Master research Internship

Bibliographic report

Privacy-preserving crowdsourcing

Domain: Databases - Cryptography and Security

Author:
Joris Duguépéroux

Supervisor:
Tristan Allard
David Gross-Amblard
DRUID

Abstract: Simply defined, crowdsourcing can be seen as a way for an entity to use the web, generally through a specialized platform, in order to reach a large amount of people and to have them work on some task, against a reward. Although the so-called task is often very simple for a human (such as identifying tags on a pictures), it can also be far more complex, such as solving a real research problem as InnoCentive proposes. In the current literature, numerous works aim at optimizing one or many aspects of crowdsourcing, such as the computation of the reward, the assignation, or the worker's motivation. However, the privacy of the workers is most of the times simply ignored, though it could quickly be compromised through the wide and systematic collection of their preferences, works, or other personal information. The goal of this internship is precisely to remedy to this lack by (1) identifying the specific modules of typical crowdsourcing processes that endanger the privacy of workers, (2) proposing privacy-preserving versions of these modules, and (3) evaluating them thoroughly. These modules will allow to trade efficiency with privacy, and be integrated into a complete privacy-preserving crowdsourcing process.

Contents

1 Introduction

Our economy is currently evolving very fast into what some already call a “gig economy”[13] to refer to the fact that more and more jobs are proposed as a single task, instead of a hiring for a longer time. Crowdsourcing, which is part of this phenomenon, can be seen as a way for tasks providers (or “requesters”) to reach a large amount of workers in order to benefit from its wealth of skills against a reward which nature can vary a lot (monetary, interest of the worker, recognition, fun, or any other). Although a few platforms are very famous and used a lot, such as InnoCentive[1], Amazon Mechanical Turk[2] Crowdflower[3] or even Uber[4], the diversity of crowdsourcing extends far beyond these firms, in domains such as science[5] or home cleaning tasks[6].

To develop and optimize the benefits of these methods, many processes and algorithms have been proposed, which deal with various aspects of a very wide range of crowdsourcing problems, from characterizing the minimum wage for a worker to produce a good job, to taking into account the workers’ preferences when assigning tasks for instance.

However, abusive behaviors have also been observed, mostly from the platforms against the workers, as we can see with Uber, causing demonstrations in France [7], or tracking of journalists whithout permission [8]. These examples of real life issues rise concerns about the way workers (and in a more general view, participants) of crowdsourcing processes are protected from the platform or even from each other.

Privacy and cryptography techniques, which offer means for preventing such misuses, have been so far very sparsely used in crowdsourcing. Thus, the objective of this internship will be to study and extend existing approaches as far as possible to protect workers and tasks requesters from the platform, while maximizing benefits for both sides.

This report is organized as follows. First, we shall focus on the definition of crowdsourcing that we adopt, which will lead us to formalizing the various parts included in a typical crowdsourcing platform. Second, we will develop the state of the art on the way these modules are being treated in current literature. Finally, we shall review the few works that tackle privacy issues in the specific context of crowdsourcing.

2 Refining the Definition of Crowdsourcing

Although the notion of crowdsourcing is rather recent since it first appears in [20], many definitions can be found in the literature that try to capture some or all aspects that stand behind the notion of “crowdsourcing”, and to distinguish different categories for it.

2.1 General Definition of Crowdsourcing

In this section, we propose an informal definition for crowdsourcing, and underline the complexity and variations that it still relies on. Our definition is mainly based on the definition from [10].

[1] https://www.innocentive.com/

[2] https://www.mturk.com/mturk/welcome

[3] https://www.crowdflower.com/

[4] https://www.uber.com/

[5] http://fold.it/portal/index.php?q=

[6] https://www.handy.com/

[7] http://www.thelocal.fr/20161223/uber-drivers-blockade-paris-airports

[8] http://www.usatoday.com/story/tech/2014/11/19/uber-privacy-tracking/19285481/

In this definition and in the following, we will use the following terms to refer to the various protagonists in crowdsourcing: “requester” denotes the person, group of persons or entity that provide the task to be done, “worker” denotes the person, group of person or entity that do the task and “platform”, when it exists, will be used to designate the third party which makes contact between requesters and workers. This definition is formulated as follows : “Crowdsourcing is a type of participative online activity in which an individual, an institution, a non-profit organization, or company proposes to a group of individuals of varying knowledge, heterogeneity, and number, via a flexible open call, the voluntary undertaking of a task. The undertaking of the task, of variable complexity and modularity, and in which the crowd should participate bringing their work, money, knowledge and/or experience, always entails mutual benefit. The user will receive the satisfaction of a given type of need, be it economic, social recognition, self-esteem, or the development of individual skills, while the [requester] will obtain and utilize to their advantage that what the user has brought to the venture, whose form will depend on the type of activity undertaken.” However, [10] also proposes other way to assert whether or not an object can be classified as crowdsourcing. In particular, it answers such questions as “who works”, “why do they work”, “what kind of reward is expected”. This definition is therefore characterized by the following criteria [10]:

- There is a clearly defined crowd
- There exists a task with a clear goal
- The reward received by the workers is clear
- The requester is clearly identified
- The compensation to be received by the requester is clearly defined
- It is an online assigned process of participative type
- It uses an open call of variable extent
- It uses the Internet

For instance, this definition considers that a contribution on Wikipedia[1] will not be crowdsourcing, since it does not show any clear requester, the reward is not clearly defined, and it does not use an open call. On the opposite, InnoCentive[2] or the Amazon Mechanical Turk[3] do fit all these criteria, and can therefore be considered as crowdsourcing according to our definition.

