Secure and Scalable Statistical Computation of Questionnaire Data in R

Kassaye Yitbarek Yigzaw, Antonis Michalas, and Johan Gustav Bellika

***Abstract*—Collecting data via a questionnaire and analyzing them while preserving respondents' privacy may increase the number of respondents and the truthfulness of their responses. It may also reduce the systematic differences between respondents and non-respondents. In this paper, we propose a privacy-preserving method for collecting and analyzing survey responses using secure multi-party computation (SMC). The method is secure under the semi-honest adversarial model.**

The proposed method computes a wide variety of statistics. Total and stratified statistical counts are computed using the secure protocols developed in this paper. Then, additional statistics, such as a contingency table, a chi-square test, an odds ratio, and logistic regression, are computed within the R statistical environment using the statistical counts as building blocks.

The method was evaluated on a questionnaire dataset of 3,158 respondents sampled for a medical study and simulated questionnaire datasets of up to 50,000 respondents. The computation time for the statistical analyses linearly scales as the number of respondents increases. The results show that the method is efficient and scalable for practical use. It can also be used for other applications in which categorical data are collected.

***Index Terms*—Bloom Filter, Privacy, Questionnaire, Statistical Analysis, Secure Multi-Party Computation, Secret Sharing**

I. Introduction

Surveys are a commonly used tool in many research areas. For example, public health researchers use surveys to study the health conditions of a given population, as well as to measure public opinion regarding the healthcare services. In addition to the health data collected at health institutions, individuals collect data, such as health and physical activity data, using mobile applications and sensors. These data have high potential for public health and clinical research [1].

In this paper, we consider collecting data via a questionnaire as an example of research data collected from individuals. The reliability of the statistical analyses of questionnaire data depends on the availability of a sufficient number of participants. The analyses also depend on the representativeness of the participants and the truthfulness of the data provided by the participants. However, disclosure of sensitive or private information to a third party may discourage participation and reduce the response rate. Nonresponse may induce nonresponse bias as a result of systematic differences between individuals who participate and individuals who do not [2]. Privacy risks may also compel participants to provide inaccurate responses. Therefore, privacy preserving data collection and analyses may promote participation and increase the accuracy of the data [3].

A simple method for collecting data via a questionnaire involves a trusted third-party (TTP) that collects the data from participants and de-identifies the collected dataset before disclosing the data to the data analyst (researcher). SurveyMonkey [4] and questback [5] use the same method to provide anonymized questionnaire data. The privacy protection depends on the de-identification technique and the security of the TTP. Any failure in the TTP incurs high privacy risks.

Secure multi-party computation (SMC) deals with the problem of a set of parties who wish to compute a certain function on their private data without revealing any private information apart from the output of the computation [6]–[8]. However, only a few SMC protocols have been developed for collecting and computing of categorical variables, which are the common type of data collected with a questionnaire [9]–[11]. These protocols have limitations for practical uses, such as privacy and scalability as the number of participating parties increases.

With the purpose of providing a practical method, in this paper we propose a privacy-preserving method for collecting and analyzing questionnaire data based on secret sharing [12] and Bloom filter [13]. The method contains SMC protocols for computing total and stratified statistical counts. Then, additional statistics, such as a contingency table, a chi-square test, an odds ratio, and logistic regression, are locally computed within the R statistical environment [14] using statistical counts as the building blocks. Moreover, we show that the method is secure under the semi-honest adversary

The paper is submitted for review on June 28, 2016. This work was supported by the Research Council of Norway grant number 248150/O70.

K. Y. Y. is with Department of Computer Science, UiT The Arctic University of Norway, 9037; Norwegian Centre for E-health Research, University Hospital of North Norway, 9019, Tromsø, Norway (e-mail: Kassaye.Y.Yigzaw@uit.no).

A. M. is with Department of Computer Science, University of Westminster, 115 New Cavendish Street, London W1W 6UW, United Kingdom (e-mail: A.Michalas@westminster.ac.uk).

J. G. B. is with Norwegian Centre for E-health Research, University Hospital of North Norway, 9019; Department of Clinical Medicine, UiT The Arctic University of Norway, 9037, Tromsø, Norway (e-mail: Johan.Gustav.Bellika@telemed.no)

model [15], where the adversary follows a protocol but may try to learn private information from the messages exchanged during the protocol execution. Theoretical and experimental evaluations demonstrated that the method is suitable for practical use.

II. USE CASE

The use case in this paper is the PAtients' TrAjectories (PAsTAs) project. The project aimed to study the relationship between healthcare services consumption and the satisfaction of patients with chronic conditions with the coordination of the healthcare services they had received. The study was approved by the Regional Committee for Medical and Health Research Ethics in central Norway (2011/2047).

The project selected a random sample of 12,502 patients with chronic conditions using health data collected by general practitioners, hospitals, and municipalities in central Norway. The patients were 18 years and older and used at least one somatic healthcare service in 2012 and 2013.

A questionnaire was sent to the selected patients, and 3,158 (25.3%) patients completed it. The questionnaire variables used to evaluate the method developed in this paper are *(1)* nominal variables, such as education, health status, and whether the respondents live alone, and *(2)* ordinal variables (on a Likert scale), such as satisfaction with the coordination of care. In addition, we used gender and age group data extracted from the respondents' healthcare service providers, which are nominal and interval variables, respectively. See Appendix A for details.

III. PROBLEM DEFINITION

Assume a cohort of N individuals who completed questionnaire Q for a particular study. The participants are denoted by $\mathcal{P} = \{p_1, p_2, \dots, p_N\}$, where p_i denotes an individual identified with a unique identifier [1] i. The questionnaire consists of n questions denoted by $Q = \{Q_1, Q_2, \dots, Q_n\}$, where Q_j has K distinct possible answers denoted by $Q_j = \{a_1^j, a_2^j, \dots, a_K^j\}$. The answers may have nominal, ordinal, or interval values. Q_j can be a multiple-response question. However, for the sake of simplicity of presentation, we describe the method proposed in this paper assuming Q_j has a single response denoted by a_l^j, where $l \in \{1,2, \dots, K\}$.

The problem addressed in this paper is to design a method for collecting questionnaire responses and analyzing the responses while protecting the respondents' privacy.

[1] The unique identifier does not have to be a regional or national unique identifier. It is sufficient that the identifier is unique within a survey. The unique identifier can be generated using different techniques depending on the survey design. For example, in the context of the use case presented in this paper, the researchers can generate identifiers to the recruited patients and send the identifier together with the invitation to the survey.

IV. BACKGROUND

In this section, we present the building blocks for the method proposed in this paper.

