

**LIINA KAMM**

Privacy-preserving  
statistical analysis using  
secure multi-party computation





**LIINA KAMM**

Privacy-preserving  
statistical analysis using  
secure multi-party computation



Institute of Computer Science, Faculty of Mathematics and Computer Science,  
University of Tartu, Estonia

Dissertation is accepted for the commencement of the degree of Doctor of Philosophy (PhD) on January 28, 2015 by the Council of the Institute of Computer Science, University of Tartu.

Supervisor:

Dr. Tech.	Sven Laur
	University of Tartu
	Tartu, Estonia

Opponents:

Prof. Ph.D	Rebecca N. Wright
	Rutgers University
	Piscataway, United States of America

Ph.D	George Danezis
	University College London
	London, United Kingdom

The public defense will take place on March 9, 2015 at 16:15 in Liivi 2-404.

The publication of this dissertation was financed by the Estonian Doctoral School in Information and Communication Technology.



European Union  
European Social Fund



Investing in your future

Copyright: Liina Kamm, 2015

ISSN 1024-4212

ISBN 978-9949-32-761-4 (print)

ISBN 978-9949-32-762-1 (PDF)

University of Tartu Press  
[www.tyk.ee](http://www.tyk.ee)

# Contents

<b>List of publications</b>	<b>8</b>
<b>Abstract</b>	<b>9</b>
<b>1 Introduction</b>	<b>11</b>
1.1 Sensitive data . . . . .	11
1.2 Claims of this work . . . . .	12
1.3 Dissertation outline and contributions of the author . . . . .	13
<b>2 Statistics and secure multi-party computation</b>	<b>16</b>
2.1 Statistical analysis . . . . .	16
2.2 Secure multi-party computation . . . . .	18
2.3 Peculiarities of statistical analysis in the secure multi-party setting	22
2.3.1 Data import . . . . .	22
2.3.2 Filtering and secure formation of case and control groups .	22
2.3.3 Statistical testing paradigms . . . . .	23
2.4 Related work . . . . .	25
2.5 Security proofs . . . . .	26
<b>3 Secure genome-wide association studies</b>	<b>31</b>
3.1 Motivation . . . . .	31
3.2 Genome-wide association studies . . . . .	32
3.3 Our solution . . . . .	33
3.4 Impact of our work . . . . .	36
<b>4 Floating-point arithmetic and satellite collision analysis</b>	<b>38</b>
4.1 Motivation . . . . .	38
4.2 Background . . . . .	38
4.3 Our solution . . . . .	39
4.3.1 Floating-point arithmetic . . . . .	39
4.3.2 Elementary functions . . . . .	43

4.3.3	Matrix operations . . . . .	46
4.4	Satellite collision analysis . . . . .	46
4.5	Impact of our work . . . . .	47
<b>5</b>	<b>A privacy-preserving statistical analysis suite</b>	<b>50</b>
5.1	Motivation . . . . .	50
5.2	Background . . . . .	51
5.3	Our solution . . . . .	52
5.3.1	Descriptive statistics and statistical tests . . . . .	52
5.3.2	Linear regression . . . . .	53
5.4	Impact of our work . . . . .	54
<b>6</b>	<b>Using the statistics suite in a real-world study</b>	<b>56</b>
6.1	Motivation . . . . .	56
6.2	Problem statement . . . . .	57
6.3	Background . . . . .	58
6.4	The PRIST project . . . . .	59
6.5	Data import and pre-processing . . . . .	61
6.5.1	Privacy-preserving aggregation . . . . .	63
6.5.2	Education dataset . . . . .	65
6.5.3	Tax dataset . . . . .	67
6.5.4	Obtaining the analysis table . . . . .	68
6.6	Statistical analysis . . . . .	70
6.7	Preparing for the study . . . . .	72
6.7.1	Achieving compliance with data protection . . . . .	72
6.7.2	Improving the tool to handle real-world data . . . . .	73
6.7.3	Code review and acceptance testing by the Tax Board . . . . .	75
6.7.4	Performance measurements . . . . .	76
6.8	Impact of our work . . . . .	77
	<b>Conclusion</b>	<b>78</b>
	<b>Bibliography</b>	<b>80</b>
	<b>Acknowledgments</b>	<b>88</b>
	<b>Kokkuvõte (Summary in Estonian)</b>	<b>90</b>
	<b>Original Publications</b>	<b>93</b>
	A new way to protect privacy in large-scale genome-wide association studies . . . . .	97
	Secure Floating-Point Arithmetic and Private Satellite Collision Analysis	107

Privacy-preserving statistical analysis on federated databases . . . . .	127
Rmind: a tool for cryptographically secure statistical analysis . . . . .	155
<b>Curriculum Vitae</b>	<b>195</b>
<b>Elulookirjeldus</b>	<b>196</b>

## **PUBLICATIONS INCLUDED IN THIS THESIS**

The publications included in this thesis describe the author's contribution to privacy preserving statistical analysis methods, and their performance results.

1. Kamm, L., Bogdanov, D., Laur, S., Vilo, J.: A new way to protect privacy in large-scale genome-wide association studies. *Bioinformatics* 29(7), 886–893 (2013).
2. Kamm, L., Willemson, J.: Secure Floating-Point Arithmetic and Private Satellite Collision Analysis. *International Journal of Information Security* pp. 1–18 (2014).
3. Bogdanov, D., Kamm, L., Laur, S., Pruulmann-Vengerfeldt, P., Talviste, R., Willemson, J.: Privacy-preserving statistical data analysis on federated databases. In: *Proceedings of the Annual Privacy Forum, APF'14*. LNCS, vol. 8450, pp. 30–55. Springer (2014).
4. Bogdanov, D., Kamm, L., Laur, S., Sokk, V.: Rmind: a tool for cryptographically secure statistical analysis. *Cryptology ePrint Archive*, Report 2014/512 (2014).

## **PUBLICATIONS NOT INCLUDED IN THIS THESIS**

This technical report by the author describes aspects of privacy preserving statistical analysis but is not included in this thesis. This report has not been published as a separate paper.

1. Bogdanov, D., Kamm, L.: Constructing Privacy-Preserving Information Systems Using Secure Multiparty Computation. Tech. Rep. T-4-13, Cybernetica, <http://research.cyber.ee/>. (2011).

## **OTHER PUBLISHED WORK OF THE AUTHOR**

This publication by the author is on a different topic.

1. Jäger, M., Kamm, L., Krushevskaja, D., Talvik, H.A., Veldemann, J., Vilgota, A., Vilo, J.: Flexible Database Platform for Biomedical Research with Multiple User Interfaces and a Universal Query Engine. In: *DB&IS. Frontiers in Artificial Intelligence and Applications*, vol. 187, pp. 301–310. IOS Press (2008).



# ABSTRACT

The analysis of sensitive personal information is a problem with which companies and governments struggle daily. Different statistical measures have been researched and adopted to help deal with the issues of leakage of personally identifiable information. However, more often, legislative measures are adopted, confidentiality agreements signed and data exported to analysts. The statistical studies are performed on sensitive information instead of using a provably secure method.

Secure multi-party computation is a means for sharing and working with data in a privacy-preserving manner where the values are protected even from the data analyst. These cryptographic protocols have been researched and developed since circuit garbling was introduced by Yao, and several practical applications have also emerged. Cryptographers have designed and implemented Boolean and integer arithmetic for various applications including statistics. However, these independent functions use different underlying protocols and are often very limited in scope.

In this dissertation, we design and implement a protocol suite for conducting statistical studies in the privacy-preserving setting using the secure multi-party computation platform SHAREMIND as the underlying framework. The designed algorithms, however, are not platform-specific and can be implemented also for other secure computation platforms that support privacy-preserving integer and floating-point arithmetic, sorting, shuffling and table join.

We start by looking at an example statistical application, namely, genome-wide association studies in the privacy-preserving setting. We implement four most frequently used statistical tests for this purpose using integer arithmetic provided by SHAREMIND. We then determine that, for more complicated statistical functions, we need the availability of floating-point arithmetic. However, we show the feasibility of privacy-preserving statistical studies on databases with millions of values.

We go on to design and implement secure floating-point numbers, addition, multiplication, and comparison for the SHAREMIND platform. We also design and implement several privacy-preserving elementary functions, namely inverse, square root, sine, natural logarithm, exponentiation of  $e$  and the error function.

Using the created arithmetic, we are able to design and implement the most often used statistical analysis operations for the privacy-preserving setting and put them together in an R-like analysis tool that we call RMIND. This tool is meant to help statistical analysts carry out studies without requiring them to understand too much about the underlying cryptography. We implement descriptive statistical methods, such as the five-number summary and histograms; simple statistical measures, like mean and standard deviation; statistical tests, like Student's t-test and the  $\chi^2$  test; and linear regression.

The final part of the dissertation describes how we use the created RMIND tool in practice, conducting a large-scale privacy-preserving statistical study on real-world data. We describe the issues of performing such a study in practice using secure multi-party computation, talk about the pitfalls and shortcomings of working with real data, detail the extract, transform, load process, and describe how RMIND will be used in the study.

# CHAPTER 1

## INTRODUCTION

### 1.1 Sensitive data

Data are collected in most areas of life. In a modern society, from the moment a person is born, a digital record is made and from there on, different sets of data are gathered and stored about different aspects of life. Whether one is swiping a customer loyalty card in a store, going to the doctor, doing taxes or even simply moving around with a mobile phone in one's pocket, sensitive data are being gathered and stored in some database.

Sometimes we give our permission for this kind of surveillance for some benefit, like using a customer loyalty card for a discount, other times we are left with a choice that is difficult in our society, like not being able to make phone calls. However, often, we are required by law to give up our sensitive information like health, income, and education data.

Once the data have been gathered, though, their controller is motivated to use them in statistical analyses or data mining to get new information, discover patterns or help make financial and governing decisions. In many countries, strong privacy laws govern the processing of sensitive personal information. These laws usually allow the employees of the organisation controlling the data to process the records only after proper confidentiality agreements have been signed. However, data sharing with other organisations is heavily restricted. This is good from the point of view of the data donors, but makes cross-database analyses a difficult process for even two closely related organisations, e.g., government institutions.

The various agencies of a modern government host hundreds of databases. They offer researchers the possibility to apply for access. For some databases, like the education information system in Estonia, authorisation can be easier to obtain than for financial databases. However, getting access to education information can also be a long-winded and tedious process even if the request for analysis comes from a government agency source.

So what is it exactly that the laws are trying to protect? Surely, knowing a general aggregation result cannot harm any individuals present in the dataset. In fact, the laws are not trying to hide the aggregation results but to keep the people working with the data from finding out sensitive information about a single subject. Often, even the widely-used pseudonymisation techniques, that replace identifiers with codes, fail to hide the individuals with more unique traits. A person can be indirectly identified by taking a set of attributes that make them unique, e.g., age, gender, nationality, education level, year of graduation, name of school. Even though this can be considered difficult for records of students graduating from their Bachelor's studies as they are mostly of the same age, it becomes fairly easy when looking at students graduating from their PhD studies.