This definition gives us precise criteria to determine whether or not a phenomenon can be considered as crowdsourcing. Furthermore, the obligation to have a clear requester and a clear reward (which prevent most open source project to be integrated to this definition) can be seen as an asset for us, since it settles a clear framework to our study.

However, these precise criteria still allow a very wide spectrum of crowdsourcing projects, and many distinction can be established to classify them. In the current literature, many papers ([4, 3, 15, 16, 43]) establish categories for crowdsourcing, so as to allow a selective treatment, and to treat them separately and more specifically. We propose here to review some of them shortly, and define these various categories with an appropriate vocabulary.

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

[2] https://www.innocentive.com/

[3] https://www.mturk.com/mturk/welcome

In [4], a distinction is done based on the objective of the crowdsourcing, by distinguishing crowdsourcing for routine tasks (for picture recognition for instance), contents (for comments, grades, opinions for instance), and for inventive activities (such as new designs or new ideas as in InnoCentive[2]).

Defining crowdsourcing as a system that "enlists a crowd of humans to help solve a problem defined by the system owners", [7] proposes a vision that includes open source world such as wikipedia[1] (contrarily to the one we use). Although it is not the main focus of this paper, it also proposes another division of workers in crowdsourcing, between four classes. The first one is called "slave", whose role is to execute the work that he is asked to do (for instance, the mechanical Turks of Amazon[3]) Then, the perspective providers, which generally reflects the case of agglomeration of grades to reach a better significant mean. The content providers, whose title is quite explicit, will add content for the requester. Finally, the component providers will have a more disparate role, which consists in various activities in the community: discussions on forums, contribution in projects, self-descriptions, or even moderation of contents. Such a role absent from some definition of crowdsourcing.

These two classifications of crowdsourcing, and of the role of the worker highlight the need for some clear assignation policy: all workers do not necessarily want to work in the same condition. Even worse, in skill-intensive cases (inventive activities for instance), the precise nature of the task matters, since it may determine whether the worker is able to do it or not.

Other distinctions are proposed in [15], according to the nature of the contribution : homogeneous/heterogeneous and emergent/non-emergent, which give birth to four categories, for each possible combination. An homogeneous crowdsourcing is defined by the fact that all contributions have the same value or not. For instance, in classification of pictures or mean computation, each contribution is weighted equally, while in any competitive work, which could be seen as an heterogeneous crowdsourcing, the winner is more heavily weighted. The distinction between emergent and non-emergent processes requires to determine whether the diversity of the crowd is a valuable asset or not. In classification of pictures for instance, since we expect every (honest) worker to give the same result, diversity is not required, whereas in competition, the diversity disappears behind the winner. On the opposite, when it comes to collaborative work, or to computing an mean of the overall appreciation of an item (say a movie for instance), diversity leads to better results.

Finally, the nature of the task is also separated into three categories in [43]: a collection, a competition or a collaborative work. These three kind of crowdsourcing have to be distinguished since they do not have the same properties in many regards, including privacy: for instance, in collaborative works, evaluation of workers will not have the same impact as in competitive works or collections, and on the opposite, while it is sometimes essential to make the accomplished work available for other workers in collaborative works, it is rarely the case in competitive works or collections.

These aspects, and the way they transform the results and its interpretation underlines that crowdsourcing requires some specific treatment of the received work: for instance, one can neither combine similarly collaborative works and competitive works, nor homogeneous and heterogeneous outputs.

Looking at such considerations, we can notice that various challenges rise from privacy concerns depending on the category we study : for instance, with "slaves", the issue may focus on the selection of workers, while in the perspective of content providers, the safety of the provided work may be important as well. For component providers, it might be very hard to draw general scheme of privacy since the systems can be subject to many variations.

2.2 Privacy-driven Functional Architecture of a Typical Crowdsourcing Platform

We propose here to expose the various problems we are to be confronted with while dealing with privacy in crowdsourcing such as defined above. Indeed, we can distinguish many interactions in this context, which involve three main entities, as mentioned in [43]: the requester, the platform and the worker. In this situation, the platform is often seen as a third party which works consists into intermediating between requester and worker. As explained in [3], this third party is often useful as it decreases the cost for the requester and the worker. Furthermore, we will see that its role can encompass other activities related to crowdsourcing. Given these three parties, [5] explains that the crowdsourcing process can be divided into the following modules: Task Management, Incentive Design, Crowd Filtering, Profile Management, Task Assignment and Result Management. In the scope of our study, and using the previous definition, we propose a close classification which is driven by privacy concerns. This classification will rely on two inputs: tasks and profiles, and on three modules: *Incentive Design*, *Task Assignment*, and *Crowd Filtering and Quality Control*.