A. Secret sharing scheme

Secret sharing is a commonly used building block in SMC. The most common secret sharing scheme includes additive and Shamir's secret sharing schemes. In general, it is a method by which a secret value is split into L shares. Additive secret sharing scheme requires all shares to reconstruct the secret value, whereas Shamir's secret sharing requires a threshold number of shares [12].

The shares of a secret value $s \in \mathbb{Z}_{2^p}$ are denoted as $[\![s]\!] = (s_1, s_2, \dots, s_L)$, where $\mathbb{Z}_{2^p}$ is a finite ring of p-bit integers, and $s = \sum_{i=1}^{L} s_i \ mod\ 2^p$. For example, in additive secret sharing, the shares are computed by selecting $L-1$ random values $s_i \in \mathbb{Z}_{2^p}$, where $i \in \{1,2, \dots, L-1\}$, and s_L is computed as $s_L = s - \sum_{i=1}^{L-1} s_i \ mod\ 2^p$ [12].

Additive and Shamir's secret sharing are linear secret sharing schemes. Therefore, they have homomorphic property for addition, subtraction, and multiplication by a public constant value [16]. Let us assume two secret values x and y and their secret shared values are denoted by $[\![x]\!] = (x_1, x_2, \dots, x_L)$ and $[\![y]\!] = (y_1, y_2, \dots, y_L)$, respectively. The addition of x and y can be computed on their secret shared values by adding each pair of shares, $[\![x+y]\!] = (x_1 + y_1, x_2 + y_2, \dots, x_L + y_L)$. Secret share subtraction between x and y is also computed similarly. The output shares $[\![x+y]\!]$ are independent of the input shares. Therefore, revealing $[\![x+y]\!]$ to a third party does not disclose any information about x and y.

Linear secret sharing schemes are not multiplicatively homomorphic [16]. Therefore, complex protocols are developed for secret share multiplication [17]. We denote the secret share multiplication of x and y by $[\![x * y]\!]$.

B. Bloom filter

The Bloom filter (BF) is an efficient data structure that encodes a set $\mathcal{X}$ of n elements [13]. BF is an array that is m long, and each array position is a one-bit counter. We denote this array as $BF = \{c_1, c_2, \dots, c_m\}$. It supports the insertion and membership query of an element $x \in \mathcal{X}$. Each operation requires $O(1)$ time.

The insertion and membership query of x is performed using k hash functions $H_i(.)$, where $1 \le i \le k$. First, the hash of x is computed, using each hash function $H_i(x)$. Second, modulo m of each hash value is computed, $b_i(x) = H_i(x) \ mod\ m$, to get k array positions of BF, $b_i(x) \in [0, m-1]$. Then, x is inserted into BF by setting all the counters at positions $b_i(x)$ of the array to 1. Similarly, x is concluded to be a non-member of BF if at least one counter at one of the positions $b_i(x)$ of the array is 0.

A membership query result can be a false positive due to a hash collision that occurs when all the array positions of BF associated with x have been set to 1 as a result of the insertion of other elements. After n elements are inserted into BF using

the optimal number of hash functions, the probability that a membership query returns a false positive is $P\ (false\ positive)\ \approx (0.6185)^{m/n}$ [18].

V. Related work

In this section, we review secure methods for computing categorical data collected by individuals, which is also the focus of this paper. We also review secure methods for computing data distributed across data custodians using the same computation model as in this paper. The review focuses on methods that are secure against the semi-honest adversarial model.

Bogetoft *et al*. [19] presented a method for a secure double auction based on Shamir's secret sharing scheme [20] with three semi-honest third parties that collect and analyze the secret shares of bidders' bids. Several similar methods have been proposed for a wide variety of statistical computations, such as SHAREMIND [21], SEPIA [22], and Chida *et al*.'s [9] methods. The methods have a similar computation model with different secret sharing schemes and numbers of semi-honest third parties. The methods proposed by Bogetoft at al. [19], Bogdanov *et al*. [21], and Burkhart *et al*. [22] are for computing numerical variables, whereas the method proposed in [9] contains protocols for computing both numerical and categorical variables.

The protocol implemented in [9] for computing categorical data is based on a random shuffle protocol [23] in which the semi-honest third parties randomly shuffle the secret shares of the required variables. Then, the data analyst collects and processes the result to find the number of individuals in each category.

Drosatos *et al*. [10] proposed a method for distributed statistical computation of biomedical sensor data using Paillier Homomorphic encryption [24] where a personal software agent aggregates individuals' data through the exchange of encrypted data in a ring or tree topology. The proposed method supports computation of categorical and numeric variables.

Chen *et al*. [11] proposed a method for statistical count of categorical data distributed across individuals, and the computation results satisfy the differential privacy model [25]. In this method, the individuals send their responses and random noises encrypted using bit encryption [26] and the analyst's public key to a semi-honest third party. The third party also blindly inserts bit-encrypted noises and sends them to the data analyst. Then, the data analyst decrypts and finds the statistical count.

The methods proposed in [10], [11] require the devices used by the participating individuals to be online during the computation. However, the same set of individuals may not be online during consecutive computations on mobile devices due to limited computation resources and network connection. Thus, consecutive statistical computations may not give comparable results. Depending on the selection policy for participants, it may take a long time until all the selected individuals go online. The method proposed in [10] also requires communication between the participants' devices. As a result, both methods have a long runtime that limits the scalability of the method to a large number of participants.

VI. Materials and methods

In this section, we present the requirements of the method developed in this paper as well as the design, threat model, and sub-protocols.

A. Requirements

Based on our experience, the following requirements are specified for a method that collects and analyzes questionnaire data while protecting respondents' privacy.

R1 Security: The method should be secure against realistic adversarial attacks, and the security should be proved using a formal security analysis. The basic security properties are the following:

Privacy: No party should learn private information about the respondents based on their participation in a survey. It should be possible to query only the statistics in the questionnaire data.

Correctness: The statistical results must be accurate or have acceptable level of accuracy.

*R2 Efficiency and scalability***:** We assumed that the respondents complete a questionnaire online on mobile or stationary devices under their control. From now on, we refer to these devices as clients. These clients have heterogeneous computation resources. For example, the mobile devices may have constraints on computation resources (e.g., CPU and battery life) and network connectivity. Therefore, the method should tolerate clients that may shut down and go offline anytime. In addition, the clients' local computation requirement should be efficient so that even clients with limited computational power are able to complete the entire process.

SMC protocols that involve bi-directional communication between clients may not be completed or may require a long computation time if at least one of the clients is offline. Since a large number of respondents are often required in public health surveys, the computation time may increase significantly with the number of clients. Therefore, SMC protocols should scale with a large number of clients.

In addition, direct communication between clients requires knowledge of each other. This raises privacy concerns when participating in a survey reveals sensitive information based on the membership of the participants in the survey, which is called membership disclosure. Of course, anonymous communication protocols [27] can be used to maintain participants' anonymity, but these protocols add extra communication rounds to a computation.

