma_i^\theta(\epsilon)$.

Based on the FIM output, the indicator values can be quantified as follows. Group Size (GS) = $\left|\gamma_i^\theta(P)\right|$, Group Support Count (GSC) = $\left|\gamma_i^\theta(T)\right|$, and Group Time Window (GTW) = $\gamma_i^\theta(\epsilon)$.

We consider a group of collaborators collusive, if all the following conditions are met:

1. If the Group Size (GS) is greater than a predefined threshold th_1.
2. If the Group Support Count (GSC) is greater than a predefined threshold th_2.
3. If Group Time Window (GTW) is smaller than th_3.

The member m_i will be selected to contribute to the task θ if no group with the above mentioned conditions is created as a result of this selection.

5 Experimentation and Evaluation

In this section, we conduct a simulation-based evaluation to analyze the behavior of our proposed framework. First, we explain experimentation setup, the metrics we use for performance evaluation and the datasets we used in experiments in section 5.1. Then, we compare our proposed framework with other methods in Section 5.2. Then, we analyze the behavior of our framework in Section 5.3 in order to find an optimum configuration. Finally, in Section 5.4, we investigate the efficiency of our proposed collusion prevention method.

5.1 Experimentation Setup

To undertake the preliminary evaluations outlined herein, we chose to conduct simulations, since real experiments in social participatory networks are difficult to organise. Simulations afford a controlled environment where we can carefully vary certain parameters and observe the impact on the system performance. Our simulations have been conducted on a PC running Windows 7.0 professional and having 4GB of RAM. We used Matlab R2012 for developing the simulator.

Dataset

The dataset that we use for our experiment is the real web of trust of Advogato.org [34]. Advogato.org is a web-based community of open source software developers in which, site members rate each other in terms of their trustworthiness. Trust values are one of the three choices master, journeyer and apprentice, with master being the highest level in that order. The result of these ratings among members is a rich web of trust, which comprises of 14,019 users and 47, 347 trust ratings. The Advogato web of trust may be viewed as a directed weighted graph, with users as the vertices and trust ratings as the directed weighted edges of the graph. So, it is in perfect match with our assumptions related to participants and their trust relations in social participatory network. The distribution of trust values in the Advogato web of trust is as follows: master: 17,306, journeyer: 21,353, and apprentice: 8688. The instance of the Advogato web of trust referenced in this paper was retrieved on October 13, 2007. In order to conform the Advogato web of trust to our framework, we map the textual ratings in the range of [0, 1] as master = 0.8, journeyer = 0.6, and apprentice = 0.4. We also pre-processed the dataset in order to remove the isolated nodes that have no connection.

174 nodes were identified as isolated and were removed. We also have enriched the dataset in order to adapt it with our simulation scenario. To do so, we have computed a reputation score for each member by calculating the average of all pairwise trust scores the member receives from his friends. The reputation score is a number in the range of [0, 1]. We also assign each member a set of expertise attributes in order to use them for measuring the member's suitability score. We assume that there exist 10 different expertise attributes in the system. Each expertise attribute is an integer number in the range of [1,10]. The total number of expertise attributes for each member is chosen randomly according to the reputation of the member. In other words, in the enriched dataset, those with higher reputation scores are likely to have greater number of expertise attributes. As mentioned before, the nomination module calculates the privacy score based on the number of tasks a participant has attended for a particular requester. Therefore, we randomly and uniformly selected numbers in the range of [1, 100] as the number of tasks completed by each member for each requester.

In order to evaluate the performance of our proposed collusion prevention method, we utilized the Wikipedia voting dataset. In Wikipedia[1], the voting process is used to elect administrators[2]. Every registered user can nominate himself or another user as an administrator in Wikipedia and initiate an election. The other users participate in the election and cast their votes on the eligibility of nominee. If the majority of users recognize a user as eligible, he then will become a Wikipedia administrator. In order to incorporate this dataset in the context of our framework, we employ the following mapping. The requester is the nominee, the worker is the voter, the task is evaluating the eligibility of the nominee as an administrator in Wikipedia and the contribution is the worker's vote. We use the log of Wikipedia Adminship Election[3] which is collected by Leskovec et al. for behavior prediction in online social networks [33], referred to as WIKILog. WIKILog contains about $2,800$ elections (tasks) with around $100,000$ total votes and about $7,000$ users participating in the elections either as a voter or a nominee. We use the WIKILog to demonstrate the efficacy of our proposed framework to detect collusion.

Evaluation Method and Metrics

In order to evaluate the performance of our proposed framework, we run the experiment for a set of rounds. A simple experimentation round contains the following steps: In the first step, we choose a requester out of the members of Advogato community. This selection is performed uniformly, meaning that all members have the same chance to be chosen as the requester. Then, a task is generated to be advertised to the community. Each task contains a set of attributes, mainly, a minimum accepted reputation score, a set of at most 5 required expertise attributes, and the maximum number of required participants. Once the requester is chosen and the task is generated, the nomination algorithm (Algorithm 1) is executed in order to find and invite suitable nominees. Then the selection module, explained in Section 4.2, chooses a subset of nominees as selected participants to contribute to the task. We assumed that at least 50% of nominees apply to the task for contribution.