2.2.1 Input: Task and Profile

In crowdsourcing, a task consists in a job which can be accomplished by one or many workers, which complexity can vary a lot (from picture recognition to research topics). Task management, in a general acceptation, covers the creation of a task, with both its transcription from a real-life problem into a well-formalized problem that can be crowdsourced and the way tasks are distributed among the workers (in parallel, sequentially, through a divide and conquer point of view...). In the following, we will suppose the Task Management to be done by the requester. Although this role is not trivial, it is not our point to develop advice or methods to do it.

The profile of a worker denotes the data possessed by the platform on the worker, which can includes preferences, personalization or history of works, email address, pseudonym, or real-life name for instance. The management of these profiles does impact the way the crowdsourcing system is perceived, and the way it proceeds. For instance, it is not possible to do the same things if we know the task and success history of workers, their preferences, both, or none. The use of these data, and the algorithms behind them are all the more essential to incentive, recommendations and assignments. Furthermore, this role and the choice to make some informations public, partially public (only for requesters, or other workers for instance) or strictly private may also be determinant for collaborative works, or in the ways the requester can select some specific profiles instead of others.

2.2.2 Module 1: Incentive Design

The incentive design defines the way the platform and the requester use to attract as many workers as possible. This idea mainly relies on two variations : extrinsic motivation (mainly monetary) or intrinsic motivations (based on fun or interest of the task for instance) ([28, 43, 5]). However, to obtain an interesting design, personalization and recommendation are often used thanks to various data about the user: we often attract more workers with higher budgets, and the money desired may vary according to the preferences of each worker. Hence, various personalisations are used to optimize the intrinsic interest of the task, or to lower the price that cannot be achieved without prior knowledge.

The Incentive Design module relies on techniques that make use of sensitive information, such as preferences in topics, skills or even expected wages, contained in the workers' profiles. It is thus necessary to perform it in a privacy-preserving manner.

2.2.3 Module 2: Task Assignment

The "Task Assignment" module consists into establishing an assignation between tasks and workers, and to optimize it such that tasks are correctly dealt with, and workers are as satisfied as possible.

This module leads to issues to the previous one since the aim is to select the most appropriate workers for a task, according to informations on both workers, tasks, and possibly requesters. Hence, the preferences of the worker or the nature of the task (number of workers required, complexity of the task, required skills) must be taken into account, and so are the preferences of the requester (for instance, for some surveys, the requester may want to focus on a specific subgroup of the workers, or prefer to promote some criteria which are not related to the nature of the task).

If this module has similarities with the "Incentive Design" module through the use of personal informations, it also has similar privacy issues. Although the focus is generally not done on the same informations within the workers' profiles, targeting workers according to their skills, preferences, history or other personal informations remains an issue that has to be dealt with.

2.2.4 Module 3: Crowd Filtering and Quality Control

Filtering the crowd consists in avoiding to assign tasks to workers that can not deal with them correctly. If a naive approach is to make a fully open call, and to consider only honest and skillful workers, this appears to be insufficient both when the task is simple and when the task is complex [5]. When the task is simple, it is often necessary to avoid abusers, who may want to earn a reward easily by automatizing the task, even if it implies giving wrong answers. On the opposite, for a complex task, selecting the users according to their skills is also necessary in order to avoid bad results. Hence, a filtering of the crowd so as to avoid useless or counterproductive workers may rely on various mechanisms, from reputation or trust systems to skill measures [5, 1].

The quality control denotes the ways to ensure that the result given for a task is correct. This control and the issues it brings are very close to the crowd filtering, since the aim remains to avoid bad quality results. Majority voting, aggregation of results and other techniques are widely studied in current literature to answer these issues. To this aspect, we can also add the fact that feedbacks provided by the requester to the worker are important and often appreciated by the workers [1]. Peer evaluation or reviewing, mentioned in [16], are also interesting both to allow some progression of the worker, and to enable various levels of efficiency to be detected and allocated according to the complexity of the task.

In both of these aspects, privacy concerns are risen. First, when we use methods to discriminate "good" workers from "bad" workers, or even to create them profiles, but also in the reviews and evaluations they received, that could legitimately be considered as private to some resort. Reciprocally, evaluating workers and the accomplishment of tasks with completely opaque and anonymised profiles appears not to be trivial.

3 State-of-the-art on Unprotected Crowdsourcing Algorithms

We propose here a non-exhaustive review of methods that are currently used or proposed to deal with the previous points. We will not develop the profile management aspect due to the fact that there is currently no literature dealing with this specific point as it is often included in other aspects of crowdsourcing. We do not attend to present a detailed view of the algorithms, but to highlight the main advantage and drawbacks of these methods.

3.1 Incentive Design

As explained in [5], there are two types of incentive designs: monetary and non-monetary. However, since they are often seen as the most efficient to motivate workers ([40, 17, 19, 26]), we focus here on monetary incentives. In this context, various methods have been proposed and reviewed in [5], with their respective advantages and drawbacks.

Some methods propose to take into account the reputation, and the way workers have behaved in previous tasks in order to prevent issues. In [42], a method is proposed that considers paying the worker before the task is accomplished. The common issue with this method is that the worker is likely to take the task less seriously than if he hadn't been paid. This problem is avoided by introducing a variation of wages according to the reputation of the worker. However, this systems has an other drawback, since it considers all tasks as similarly complex, and all workers as equally skilled.