R3 Usability: Data analysts use certain familiar statistical computation tools, such as R. Therefore, the method should allow analysts to work with familiar tools.

B. Secret shared Bloom filter

Once a respondent p_i completes the questionnaire, her response for each question Q_j is coded as a_i^j. The codes for all

responses are inserted into a Bloom filter denoted as BF_i. Then, each counter value of $BF_i = \{c_1, c_2, \dots, c_m\}$ is secret shared as shown in Fig. 1. The secret share of counter c_k of BF_i is denoted as $[\![c_k]\!] = (c_1^k, c_2^k, \dots, c_L^k)$, where $k \in [1, m]$. The secret share of BF_i is denoted as $[\![BF_i]\!] = (BF_1^i, BF_2^i, \dots, BF_L^i)$, and each share is denoted by $BF_j^i = \{c_j^1, c_j^2, \dots, c_j^m\}$, where c_j^k contains a share of the counter c_k of BF_i.

C. Architectural design

The architectural design of the method proposed in this paper is shown in Fig. 2. It contains three components: *N clients*, *L data miners*, and the R statistical environment. Each client runs on the respondent's device used to complete the questionnaire. A data miner is a third party that satisfies the threat model described in section VI-D. A data analyst works within the R statistical environment to analyze the questionnaire data.

Each client encodes a respondent's responses as secret shared Bloom filters and distributes them to the data miners together with the respondent's unique identifier. The data miners store the secret shared Bloom filters locally. They are blind to the respondents' answers; however, they jointly compute the secure protocols proposed in section VI-E to answer a data analyst's statistical query.

The data analyst uses the R package developed in this paper to make statistical queries against the secret shared Bloom filters. We refer to these queries as private queries, because the queries are computed on individual-level responses. In contrast, public queries are locally computed within R on the results of private queries.

D. Threat model and assumptions

We assumed that respondents create valid secret share Bloom filters. However, for some surveys, malicious respondents may create corrupted secret shared Bloom filters to alter the accuracy of the surveys' statistical outputs. Therefore, in Appendix B, we discuss how a malicious respondent could try to alter statistical outputs, and propose a secure and very efficient protocol for verifying whether a respondent's secret shared Bloom filters are legitimate.

We considered a *semi-honest* (*honest-but-curious*) adversarial model in which the *data miners* follow a protocol specification using the correct secret shared Bloom filters. However, the data miners might try to use the messages exchanged in the protocol to learn private information.

We assumed that there exists a secret share multiplication protocol that is secure under the semi-honest adversarial model. We also assumed that the protocol is universally composable [28] in which multiple executions of the protocol remain secure.

We assumed that communications between two entities that participate in a protocol are secure. Therefore, an adversary cannot read messages sent between two entities and the integrity of the messages is verified.

During the execution of the secure protocols, we assumed that the communications between the data miners are asynchronous. Therefore, sending and receiving a message may take an arbitrary amount of time. However, we assumed that there is no message loss as fault tolerance is beyond the scope of the paper.

E. Secure multi-party computation protocols

In this section, we presented the sub-protocols we developed for analyzing the respondents' answers, while preserving the respondents' privacy.

1) Secure protocols for the logical operators

Let us consider the counters c_k and c_{k+1} of BF_i, which have binary (1 or 0) values. Their secret shares $[\![c_k]\!]$ and $[\![c_{k+1}]\!]$, respectively, are distributed across the data miners. Let us also assume that a secret share of 1 is distributed across the data miners. In this section, we describe protocols for computing the logical operators, such as AND, OR, and NOT, of the secret shared values. The result of the protocols after computing the logical operators is a secret share of 1 distributed across the data miners when the expression is true, and 0 otherwise.

The logical AND of c_k and c_{k+1} can be computed on their secret shared values using a secret share *multiplication* protocol.

The logical OR of c_k and c_{k+1} can be described as $(c_k + c_{k+1}) - (c_k * c_{k+1})$. Thus, the data miners jointly compute the logical OR on their secret shared values using a combination of the secret share *addition*, *multiplication*, and *subtraction* protocols.

The logical NOT of c_k can be described as $(1 - c_k)$. Thus, the data miners compute the logical NOT on the secret share value of c_k and 1 using only the secret share subtraction protocol.

2) Secure count protocol

Let us consider question Q_j of the questionnaire and its answer a_l^j. The objective of the secure count protocol is to find a secret shared value $[\![n]\!]$ of the number of respondents who chose a_l^j. For example, the query can be finding the number of female respondents.

The responses of all survey participants are distributed across the data miners as a secret shared Bloom filter $[\![BF_i]\!]$, $i \in \{1,2, \dots, N\}$. The counter positions of a_l^j in BF_i is denoted as $b_h(a_l^j) \in [0, m-1]$, where $h \in \{1,2, \dots, k\}$ and k is the number of hash functions.

For each counter position $b_h(a_l^j)$, the data miners execute the secret share addition protocol of the counter values for all participants' secret shared Bloom filters. We denote the result as $[\![c_h]\!]$, which is a secret shared value distributed across the data miners. The results of the entire counter positions are denoted as $[\![c]\!] = \{[\![c_1]\!], [\![c_2]\!], \dots [\![c_k]\!]\}$. The minimum value of $[\![c]\!]$ is equal to $[\![n]\!]$ [29].

3) Secure membership query protocol

The objective of the protocol is to find a secret shared value $[\![r_{l,i}^j]\!]$ that is equal to 1, if a_l^j is a member of BF_i (in other words, the respondent p_i answered a_l^j), and 0 otherwise. For example, the protocol can be used to identify whether p_i is a

female. However, the result is secret shared between the data miners so that no one learns any information about the gender of p_i.

a_l^j is considered a non-member of BF_i if at least one of the positions $b_h(a_l^j)$ is 0. Therefore, as presented in *Algorithm 1,* the logical AND between the counter values at positions $b_h(a_l^j)$ of $[\![BF_i]\!]$ is equal to $[\![r_{l,i}^j]\!]$.

Algorithm 1: *SecureMembershipQuery* ($[\![BF_i]\!]$, a_l^j)

Input: The counter positions $\{b_1, b_2, \dots, b_k\}$ of a_l^j in $[\![BF_i]\!]$

Output: Secret shared value $[\![r_{l,i}^j]\!]$ of 1 if a_l^j exists in BF_i, and 0 otherwise.

//the data miners run the secure logical AND protocol
$[\![r_{l,i}^j]\!] = [\![c_{(b_1)} * c_{(b_2)}]\!]$
for $h = 3$ to k **do**
  //the data miners run the secure logical AND protocol
  $[\![r_{l,i}^j]\!] = [\![r_{l,i}^j * c_{(b_h)}]\!]$
end

The protocol is extended to perform a secure *non-membership* query of whether p_i did not answer a_l^j. This query is computed using the logical NOT protocol on the result of the membership query $[\![r_{l,i}^j]\!]$.