## 1.2 Claims of this work

Secure multi-party computation is a way of performing computations without seeing the data that are being worked on. Combining this cryptographic technology with the statistical analysis problem is a logical step forward. By now, secure multi-party computation has been used in practice for several real-life applications. The earlier concerns of slow running time are being worked on constantly and more feasible benchmark results keep appearing.

This dissertation describes the author's work on privacy-preserving statistical analysis using secure multi-party computation. We claim that it is possible and feasible to use general statistical functions that are not problem-specific to perform statistical analyses and make the process similar to the existing procedures. To be more specific, we make three claims. First, we claim that secure multi-party computation can be employed to design and implement the most commonly used statistical analysis algorithms, without knowing the problem or setting in which the analysis will be performed. Second, we claim that these algorithms can be incorporated into a statistics suite, that is similar to existing statistics packages, is approachable to the analyst and does not require the analyst to have a thorough understanding of SMC techniques and computation protocols. Third, we claim that these analyses are feasible enough to be used in practice even with large datasets.

We start our research with a proof-of-concept by implementing the privacy-preserving version of genome-wide association studies. We find that what is lacking from a comprehensive solution is the availability of secure floating-point arithmetic. We design and implement this in order to securely compute elementary functions like inverse and square root. Finally, we choose the most commonly used statistical analysis methods and go on to design and implement these in the privacy-preserving setting as a general tool. We provide benchmark results for

all of the designed and implemented algorithms and validate our results in a real-world statistical study.

### 1.3 Dissertation outline and contributions of the author

The author has designed all of, and implemented some of the privacy-preserving versions of the statistical analysis methods discussed in this dissertation. This work was done using the SECREC language [11] for the SHAREMIND secure multi-party computation platform [6]. In addition, the author implemented secure floating-point arithmetic for the SHAREMIND system based on algorithms designed in collaboration with Jan Willemson. In the following, we introduce the dissertation chapter by chapter.

**Chapter 2** provides an overview of the terms and principles of secure multi-party computation that are needed as a basis for the work in this dissertation. We also give a survey of previous work in the field of privacy-preserving statistical analysis and briefly describe the secure multi-party computation system SHAREMIND that the implementations and performance results presented in the papers are based on.

**Chapter 3** introduces our first contribution to the world of privacy-preserving statistical analysis. Namely, we designed and implemented four methods for finding trends in genotype and phenotype data. We deployed the methods on sample data and found the performance results almost feasible.

The chapter refers to the following paper included in this dissertation.

- Kamm, L., Bogdanov, D., Laur, S., Vilo, J.: A new way to protect privacy in large-scale genome-wide association studies. *Bioinformatics* 29(7), 886–893 (2013).

This paper [48] contains the application scenarios of secure multi-party computation in large-scale genome-wide association studies. It describes the specifics of adapting existing statistical analysis methods for use on secret-shared integer data. The performance results are also included in the paper. The author’s contributions include the adaptation and implementation of the statistical tests, design of optimised algorithms that make use of the optimisation features of secure multi-party computation, and proposal of application models. The author also conducted benchmarking experiments and analysed performance results. The author of this dissertation is the first author of the paper.

**Chapter 4** describes our work on the shortcomings that we found in the SHAREMIND system when we were implementing the secure genome-wide association

study methods, namely the lack of support for floating-point arithmetic and elementary functions. We test the design and implementation of the new arithmetic in satellite collision prediction analysis.

The chapter refers to the following paper included in this dissertation.

- Kamm, L., Willemson, J.: Secure Floating-Point Arithmetic and Private Satellite Collision Analysis. *International Journal of Information Security* pp. 1–18 (2014).

This paper [49] introduces the floating-point data type and contains the algorithms for floating-point addition and multiplication, and elementary functions like inversion, square root and exponentiation of  $e$ . It also gives the privacy-preserving adaptation of an algorithm that calculates the chance of a pair of satellites colliding in Earth’s orbit. The performance results of floating-point arithmetic, elementary functions, as well as the satellite collision analysis are included in the paper. The author adapted floating-point arithmetic for the privacy-preserving setting in collaboration with Jan Willemson. The author implemented and optimised the privacy-preserving versions of the arithmetic. The author also designed, implemented and optimised the privacy-preserving versions of the elementary functions in collaboration with Jan Willemson. In addition, the author implemented and optimised the privacy-preserving satellite collision analysis method, conducted the benchmarking experiments and analysed performance results. The author of this dissertation is the first author of the paper.

**Chapter 5** describes how we designed and implemented a statistical suite for privacy-preserving statistical analysis.

The chapter refers to the following papers included in this dissertation.

1. Bogdanov, D., Kamm, L., Laur, S., Pruulmann-Vengerfeldt, P., Talviste, R., Willemson, J.: Privacy-preserving statistical data analysis on federated databases. In: *Proceedings of the Annual Privacy Forum, APF’14*. LNCS, vol. 8450, pp. 30–55. Springer (2014).

This paper [8] illustrates how using privacy-preserving statistical analysis in a federated database setting can be beneficial. The paper describes the Estonian X-Road environment that connects all governmental databases and how to combine it with secure multi-party computation for privacy-preserving statistical analysis and decision making. The paper also refers to an end-user study, that the authors carried out, validating the need for such a solution. The practical implementation of the solution and a pilot project are described, and some algorithms for secure multi-party statistical analysis are given, along with benchmark results. The author designed and

implemented the privacy-preserving statistical analysis methods and systematised the deployment models. The author is the first author of Sections 5–7 of this paper and shares first authorship with Riivo Talviste, who is the first author of Sections 2–4 of this paper.

2. Bogdanov, D., Kamm, L., Laur, S., Sokk, V.: Rmind: a tool for cryptographically secure statistical analysis. Cryptology ePrint Archive, Report 2014/512 (2014).

This paper [9] contains the descriptions of a comprehensive set of algorithms for a standard statistical study. The design and implementation details of a tool for privacy-preserving statistical analysis—RMIND—are given along with performance results. The author designed the privacy-preserving versions of the algorithms and implemented about half of them. The author of this dissertation is the first author of the paper.

**Chapter 6** describes the practical culmination of the work done for this dissertation. The SHAREMIND team is using the implemented statistics suite in a national-scale privacy-preserving joint statistical study. We give an overview of this study and the work that has been done to make this unprecedented project possible. The author has supervised further development of the statistical suite. She has collaborated with the project team in the analysis and design of this statistical study. Her most significant contribution is the design of the algorithms and methods used in the privacy-preserving extract, transform, load (ETL) process.

# **CHAPTER 2**

## **STATISTICS AND SECURE MULTI-PARTY COMPUTATION**

### **2.1 Statistical analysis**

Statistical analysis is a way of finding patterns and trends about the subjects of the data. There are different ways of obtaining data, such as asking people to fill out questionnaires, applying for access to central registries, measuring objects or processes and their different results, and analysing data logs.

In this dissertation, we mostly focus on the comparison of two groups or cohorts, called case and control groups. These two groups are formed based on some traits that differentiate the two groups, such as age, gender, the presence or severity of disease, education level for people, and successful or failed execution for processes. The statistical analyst poses a hypothesis about the difference of these two groups based on some attribute other than the ones that distinguish the two groups. For example, the hypothesis can state that students working during their studies graduate later than their fellows who are not working.

Based on the assumptions known to hold for the available data, different tests can be chosen to compare these two groups and confirm whether the hypothesis holds or not. In the standard setting, the testing process usually works in the following way: data are analysed according to the testing algorithm, the test statistic is returned, the p-value is calculated based on the test statistic and compared to the significance threshold specified by the statistical analyst. The probability of successfully accepting the alternative hypothesis when the null hypothesis does not hold is called the power of the test. The power depends on how different the alternative hypothesis is from the null hypothesis and how well the chosen test is able to distinguish between the two.

A test statistic gives the analyst a measure that summarises the data in one number. The test statistic is used to distinguish the null hypothesis from the alter-



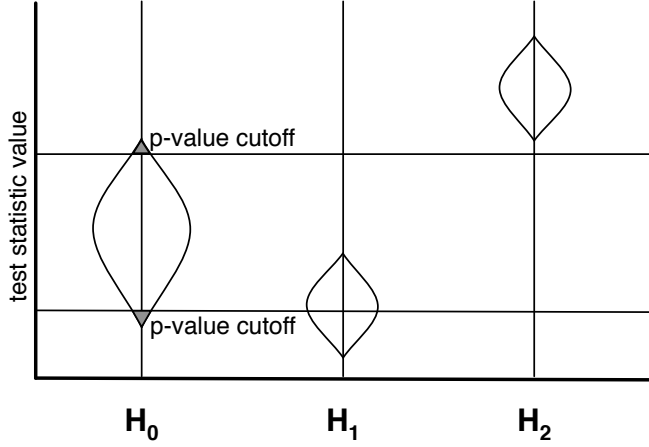


Figure 2.1: The mechanics behind statistical testing

native hypothesis posed by the analyst. The test statistic is computed differently for each different statistical test. For instance, the t-test compares the means of two populations, whereas the  $\chi^2$  test statistic is based on the difference of the actual distribution and the expected distribution in the groups.

To interpret the different test statistics, social scientists and statistical analysts use p-values. The p-value shows how probable it is to compute a test statistic at least as extreme as the one received as a result of the statistical test if, in actuality, the null hypothesis holds. If this probability is lower than a threshold determined by the analyst (usually 0.05 for single tests and  $10^{-5}$  in genome-wide association studies, where multiple tests are done at once), the alternative hypothesis can be considered more plausible than the null hypothesis.

Figure 2.1 illustrates the notion of hypothesis testing using the p-value. These violin plots depict the distribution of the test statistic under different hypotheses. The null hypothesis  $H_0$  has a p-value cut-off and alternative hypotheses that have p-values at least as extreme as that of the null hypothesis are considered plausible. In this particular case, we consider two alternative hypotheses  $H_1$  and  $H_2$ . If  $H_1$  holds, a large part of the distribution of the test statistic is between the cut-off lines and, thus, the test is not well suited for distinguishing between hypotheses  $H_0$  and  $H_1$ . However, when  $H_2$  holds, the test statistic is above the cutoff threshold and the test can distinguish between hypotheses  $H_0$  and  $H_2$  well.

Personal data processing is always closely connected with issues of privacy. Whether we are dealing with income, health, behaviour or location data, we are working with sensitive information. Often pseudonymisation techniques are applied to deal with this issue, other times data controllers pre-aggregate the data or add noise before releasing them for analysis. This dissertation investigates

whether secure multi-party computation can be used as a method for preserving the privacy of the individuals contributing their data.