In order to evaluate the effectiveness of framework modules, we define four evaluation metrics. The first metric is the *number of nominees*. The ability to identify more

[1] http://www.wikipedia.org/

[2] http://en.wikipedia.org/wiki/Wikipedia:Requests_for_adminship

[3] http://snap.stanford.edu/data/wiki-Elec.html

suitable nominees is a desirable property of the nomination module. The second evaluation metric is the *overall suitability score of the nominees*, which is the average of all nominees' suitability scores. A larger value for this metric suggests that the nomination module is able to recruit well-suited participants. We have two similar metrics to evaluate the performance of the selection module: the *number of selected participants* and the *overall suitability score of selected participants*. In the following, we will use these four metrics to evaluate the performance of our framework. All results shown in charts are the average of outcome of running the experiment for 1000 independent rounds.

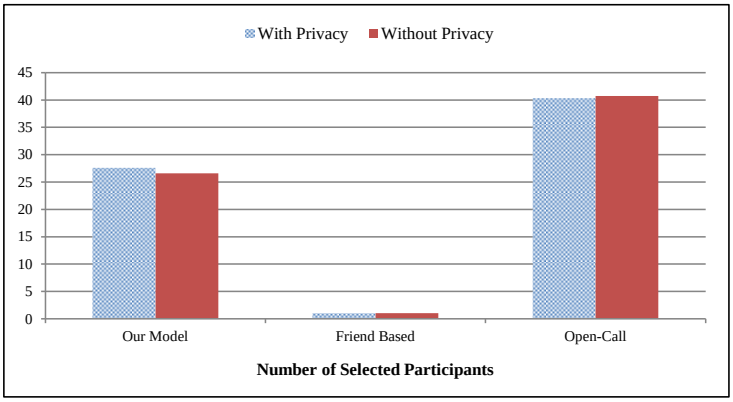
5.2 Performance Comparison

In this section, we compare the performance of our framework with two well-known recruitment methods: (i) *Open-call* which is used in most existing crowdsourcing platforms such as Amazon Mechanical Turk [4] , CrowdFlower [5], etc. Recall that in this scheme, the requester broadcasts the task to all members in the community and everyone is able to contribute to the task (ii) *Friend-based* which is widely used in social networks and related work such as [21, 26], wherein, the requester advertises the task to his friends.

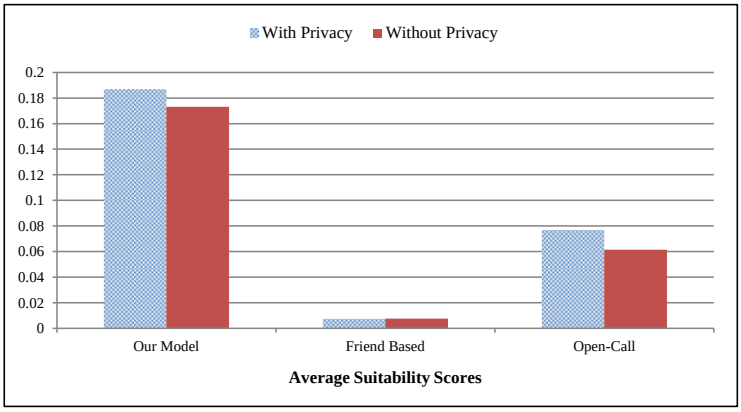
It should be noted that neither of these methods consider the privacy preservation in their recruitment methods. So, in order to have a fair comparison, we consider each of the compared methods with two separate configurations: privacy-aware and non privacy-aware. In privacy-aware configuration, the weights of reputation score, expertise, privacy score and requester's preference in the computation of the suitability score are 0.5, 0.3, 0.1 and 0.1, respectively (refer to the Section 4.1). In non privacy-aware recruitment, the weight of privacy score is zero and reputation, expertise and preference are taken into account with weights of 0.55, 0.35, and 0.1 respectively. Another important point is that the simulation results illustrated in the following figures have been scaled with the number of participants in order to have reasonable comparisons. For example, the number of selected participants for each of the aforementioned methods has been scaled with the corresponding maximum number of possible participants. In order to evaluate the performance of methods in real situations, we run the simulation in three different situations: (i) when the requester has few friends, (ii) when the requester has large number of friends, and (iii) when we select the requester randomly, regardless of his number of friends. Note that the concepts such as 'few' or 'large' are relative and depend on the characteristics of the underlying social network. In order to consider these situations, we first arrange all members (i.e., Advogato members) in ascending order according to their number of friends (outgoing links). For the first situation, the requester will be selected from the first one third of the members, and for the second situation, the requester will be selected from the last one third. The last situation will be the case when the requester is selected randomly from the unordered list. In Advogato, the range of number of friends for the first group is between 3 and 1000, and for the second group is between 3000 and 4000. To come up with dependable results, we run the simulation for 1000 rounds.