The protocols for logical operators can be used to join the results of multiple membership queries with logical operators, such as AND and OR. For example, we may want to query whether p_i is female AND > 65 years old. No one learns about the result, as it is secret shared.

Let us consider answers a_l^j and a_l^{j+1} to questions Q_j and Q_{j+1}, respectively. The membership queries of a_l^j and a_l^{j+1} in $[\![BF_i]\!]$ are denoted as $[\![r_{l,i}^j]\!]$ and $[\![r_{l,i}^{j+1}]\!]$, respectively. For example, a query about whether p_i answered a_l^j and a_l^{j+1} can be computed by executing the logical AND protocol on $[\![r_{l,i}^j]\!]$ and $[\![r_{l,i}^{j+1}]\!]$.

In general, any query criteria can be build by concatenating the logical operators, and the secret share of the result of a query on the responses of p_i is denoted as $[\![r_i]\!]$.

4) Secure conditional count protocol

The objective of the secure conditional count protocol is to find a secret shared value $[\![n]\!]$ of the number of individuals who satisfy a conditional query criteria, without revealing an individual's query results. An example query can be the number of respondents who are female AND > 65 years old.

The protocol computes as follows. First, for each respondent p_i, the data miners execute the query using the secure membership query protocol. Second, they compute secret share addition protocol of the query result $[\![r_i]\!]$, where $i \in \{1,2, \dots, N\}$, for all respondents, which gives the secret shared value $[\![n]\!]$.

VII. Statistical computations

The appropriate statistical analyses for questionnaire data depend on the objectives of the survey and the types of variables used in the questionnaire, such as nominal, ordinal, and interval.

As described in section VI-E-2 and VI-E-4, the results of the secure count and secure conditional count protocols are secret shared between the data miners. Then, the data miners send their shares to the R package that constructs the count values. These results are total and stratified statistical counts, respectively. As described next, a wide variety of statistical analyses can be computed using the statistical counts as building blocks.

The frequency table of a variable can be created by querying the total statistical count of each category of the variable. For an interval variable, it is possible to compute statistics, such as average, variance, and standard deviation, based on the frequency table.

A contingency table (cross-tabulation) of two or more categorical variables can be constructed by computing stratified statistical counts of each combination of the answers of the variables. Table I shows a 2×4 contingency table of age group and gender, where cell $n_{i,j}$ is the number of respondents who chose the values of row i and column j.

Statistics, such as a chi-square test and an odds ratio, can be computed on contingency tables. Logistic regression can also be computed on a multi-way contingency table for a binary or dichotomous outcome variable, which has two categories. Similarly, multinomial logistic regression can be computed for an outcome variable that has more than two categories.

VIII. Results

A. Security analysis

In this section, we prove the security for the protocols based on the threat model and the assumptions described in section VI.D.

Theorem 1 *The secure count protocol is secure for semi-honest data miners.*

Proof: The security proof of the protocol is simple following the security of the secret sharing scheme.

The protocol uses the homomorphic addition property of linear secret sharing schemes. Therefore, the protocol is secure, as the secret sharing scheme is secure. In the case of the additive secret sharing scheme, the protocol is secure with a collusion of up to $L - 1$ dishonest data miners.

Theorem 2 *The secure conditional count protocol is secure for semi-honest data miners.*

Proof: The security proof of the protocol is simple following the universal composability theorem.

The notion of universal composability allows us to reduce the security proof of the protocol to the security proof of its sub-protocols. Multiple executions of a universally composable protocol remain secure [28]. The protocol uses homomorphic addition and subtraction properties of a linear secret sharing scheme and a multiplication protocol. Therefore, the secure membership query protocol and the protocols for the logical operators are secure for a universally composable secret share multiplication protocol. Consequently, the secure conditional count protocol is secure,

and it provides the same security guarantee as the multiplication protocol.

As a result, the protocols successfully protect the respondents' privacy. Therefore, the method satisfies requirement R1 (Security).

B. Implementation

We implemented a prototype of the proposed method in Java. The method is agnostic to a linear secret sharing scheme and a secret share multiplication protocol. However, the prototype was implemented using the additive secret sharing scheme and the multiplication protocol proposed in [17]. The multiplication protocol was designed for three data miners. It has been shown that the protocol is universally composable under the semi-honest adversarial model. Consequently, all experiments performed in this paper used three data miners.

As we did not run a survey, the questionnaire data used in the experiments were stored in a csv file, where each record is an individual's questionnaire response. Therefore, to emulate the *client* application, we created a simple client that encoded each record of a csv file as secret shared Bloom filters of a respondent and distributed the secret shared Bloom filters to the data miners.

The format of a secret shared Bloom filter varies with the survey design, such as the questionnaire and the expected number of participants. Thus, a data miner could simply store a secret shared Bloom filter in a relational database as a text. However, it will require the data miners to process the whole secret shared Bloom filter, although a given query is computed only on the counter values at specific counter positions. Therefore, our implementation of data miners used Redis as a database and the Jedis client for persisting and querying secret shared Bloom filters [30]. The array of a secret shared Bloom filter was stored as a list using the respondent identifier as a key.

We developed an R package (SecureStat) that contains functions, such as *sec.count()* and *sec.conditional.count()*, for the secure statistical count and conditional count queries. We assumed that data analysts are familiar with the R statistical environment and perform further statistical computations of the results of secure counts using existing R packages.

We used the JavaScript Object Notation (JSON) format for message communication. The JSON messages were sent through the Extensible Messaging and Presence Protocol (XMPP) [31]. Each entity used an XMPP client to connect to a local XMPP server. Then, a message between two entities was sent through a server-to-server connection. All messages were compressed using the Lz4 [32] lossless compression algorithm to reduce the overall size. After transmission, each message was decompressed before actual use.

C. Analytical evaluation

A protocol is efficient if it is able to compute with good performance, which is often expressed by the communication (i.e., communication rounds and size of messages) and computation complexity. Scalability is measured in terms of the change in efficiency as the number of participants increases.

The secure count protocol does not require communication between the data miners. The local computation performed by a data miner is simple arithmetic that is linear with the number of survey respondents (N), number of hash functions (k), and the number of bits p of the secret shares.

The conditional count protocol uses a secure multiplication protocol that requires communication between the data miners. The size of the messages is linear with the values of N, k, and p. For query criteria that contain v variables, the number of multiplications is equal to $(k * v) - 1$. The value of k can be minimized by increasing the expected number of elements n of the Bloom filter for the same false positive probability. As a result, the number of multiplications significantly reduces.