An other model is proposed in [39], that takes into account the diversity of skills within workers. In this scheme, the reward is divided among workers whose work has been approved by the requester. Here, the reputation is used in the assignation phase, where it impacts the probability that a worker is assigned to some tasks.

Other metrics to set the best price for a work are also proposed in [17] and [12], that do not take reputation into account, but these prices are mostly based on the nature of the work (time required, complexity), so that they do not really take into account the expertise of the worker.

The interest of taking reputation into account is studied in [29], which proposes crowdsourcing on IQ tests with various parameters, including variation of the retribution and reputation variations. Although this study has some serious limitations, such as the absence of diversity in tasks, and the modeling of reputation by a single method (a maximum failure threshold not to reach), the results are quite interesting. In this study, it is concluded that taking reputation into account both rises the mean IQ of the crowd, and decreases participations. In a similar idea, rising the retribution first rises the mean IQ, but decreases it after some point, while the participation doesn't cease to increase.

3.2 Task Assignment

The Task Assignment module is probably the most widely studied part of crowdsourcing. In [5], a distinction is made between offline and online task assignment. In the case of offline task assignment, we suppose that we have informations on both tasks and workers. The nature of these informations may vary according to the model, but the algorithms of offline assignment focus on optimizing the assignation between users and tasks, so as to satisfy each party as much as possible. On the opposite, in online task assignment, there is no prior knowledge on the workers: the aim is to focus on acquiring it as fast as possible. To do so, the idea is to use a budget of questions or tasks, and to use it to characterize a user as precisely as possible. If the workers remain in the systems, this online task assignment introduces the offline task assignment. We will here focus on the so-called "offline" task assignment problems, since prior knowledge is often available on users or quickly acquired. We refer the interested reader to [5] for further details about online task assignment problems.

As an offline method, [32] proposes to associate to each user a weighted taxonomy of skills, and to each task a specific skill (which could be justified by the fact that in crowdsourcing, a task are often simple). In this context, the use of Hungarian method [30] to establish a bi-partite matching bring the optimal result, but has a cubic cost in the number of workers and tasks. Hence, two heuristics are proposed in [32] that reach results close to the optimal solution, with reasonable computation time. However, the limitation of a task to a single skill and the fact that no mention is done to the price according to the complexity of the task may introduce some difficulties to use it in real-life settings.

Using both the history of success and the history of research of the workers, [41] proposes an approach which aims at proposing the most appealing tasks to each users, by using factorisation of preference matrices, similarly to some recommendation systems. However, as the tasks are likely to change along the time, the use of recommendation systems based on similarities between workers and researches is not trivial to apply in real-life cases.

Although it is considered as "offline" by the review in [5], the algorithms presented in [2] consider the workers to come one by one (but we consider prior knowledge on them, in particular, the tasks they prefer, and the bid the want). Considering the min-cost flow as a subroutine, this paper proposes a solution based on the modeling of workers and tasks as opposite vertices of a bipartite graph, and the edge assimilated to costs. However, as stated in the paper, the upper bound of this algorithm "crucially depends on the large market assumption where we assume that the bids of the workers are small compared to the budget", which is not always the case in real-life scenarios.

3.3 Crowd Filtering and Quality Control

In this part, we will focus on the filtering, and the various ways that have been developed to ensure that none of the worker can cheat on the system. In particular, we will not review the various ways that exist to optimize the results out of the outputs since at this point, the platform is dispensable so that the requester can handle the management of outputs on her own.

Cheater detection in general is a hard issue. In most cases, this procedure is done by using questions or tasks which result is already known, to detect whether a worker answers correctly or not[18]. However, it is not always possible to have such clear questions (in complex domains, such as translations for instances), and even when we have, it may be hard to distinguish an unskilled worker who is honest from a cheater. This kind of approaches is more developed in [27], which also takes into account the difficulty of these trap questions.

Other methods such as [38, 25], are based on the idea of proposing the same task to several workers, in order to detect diverging ones. An other approach of this method proposed in [11] proposes an adaptive framework in order to take into account similarities between tasks to create personalized profiles of workers. Hence, each worker is here characterized by a probability of success on various tasks. However, these kind of methods relying on majority voting are rarely robust against a sufficient number of coordinate spammers (which would give the same answer to same questions).

In [31], an estimation of the expertise of each worker is computed, so that contribution of expert workers can be more heavily weighted. However, this score is unique, so that it does not take into account the fact that workers can have various expertise levels on different tasks.

Following a trend first explored in [6], the authors of [35] propose an evaluation of the skill of a worker, given her past collaborative works. In this model, we consider that tasks are binary labelling which value is not known. The main idea is to determine the result of a task by using a weighted mean of all answers (or workers), and modifying the weights through successive iterations until convergence. The weights of workers are to be considered as an evaluation of expertise. This evaluation takes into account the various tasks previously done by the worker, but also the teams she has been involved with. Although this work focuses on the estimation of a single skill, if all skills associated with a task are independent from each other, this approach can easily be extended to more complex structures.

3.4 Related Issues

Some issues are very related to the previous ones but are almost ignored from the crowdsourcing literature. We propose here to mention them, and to give some of the reasons why they appear so little in current literature.