Figures 5.1(a) and 5.1(b) depict the results of comparing three methods for the case in which, the requester has few friends. As it is evident from the charts, open-call outperforms other two methods in terms of the number of selected participants. This is an

[4] http://mturk.com
[5] http://crowdflower.com



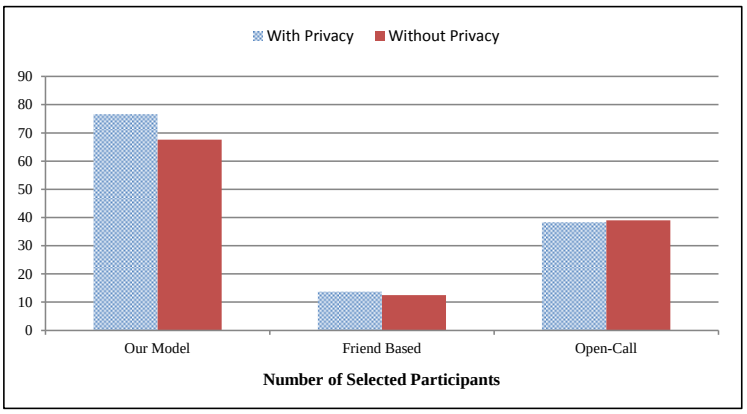
(a)



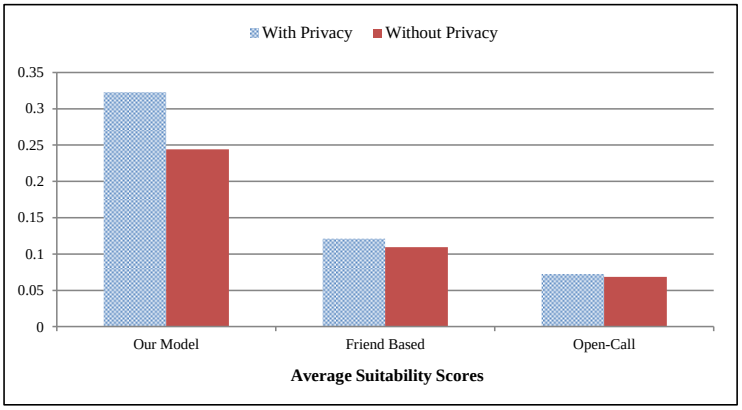
(b)

Figure 5.1: Performance of three methods in the case of requesters with few friends

expected result since in this method, it is possible to select the participants from everywhere inside the social graph, whereas there are restrictions in participant selection in two other methods. In our proposed method, a potential limitation is due to λ which restricts the recruitment domain. Moreover, having few friends will result in few numbers of random surfers, which results in less selected participants. The friend based method is also limited to recruiting one-hop friends. But when it comes to overall suitability score of the selected participants, the best performance belongs to our framework. This is because our method considers the suitability scores for in selection module and tries to assign higher selection probability to participants with higher suitability scores. This better performance is of great importance since it demonstrates a valuable achievement for the case of having sparse friendship network, which, as mentioned in Section 1, is currently an issue in existing online social networks. The relative order of these 3 methods is consistent in both privacy-aware and non privacy-aware scenarios. Figures 5.2(a) and 5.2(b) illustrate the performance of three methods for the case in which, the requester has large number of friends. In this case, as it is expected, the performance of the friend-based method improves since the number of friends (potential



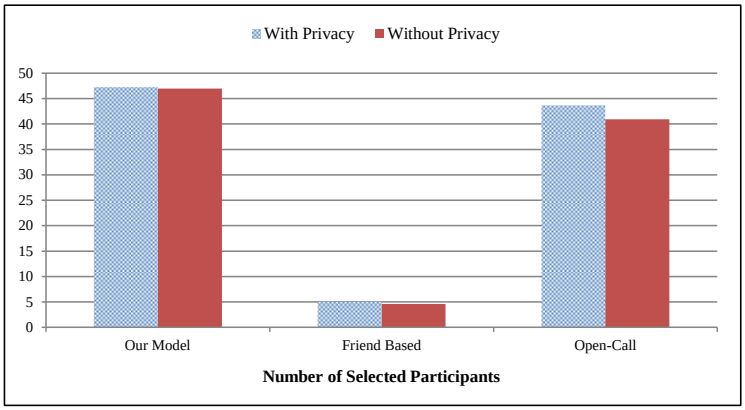
(a)