The client encodes a respondent's answers as a Bloom filter and then secret shares each array position of the Bloom filter, which is computationally very efficient. The size of the message a *client* sends to data miners is linear with the Bloom filter parameters. The number of messages sent by a client is equal to the number of data miners, which is very few.

D. Experimental evaluation

We deployed the prototype in a local area network connected through a Gigabit switch. The data miners were deployed on three machines equipped with Intel i3-5010U dual core 2.10GHz CPUs, 8GB RAM, and Windows 10 Pro. The client and the R statistical environment were deployed on an Intel dual core i7 2.9 GHz CPU, 8GB RAM and Mac 10.10.5.

We ran experiments to evaluate the runtime of the protocol to compute a statistical count and a 3 × 6 contingency table. Each experiment was run 20 times, and the average runtime was recorded. The parameters used for the experiments are presented in Appendix C.

We ran the first experiments on the questionnaire data of 3,158 respondents described in section II. The runtime of a statistical count completed within 0.4 seconds. A query of a 3 × 6 contingency table was completed within 5.437 seconds.

To evaluate the scalability of the proposed method, the second experiment was run on simulated questionnaire data of 50,000 respondents. Fig. 3 and Fig. 4 illustrate the total runtime of a statistical count and a 3 × 6 contingency table query, respectively, as the number of respondents increases.

We also ran experiments to evaluate of the size of message and the computation times of a client as the number of questions on the questionnaire increases. Table II shows the size of the message a client sends to data miners. Fig. 5 shows the client's computation time.

IX. Discussion

We designed and evaluated a secure method for collecting and analyzing questionnaire data. The proposed method protects respondents' privacy under the semi-honest adversarial model, and the computation results were correct. Therefore, the method satisfies requirement R1.

Unlike the methods proposed in [10], [11], the computation model of the method does not require a *client* to be online after the respondent completes the questionnaire and the responses are distributed to the data miners as secret shared Bloom filters. When a client goes offline or shuts down before completing the questionnaire or distributing the secret shared Bloom filters, the task can be completed anytime within the survey's data collection period or the individual will be considered a non-respondent. In addition, our method does not require peer-to-peer communication between clients in contrast to the method proposed in [10].

The experimental results showed that the proposed protocols for statistical analyses are efficient, and they linearly scale with the addition of respondents. The runtime of the protocols are significantly faster than the protocols proposed in [10], [11]. The runtime of our protocols are comparable to the protocols implemented in [9], but our protocol for creating a contingency has security advantage as discussed later.

In addition, in our method, the computation complexity of a client is very minimal that makes it suitable for usage on mobile devices. Therefore, requirement R2 is satisfied.

An R package was developed for the secure statistical count and conditional count queries of the secret shared data distributed across data miners. Various statistics are computed within R using the statistical count as a building block. Requirement R3 was satisfied as the method allows a data analyst to work within a familiar environment. In addition, the use of R as a front-end tool avoided the need to implement advanced statistics.

The paper discussed the proposed method in terms of questionnaire data. However, the method can be used for other data sources, such as mobile application and medical sensor data, where the types of variables are nominal, interval, and ordinal.

The client application can be implemented as a web application and/or a mobile application. Bogdanov *et al*. [33] discussed the state of the art for web technologies for privacy-preserving computations.

The method proposed in the paper is secure for any entities selected as data miners as long as they satisfy the threat model. Data miners cannot be able to obtain an individual's personal information, even if they try to do so. However, practical uses of the method require public awareness of how the method works and how much privacy protection is provided. The use of public entities trusted by the community, such as healthcare institutions, patient organizations, and privacy advocates, as data miners may also increase public trust.

The business process of running a survey using this privacy-preserving method will not be much different from existing online survey systems. The proposed method starts with a survey design that includes the preparation of a questionnaire and coding the questions (see Appendix A for the questionnaire used in the experiments). The survey design also includes setting the appropriate parameters (see Appendix C for the parameters used in the experiments). Participants are invited and data are collected after the survey is designed. During the data collection phase, data analysts may be interested only in querying the number of respondents. However, after the data collection phase is completed, data analysts run statistical analyses.

The security analysis proved that the protocols proposed in this paper are secure. In general, SMC protocols compute without revealing any information apart from the computation results. However, the computation results might lead to inferential disclosure given prior knowledge and repeated queries. In the following sections we present techniques that can be used in our method to avoid or minimize inferential disclosure. However, the methods presented in [9], [10] lacks inferential disclosure techniques.

Query restriction techniques are developed in the statistical database research area to limit inferential disclosure, where queries that can lead to a compromise are denied [34]. For example, SHRINE, a distributed health research network, limits the smallest value of a statistical count query result to ten [35]. In our method, a secret share less than protocol [17] can be used to evaluate whether results of secure count and secure conditional count protocols is less than the threshold t, which gives a secret share of 1 if the count is less than t, and 0 otherwise. Therefore, it is possible to extend our method to make sure that statistical counts of less than t are not returned to a data analyst.

Perturbation techniques were also developed to limit inferential disclosure techniques, by adding noise into the query results in a way that protects individuals' privacy [34]. The method proposed in [11] uses the differential privacy model [25] for inferential disclosure limitation. For our experiments, we chose a small false positive probability that led to computation of the exact statistical results. However, it is possible to use the false positive probability of the Bloom filter, which is selected during the survey design phase, as a parameter to insert noise into the statistical results.

In the current implementation of the secure questionnaire method, statistical counts are returned to a data analyst who performs additional analysis on the statistical count results. However, as the results of the secure count and secure conditional count protocols are secret shared, advanced statistical computations can be implemented without revealing the intermediate results. Therefore, inferential disclosures can be minimized or avoided.

X. Conclusion

The privacy-preserving method proposed in this paper is efficient and scalable for practical use. The method may increase the number of survey respondents and the accuracy of their responses. The method may also minimize or avoid the non-respondent bias that occurs as a result of differences between survey respondents and non-respondents. However, further research is required to measure the effects of this method on a survey compared to existing anonymous survey method.

Currently, the prototype was evaluated between data miners connected through a local area network. We plan to evaluate

this method with data miners connected through the Internet. In addition, we plan to enhance the prototype with the techniques discussed in Section IX for avoiding statistical disclosures.

APPENDIX A

In this appendix, we present the subset of the questionnaire of the PAsTAs project that was used to evaluate the method presented in this paper. The questionnaire was originally in Norwegian. However, we present an English translation of the questions. In addition, the questions are coded according to the notations described in section III.