First, the "college admissions problem". First solved in [14] with the so-called Gale-Shapley algorithm, and still used in many cases such as college assignation for students, this problem is a specific assignation problem, in which two sets (namely students and colleges for instance) formulate their preference order. The colleges can accept students up to their respective limits. Simply said, stability is reached (and the problem is solved) when there is no student S and college C such that S is either unmatched or prefers C over its current assignation, and C has a free slot, or would prefer S over one of the students it has been assigned. An extension of this problem is developed and solved in [36], which takes into account couples which would prefer to stay together for instance. However, these problems both states clear lists of preferences, which are not possible in a crowdsourcing context. Even if, in some cases, such a list could be established on the worker side, it is not realistic to suppose that each tasks has an ordered list of preference of workers. Furthermore, these assignment issues also rise problems of complexity, since the simplest variation of this issue is quadratic in the number of entities.

Recommendation issues are also related, and, to the best of our knowledge, only little explored so far in the context of crowdsourcing. The main issue with the use of recommendation systems is the fact that there are only little way to know whether a worker appreciates the task she did, since the success/failure is not necessarily related to the appreciation. Some feedback from the worker on the task, or, as mentioned in [41], the history of research, could help identifying the interests of the workers. An other issue is that many tasks are also to be completed in a limited amount of time. Hence, in the absence of similarity measure between tasks in general case, the presence of history is hard to exploit on data that is perpetually renewed.

4 State-of-the-art on Privacy-Preserving Crowdsourcing Approaches

In this section, we review the main existing privacy-preserving crowdsourcing techniques. First, we will introduce the current standard definition used in privacy preservation, then, the main work produced on general crowdsourcing approaches. Third, we survey the works tackling the specific issues oriented to geospatial crowdsourcing. Finally, we develop related works, with solutions we might draw some inspiration from.

4.1 Pre-Requisite: Differential Privacy

Defined in [9], *Differential Privacy* is the state-of-the-art standard when it comes to data privacy and sanitizing. The initial Differential Privacy model allows interactions with a database only through aggregations of results (means, sums, counts...). The main idea is to add noise in the query results in order to preserve privacy. Formally stated, the definition is formulated in Definition 1 (from [9]).

Definition 1 *A randomized function K gives ϵ-differentially privacy if for all data sets D_1 and D_2 differing on at most one element, and all $S \subseteq Range(K)$,*

$$Pr[K(D_1) \in S] \leq exp(\epsilon) \times Pr[K(D_2) \in S]$$

Informally, the parameter ϵ can be seen as a privacy budget, with small values corresponding to high privacy. To achieve ϵ-differential privacy, the amount of noise required is proportional to the *sensitivity* of the query, defined in 2.

Definition 2 *For $f : D \to R^d$, the* L1-sensitivity *of f is*

$$\Delta f = \max_{D_1, D_2} ||f(D_1) - f(D_2)||_1$$

for all D_1, D_2 differing in at most one element.

Among the interesting properties of differential privacy, we will only mention the properties 1 and 2.

Theorem 1 *Let A_i be a set of analyses such that each provides ϵ_i-differential privacy. Then, running in sequence all analyses A_i provides $(\sum_i \epsilon_i)$-differential privacy.*

Theorem 2 *If D_i are disjoint subsets of the original database, and A_i is a set of analyses each providing ϵ-differential privacy, then applying each analysis A_i on partition D_i provides $max(\epsilon_i)$-differential privacy.*

4.2 General Privacy-Preserving Crowdsourcing

To the best of our knowledge, the focus on privacy-preserving issues in general case crowdsourcing has not been widely studied. The main contributions to this topic are from the PhD [22], and two papers from the same authors, [24, 23], which focus on two of the three main points of [22]: worker privacy preservation, privacy-preserving task assignment and instance privacy preservation.

4.2.1 Worker Privacy Preservation

In order to preserve the privacy of workers while maintaining a good quality control, [23] proposes a scheme which aims at maintaining good results while keeping workers' informations safe. The basis for quality control which is explored is the one exposed in [6] (see section 3.3).

Here, weights used are related to the workers, since it reflects the probability that a worker gives the good answer to a question. It can also be seen as a grade, and one of the aim is to protect this value from any other party. Two main points can be pointed out in the contribution of [23]. First, the use of as much decentralization as possible allows some part of the mean and the maximization to be computed on workers' devices, which avoids some issues. Then, the use of additive homomorphic cryptography[34] allows sums to be computed without disclosing the terms of the sum.

Although some issues are noticed, such as few more approximations compared to the initial method, or higher computation time which remain reasonable for low number of workers (the computation takes $789.3ms$ for 20 workers), the main drawback that can be noticed comes from the initial method. Indeed, by considering tasks which answers are not previously established and taking as a result a weighted mean, this approach is still weak against coordinated spammers giving similar but random answers.