Before moving on, note that there are two orthogonal aspects of privacy we can consider. On one hand, how much side information in addition to the desired results is leaked, and, on the other hand, what the desired results leak about individual data records. The former can be addressed using cryptographic methods. The latter is described by output privacy. Intuitively, a study is considered output private if the results reveal nothing important about the input values. More formally, we consider a statistical study output-private, if the results reveal nothing about the input values [6]. This is a very strong statement, therefore we consider different levels of output privacy. On one end, the *ideal level* leaks no information about the input values, on the other end, on the *no level* everything about the inputs can be deduced from the outputs. What we actually try to achieve is the *reasonable level* where nothing important about the inputs can be deduced from the outputs.

The question whether a statistical function offers output privacy is a topic that has been well studied in statistics. Methods, such as  $k$ -anonymity [70],  $l$ -diversity [57],  $t$ -closeness [55] and differential privacy [29] are often employed for this purpose. Of these methods, differential privacy is specially designed to achieve output privacy, the other three contribute indirectly by adding noise to the inputs. Although we consider this an important direction of research, we do not handle these topics in this dissertation. In our experimental studies, we assume that a privacy risk assessment is conducted to evaluate the leakage of private information through the outputs and modify them on a case-by-case basis.

Instead, we look at cryptographic privacy. We consider a statistical study to be cryptographically private if the results reveal only the values that are desired as outputs. Our algorithms return values in a privacy-preserving format. Which of these values can be opened and given as a result to the analyst must, at present, be determined by the study plan. We hope to find a more automatic solution to this topic during our future research into ensuring output-privacy in secure multi-party computation. The task is not as simple as applying one of the studied methods, as we have to take into account the output precision and privacy leakage trade-off that these methods entail.

## 2.2 Secure multi-party computation

Secure multi-party computation (SMC) is a technology with which two or more parties compute a function without seeing any private input values but their own. There are different flavours of general SMC. The underlying technology of current methods can be secret sharing [5, 21], garbled circuits [73], or homomorphic

Functionality	Notation
Protected storage of a private value $x$ (signed integer, floating-point, boolean)	$\llbracket x \rrbracket$
Conversion to the private form	$\llbracket x \rrbracket \leftarrow x$
Support for value vectors and matrices	$\llbracket \mathbf{x} \rrbracket$ and $\llbracket \mathbf{M} \rrbracket$
Privacy-preserving binary operations (signed integer, floating point, boolean)	$\llbracket z \rrbracket \leftarrow \llbracket x \rrbracket \circledast \llbracket y \rrbracket$
Privacy-preserving functions	$\llbracket y \rrbracket \leftarrow f(\llbracket x \rrbracket)$
Declassify value to computing parties	$\text{declassify}(\llbracket x \rrbracket)$
Publish value to result parties	$\text{publish}(\llbracket x \rrbracket)$
Private shuffling, linking and sorting	—

Table 2.1: Secure computation capabilities and notation for statistical analysis algorithms.

encryption [60, 37]. The main idea is to perform operations on data without seeing their values. We also introduce three different party roles to help with explaining which parties can see what and who is in charge of certain operations. These three roles include *input parties*, who secret-share or encrypt the data. They send the values to the *computing parties*, who perform the operations that have been agreed upon on the hidden data without seeing the values of the private inputs. They send the encrypted or secret-shared data to the *result parties*, who decrypt or declassify the values to see the results. A party can take on one or several of these roles.

Even though the protocols in this dissertation can be used in every setting that supports the underlying basic operations that we need for our computations, our implementation and the benchmark results are done using additively homomorphic secret sharing.

Our algorithms assume that the privacy-preserving operations listed in Table 2.1 from [9] are available for use. These secure operations on secret-shared data are used as building blocks in the algorithms proposed in this dissertation.

Let us look at this setting more closely. Let  $Z$  be a quotient ring. To secret-share a value  $x \in Z$  among  $n$  computing parties, the input party uniformly selects  $x_1, \dots, x_{n-1}$  from  $Z$  and computes the  $n$ -th share as  $x_n = x - x_1 - \dots - x_{n-1}$ . Each share is sent to the corresponding computing party who learns nothing about the received uniform value without colluding with other computing parties. There are protocols that help the computing parties process the data to obtain the results that the result parties ask for. Some of these protocols are fairly straightforward, e.g., addition and multiplication, but some, e.g., bit extraction, are more complex requiring the use of subprotocols. We also require that the protocols be universally composable, meaning that the protocol remains secure even if side computations

are done in parallel, e.g., another protocol, potentially using the same data, is scheduled to run at the same time.

There are several practical implementations of secure multi-party computation systems. Secure integer arithmetic is available in Fairplay [58], SEPIA [16], TASTY [42], VIFF [36], SPDZ [25], SHAREMIND [6], and PCF [52]. Floating point arithmetic has been considered in [20, 35]. Shuffling and sorting have been implemented by the authors of [40] and linking has been implemented in [30]. We use the SHAREMIND platform [6] for our practical experiments. The SHAREMIND protocols are described in [12, 13] and the detailed composability proofs are given in [10].

Several practical prototype applications have also been implemented. Table 2.2 from [8] describes them within the party role paradigm that we introduce.

We often need to work with private and public data in parallel, as some values do not need to be secret-shared, such as classifier element descriptions or other publicly available information. For this, we bring in the distinction between these values in algorithms. All the public values are denoted in the usual manner as  $x$ , and private values are denoted as  $\llbracket x \rrbracket$ . Similarly, we denote private vectors as  $\llbracket \mathbf{y} \rrbracket$  and matrices as  $\llbracket \mathbf{A} \rrbracket$ .

The SHAREMIND platform has its own language SECREC [11] that can be used to implement algorithms on the SHAREMIND platform. This language provides the programmer with the means to distinguish between private and public data types and the appropriate protocols for the operations are chosen automatically, so the programmer does not have to worry about this.

An interesting aspect of solutions such as SHAREMIND is the fact that parallelisation can greatly speed up the working time of algorithms. This is due to the fact that most protocols require the computing parties to send some data between each other and the network messages are often not filled optimally so the running time does not grow linearly in the number of inputs. When the network channel becomes saturated, the running time grows linearly with respect to the number of inputs. The number of inputs at which the protocol saturates the network channel is called the saturation point and knowing its estimated value in certain network conditions helps the system to optimise the running time of secure computation. At the other end, there is a point where all resources are used up and computations become very slow. To avoid reaching this point, SHAREMIND uses batches that are created with sizes based on the network connection parameters. See Figure 2.2 taken from [9] for an illustration of this concept.

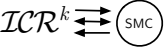



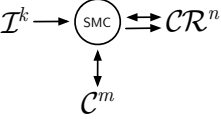
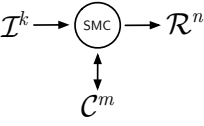
Basic deployment model	Example applications
	<p><b>The classic millionaires' problem [73]</b>  <i>Parties:</i> Two—Alice and Bob (both <math>ICR</math>)  <i>Overview:</i> Millionaires Alice and Bob use SMC to determine who is richer.</p> <p><b>Joint genome studies [48]</b>  <i>Parties:</i> Any number of biobanks (all <math>ICR</math>)  <i>Overview:</i> The biobanks use SMC to create a joint genome database and study a larger population.</p>
	<p><b>Studies on linked databases [8]</b>  <i>Parties:</i> Ministry of Education, Tax Board, Population Register (all <math>IC</math>) and Statistics Bureau (<math>R</math>).  <i>Overview:</i> Databases from several government agencies are linked to perform statistical analyses and tests.</p>
	<p><b>Outsourcing computation to the cloud [37]</b>  <i>Parties:</i> Cloud customer (<math>IR</math>) and cloud service providers (all <math>C</math>).  <i>Overview:</i> The customer deploys SMC on one or more cloud servers to process her/his data.</p>
	<p><b>Collaborative network anomaly detection [16]</b>  <i>Parties:</i> Network administrators (all <math>IR</math>) a subset of whom is running computing servers (all <math>ICR</math>).  <i>Overview:</i> A group of network administrators uses SMC to find anomalies in their traffic.</p>
	<p><b>The sugar beet auction [15]</b>  <i>Parties:</i> Sugar beet growers (all <math>I</math>), Danisco and DKS (both <math>CR</math>) and the SIMAP project (<math>C</math>).  <i>Overview:</i> Sugar beet growers and their main customer use SMC to agree on a price for buying contracts.</p>
	<p><b>The Taulbee survey [34]</b>  <i>Parties:</i> Universities in CRA (all <math>I</math>), universities with computing servers (all <math>IC</math>) and the CRA (<math>R</math>).  <i>Overview:</i> The CRA uses SMC to compute a report of faculty salaries among CRA members.</p> <p><b>Financial reporting in a consortium [14]</b>  <i>Parties:</i> Members of the ITL (all <math>I</math>), Cybernetica, Microlink and Zone Media (all <math>IC</math>) and the ITL board (<math>R</math>).  <i>Overview:</i> The ITL consortium uses SMC to compute a financial health report of its members.</p>

Table 2.2: SMC deployment models and example applications

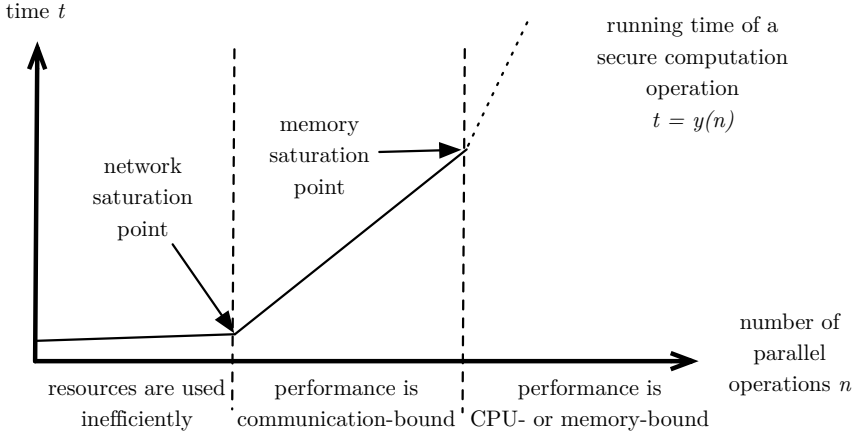


Figure 2.2: Performance model of secure computation protocols based on secret sharing.

## 2.3 Peculiarities of statistical analysis in the secure multi-party setting

### 2.3.1 Data import

When importing data from different sources, the issue of missing values arises quickly. There is a distinct difference between whether a value is zero or simply missing. Whereas in the usual setting, missing values can be denoted as empty fields or by entering NA (not available) instead of the value, this is not so straightforward in the privacy-preserving setting. Even though it is possible to encode the missing value as not available using some chosen value, finding these values later using the private comparison is expensive time-wise.