Q_1. What is your gender?

a_1^1. Female a_2^1. Male

Q_2. What is your age?

a_1^2. <18 a_2^2. Between 18 and 45

a_3^2. Between 46 and 65 a_4^2. >65

Q_3. What is the highest education you have completed?

a_1^3. Primary education

a_2^3. Lower secondary education

a_3^3. Upper secondary education

a_4^3. Less than 4 years college/university education

a_5^3. Four years or more college/university education

Q_4. In general, would you say your health is

a_1^4. Excellent

a_2^4. Very good

a_3^4. Good

a_4^4. Fair

a_5^4. Poor

Q_5. Do you live together with someone?

a_1^5. Yes a_2^5. No

Q_6. Think of the health services you have received in 2012 and 2013 and indicate your level of agreement with the following statement:

Overall, I am satisfied with the healthcare services I have received.

a_1^6. Completely disagree

a_2^6. Somewhat disagree

a_3^6. Neutral

a_4^6. Somewhat agree

a_5^6. Completely agree

a_6^6. Not applicable

APPENDIX B

A malicious respondent may try to affect the accuracy of a survey's statistical analysis results although it is not possible to get any other benefit (e.g., cannot learn other respondents' private information). The array positions of a Bloom filter are set to 1 based on a respondent's answers, and 0 otherwise. However, a respondent may create a corrupted Bloom filter with all the array positions set to 1. Consequently, the secret shared Bloom filters generated from the Bloom filter become corrupted. Such corruption of a malicious respondent increases the result of a statistical count by only one, which is not significant in most applications.

A statistical count of the questionnaire data will be significantly distorted if the array positions of the Bloom filter are set to large values. A zero-knowledge range proof [36] can be used to protect against such attacks as a verifier will be able to confirm that each array position of a respondent's Bloom filter is either 0 or 1. However, zero-knowledge proofs are computationally expensive.

As an alternative, we propose a simple and efficient protocol for identifying the corrupted Bloom filter of a malicious respondent and consequently remove his secret shared Bloom filters from the survey. For a given survey design it is possible to estimate the maximum number of array positions of a Bloom filter that can be set to 1, and we denote it as M.

For each secret shared Bloom filter $[\![BF_i]\!]$, the data miners execute the secret share addition of all counter values, which approximates a secret share of the number of array positions that have value 1 denoted as $[\![M_i]\!]$. The data miners exchange the shares of $[\![M_i]\!]$ to reconstruct M_i. Finally, each data miner locally evaluates whether $M_i \leq M$ and verifies that the secret shared Bloom filter is legitimate.

APPENDIX C

Kirsch et al. [37] demonstrated that any BF can be effectively implemented with only two hash functions $H_1(x)$ and $H_2(x)$. The k counter positions of the k hash functions are simulated with the form $b_i(x) = (H_1(x) + iH_2(x)) \; mod \; m$ without affecting the false positive probability. A hash function $H(.)$ and two secret keys k_1 and k_2 can be used to instantiate the hash values of the two hash functions as follows, $H_1(x) = H(k_1 \parallel x)$ and $H_2(x) = H(k_2 \parallel x)$. The hash function can be a non-cryptographic hash function, which is more efficient than cryptographic hash functions. For all the experiments in this paper we used MurmurHash 2, which is a non-cryptographic hash function.

For all the experiments, we chose the number of hash functions $k = 1$ and false positive probability $P\,(false\ positive) = 0.01$. The expected number of elements is equal to the number of questions of questionnaire n. Then, we chose a Bloom filter size m that provides the false positive probability of $P\,(false\ positive) = 0.01$ for a given n and $k = 1$.

We also used secret sharing with a finite ring of integers $\mathbb{Z}_{2^p}$, where $p = 16$. Therefore, the result of a statistical count query is a 16-bit integer. The upper bound of the result of a statistical count query is equal to the number of survey participants N. Therefore, the number of bits p should be able to represent N. Let us consider $N = 50{,}000$ and question Q6 of the questionnaire presented in Appendix A. In an ideal scenario, all respondents were satisfied with the healthcare services they had received and responded: "Completely agree." Consequently, a statistical count of the number of respondents who "Completely" satisfied with the healthcare services they had received is 50,000, which requires 16-bit.

ACKNOWLEDGMENTS

We thank Gro Berntsen and Aslak Steinsbekk for the discussion about the use case used in the paper and for allowing us to use the PAsTAs project data to evaluate the method proposed in this paper. We would like to thank Gunnar Hartvigsen and Fred Godtliebsen for invaluable discussions.

We also would like to thank Torje Henriksen and Joseph Hurley for invaluable discussions and supports during the implementation of the proposed method.

REFERENCES

[1] European Commission, "Green Paper on mobile health ('mHealth')," European Commission, COM(2014) 219 final, 2014.

[2] R. M. Groves, "Nonresponse Rates and Nonresponse Bias in Household Surveys," *Public Opin. Q.*, vol. 70, no. 5, pp. 646–675, Jan. 2006.

[3] A. D. Ong and D. J. Weiss, "The Impact of Anonymity on Responses to Sensitive Questions," *J. Appl. Soc. Psychol.*, vol. 30, no. 8, pp. 1691–1708, Aug. 2000.

[4] SurveyMonkey Inc., "SurveyMonkey: Free online survey software & questionnaire tool," 2016. [Online]. Available: https://www.surveymonkey.com/. [Accessed: 19-Jun-2016].

[5] questback, "Online Survey Software - Customer Feedback Solutions - Questback," 2016. [Online]. Available: https://www.questback.com/. [Accessed: 19-Jun-2016].

[6] M. Kantarcioglu, "A survey of privacy-preserving methods across horizontally partitioned data," in *Privacy-preserving data mining*, C. C. Aggarwal and P. S. Yu, Eds. Springer US, 2008, pp. 313–335.

[7] J. Vaidya, "A Survey of Privacy-Preserving Methods Across Vertically Partitioned Data," in *Privacy-Preserving Data Mining*, C. C. Aggarwal and P. S. Yu, Eds. Springer US, 2008, pp. 337–358.

[8] M. A. Hailemichael, K. Y. Yigzaw, and J. G. Bellika, "Emnet: a tool for privacy-preserving statistical computing on distributed health data," in *Proceedings from The 13th Scandinavian Conference on Health Informatics*, Tromsø, Norway, 2015, pp. 33–40.

[9] K. Chida, G. Morohashi, H. Fuji, F. Magata, A. Fujimura, K. Hamada, D. Ikarashi, and R. Yamamoto, "Implementation and evaluation of an efficient secure computation system using 'R' for healthcare statistics," *J. Am. Med. Inform. Assoc.*, vol. 21, no. e2, pp. e326–e331, Oct. 2014.

[10] G. Drosatos and P. S. Efraimidis, "User-centric privacy-preserving statistical analysis of ubiquitous health monitoring data," *Comput. Sci. Inf. Syst.*, no. 0, pp. 22–22, 2014.