4.2.2 Privacy-Preserving Task Assignment

In the focus done on assignment, [22] proposes a scheme that takes into account both features of tasks, and associated skills of worker, and tries to protect both of them. Noticing that this problem can be reduced to a maximum flow problem, their methods proceeds into two steps: the construction of the assignment network in which the maximum flow corresponds to an optimal assignment, and then, the computation of the maximum flow. These two steps are done using the Paillier cryptosystem [34], so that the platforms learns the optimal assignment only, and none of the other entities learn anything. The main asset of this method is that it preserves both privacy and the optimality of the solution, without any perturbation. However, the main drawback of this method is not to be ignored. Indeed, the price for using these cryptographic tools is the computation time. With the original settings, the authors report that this approach takes approximately 1.34×10^4 centuries, for as little as 100 workers and as many tasks. After few acceleration techniques, the result is far more acceptable, but still unusable in real-life application, since it takes 4 to 5 days of computation (again, for 100 workers and as many tasks).

4.2.3 Instance Privacy Preservation

The issue highlighted by [24] is related to the disclosure of sensitive data through tasks. For instance, in picture recognition, the presence of faces or names could introduce privacy concerns. Therefore, a framework to quantify the preservation of privacy and of an example of perturbation are introduced. The framework consists in defining, for each application, a privacy definition. With this definition, empirical estimation is made (by humans) of the privacy leakage, and of the information loss. Using this framework, they propose to evaluate a method to protect privacy of people on pictures to be labelled in crowdsourcing. In this example, the aim of the task was to determine whether a human face was on a picture, and the privacy was considered to be threatened when both a face and an action are identified.

Their method, which basically consists in using clipped pictures, is not very important to our study, but the overall framework is interesting since it introduces the fact that in some cases, general privacy definition are not feasible or suitable. In these cases, allowing a more general view, with customizable definition, might be a good option.

4.3 Spatial Crowdsourcing Techniques

Spatial crowdsourcing is a specific kind of crowdsourcing that require spatio-temporal data on users, for some specific tasks both spatially and temporally localized. Exploring the privacy issues that occurs when participating to such crowdsourcing processes, [37] focuses on protecting both the location and the identity of workers during the assignation phase. After that, the idea is that some specific channel can be opened to enable communication, with other kind of protection that already exist if needed (TOR for instance).

The interesting point here is that neither the task requester nor the platform are here considered as trust reliable toward the privacy concerns. To allow such a scheme, the locations are revealed to an other third party, independent from the platform, for instance, the cellular service provider, that already has access to the worker locations (e.g., through cell tower triangulation). Hence, no information is revealed that was not already disclosed. The role of this service provider will be to perturb the workers' locations in a differentially private manner before sending them to the platform.

In this optic, the main idea is to add random Laplace noise into the counting of workers for each possible zone. This procedure exploits the theorem 2 to ensure that the composition of these zones will maintain private enough results. With this method, workers can be created that do not really exist (i.e., positive noise), and others may be ignored although they could have done a task (i.e., negative noise). Therefore, the main issue will be to obtain a correct assignment with a probability as great as possible. We do not explain here as they are specific to spatial crowdsourcing, which is not our main focus. The evaluation of this method, based on various measures for the success rate of a query (assignment success rate, worker travel distance, and system overhead) show that this scheme is affordable in real-life settings despite the information loss due to the differentially-private perturbation.

4.4 Related Work

Privacy-Preserving Data Publishing: As explained in [21], publishing an exact matching between goods and agents (or, in our case, tasks and workers), while maintaining differential privacy (see 1) is impossible. Indeed, this allocation would rely on the preference order of each participant, which are precisely what we want to be hide. The authors propose to relax the adversarial assumptions of differential privacy model with the following condition "for any agent i, no coalition of agents excluding i should be able to learn about the valuation function of agent i". Contrary to the original differential privacy, the possible attacks do not come from requests but from coalition of attackers. In this variation, allocations are no longer published, but each worker knows which task is the most suitable. However, this approach, which numerous theoretical limits are underlined (such the necessity of a small number of identical copies of each good), doesn't seem to be adapted to a crowdsourcing context. Indeed, even in the cases where the hypothesis are respected, the protection of workers against coalitions of other workers doesn't seem to be the main privacy issue. The fact that all these data and preferences transit through an identical third party is more likely a weakness, and this aspect isn't dealt with in [21].

Recommendation systems: Since most recommender systems are based on the data of other users, the results obtained with one or many accounts with various preferences may be used to infer some specific data on these other users. Many works propose privacy protecting recommendation systems. We decide to focus specifically on [33], since it proposes an approach which consists into adapting existing recommendation systems into privacy protecting ones. The main idea was to show that it is possible to adapt relatively simple recommender systems into more complex ones, by adding noise in counts. However, this solution is not adapted without trusted third party, and the limitations of recommender systems for crowdsourcing systems is explained in 3.4.

Secure Stable Matching: This protection against a third party is dealt with in [8], which proposes algorithms to ensure protection against a third party for stable marriage problem. Although these issues are not in our scope, as underlined in 3.4, it is interesting to notice what exists in fields strongly related with ours. In this approach, the main idea is to use two servers and to spread the preferences of users between them, either by splitting the data, or by XOR-sharing, and before proceeding to the computation with new data structure they introduce. As said in the paper, the resulting protocol is efficient enough to handle about 30000 of participants and similar number of residencies slots within about 18 hours. Cumulating the cost of their structure and the cost of their algorithm, the overall asymptotic complexity reaches $O(n^2 * log^3 n)$ for the Gale-Shapley algorithm. This cost is even higher for more general variations mentioned in 3.4. We can notice that, if this cost is interesting for occasional computations, it is not possible to use it in real-life situations for the case of crowdsourcing.