As data storage is cheap in comparison with processing power, we propose an alternative solution of adding an extra column for each column that can contain missing values. We call this mask vector the *availability vector*. More specifically, if a column can contain missing values, i.e., it is not a key column, a separate Boolean vector will be created for each such column when data is secret-shared by the input parties. This so-called availability vector is, essentially, a mask vector that contains 0 if the value is missing and 1 if the value is available. Storing the information in such a manner makes it easier to work with the data and eliminate the missing values on the fly.

### 2.3.2 Filtering and secure formation of case and control groups

Similarly to the availability vector, we use the mask vector idea for filtering. As most times we do not want to reveal the size of the dataset after it has been fil-

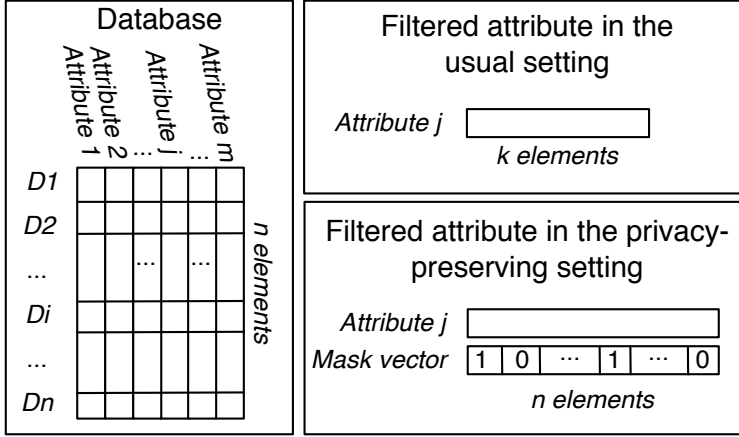


Figure 2.3: Difference of database filtering in the usual and privacy-preserving settings

tered, i.e., we do not want to show how many records correspond to the specified filter conditions, our solution differs from the methods used in standard databases. Usually, filtering returns a set of records that correspond to the filter, whereas our solution returns a secret-shared mask vector that can later be used in computations together with the original dataset. See Figure 2.3 for a visual representation of the differences between the two settings. Only the records that have the mask value 1 will be used in the computations and the user will not see which records are used.

Sometimes, however, the number of records must be revealed to the user because the analysis or study design requires it. In such cases, the set of records corresponding to the filter can be compiled obviously, so that no additional information about the values is revealed, except for the number of filtered records. In the following, we refer to this type of filtering as cutting the dataset or using the **cut** function described in [9, Algorithm 1].

### 2.3.3 Statistical testing paradigms

We introduce three new paradigms for statistical testing in the privacy-preserving setting. Figure 2.4 is based on the work in [9] but we expand this a bit and add Option 3 to the privacy-preserving setting.

As mentioned before, the testing process in the usual setting is quite straightforward, starting with the test statistic and the p-value being computed and compared to the significance threshold set by the analyst. In the privacy-preserving setting, there can be different concerns for data privacy that require the testing process to be modified to some extent. Let us look at the three options described on Figure 2.4 that can be used in the privacy-preserving setting.

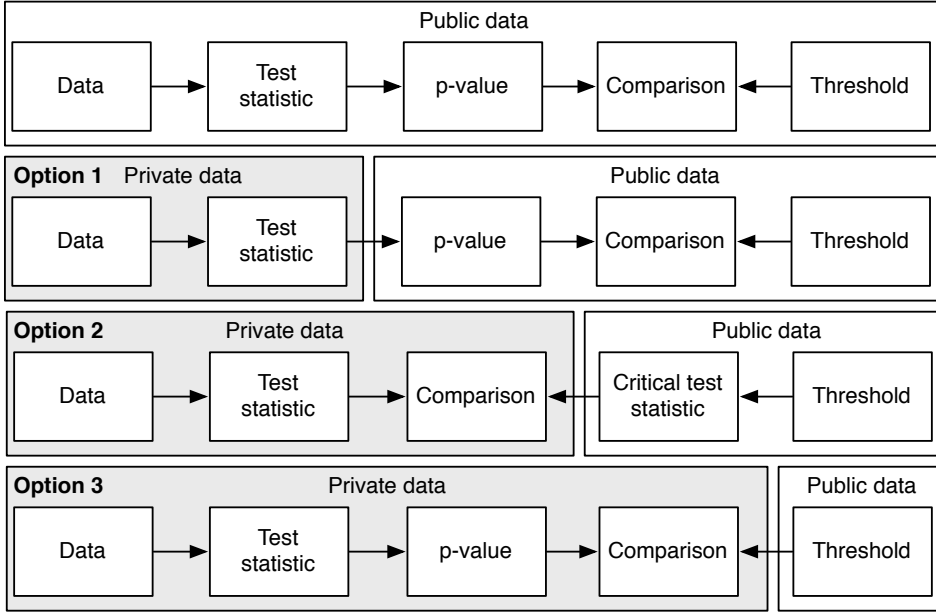


Figure 2.4: Different statistical testing paradigms in the public and private setting

As with most issues in the privacy-preserving setting, these options offer a time-privacy trade-off. Option 1 can be used when the test statistic and sample sizes can be declassified. The p-value can then be computed in public, making statistical testing significantly faster. Option 2 offers somewhat better privacy by revealing only the sample sizes but not the computed test statistic. Using the sample sizes and the chosen significance threshold, the analyst can look up or calculate the critical test statistic based on the test being performed. The critical statistic will then be compared to the test statistic received during the testing process. This option is important, for example, in genome-wide association studies, where revealing the p-values can already leak whether a donor's sample is in the chosen dataset or not [43]. This problem will be further discussed in Chapter 3.

Option 3 offers the best privacy guarantee, as only the comparison result is revealed to the analyst. However, this reveals very little information and might not always be acceptable to statistical analysts who are used to seeing p-values as well. In addition, this option is the slowest as the p-value has to be computed in the privacy-preserving setting. The third option is also the closest to the standard setting with the exception that all operations are done in a privacy-preserving manner.



## 2.4 Related work

Statistical analysis tasks have been attempted and successfully carried out in a secure multi-party setting before. We look at the most prominent statistical analysis protocols that have been published along with practical implementations. Most of these prototype applications work in the deployment model where all parties have all party roles, i.e., they all input data, perform computations and receive the results. This is the first model in Table 2.2.

One notable exception is the system proposed by Chida et al. [22] where the computing parties are separated from the others similarly to the third model in Table 2.2. Their system includes descriptive statistics, filtering, cross-tabulation and versions of the t-test and  $\chi^2$  test. They implement their set of privacy-preserving statistical analysis operations based on Shamir’s secret sharing scheme as a back-end for the R environment, meaning that the analyst who is familiar with R can start working on the data immediately with a short study period. Additionally, they allow both public and private computations and the performance results are impressive. However, only the commands of their R module can be used on private data, and other R commands work on public data. In addition, they only support private integer operations and cannot, therefore, use more complex statistical tools—such as linear regression—that are most commonly used by statistical analysts. Their implementation also lacks the ability to link different database tables.

El Emam et al. [30] describe protocols for linking databases and computing the  $\chi^2$  test, odds ratio and relative risk. They base their work on the Paillier cryptosystem [60] and their protocols work on integer data, letting the result party perform the final division in the standard setting.

Different components of a set of statistical analysis tools have been implemented by different authors. Multiple ideas for implementing weighted sums, mean and variance of integer data are considered in [19] using private information retrieval (PIR), homomorphic encryption, and general secure multi-party computation. Special protocols for scalar products using oblivious transfer and homomorphic encryption are given in [27]. The latter, however, were proven to be insecure in [38].

In [28], the authors look at multivariate linear regression and classification in the two-party setting, where the data are vertically partitioned and distributed between the two parties, making the approach very setting-specific. The computation of mean in the two-party setting is investigated in [50]. The authors of this paper also use a two-party protocol that is tailored for this setting.

Filtered sums are considered in [69] and scalar products are similarly investigated in [72]. The Paillier homomorphic cryptosystem is used for both implementations. Another reformulation of scalar product as weighted sums over ho-

homomorphically encrypted data is considered in [47]. Private data aggregation of streaming data has been considered in [67, 56].

It is easy to see, that even though many different statistical analysis functions have been considered previously, it would be difficult, to incorporate all of these protocols into one simple statistical analysis suite. In addition, only a few of these protocols provide cryptographic privacy. Therefore, we find that our research into combining a comprehensive tool for privacy-preserving statistical analysis is justified.

## 2.5 Security proofs

This subsection gives the security framework in which this dissertation operates. We do not provide detailed proofs for all our algorithms, but rather we rely on the security of the underlying protocols and their property of being universally composable.

First, let us look at universal composability. A protocol is considered to be universally composable if it remains secure even if side computations are done in parallel [18]. This means that other instances of the same protocol or other protocols can be scheduled to run at the same time or in sequence and this will not leak additional information about the inputs. Protocols that use universally composable subprotocols are universally composable as well [18].

Second, we assume, that the evaluation of arithmetic circuits is secure and universally composable. On the SHAREMIND platform the proof framework for the arithmetic circuits and universal composability is presented in [10]. This provides us with cryptographic security on the condition that only the desired output is declassified during computation.

Let us look at the situation where we need to declassify some intermediate values during computation to reduce the number of branching decisions in the protocol. In SMC, each branch has to be completed and then the right result will be chosen using oblivious choice. Consider the following example:

```
a ← {1, 2, 3, 4, 5}
b ← {1, 2, 3, 4, 5}
if (a == 5) {
    b = b + 1
    a = a + b
} else {
    a = b + 7
}
```

In this example, there is a branching decision based on whether  $a$  is equal to five, and two branches are created. One possible solution to performing this kind of computation without declassifying the comparison result, is to convert the program with branching into an arithmetic circuit where both branches are evaluated and the output of the right branch is selected obliviously. For our example, one possible solution is the following:

```

a ← {1, 2, 3, 4, 5}
b ← {1, 2, 3, 4, 5}
c = b + 1
d = a + c
e = b + 7
s = (a == 5)
a = s · d + (1 - s) · e
b = s · c + (1 - s) · b

```

Computations in all branches have to be performed, otherwise the algorithm leaks data about the condition used in the branching decision.

Often, there are too many branches to go through in feasible time. Then the condition on which a branching decision is made is explicitly declassified and the decision is made based on a public condition. In the declassification case, the algorithm explicitly states, that a private value is made public and this makes it clear that the decision leaks something. This information can be used in the security analysis of the algorithm. For each case, where this premature declassification method is used, we need to provide a separate reasoning for why the protocol remains secure, as we can no longer rely on the security proofs of the underlying protocols.