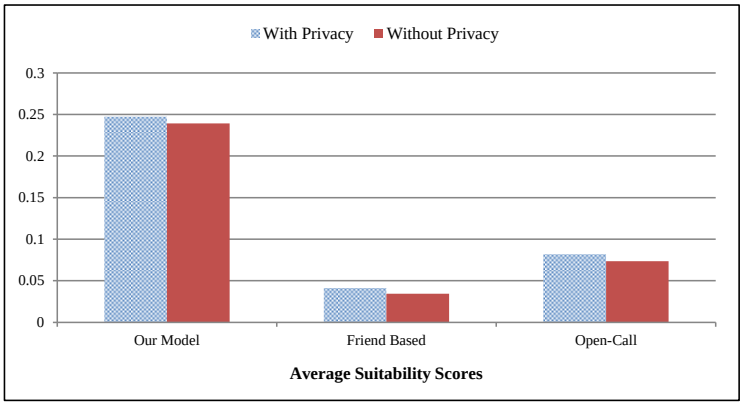
(b)

Figure 5.2: Performance of three methods in the case of requesters with large number of friends

participants) has increased for both (privacy-aware and non-privacy-aware) scenarios. The best performance still belongs to our framework. Remember that in the previous case where the requester has few friends, the open-call method outperforms ours in terms of number of selected participants due to the limitation occurred by the propagation factor and number of random surfers. Here, our method outperforms the open call method even in terms of number of selected participants. This is due to the large number of friends in the requester's friendship graphs, which in turn, increases the number of random walks and consequently, the number of selected participants. Finally, Figures 5.3(a) and 5.3(b) show the results of our experiments when we selected a requester from the community, regardless of his number of friends. In this case, as it is expected, due to the scarcity of the social network, the overall performance of the open-call is better than friend-based method. As for our proposed framework, it is slightly better than open-call in terms of the number of selected participants. This small difference between our method and the open-call method is due to the improved performance of open-call in cases where the selected requester has few friends and better performance of our method in cases where the selected requester has large number of friends.



(a)



(b)

Figure 5.3: Performance of all methods regardless of requesters' number of friends

As all above figures show, the number of nominees and selected participants decreases when privacy considerations are taken into account, since such considerations will result in tighter restrictions in selection module. The important point is that relative ordering of the methods in terms of performance remains unchanged in both the privacy-aware and non-privacy-aware scenarios.

5.3 Sensitivity Analysis

In this section, we run a series of experiments to reach to an optimal setting for our proposed recruitment framework. In particular, we first obtain the optimal value for λ (propagation factor), and then evaluate the performance of our framework in the presence/absence of privacy considerations and participant selection process.

Optimum Value of λ

One of the important parameters which impacts the performance of the random surfer is the propagation factor, denoted by λ. In fact, λ is a system-dependant parameter

which denotes how deep the random surfer can explore the graph to find suitable participants. In order to assess the performance of our framework, we need to find an optimum value for λ. Note that λ is a system-dependant parameter and its optimum value totally depends on the characteristics and the size of social graph. We conducted an experiment to test the framework on Advogato graph with different values of λ in the range of 1 to 150. For each λ value, we generated 500 tasks and then investigated the outcomes. Based on various runs of this experiment, the highest value of overall suitability score for selected participants is obtained when λ is equal to 100. So, we select this value for λ as the optimum value for future experiments.

Performance Analysis of Framework Components

In addition to the value of λ, we investigate the impact of two other aspects on the performance of our proposed framework. The aim of these experiments is to obtain the best configuration for our proposed framework.

At first, we try to investigate the effect of privacy score in the evaluation of suitability score. As mentioned before in Section 4.1, the probability of a selecting a non-friend participant for further tasks of a particular requester has an inverse relation to the number of the tasks he has been involved for that requester, due to the reduction in his privacy score. In other words, taking the privacy into consideration, while valuable in terms of members' security, will inherently decrease the number of potential participants. In the following experiments, we aim at investigating how the privacy score consideration will affect the framework performance in terms on number of nominees and selected participants.

Next, we aim at observing the performance of our framework with and without the selection module. We expect that including the selection module will increase the overall suitability score, but at the same time, will decrease the number of final participants, since it tightens the criteria of participant selection.