[11] R. Chen, A. Reznichenko, P. Francis, and J. Gehrke, "Towards Statistical Queries over Distributed Private User Data.," in *NSDI*, 2012, vol. 12, pp. 13–13.

[12] A. Beimel, "Secret-sharing schemes: a survey," in *Coding and Cryptology*, Y. M. Chee, Z. Guo, F. Shao, Y. Tang, H. Wang, and C. Xing, Eds. Springer Berlin Heidelberg, 2011, pp. 11–46.

[13] B. H. Bloom, "Space/Time Trade-offs in Hash Coding with Allowable Errors," *Commun. ACM*, vol. 13, no. 7, pp. 422–426, Jul. 1970.

[14] R Core Team, *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, 2016.

[15] Y. Lindell and B. Pinkas, "Secure multiparty computation for privacy-preserving data mining," *J. Priv. Confidentiality*, vol. 1, no. 1, p. 5, 2009.

[16] J. Cohen Benaloh, "Secret sharing homomorphisms: keeping shares of a secret secret," in *Proceedings on Advances in cryptology—CRYPTO '86*, Berlin, Heidelberg, 1987, pp. 251–260.

[17] D. Bogdanov, M. Niitsoo, T. Toft, and J. Willemson, "High-performance secure multi-party computation for data mining applications," *Int. J. Inf. Secur.*, vol. 11, no. 6, pp. 403–418, Nov. 2012.

[18] S. Tarkoma, C. E. Rothenberg, and E. Lagerspetz, "Theory and practice of bloom filters for distributed systems," *Commun. Surv. Tutor. IEEE*, vol. 14, no. 1, pp. 131–155, 2012.

[19] P. Bogetoft, D. L. Christensen, I. Damgård, M. Geisler, T. Jakobsen, M. Krøigaard, J. D. Nielsen, J. B. Nielsen, K. Nielsen, J. Pagter, M. Schwartzbach, and T. Toft, "Secure Multiparty Computation Goes Live," in *Financial Cryptography and Data Security*, R. Dingledine and P. Golle, Eds. Springer Berlin Heidelberg, 2009, pp. 325–343.

[20] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[21] D. Bogdanov, S. Laur, and J. Willemson, "Sharemind: A framework for fast privacy-preserving computations," in *Computer Security-ESORICS 2008*, Springer, 2008, pp. 192–206.

[22] M. Burkhart, M. Strasser, D. Many, and X. Dimitropoulos, "SEPIA: Privacy-preserving Aggregation of Multi-domain Network Events and Statistics," in *Proceedings of the 19th USENIX Conference on Security*, Berkeley, CA, USA, 2010, pp. 15–15.

[23] S. Laur, J. Willemson, and B. Zhang, "Round-efficient oblivious database manipulation," in *Information Security*, Springer, 2011, pp. 262–277.

[24] P. Paillier, "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes," in *Advances in Cryptology — EUROCRYPT '99*, J. Stern, Ed. Springer Berlin Heidelberg, 1999, pp. 223–238.

[25] C. Dwork, "Differential privacy," in *Automata, languages and programming*, M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, Eds. Springer Berlin Heidelberg, 2006, pp. 1–12.

[26] S. Goldwasser and S. Micali, "Probabilistic encryption," *J. Comput. Syst. Sci.*, vol. 28, no. 2, pp. 270–299, Apr. 1984.

[27] M. Edman and B. Yener, "On anonymity in an electronic society: A survey of anonymous communication systems," *ACM Comput. Surv.*, vol. 42, no. 1, pp. 1–35, Dec. 2009.

[28] R. Canetti, "Universally composable security: a new paradigm for cryptographic protocols," in *42nd IEEE Symposium on Foundations of Computer Science, 2001. Proceedings*, 2001, pp. 136–145.

[29] S. Cohen and Y. Matias, "Spectral Bloom Filters," in *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, New York, NY, USA, 2003, pp. 241–252.

[30] S. Sanfilippo, "Redis." [Online]. Available: http://redis.io/. [Accessed: 17-Jun-2016].

[31] P. Saint-Andre, K. Smith, and R. Tronçon, *XMPP: The Definitive Guide: Building Real-Time Applications with Jabber Technologies*, First Edition. Sebastopol, CA: O'Reilly Media, Inc., 2009.

[32] Y. Collet, "RealTime Data Compression: LZ4 explained," 26-May-2011. .

[33] D. Bogdanov and R. Talviste, "A prototype of online privacy-preserving questionnaire system," Technical report, University of Tartu, Institute of Computer Science, 2010.

[34] N. R. Adam and J. C. Worthmann, "Security-control Methods for Statistical Databases: A Comparative Study," *ACM Comput Surv*, vol. 21, no. 4, pp. 515–556, Dec. 1989.

[35] G. M. Weber, S. N. Murphy, A. J. McMurry, D. MacFadden, D. J. Nigrin, S. Churchill, and I. S. Kohane, "The Shared Health Research Information Network (SHRINE): A Prototype Federated Query Tool for Clinical Data Repositories," *J. Am. Med. Inform. Assoc.*, vol. 16, no. 5, pp. 624–630, Sep. 2009.

[36] T. Dimitriou and A. Michalas, "Multi-party trust computation in decentralized environments in the presence of malicious adversaries," *Ad Hoc Netw.*, vol. 15, pp. 53–66, Apr. 2014.

[37] A. Kirsch and M. Mitzenmacher, "Less hashing, same performance: Building a better Bloom filter," *Random Struct. Algorithms*, vol. 33, no. 2, pp. 187–218, Sep. 2008.

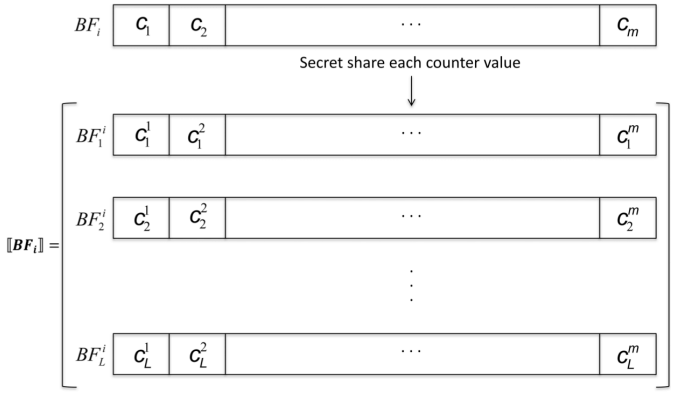

Fig. 1. Secret share of Bloom filter BF_i that encodes the responses of respondent p_i