It is not always clear whether additional declassifications weaken the security or not as can be seen from the following example. Consider the situation where we want to know how many elements in a secret-shared vector  $\llbracket a \rrbracket$  correspond to a condition. The easiest way is to declassify vector  $\llbracket a \rrbracket$  and compare the values to the condition in public. It is clear that this leaks both the values and the records that corresponded to the condition. Taking this further, we can perform the comparison in the privacy-preserving setting if the comparison protocol is available. Let us denote the Boolean vector of comparison results by  $\llbracket b \rrbracket$ . We can now declassify this vector. This does not leak the original values in the vector. However, it leaks which records corresponded to the condition, and thus leaks information about those records. To counter this, we can shuffle  $\llbracket a \rrbracket$  before comparison, and later declassify the comparison results. This does not leak the values of the original vector nor does it leak the locations of the values that corresponded to the condition. Alternatively, we can use the circuit based solution, where we obli-

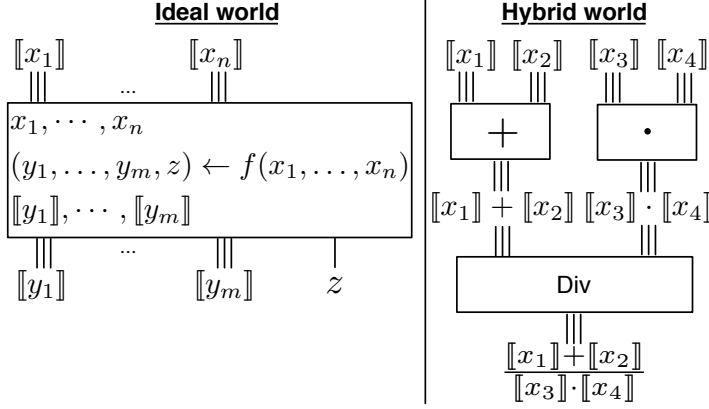


Figure 2.5: Ideal world and hybrid world

ously sum the elements of the unshuffled  $\llbracket b \rrbracket$  and declassify the sum as the desired result  $s$ . This gives us the result we need without having to use additional classification during the execution of the algorithm.

The proofs for the additional declassifications given in the paper [9] work in the hybrid setting. Proofs using the ideal and hybrid model setup are standard tools in cryptography [17, 19, 39, 24]. We give an illustrative explanation of the concept using Figure 2.5. In this setup, we look at two different functionalities—the ideal functionality and the hybrid functionality. The ideal functionality is an incorruptible party that gets all inputs, computes the function and outputs the results. In the hybrid world we have modules of secure functionality, such as addition, multiplication, comparison, division, declassification, shuffle, sort. We assume that the security of these protocols has been proven and that they can be composed into a more complicated algorithm without breaching that security.

For the example in Figure 2.5, we assume that we have three parties participating in our SMC protocol. The triple wires represent a secret shared value being transmitted, each wire comes from a different party. The single separate wire that can be seen in the example in the ideal world, represents an unclassified value being transmitted. The ideal world, therefore, outputs a tuple of secret-shared values  $(y_1, \dots, y_m)$  and a public value  $z$  in the figure. A semi-honest adversary in this setting can be viewed as a party that can see the values being transmitted on the wires that correspond to corrupted computing parties. In the malicious model, the adversary can also change the values on these wires.

The hybrid model gives a circuit that uses the different building blocks that are provided by secure protocols. An explicit branching choice in the hybrid world is hard to depict on such a structure without introducing public multiplexing. Alternatively, we can model public branching by dynamically constructing the circuit

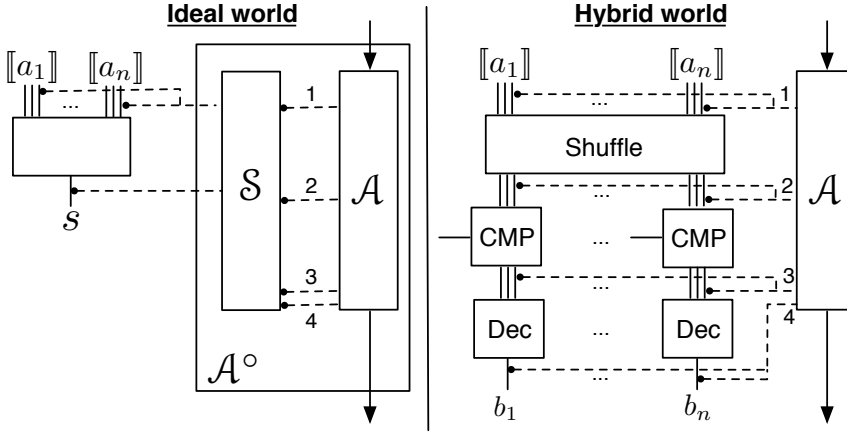


Figure 2.6: Adversaries in the ideal and hybrid world

based on the public decisions made with the help of premature declassification.

To prove that an algorithm that employs premature declassification is private, we show that the protocol in the hybrid world is equivalent to the ideal world from the point of view of the adversary  $\mathcal{A}$ . Essentially, we need to show that we are able to simulate the declassified values to the adversary  $\mathcal{A}$  so that  $\mathcal{A}$  is not able to distinguish between the two worlds. In the examples we discussed, it is easy to see that for the first case, knowing the number of elements corresponding to a condition does not give us the capability of simulating the values in the original vector. In the second case where we declassify the comparison result vector  $\llbracket b \rrbracket$ , we cannot simulate the correct order knowing the number of elements. In the final example, the simulation is trivial, as the sum is the number of vector elements that correspond to the condition.

However, let us look more closely at the third example where we shuffled the input vector, performed the comparison and then declassified the results. Figure 2.6 shows this example in the ideal and real world. In the ideal world, we input the vector  $\llbracket a \rrbracket$  and as a result, we receive the number of elements  $s$  that corresponded to our condition. In the hybrid world, we use the blocks for shuffle, comparison (CMP) and declassification (Dec) to compose our algorithm. Let  $\mathcal{A}$  be a static semi-honest adversary in the hybrid world and let us assume that  $\mathcal{A}$  is corrupting the third party in our example. This means that  $\mathcal{A}$  can see all of that party's inputs and outputs in addition to all of the public values. The adversary  $\mathcal{A}$  can also have auxiliary information about the world that is based on something outside the current algorithm, e.g., they can have additional information about some inputs. This is depicted as the topmost arrow going into  $\mathcal{A}$ .

We introduce the same adversary to the ideal world and create a simulator

$\mathcal{S}$ . This simulator also sees the inputs of the third party in addition to the public values. In our example, this means that  $\mathcal{S}$  sees the input shares of the third party and the returned value  $s$ . There are four inputs that  $\mathcal{S}$  needs to simulate to  $\mathcal{A}$ . We do not consider that the auxiliary input is given to  $\mathcal{S}$  as we want  $\mathcal{S}$  to be a general simulator that works with any adversary  $\mathcal{A}$  and, thus, interpreting the auxiliary input directly without the help of  $\mathcal{A}$  is infeasible. For the first input,  $\mathcal{S}$  can relay the input values of the third party. For the second input,  $\mathcal{S}$  can simulate the shuffled values by uniformly choosing shares, because the shuffle protocol also reshapes the values in addition to changing their order. For the adversary, this will be indistinguishable from the hybrid world if the distribution of shares is equivalent. The third input can be simulated similarly to the second one.

The fourth input is the public result of the algorithm in the hybrid world. As the simulator knows the number of values  $s$  corresponding to the condition,  $\mathcal{S}$  can randomly assign  $s$  ones to a zero-one vector. As the original values were shuffled, the adversary does not know which comparison results correspond to which input values. Hence, even if  $\mathcal{A}$  has additional information about the inputs, their view is still indistinguishable from the view in the hybrid world. The simulator and hybrid world adversary together form the ideal world adversary  $\mathcal{A}^\circ$ .

The described simulation construction shows only that the adversary cannot learn anything during the protocol execution. However, after the protocol, additional information about the secret-shared protocol outputs might become available to the adversary  $\mathcal{A}$ . To show that this does not give an additional advantage to the adversary compared to the ideal model, we must show that the joint output distribution of all parties in the ideal and hybrid world coincides. However, when the protocol is correct and we are working with a static semi-honest adversary, this is not a problem as the outputs of the honest parties are the same in both models. In other settings, the formal reasoning is more difficult, as the adversary can influence the outputs of honest parties in the hybrid model. Further details can be obtained from [61, 39, 24].

# CHAPTER 3

## SECURE GENOME-WIDE ASSOCIATION STUDIES

### 3.1 Motivation

The need to merge gene banks of different countries for joint genome-wide association studies (GWAS) has been a topic of discussion for several years. The Estonian Genome Center at the University of Tartu (EGCUT) has collaborated with scientists from different countries by giving them genotype and phenotype data that have been gathered from Estonian donors<sup>1</sup>. This kind of sharing, however, requires rigorous procedures to be followed as the personal data being processed is considered highly sensitive medical information [31, 63]. This process usually involves the participating institutions to apply for an Ethics Committee (EC) approval for the studying and sharing of these data.

The EC approvals take time to obtain and the study has to be very well defined beforehand. But what if the study is observational, what if the scientists do not yet know whether there is a basis for their hypotheses or whether they are so improbable that there is no point in looking in the chosen direction? It would be useful if there were a technology with which to do this preliminary testing for trend to see if there is a basis for the posed hypothesis or not. Only after a positive preliminary result is found, will the study be fully specified, the EC approval obtained and the analyses carried out.

Usually, statistical studies benefit from extra data as new dimensions can be added to the analyses. With this in mind, the National Institute of Health (NIH) in the USA published ratios of SNP alleles used in different case-control studies [23] assuming that it is virtually impossible to split a DNA mixture into individual genotypes. However, it has been shown that it is indeed possible to identify,

---

<sup>1</sup>Research projects of EGCUT: <http://www.geenivaramu.ee/en/projects>

with high probability, whether a specific donor’s DNA is present the mixture [43]. Now consider that case-control groups are formed based on phenotype data. For instance, the study could be looking at a specific age group in different genders. In most cases, this is not harmful, as the attackers are looking for a certain person and most probably already have this information about their target as for most people this information is inferable. On the other hand, case and control groups can be formed based on sensitive health or disease information, which many people would prefer to keep private. Again, we see the need for a technology with which data can be shared and analysed without revealing their contents.

Secure multi-party computation (SMC) seemed to be a good solution to this problem in theory. However, a practical solution using this technology adds a significant overhead to computation time. SMC had been successfully used in practice for Danish sugar beet auctions [15] and for analysing financial data [14]. We wanted to go a step further to find out whether this technology could also be applied to provide bioinformaticians with the possibility to carry out privacy-preserving large-scale GWAS. The problem was twofold—firstly, we wanted to see whether the data analysis algorithms could be implemented using existing technology which is based on integer arithmetic, and, secondly, whether the implemented methods would be fast enough to be feasible in practical use with large genotype databases.