In order to evaluate the effect of these two components, we conducted an experiment in which, the performance of our framework is evaluated with the following four scenarios:

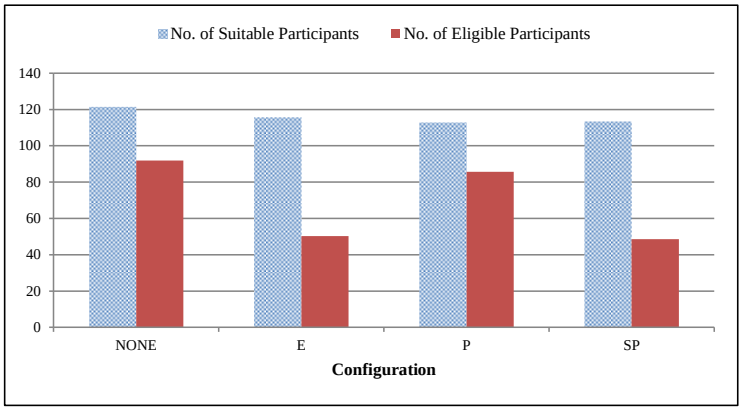
1. In the first scenario, we neither take privacy nor selection module into account. In other words, the suitability score of nominees is only calculated based on their reputation, expertise and the requester's preferred and blocked list. Also, we deactivate the selection module. In our illustrations in Figures 5.4(a) and 5.4(b), we represent this scenario by '*NONE*'.

2. In the second scenario, the selection module is active and working. The privacy does not affect the suitability score. We denote this scenario by '*S*'.

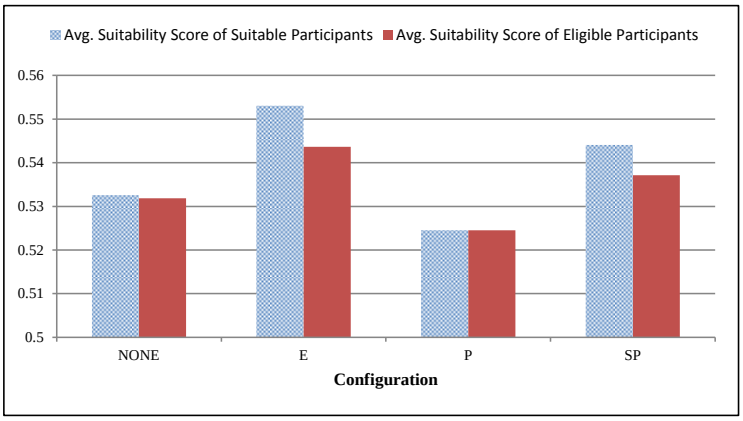
3. In the third scenario, the privacy score is considered in the evaluation of suitability scores. The selection module, however, is not included, meaning that when a nominee applies to do a task, no restriction will be applied and he will be directly accepted if there are still vacant places. This scenario is denoted by '*P*'.

4. The forth scenario, is our proposed framework where both privacy and selection aspects are taken into account. We denote this scenario by '*SP*'.

The evaluation results are depicted in Figures 5.4(a) and 5.4(b). As shown in Figure 5.4(a), the overall number of nominees and selected participants is the highest



(a)



(b)

Figure 5.4: Evaluation of our framework with different scenarios

in the first scenario (when we have neither selection module nor privacy considerations). This is because both the privacy consideration and the involvement of selection module pose limitations in the number of nominees and selected participants. However, as Figure 5.4(b) reveals, the overall suitability score in this scenario is too low, since there is no suitability check in the selection process. So, it cannot be deemed as a good configuration. The same argument can be applied to the third scenario as well. In this scenario, the number of selected participants is greater than those methods which include the selection module, since there is no limitation for participant selection. However, the average suitability score in this scenario is the least, compared to other scenarios, since it does not consider the suitability score as a dominant factor. So the optimum configuration is to be selected from the second and forth scenarios (*S* and *SP* scenario). In both *S* and *SP* scenarios, the selection process is applied; but privacy is considered only in *SP*. The overall number of nominees and selected participants in both settings are approximately the same, but the overall suitability score in *S* scenario is slightly (about 0.009) higher than the suitability score in *SP*. Therefore, we conclude that *SP* configuration is the best for the privacy-aware systems and *S* configuration is

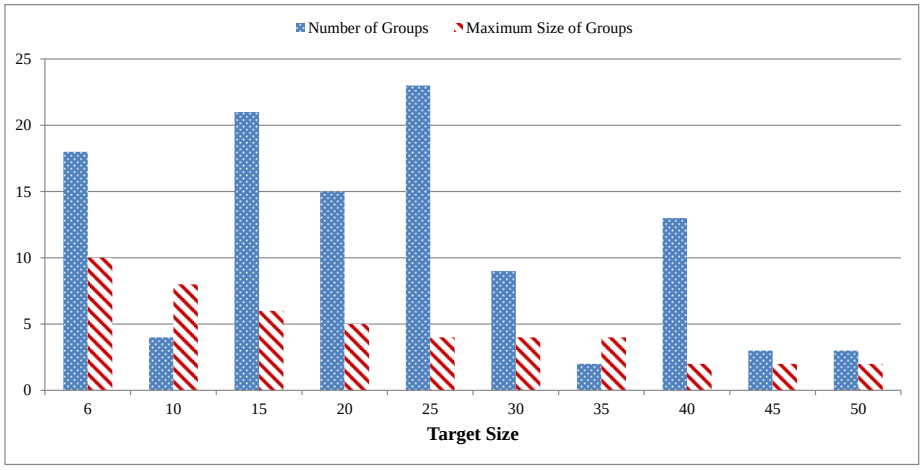


Figure 5.5: Evolution of number of groups and their maximum size according to the target size

appropriate for the rest.