5 Conclusion

In this report, we have introduced the notion of crowdsourcing and identified the privacy concerns that are related. In order to allow a general view of what matters in crowdsourcing, from a privacy point a view, we also established a division of crowdsourcing into three main modules to focus on. The state-of-the-art about these three modules, and the current ways to deal with them, have been developed first from an unprotected view, with state-of-the-art algorithm focusing on efficiency rather than protection of the parties. Then, we also studied the part of the literature dedicated to privacy-aware crowdsourcing. In this latter, many aspects have been noticed to require further improvement.

Indeed, to the best of our knowledge, there are for instance no study on privacy-aware incentive design, nor efficient task assignment algorithms that does not disclose private informations at least to a third party. Although incentive design is very close to task assignment in many regards, some specificities are still to be noticed, since many ways of making incentives have been studied. It might be that no such privacy aware incentive could be designed, since incentive and gratification require clear identification of workers, but the same could be said about task assignment, which has been studied with various success. Privacy aspects of task assignments are also to be improved: current studies either focus on close but dissimilar problems, or use cryptographic protocols that make the computation impossible to use in real-life applications.

Finally, although quality control is handled in [23], some improvement could still be studied bring some further robustness against large numbers of attackers, by adapting approaches that exist in unprotected contexts.

Hence, we can see that there are still many aspects to improve to reach a whole privacy-protective crowdsourcing design. In this internship, we propose a focus on task assignment, by using differential privacy techniques along with cryptographic protocols to relax the security model while maintaining high protection standards. Incentive designs, could also be studied to determine precisely to what extent a privacy-preserving incentives and remuneration is feasible and related to task assignment or not.

References

[1] Sihem Amer-Yahia and Senjuti Basu Roy. Toward worker-centric crowdsourcing. *IEEE Data Eng. Bull.*, 39(4):3–13, 2016.

[2] Sepehr Assadi, Justin Hsu, and Shahin Jabbari. Online assignment of heterogeneous tasks in crowdsourcing markets. In *Third AAAI Conference on Human Computation and Crowdsourcing*, 2015.

[3] Thierry Burger-Helmchen and Julien Pénin. The limits of crowdsourcing inventive activities: What do transaction cost theory and the evolutionary theories of the firm teach us. In *Workshop on Open Source Innovation, Strasbourg, France*, pages 1–26, 2010.

[4] Thierry Burger-Helmchen and Julien Pénin. Crowdsourcing: définition, enjeux, typologie. *Management & Avenir*, (1):254–269, 2011.

[5] Anand Inasu Chittilappilly, Lei Chen, and Sihem Amer-Yahia. A survey of general-purpose crowdsourcing techniques. *IEEE Trans. Knowl. Data Eng.*, 28(9):2246–2266, 2016.

[6] Alexander Philip Dawid and Allan M Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Applied statistics*, pages 20–28, 1979.

[7] Anhai Doan, Raghu Ramakrishnan, and Alon Y Halevy. Crowdsourcing systems on the world-wide web. *Communications of the ACM*, 54(4):86–96, 2011.

[8] Jack Doerner, David Evans, et al. Secure stable matching at scale. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1602–1613. ACM, 2016.

[9] Cynthia Dwork. Differential privacy. In *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II*, pages 1–12, 2006.

[10] Enrique Estellés-Arolas and Fernando González-Ladrón-De-Guevara. Towards an integrated crowdsourcing definition. *Journal of Information science*, 38(2):189–200, 2012.

[11] Ju Fan, Guoliang Li, Beng Chin Ooi, Kian-lee Tan, and Jianhua Feng. icrowd: An adaptive crowdsourcing framework. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1015–1030. ACM, 2015.

[12] Siamak Faradani, Björn Hartmann, and Panagiotis G Ipeirotis. What's the right price? pricing tasks for finishing on time. *Human computation*, 11:11, 2011.

[13] Gerald Friedman. Workers without employers: shadow corporations and the rise of the gig economy. *Review of Keynesian Economics*, (2):171–188, 2014.

[14] David Gale and Lloyd S Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962.

[15] David Geiger, Michael Rosemann, Erwin Fielt, and Martin Schader. Crowdsourcing information systems - definition, typology, and design. In *Proceedings of the International Conference on Information Systems, ICIS 2012, Orlando, Florida, USA, December 16-19, 2012*, 2012.

[16] David Geiger, Stefan Seedorf, Thimo Schulze, Robert C Nickerson, and Martin Schader. Managing the crowd: Towards a taxonomy of crowdsourcing processes. In *AMCIS*, 2011.

[17] Christopher Harris. You're hired! an examination of crowdsourcing incentive models in human resource tasks. In *Proceedings of the Workshop on Crowdsourcing for Search and Data Mining (CSDM) at the Fourth ACM International Conference on Web Search and Data Mining (WSDM)*, pages 15–18, 2011.

[18] Kurtis Heimerl, Brian Gawalt, Kuang Chen, Tapan Parikh, and Björn Hartmann. Communitysourcing: engaging local crowds to perform expert work via physical kiosks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1539–1548. ACM, 2012.