## 3.2 Genome-wide association studies

A single-nucleotide polymorphism (SNP, pronounced *snip*) in humans is a variation in the DNA sequence where a single nucleotide differs in paired chromosomes, see Figure 3.1. Although all four nucleotides (adenine (A), cytosine (C), guanine (G), and thymine (T)) can be located in a SNP site, usually only two alleles A and B are considered instead. The first (A) corresponds to the reference sequence and the second (B) represents potential mutations. SNP data is stored as pairs of chromosomes in text format. The possible data pairs are AA, AB, BB, and NN if the measurement could not be completed for some reason.

First, we found out what the most commonly used statistical analysis methods are for GWAS. Most often, the dataset is divided into case and control groups and then a standard  $\chi^2$  test for trend is applied. We also looked into three more specific tests that, for reasonable sample sizes, are also distributed according to the  $\chi^2$  distribution. Namely, we chose the  $\chi^2$  test that looks at the equiproportionality of allele A in both groups [71], the Cochran-Armitage test for trend [4, 66] and the transmission disequilibrium test [68] that is applicable only in a more distinct case, specifically when we are dealing with parent-child trios.



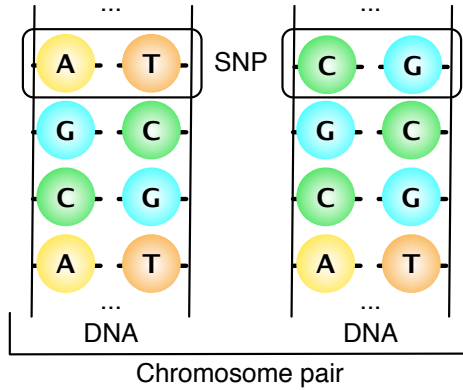


Figure 3.1: Single-nucleotide polymorphism in paired chromosomes

### 3.3 Our solution

Our solution is described in detail in the article [48]. For our specific technological platform, we decided to use the SHAREMIND secure multi-party computation system. Our aim was to show that secure computations for specific problems can be performed fast when implemented using a generalised platform without spending too much time designing a very problem-specific program.

The first obstacle we faced was how to encode the data into the database. At that time, the SHAREMIND database could only hold integer data, but as mentioned in Section 3.2, SNP data are stored in text format. Today, it is possible to store also floating-point numbers, Boolean values and text values in a SHAREMIND database. However, the actual statistical tests are done not on the text but on the counts of alleles A and B in each SNP. So even with the capabilities of the platform available today, we would not store the data as text but would opt for a categorical data representation instead.

If the SNP data contained only A and B, it would be possible to encode these data as one vector, where the element holds the count of allele A and the count of allele B can be calculated by subtracting the count of allele A from 2. However, as not all SNPs can be measured for all donors, we decided to use two vectors instead of one to store these data.

Consider a pair of integers  $(A, B)$  where  $A$  is the count of allele A and  $B$  is the count of allele B in the current SNP. This means that pairs AA, AB, BB and NN are encoded as  $(2, 0)$ ,  $(1, 1)$ ,  $(0, 2)$ , and  $(0, 0)$ , respectively. Hence, we have twice the number of data vectors that we would have if we stored the data as text. See Tables 3.1 and 3.2 for an example. In this format, the allele counts for the  $\chi^2$  tests can be calculated easily, however, the calculation will take extra time for the

Donor ID	SNP <sub>1</sub>	SNP <sub>2</sub>	SNP <sub>3</sub>
123	AA	AB	NN
124	AA	AB	AB
125	BB	BB	AB

Table 3.1: SNP data in text format

Donor ID	SNP <sub>1</sub> A	SNP <sub>1</sub> B	SNP <sub>2</sub> A	SNP <sub>2</sub> B	SNP <sub>3</sub> A	SNP <sub>3</sub> B
123	2	0	1	1	0	0
124	2	0	1	1	1	1
125	0	2	0	2	1	1

Table 3.2: SNP data given as counts of alleles A and B

Cochran-Armitage test.

Having found a solution for the data representation issue, we went on to implement the tests in SECUREC. For our test data, we chose 270 publicly available genotypes from the HAPMAP project [45]. Each of the donors in that dataset has 262264 measured SNPs. The chosen test dataset gave us a real-world dataset size and with it we ran into another problem area—we had never tested our SHAREMIND for such a large-scale analysis problem. The issue that we ran into first was with how to store such a large dataset in the database as we needed each SNP to be available as a separate element.

It turns out not to be a trivial task to find a database engine that can hold 270 rows and 524528 columns. Therefore, we transposed the database and created 270 columns and 524528 rows, instead. Essentially, such a step changes nothing in the computation process, the only constraint is, that the underlying database engine must be capable of querying row vectors in acceptable time. As most statistical analyses are focused on aggregations based on attribute columns (e.g., count of allele A, average age, average salary of all data donors), database engines are also quicker with these queries. The versions of SHAREMIND in use today can also support 500000 columns, so this modification need not be done in order to perform analyses on large-scale data any more.

The rest of the implementation was mostly straightforward as SECUREC is a C-like language. However, as mentioned in Section 2.3, filtered data in the privacy-preserving setting are handled, where possible, not as a vector of values that correspond to a filter but as a pair consisting of the original private value vector and a private mask that indicates which values correspond to the filter. Hence, all of the algorithms had to be composed bearing this in mind.

Fortunately, this does not complicate matters too much, as we only need to find the count of the occurrences of alleles A and B in each SNP in different groups. If we multiply the data vector with the mask vector, we get the value 0, where the element did not correspond to the filter, and the original value, where it did. When we sum the elements of the obtained vectors, we get the needed sums as unnecessary elements are obviously changed to 0.

Another aspect that we needed to address was the fact that computing all of the chosen test statistics  $T$  requires two values to be divided and then compared to a significance threshold  $T_\alpha$ . As we only had integer arithmetic available (without division), we rewrote the equation  $T \geq T_\alpha$  in terms of integer operations. Let  $T_\alpha = \frac{p}{q}$  and  $T = \frac{x}{y}$ , then the condition can be written as

$$T \geq T_\alpha \iff xq \geq yp$$

and we can do all our calculations using only integer arithmetic.

Even though not all statistical functions can be transformed to work on integer data, in the case of GWAS, integers are enough to perform the most widely-used statistical tests. Moreover, as only one division is needed at the end of these algorithms, most of the necessary calculations can be done on integers even when division is available.

As mentioned in Section 2.2, a very important aspect of SHAREMIND is using parallelisation in algorithms. This allows SHAREMIND to achieve greater speeds during computation. Luckily, the calculation of statistics for GWAS can be optimised very well. Namely, we can perform frequency calculation and hypothesis testing on all of the 262264 SNPs in parallel. To further emphasise this point, we measured performance results for the original 270 donors and also 540, 810 and 1080 donors.

It is clearly visible in [48, Table 4], that the majority of performance time is spent on frequency calculation, which makes perfect sense, considering that the filters have to be applied twice to all SNPs of all donors. The number of necessary multiplications exceeds the saturation point and, therefore, the performance results also show linear growth w.r.t. the number of donors. The evaluation methods, however, have to be applied to less data and the saturation point is not reached, which can be seen from the corresponding performance results in [48, Table 5].

Unfortunately, the performance results are not as good as we would have hoped, but they are not infeasible either. The results are 60 times slower than the results we achieved with the public version. On one hand, waiting approximately two and a half hours instead of 15 seconds for test results is quite some time. On the other hand, applying and waiting for the Ethics Committee approval for a pilot study takes more than a month. If we go back to the scenario, where

we are dealing with a pilot study to find out whether there is cause to look into the data more deeply, the solution can be considered feasible in practice. Moreover, with the new developments in the SHAREMIND platform, we estimate, that the computations can be done at least 10 times faster.

In addition, as the performance results indicate, the majority of the computation time was spent on the calculation of frequency tables. A hybrid study can be conducted, where each partner submits already pre-calculated frequencies into the database for joint analysis of cohorts. This version is not as powerful and general as we would like, but it speeds up the calculation drastically.

### 3.4 Impact of our work

Figure 3.2 depicts the algorithms we created and implemented for SHAREMIND using SECREC. All of these algorithms are available in the supplementary materials of the paper [48]. Some of the algorithms and their implementations (i.e., allele counting, determining a genotype distribution, allele counting for homozygous children, the Cochran-Armitage test and TDT), are data-specific and cannot, therefore, be generalised to data from other fields of life. The range mask compilation algorithm, on the other hand, is universal and can be used for any kind of data. The two  $\chi^2$  tests are specific in the sense that they are optimised for two groups (allele A and B) whereas the general  $\chi^2$  test can be used for  $n$  groups. But otherwise, these tests can also be applied to any kind grouped data.

The implemented functionality is not available as a separate program as it requires the use of the SHAREMIND platform. The filtering functionality and a generalised version of the  $\chi^2$  test have been incorporated into our statistics tool which will be discussed further in Chapter 5.

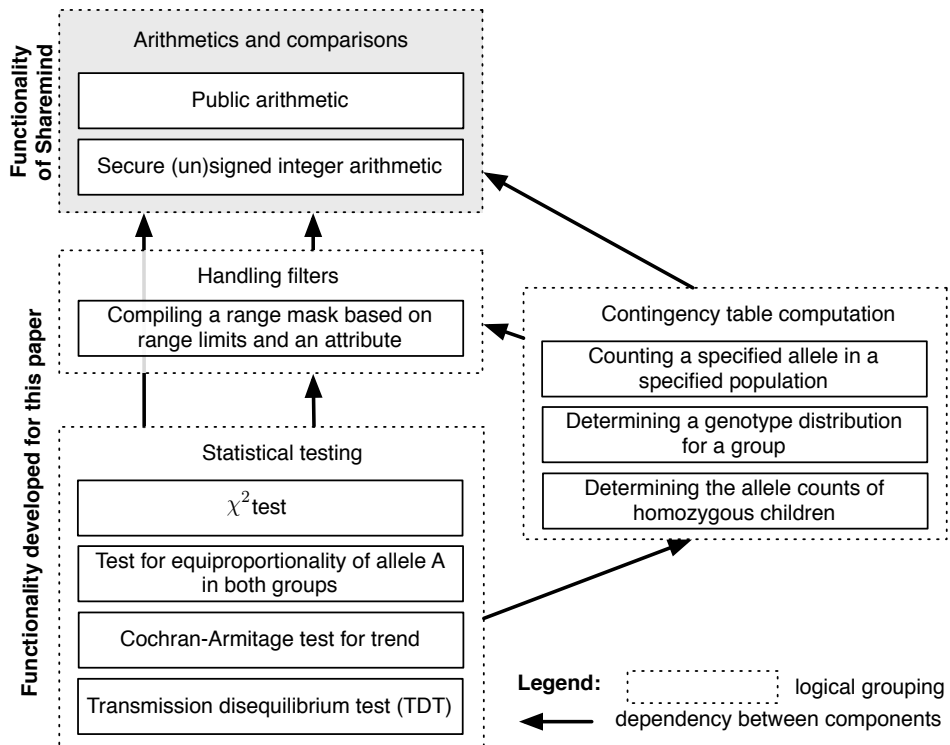


Figure 3.2: Overview of operations implemented for the privacy-preserving GWAS

# CHAPTER 4

## FLOATING-POINT ARITHMETIC AND SATELLITE COLLISION ANALYSIS

### 4.1 Motivation

Even though we were able to implement a privacy-preserving solution for genome-wide association studies using rational arithmetic implemented on top of integers, we found that such workarounds cannot easily be extended to the general case and require special attention for each separate statistical problem that one encounters. Our end goal was, however, to put together a statistical package that is similar to R<sup>1</sup> and other current data analysis tools that allow statisticians to perform analyses regardless of the nature of the data.