5.4 Collusion Prevention Analysis

As mentioned in Section 5.1, we use Wikipedia Adminship Election dataset to investigate the performance of our proposed collusion prevention method. The dataset contains the information related to 2794 tasks. The average number of participants in these tasks equals to 40. In order to obtain reliable results, we considered the tasks with number of participants greater than the average as the sample data, and randomly select 100 tasks from these. We then tested our proposed method to identify any potential colluding group among the participants. As mentioned in Section 4.2, we considered three indicators for detecting potential collusion. More precisely, we assume that a group of participants is considered as a colluding group if it has at least th_1 members who all have collaborated in at least th_2 tasks in the past, and they have all submitted their contributions to these tasks in time window not larger than th_3.

In order to find the optimum value for th_2, we set an experiment in which, the target size (i.e., number of the tasks that the group members have collaborated in them in the past) is changed. For each target size, we measure the number of groups identified, together with their size. As can be seen in Figure 5.5, the maximum size of identified groups decreases by increasing the target size. This is rational since the probability of finding groups whose members have collaborated in greater number of tasks is smaller. We believe that the best setting is the one which results in the identification of largest groups to make a considerable impact. As derived from the figure, this situation is related to the case where the target size is 6. So, we set th_2 to be equal to 6. For the sake of simplicity, we assume that th_1 equals to th_2. th_3 has also been set to the average of all time windows for all the tasks. So, our method aims at identifying the set of candidate groups who have at least 6 members, have all collaborated in at least 6 tasks in the past and submitted their contributions within a specific time window.

In order to investigate the performance of our proposed collusion prevention method,

we first utilized the FIM technique to find the candidate groups among the participants. The outcome was the discovery of 18 candidate groups with at least 10 members. We then employed our collusion prevention method and identified 9 of these 18 groups as collusive. To evaluate the efficiency and accuracy of our method, we went through a set of statistical calculations. At first, we measured the ratio of the tasks targeted with the colluding groups. The result shows that 14% of the tasks were affected by these 9 colluding groups. This means that our collusion prevention method is able to prevent 14% of the tasks from being targeted by the colluders. We then calculated the success ratio of the tasks targeted by the colluding groups as well as all 100 tasks. By success ratio, we mean the ratio of the tasks that have resulted in a decision, to the total number of tasks (note that in the Wikipedia adminship election dataset, a task (an election) is successful if it results in the selection of the user as an administrator). We observed that overall success ratio of the tasks in our dataset is 71%. This ratio is 83% for the groups identified by our collusion detection method. This means that there is a high probability that the groups identified by our method are colluding groups, since their collaboration has resulted in a considerably high success ratio. This is a significant indication that the identified groups are much likely to be collusive.

To be brief, the results show that our proposed collusion prevention method is successful in preventing the formation of colluding groups among the selected participants with high accuracy. This is due to the correct selection of indicators as well as accurate settings of the thresholds.

6 Conclusions

In this paper, we proposed a participant selection framework for social participatory sensing systems. Our system leverages a customized random surfer to crawl the multi-hop friendship relations and identify well-suited nominees. The system then selects the final participants among the nominees. The selection is done in a way that it prevents the formation of a group of colluders within the set of selected participants. Simulations demonstrated that our scheme increases the number of participants who are reputable and well-suited to contribute. It also performs well in efficient and accurate detection of colluding groups.

Bibliography

[1] Collusion in peer-to-peer systems. In *Computer Networks*, volume 55, pages 3517 – 3532. 2011.

[2] Alekh Agarwal and Soumen Chakrabarti. Learning random walks to rank nodes in graphs. In *24th international conference on Machine learning*, pages 9–16, 2007.

[3] Akram Alkouz, Ernesto William De Luca, and Sahin Albayrak. Latent semantic social graph model for expert discovery in facebook. In *11th International Conference on Innovative Internet Community Systems (IICS), USA*, pages 128–138, 2011.

[4] Mohammad Allahbakhsh, Boualem Benatallah, Aleksandar Ignjatovic, Hamid Reza Motahari-Nezhad, Elisa Bertino, and Schahram Dustdar. Quality

control in crowdsourcing systems: Issues and directions. *Internet Computing, IEEE*, 17(2):76–81, 2013.

[5] Mohammad Allahbakhsh, Aleksandar Ignjatovic, Boualem Benatallah, Seyed-Mehdi-Reza Beheshti, Elisa Bertino, and Norman Foo. Reputation management in crowdsourcing systems. In *IEEE 8th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, pages 664–671, 2012.