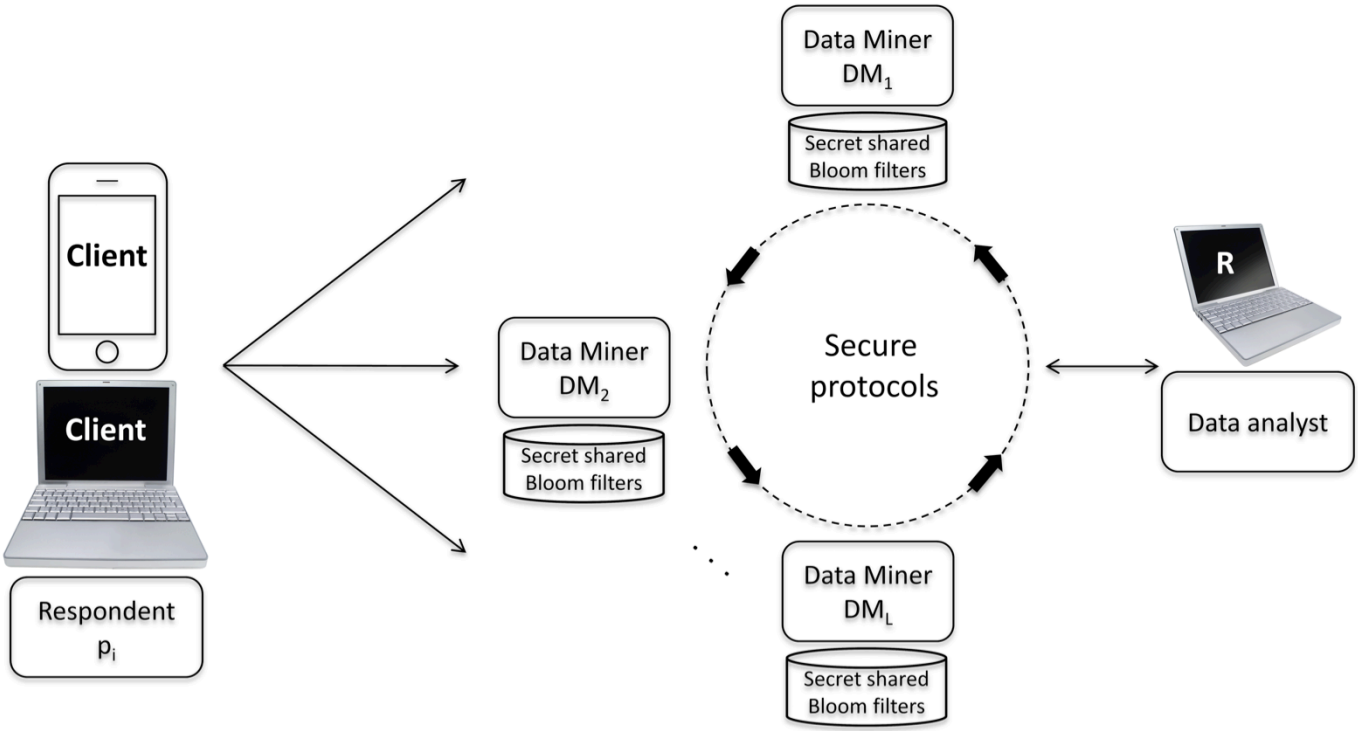

Fig. 2. Architectural design of the privacy-preserving questionnaire method

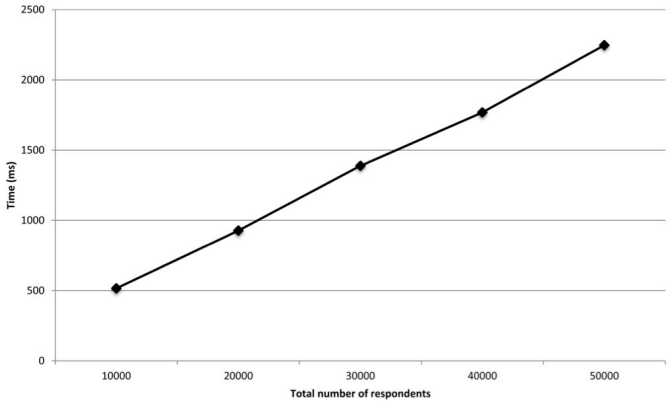

Fig. 3. The total runtime of a statistical count query as the number of respondents increases

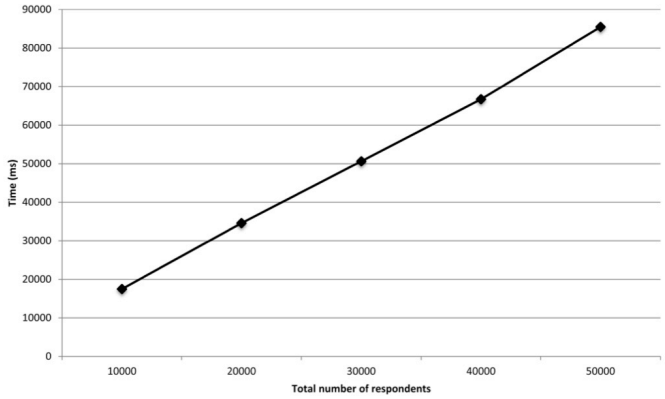

Fig. 4. The total runtime of a 3 × 6 contingency table query as the number of respondents increases

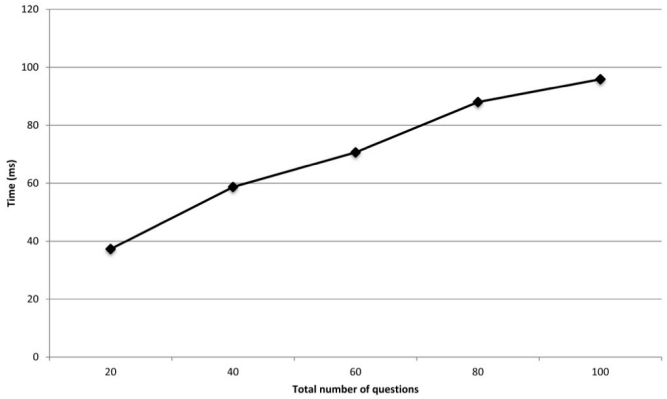

Fig. 5. The computation time of a *client* as the number of questions of the questionnaire increases

TABLE I
A CONTINGENCY TABLE OF GENDER AND AGE GROUP

Gender	Age Group			
	< 18	$18 - 45$	$46 - 65$	> 65
Male	$n_{1,1}$	$n_{1,2}$	$n_{1,3}$	$n_{1,4}$
Female	$n_{2,1}$	$n_{2,2}$	$n_{2,3}$	$n_{2,4}$

TABLE II
THE SIZE OF THE MESSAGE A CLIENT SENDS TO A DATA MINER AS THE NUMBER OF QUESTIONS INCREASES

Number of questions	20	40	60	80	100
Message size (KB)	4.77	9.22	13.69	18.15	22.62