The main challenge with this goal was the absence of efficient secure floating-point numbers and arithmetic. So we set out to design and implement floating-point arithmetic and some elementary functions on real numbers for the SHARE-MIND system, namely inverse, exponentiation, square root, natural logarithm and sine. The availability of floating-point arithmetic gives us the opportunity to also design and implement other more complex algorithms and applications that would not be possible using only integers.

### 4.2 Background

Most floating-point arithmetic implementations in modern computers follow the standard IEEE 754 [44]. There is also a more generic presentation given in Section 4.2 of the book [51]. In these approaches, the representation of a floating-point number  $N$  is split into the following parts:

- Base  $b$ ,

---

<sup>1</sup>The R Project for Statistical Computing. <http://www.r-project.org>

- Sign  $s$ ,
- Significand  $f$  representing the significant digits of  $N$ ,
- Exponent  $e$  showing the number of places the radix point of the significand needs to be shifted in order to produce the number  $N$ ,
- Bias  $q$  is a fixed number used to make the representation of  $e$  non-negative.

In most representations, the base is 2, but it can be 10 as well. The sign takes a value 0 or 1 and corresponds to plus and minus signs respectively. The significand usually belongs to a fixed interval like  $[1, 2)$  or  $[1/b, 1)$ . Instead of the actual exponent  $e$ , the floating-point representation stores a biased exponent  $E$ , such that  $e = E - q$ . These notations give us the generic representation for floating-point numbers:

$$N = (-1)^s \cdot f \cdot b^{E-q}.$$

In the standard representation, the computing entity has access to all the bits of the representation. Hence, some optimisations are introduced into the representations, such as storing the sign, exponent and significand in one machine word and leaving the leading digit of the significand out of the representation, as it is always 1 due to normalisation. Hence, before computation is carried out, the compressed values are *unpacked* and after computation the result is once again *packed*. In the standard single-precision (32-bit floating-point) representation, the sign takes up 1 bit, the exponent 8 bits and the remaining 23 bits hold the significand. The double-precision (64-bit floating-point) representation has a 1-bit sign, an 11-bit exponent and a 52-bit significand. The base and bias are fixed beforehand and are not stored in the number representation.

## 4.3 Our solution

### 4.3.1 Floating-point arithmetic

Our solution is described in detail in the article [49]. In this section we describe the interesting design and implementation details that emerged during the process of creating privacy-preserving floating-point arithmetic.

In case of an additive secret-shared representation of numbers, it is not as straightforward to access the individual bits of the value as it is in the standard representation. Separate bits can be accessed using bit extraction [12, 13] but this involves a significant amount of computation and is, therefore, time-consuming. Here, using bitwise shared values would be a solution. SHAREMIND supports bitwise shared values, but does not have all the necessary operations for manipulating these values that we need for floating-point arithmetic. Hence, we share

the parts of floating-point values additively, and keep them in separate structures inside the SHAREMIND platform. For the end-user, they will still be presented as whole floating-point values.

This decision also makes the next design issue simpler for us—if we treat the parts of floating-point numbers as separate integers, we can skip the packing and unpacking steps. It is true that this constitutes a growth in memory consumption as instead of one machine word, we now need almost twice as much space. However, considering that currently in secure multi-party computation, the problem lies in performance rather than storage, this choice was fairly easy to make.

Hence, our floating-point number consists of three separately shared values  $s$ ,  $e$  and  $f$  signifying the sign, the significand and the exponent. They are stored in 8, 16 and 32 bits, respectively. This choice was induced by the following reasoning.

Though the sign is only 1 bit, the shortest machine word in SHAREMIND is 8 bits, so a Boolean value is still stored in 8 bits. Therefore, it made sense to show this explicitly and store the sign as an 8-bit unsigned integer. Additionally, the necessary operations used to implement floating-point arithmetic were available for unsigned integers values.

The choice of a 16-bit unsigned exponent instead of using 8 bits stems from a performance concern, namely that there are two comparison protocols [13] in SHAREMIND—a faster one that is based on bit shift right and works if the most significant bit is 0, and a slower one that uses bit extraction and works in all cases. If we use the standard 8-bit representation, we cannot ensure that the most significant bit is 0, in fact in about half the cases it will not be, hence we would have to use the slower comparison protocol. However, if we use more storage, we can have a faster protocol. The tradeoff is sensible, as we want the protocol to be as fast as possible at the cost of everything but precision. In fact, if we were to compress our floating-point representation into machine words, we would need two 32-bit words or one 64-bit word anyway. So in this sense, 8 bits versus 16 bits does not make a difference.

Computations on significands in the standard representation are performed by first raising the significand to 32 bits and then truncating it after the result is obtained. Therefore, we saw no reason to keep the significand smaller than the 32-bit machine word. In fact, our computations are a bit more precise than the IEEE operations due to this decision. For a comparison of the standard presentation and our version see Figure 4.1.

When we get a standard floating-point number, we unpack it, share the different parts of the value separately, and keep it in our unpacked format while the value is shared form. Upon declassification, we declassify the parts and, finally, pack the number into the standard format.

We also allow double-precision floating-point values, where the sign and ex-



### Standard 32-bit floating-point value

1-bit sign



### SHAREMIND 32-bit floating-point value

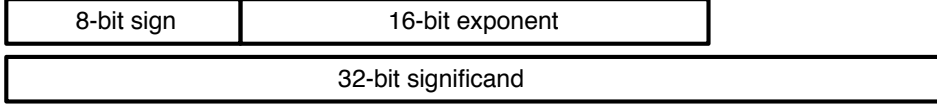
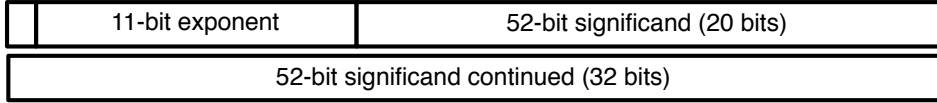


Figure 4.1: Comparison of the standard single-precision floating-point presentation and our solution

### Standard 64-bit floating-point value

1-bit sign



### SHAREMIND 64-bit floating-point value

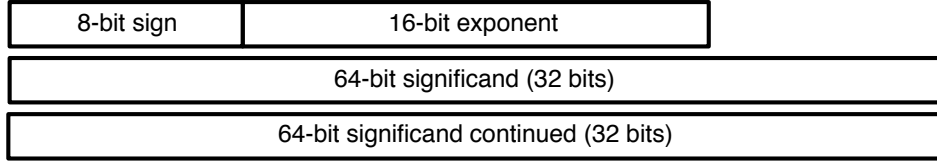


Figure 4.2: Comparison of the standard double-precision floating-point presentation and our solution

ponent are stored in 8 and 16 bits, similarly to the 32-bit floating-point values. The significand is stored using 64 bits. For this case, similar reasoning applies as for the 32-bit case. For a comparison of the standard presentation and our version see Figure 4.2.

For all components, we use unsigned integers of the mentioned length. The sign is 1 if the value is positive and 0 if negative. This reversal of the standard format allowed us to reduce the integer circuits that make up the floating-point operations. In the case of exponents, we use all of the available 15 bits for storage, leaving the highest bit free for optimisation purposes, hence introducing a bias of  $2^{14} - 1$  for both single and double-precision values.

The significand represents the real number  $\bar{f} = f \cdot 2^{-32}$ , i.e., the highest bit of  $f$  corresponds to  $\frac{1}{2}$ , the second one to  $\frac{1}{4}$  and so forth. As with the standard representation, we always normalise the significand after computations, i.e., we

adjust the exponent so that the most significant bit of the significand is 1 unless the number to be represented is zero. In the latter case,  $f = 0$ ,  $s$  corresponds to the sign of the value, and the exponent  $e$  is 0. Our single-precision floating-point values achieve 32-bit significand precision, falling between IEEE single and double-precision. The double-precision floating-point values achieve 64-bit significand precision which is more than the IEEE double-precision numbers have.

In the standard representation of floating-point numbers, there are special values that represent exponent under- and overflows. Such exceptions as under- and overflows, division by zero, square root of negative values, are problematic in the case of secure multi-party computation. Error messages in the control flow can leak information about the shared values which raises a serious issue. However, when writing a program using standard techniques and normal programming languages, the programmer can take steps to avoid these kinds of exceptions by finding out what the nature of the data is and using measures accordingly. For instance, when there is a doubt that division by zero is a possibility in the algorithm, the programmer can catch this error by comparing the divisor to zero, and at a convenient point, giving an error message based on the comparison result. This example can be seen in the algorithms for solving sets of linear equations in [9]. However, in the simple statistical measures that are described in this dissertation, these exceptions rarely, if ever, occur. Furthermore, as our exponents are larger than in the standard version, we can catch the overflow and, during declassification, set the corresponding parameters in the standard representation.

As our floating-point data structure uses separate values to keep the components, we can also introduce flags for overflow, underflow and values that are not applicable, such as when dividing by zero. These will make the data structure larger and the comparisons for over- and underflow will have to be done after each addition and multiplication, but the flags will provide stricter guarantees that these values will not be lost in some borderline cases. This will also make our floating-point values more compatible with the IEEE standard. Note, that these flags are not included in the current implementation.

We designed algorithms for adding and multiplying floating-point values that use the described secret shared format, and implemented them for the SHARE-MIND platform. Of the two, the multiplication protocol is the simpler one, as the floating-point value representation is multiplicative. Essentially, we use the comparison protocol to determine the sign of the result and add exponents to get the exponent of the result. To get the significand, we cast the  $n$ -bit argument significands to  $2n$ -bits and multiply those, truncate them to  $n$  bits again and normalise the significand. Normalisation shifts the significand so that the significant bit is one and adjusts the exponent according to the shift. For both addition and multiplication, we designed and implemented all the necessary subprotocols, including

normalisation, casting between different bit widths and shifting a value right by a secret number of bits. These subprotocols are described in [49].

Addition is a bit more complicated, but in general, we align the decimal points of the values, and either add the values, if the signs are equal or subtract them, otherwise. For addition, we need to cast the  $n$ -bit significands to  $2n$  bits and back after the addition and adjust the significand accordingly. For subtraction, we need to normalise the result at the end as this operation can end with a non-normalised significand.