[6] Mohammad Allahbakhsh, Aleksandar Ignjatovic, Boualem Benatallah, Seyed-Mehdi-Reza Beheshti, Elisa Bertino, and Norman Foo. Collusion detection in online rating systems. In *Proceedings of the 15th Asia Pacific Web Conference (APWeb 2013)*, pages 196–207, 2013.

[7] Mohammad Allahbakhsh, Aleksandar Ignjatovic, Boualem Benatallah, Seyed-Mehdi-Reza Beheshti, Norman Foo, and Elisa Bertino. Representation and querying of unfair evaluations in social rating systems. *Computers & Security*, 2013.

[8] Haleh Amintoosi and Salil Kanhere. Privacy-aware trust-based recruitment in social participatory sensing. In *Mobile and Ubiquitous Systems: Computing, Networking, and Services (MobiQuitous)*, in press.

[9] Haleh Amintoosi and Salil S. Kanhere. A trust-based recruitment framework for multi-hop social participatory sensing. In *9th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 266–273, 2013.

[10] Haleh Amintoosi and Salil S Kanhere. A trust framework for social participatory sensing systems. In *Mobile and Ubiquitous Systems: Computing, Networking, and Services (MobiQuitous)*, pages 237–249. 2013.

[11] Haleh Amintoosi and SalilS. Kanhere. A reputation framework for social participatory sensing systems. In *Mobile Networks and Applications (MONET)*, volume 19, pages 88–100. 2014.

[12] Lars Backstrom and Jure Leskovec. Supervised random walks: predicting and recommending links in social networks. In *4th ACM international conference on Web search and data mining*, pages 635–644, 2011.

[13] Jeffrey A Burke, Deborah Estrin, Mark Hansen, Andrew Parker, Nithya Ramanathan, Sasank Reddy, and Mani B Srivastava. Participatory sensing. In *WSW workshop, SenSys*, 2006.

[14] Jesse Chandler, Gabriele Paolacci, and Pam Mueller. Risks and rewards of crowdsourcing marketplaces. In *Handbook of Human Computation*, pages 377–392. 2013.

[15] Georgios Chatzimilioudis, Andreas Konstantinidis, Christos Laoudias, and Demetrios Zeinalipour-Yazti. Crowdsourcing with smartphones. In *IEEE Internet Computing*, volume 16, pages 36–44. 2012.

[16] Murat Demirbas, Murat Ali Bayir, Cuneyt Gurcan Akcora, Yavuz Selim Yilmaz, and Hakan Ferhatosmanoglu. Crowd-sourced sensing and collaboration using twitter. In *IEEE International Symposium on World of Wireless Mobile and Multimedia Networks (WoWMoM)*, pages 1–9, 2010.

[17] Anhai Doan, Raghu Ramakrishnan, and Alon Y Halevy. Crowdsourcing systems on the world-wide web. In *Communications of the ACM*, volume 54, pages 86–96. 2011.

[18] Anhai Doan, Raghu Ramakrishnan, and Alon Y Halevy. Crowdsourcing systems on the world-wide web. volume 54, pages 86–96. 2011.

[19] Akshay Dua, Nirupama Bulusu, Wu-Chang Feng, and Wen Hu. Towards trustworthy participatory sensing. In *Usenix Workshop on Hot Topics in Security (HotSec)*, 2009.

[20] Kate Ehrlich, Ching-Yung Lin, and Vicky Griffiths-Fisher. Searching for experts in the enterprise: combining text and social network analysis. In *ACM International conference on supporting groupwork, USA*, pages 117–126, 2007.

[21] Yuval Emek, Ron Karidi, Moshe Tennenholtz, and Aviv Zohar. Mechanisms for multi-level marketing. In *12th ACM conference on Electronic commerce*, pages 209–218, 2011.

[22] Gösta Grahne and Jianfei Zhu. Fast algorithms for frequent itemset mining using fp-trees. *Knowledge and Data Engineering, IEEE Transactions on*, 17(10):1347–1362, 2005.

[23] Kuan Lun Huang, Salil S Kanhere, and Wen Hu. On the need for a reputation system in mobile phone based sensing. *Ad Hoc Networks*, 2011.

[24] Kuan Lun Huang, Salil S Kanhere, and Wen Hu. A privacy-preserving reputation system for participatory sensing. In *e7th IEEE conference on Local Computer Networks (LCN)*, pages 10–18, 2012.

[25] Salil S Kanhere. Participatory sensing: Crowdsourcing data from mobile smartphones in urban spaces. In *Distributed Computing and Internet Technology*, pages 19–26. 2013.

[26] J. Kleinberg and P. Raghavan. Query incentive networks. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 132–141, 2005.

[27] Ioannis Konstas, Vassilios Stathopoulos, and Joemon M Jose. On social networks and collaborative recommendation. In *32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 195–202, 2009.