[19] John Joseph Horton and Lydia B Chilton. The labor economics of paid crowdsourcing. In *Proceedings of the 11th ACM conference on Electronic commerce*, pages 209–218. ACM, 2010.

[20] Jeff Howe. The rise of crowdsourcing. *Wired magazine*, 14(6):1–4, 2006.

[21] Justin Hsu, Zhiyi Huang, Aaron Roth, Tim Roughgarden, and Zhiwei Steven Wu. Private matchings and allocations. In *Proceedings of the Forty-sixth Annual ACM Symposium on Theory of Computing*, STOC '14, pages 21–30, New York, NY, USA, 2014. ACM.

[22] Hiroshi Kajino. *Privacy-Preserving Crowdsourcing*. PhD thesis, 2015.

[23] Hiroshi Kajino, Hiromi Arai, and Hisashi Kashima. Preserving worker privacy in crowdsourcing. *Data Mining and Knowledge Discovery*, 28(5-6):1314–1335, 2014.

[24] Hiroshi Kajino, Yukino Baba, and Hisashi Kashima. Instance-privacy preserving crowdsourcing. In *Second AAAI Conference on Human Computation and Crowdsourcing*, 2014.

[25] David R Karger, Sewoong Oh, and Devavrat Shah. Iterative learning for reliable crowdsourcing systems. In *Advances in neural information processing systems*, pages 1953–1961, 2011.

[26] Gabriella Kazai. An exploration of the influence that task parameters have on the performance of crowds. *Proceedings of the CrowdConf*, 2010, 2010.

[27] Faiza Khan Khattak and Ansaf Salleb-Aouissi. Quality control of crowd labeling through expert evaluation. In *Proceedings of the NIPS 2nd Workshop on Computational Social Science and the Wisdom of Crowds*, 2011.

[28] Frank Kleemann, G Günter Voß, and Kerstin Rieder. Un (der) paid innovators: The commercial utiliza-tion of consumer work through crowdsourcing. *Science, technology & innovation Studies*, 4(1):PP–5, 2008.

[29] Michal Kosinski, Yoram Bachrach, Gjergji Kasneci, Jurgen Van-Gael, and Thore Graepel. Crowd iq: Measuring the intelligence of crowdsourcing platforms. In *Proceedings of the 4th Annual ACM Web Science Conference*, pages 151–160. ACM, 2012.

[30] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[31] Zhiquan Liu, Luo Luo, and Wu-Jun Li. Robust crowdsourced learning. In *Big Data, 2013 IEEE International Conference on*, pages 338–343. IEEE, 2013.

[32] Panagiotis Mavridis, David Gross-Amblard, and Zoltán Miklós. Using hierarchical skills for optimized task assignment in knowledge-intensive crowdsourcing. In *Proceedings of the 25th International Conference on World Wide Web*, pages 843–853. International World Wide Web Conferences Steering Committee, 2016.

[33] Frank McSherry and Ilya Mironov. Differentially private recommender systems: building privacy into the net. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 627–636. ACM, 2009.

[34] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 223–238. Springer, 1999.

[35] Habibur Rahman, Saravanan Thirumuruganathan, Senjuti Basu Roy, Sihem Amer-Yahia, and Gautam Das. Worker skill estimation in team-based tasks. *Proceedings of the VLDB Endowment*, 8(11):1142–1153, 2015.

[36] Alvin E Roth and Elliott Peranson. The redesign of the matching market for american physicians: Some engineering aspects of economic design. Technical report, National bureau of economic research, 1999.

[37] Hien To, Gabriel Ghinita, and Cyrus Shahabi. A framework for protecting worker location privacy in spatial crowdsourcing. *Proceedings of the VLDB Endowment*, 7(10):919–930, 2014.

[38] Jiannan Wang, Tim Kraska, Michael J Franklin, and Jianhua Feng. Crowder: Crowdsourcing entity resolution. *Proceedings of the VLDB Endowment*, 5(11):1483–1494, 2012.

[39] Hong Xie, John CS Lui, Joe Wenjie Jiang, and Wei Chen. Incentive mechanism and protocol design for crowdsourcing systems. In *Communication, Control, and Computing (Allerton), 2014 52nd Annual Allerton Conference on*, pages 140–147. IEEE, 2014.

[40] Guo Xintong, Wang Hongzhi, Yangqiu Song, and Gao Hong. Brief survey of crowdsourcing for data mining. *Expert Systems with Applications*, 41(17):7987–7994, 2014.

[41] Man-Ching Yuen, Irwin King, and Kwong-Sak Leung. Task recommendation in crowdsourcing systems. In *Proceedings of the First International Workshop on Crowdsourcing and Data Mining*, pages 22–26. ACM, 2012.

[42] Yu Zhang and Mihaela van der Schaar. Reputation-based incentive protocols in crowdsourcing applications. In *INFOCOM, 2012 Proceedings IEEE*, pages 2140–2148. IEEE, 2012.

[43] Yuxiang Zhao and Qinghua Zhu. Evaluation on crowdsourcing research: Current status and future direction. *Information Systems Frontiers*, 16(3):417–434, 2014.