[28] Ioannis Krontiris and Felix C Freiling. Urban sensing through social networks: The tension between participation and privacy. In *International Tyrrhenian Workshop on Digital Communications (ITWDC), Italy*, 2010.

[29] Maciej Kurant, Minas Gjoka, Carter T Butts, and Athina Markopoulou. Walking on a graph with a magnifying glass: stratified sampling via weighted random walks. In *ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*, pages 281–292, 2011.

[30] Amy N Langville and Carl D Meyer. *Google's PageRank and beyond: The science of search engine rankings*. Princeton University Press, 2006.

[31] Matthieu Latapy and Pascal Pons. Computing communities in large networks using random walks. *arXiv preprint cond-mat/0412368*, 2004.

[32] Werner Van Leekwijck and Etienne E Kerre. Defuzzification: criteria and classification. *Fuzzy sets and systems*, 108(2):159–178, 1999.

[33] J. Leskovec, L.A. Adamic, and B.A. Huberman. The dynamics of viral marketing. *TWEB*, 1(1), 2007.

[34] Raph Levien and Alexander Aiken. Attack-resistant trust metrics for public key certification. In *7th USENIX Security Symposium*, pages 229–242, 1998.

[35] Q Lian et. al. An empirical study of collusion behavior in the maze p2p file-sharing system. In *27th International Conference on Distributed Computing Systems*, pages 56–. IEEE Computer Society, 2007.

[36] E Lim et. al. Detecting product review spammers using rating behaviors. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 939–948. ACM, 2010.

[37] László Lovász. Random walks on graphs: A survey. volume 2, pages 1–46. 1993.

[38] Thomas Malone, Robert Laubacher, and Chrysanthos Dellarocas. Harnessing crowds: Mapping the genome of collective intelligence. In *MIT Sloan Research Paper*. 2009.

[39] Arjun Mukherjee, Bing Liu, and Natalie Glance. Spotting fake reviewer groups in consumer reviews. In *Proceedings of the 21st international conference on World Wide Web*, WWW '12, pages 191–200, New York, NY, USA, 2012. ACM.

[40] San Murugesan. Understanding web 2.0. In *IT professional*, volume 9, pages 34–41. 2007.

[41] Swaprava Nath, Pankaj Dayama, Dinesh Garg, Yadati Narahari, and James Zou. Mechanism design for time critical and cost critical task execution via crowdsourcing. In *Internet and Network Economics*, pages 212–226. 2012.

[42] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: bringing order to the web. In *Technical report, Stanford Digital Library Technologies Project*. 1999.

[43] Sasank Reddy, Deborah Estrin, and Mani Srivastava. Recruitment framework for participatory sensing data collections. In *Pervasive Computing*, pages 138–155. 2010.

[44] Adam Sadilek, John Krumm, and Eric Horvitz. Crowdphysics: Planned and opportunistic crowdsourcing for physical tasks. volume 21, pages 125–620. 2013.

[45] Benjamin Satzger, Harald Psaier, Daniel Schall, and Schahram Dustdar. Auction-based crowdsourcing supporting skill management. *Information Systems*, 38(4):547 – 560, 2013.

[46] Frank Spitzer. *Principles of random walk*. Springer, 2001.

[47] Y.(. Sun and Y. Liu. Security of online reputation systems: The evolution of attacks and defenses. In *Signal Processing Magazine, IEEE*, volume 29, pages 87 –97. 2012.

[48] James Surowiecki. *The wisdom of crowds*. Random House Digital, Inc., 2005.

[49] Johan Ugander, Brian Karrer, Lars Backstrom, and Cameron Marlow. The anatomy of the facebook social graph. *arXiv preprint arXiv:1111.4503*, 2011.

[50] Maja Vukovic and Claudio Bartolini. Towards a research agenda for enterprise crowdsourcing. In *Leveraging Applications of Formal Methods, Verification, and Validation*, pages 425–434. 2010.

[51] Mohammad Hossien Yaghmaee, Mohammad Bagher Menhaj, and Hale Amintoosi. Design and performance evaluation of a fuzzy based traffic conditioner for differentiated services. In *Computer Networks*, volume 47, pages 847–869. Elsevier, 2005.

[52] Shuang-Hong Yang, Bo Long, Alex Smola, Narayanan Sadagopan, Zhaohui Zheng, and Hongyuan Zha. Like like alike: joint friendship and interest propagation in social networks. In *20th international conference on World wide web*, pages 537–546. ACM, 2011.

[53] Jing Zhang, Jie Tang, and Juanzi Li. Expert finding in a social network. In *Advances in Databases: Concepts, Systems and Applications*, pages 1066–1069. 2007.

[54] Elena Zheleva and Lise Getoor. Privacy in social networks: A survey. In *Social Network Data Analytics*, pages 277–306. 2011.