### 4.3.2 Elementary functions

When we had privacy-preserving composable floating-point arithmetic, we were able to go on to implement some elementary functions, namely inversion, square root, sine, exponentiation of  $e$ , natural logarithm and error function. We used Taylor series expansion for algorithm design and implementation as polynomial evaluations use only addition and multiplication operations, and are, therefore, perfect for our solution. They also offer data-independence which is important in the privacy-preserving setting. Standard implementations of elementary functions offer precision within a small relative error. Taylor series expansion allows us to do the same.

The Taylor series of an elementary function is an infinite sum of terms that can be used to approximate the corresponding elementary function at a given point. The approximation is done by computing a finite number of terms. This allowed us to introduce a time-precision trade-off. With Taylor series expansion, the more elements are in the series, the more precise the result. Even though computing fewer elements is significantly faster, it leaves us with a result that cannot be considered precise enough.

In most computers today, elementary functions such as square root are calculated in hardware. However, early personal computer processors such as Zilog Z80 CPU as used in the Spectrum ZX often used Chebyshev polynomials [3] to approximate the calculation of elementary functions. If arguments fall into fixed intervals then Chebyshev polynomials can be used to achieve better precision with a smaller number of elements. The coefficients for the polynomials are pre-calculated based on the desired precision and are faster to use in later calculations. As the current performance of secure computation is comparable with that generation of computing hardware, we were inspired to try these polynomials in our platform as well.

We implemented Chebyshev polynomials for inverse and exponentiation of  $e$  as division is most commonly used in algorithms and the error function is the slowest of the elementary functions. As Chebyshev polynomials exist for the other functions as well, they can be easily implemented when need arises.

---

**Algorithm 1:** Protocol for finding the sine of a floating-point value  $\llbracket N \rrbracket$  with precision  $p$

---

**Data:** Shared floating-point value  $\llbracket N \rrbracket$

**Result:** Shared floating-point value  $\llbracket N' \rrbracket$  such that  $N' = \sin(N)$  with precision  $p$

- 1 Evaluate the Taylor series  $\llbracket N' \rrbracket \leftarrow \sum_{n=1}^p \frac{(-1)^{n-1}}{(2n-1)!} \cdot \llbracket N \rrbracket^{2n-1}$
  - 2 **return**  $\llbracket N' \rrbracket$
- 

In theory, higher order Chebyshev polynomials also provide better precision similarly to Taylor series. In our implementation, however, we only provide algorithms using a fixed number of elements, hence, the user cannot ask for better precision at the cost of running time with the algorithms using Chebyshev polynomials. For instance, Taylor series evaluation for inverse requires at least 32 summation terms to achieve the precision we need so we also implemented Chebyshev polynomials for inverse using 7 elements. The significant performance gain justifies the use of Chebyshev polynomials, however, we still leave the option of using the slower Taylor series, if precision is paramount in an application.

As mentioned, the SHAREMIND platform uses parallelised computations to achieve better performance. Thus, in our implementation, we perform as many polynomial evaluation computations in parallel as possible.

Before moving to the specific functions, let us look more closely at how Taylor series expansion works. With some clever tricks, we can modify our problem so that we do not need to approximate the series over the whole real number range. We call the range that interests us the region of approximation. We need to have enough elements in the Taylor series, so that the approximation in the chosen region has a small relative error.

*Sine* is the most straightforward of the elementary functions. It is a periodical function, with its period spanning  $[-\pi, \pi]$ , so its Taylor polynomial converges for all real numbers modulo  $2\pi$  already at a few elements. Hence, we decided to start with implementing this function to try out Taylor series expansion in practice. As this algorithm is not given in [49], we include it here for completeness as Algorithm 1.

*Inversion* is needed to implement division which is an essential operation in statistical analysis and most other real-life applications. Division itself works, as expected, by inverting the divisor and multiplying the result with the dividend. Inverse converges best in the interval  $(0, 1]$ , so we separate the significand  $f \in [\frac{1}{2}, 1)$  from the given floating-point number and, essentially, based on the

---

**Algorithm 2:** Protocol for finding the natural logarithm of a floating-point value  $\llbracket N \rrbracket$  with precision  $p$

---

**Data:** Shared floating-point value  $\llbracket N \rrbracket$

**Result:** Shared floating-point value  $\llbracket N' \rrbracket$  such that  $N' = \ln(|N|)$  with precision  $p$

- 1 Let  $\llbracket f \rrbracket$  be the significand of  $\llbracket N \rrbracket$
  - 2 Set  $\llbracket x \rrbracket \leftarrow \llbracket f \rrbracket - 1$
  - 3 Evaluate the Taylor series  $\llbracket \ln_f \rrbracket \leftarrow \sum_{n=1}^p \frac{(-1)^{n-1}}{n} \cdot \llbracket x \rrbracket^n$
  - 4 Set  $\llbracket \ln_e \rrbracket \leftarrow \llbracket e \rrbracket \cdot \ln(2)$
  - 5 **return**  $\llbracket N' \rrbracket = \llbracket \ln_f \rrbracket + \llbracket \ln_e \rrbracket$
- 

following equation, only inverse the significand using Taylor series:

$$\frac{1}{f \cdot 2^e} = \frac{1}{f} \cdot 2^{-e} .$$

*Natural logarithm* Taylor series only converge in the interval  $(0, 2]$ . Hence, we separate the significand  $f \in [\frac{1}{2}, 1)$  and base our algorithm on the fact that

$$\ln(f \cdot 2^e) = \ln(f) + e \cdot \ln(2) .$$

Similarly to the protocol for finding sine, we include the algorithm for finding the natural logarithm in this dissertation for completeness. Algorithm 2 shows how the natural logarithm is computed. First, we deal with the natural logarithm of the significand. On line 1, a float  $\llbracket f \rrbracket$  is created containing the absolute value of the significand of the original value  $\llbracket N \rrbracket$ . In our underlying representation this can be achieved simply by setting the sign bit to be 1, leaving the significand of  $\llbracket N \rrbracket$  untouched and setting the exponent to be 0.

*Square root* is the most complex of the elementary functions we implemented. The convergence interval of the series is  $(0, 2)$ . Thus, we first computed the square root of the significand and then multiplied the result with a correcting exponential term:

$$\sqrt{f \cdot 2^e} = \sqrt{f} \cdot 2^{e/2} .$$

In addition, to avoid divergence for the square root of zero, the corresponding result is obviously set to zero at the end of the algorithm.

*Exponentiation of  $e$*  converges for all real numbers, however, the larger the value, the higher the number of elements in the Taylor series has to be to keep acceptable precision. As expected, this increases the working time of the algorithm. As a solution we changed bases from  $e$  to 2:

$$e^x = (2^{\log_2 e})^x = 2^{(\log_2 e) \cdot x} \text{ and } y = (\log_2 e) \cdot x \approx 1.44269504 \cdot x .$$

Now, knowing that  $2^y = 2^{[y]+\{y\}} = 2^{[y]} \cdot 2^{\{y\}}$ , where  $[y]$  denotes the integer part and  $\{y\}$  the fractional part of  $y$ , we separated the integer and fraction parts of the power  $y$ . As our floating-point works on base 2, we were able to simply construct  $2^{[y]}$  and compute  $2^{\{y\}}$  which is always in the interval  $(0, 2)$ .

*Error function* is a function that describes diffusion and is defined as

$$\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt.$$

It is used for approximating numerical integration which is needed for the satellite collision analysis algorithm. At larger values, several elements are needed for the Taylor series, hence, we did a rough approximation and checked whether the exponent is larger or smaller than 3. This shows us whether  $x \in [-2, 2]$  and we were able to obviously set the result as  $\pm 1$  according to the original sign  $\llbracket s \rrbracket$ .

### 4.3.3 Matrix operations

We implemented some of the most commonly used matrix operations, as these are also often used in statistical analyses. Unit vector computation, scalar product, cross product for vectors of length 3 and multiplication of two-dimensional matrices were straightforward to implement in the privacy-preserving setting. We also implemented a specific algebraic algorithm for computing the eigenvalue and eigensystem for  $2 \times 2$  symmetric matrices.

## 4.4 Satellite collision analysis

As a validation of large-scale usage of floating-point arithmetic and the elementary functions, we used our brand new capabilities for implementing an algorithm for satellite collision analysis in SECREC. Satellite collision analysis is more of a geometric than a statistical problem, however parts of the analysis (mainly matrix manipulations) can also be useful in statistical analysis.

In pragmatic terms, we chose satellite collision analysis as a prototype application for floating-point because of its complexity. This analysis works on floating-point data, uses most of the elementary functions we implemented, and needs both high precision and relatively fast performance. With GWAS, the only thing that can run out is the analyst's patience. Satellite collision analysis, however, must be performed on a fixed number of satellite pairs during a fixed period, e.g., in 24 hours, to find all possible collisions for the next period. Hence, we are dealing with a more performance-critical problem, as delayed predictions can leave too little time for avoiding a collision. Implementing this complex algorithm helps us validate whether privacy-preserving computation can be used in a performance-critical setting.

On the other hand, we chose this problem, because it is an ideal setting for secure multi-party computation—no country is willing to let other countries know the location of their intelligence satellites, but they would also like their investment to stay safe. Secure multi-party computation could be used to analyse the locations and let satellite owners know whether they need to steer their satellites away from potential collision situations. Even though the privacy-preserving solution is perfect for this problem, its use is a very delicate political decision for countries for national security reasons. One of the difficulties is the decision of how many collision probability queries and satellites can be allowed for one country so that they could not probe for the location of other satellites. Hence, it will be some time until this solution will be used in practical applications.

We based our implementation on the algorithms from [1, 2, 41] that compute the probability of collision between two spherical objects. For this, the support for floating-point addition, multiplication, division, square root, exponentiation of  $e$ , matrix and vector operations and integral computation is needed which requires the use of error function computation.

## 4.5 Impact of our work

Figure 4.3 depicts the algorithms we created and implemented for SHAREMIND using C++ and SECREC. The first layer in the new functionality section is implemented in C++ and is an integrated part of code of the SHAREMIND platform. The second layer contains the functionality necessary for performing privacy-preserving satellite collision analysis and is implemented in SECREC as a standard library.

For finding the collision probability, we need to compute a double integral. We did this by using Simpson’s one-third rule for numerical integration. This rule transforms the probability formula into a form which we can compute if we have access to floating-point operations for multiplication, addition, division, square root, exponentiation of  $e$  and the computation of the error function. Though we implemented a version of this rule that is optimised this for the application specific function, it can be modified to accept other functions when needed.

Detailed algorithms for our floating-point arithmetic and all elementary functions except sine and natural logarithm are available in the paper [49]. These two algorithms are included in this dissertation for completeness and their implementations are included in the SHAREMIND system along with the rest. The benchmark results indicate that our privacy-preserving application is fast enough to work on the satellite collision problem in real time.

Today, the described floating-point arithmetic is used by the SHAREMIND platform. The elementary functions have been further developed using fixed point

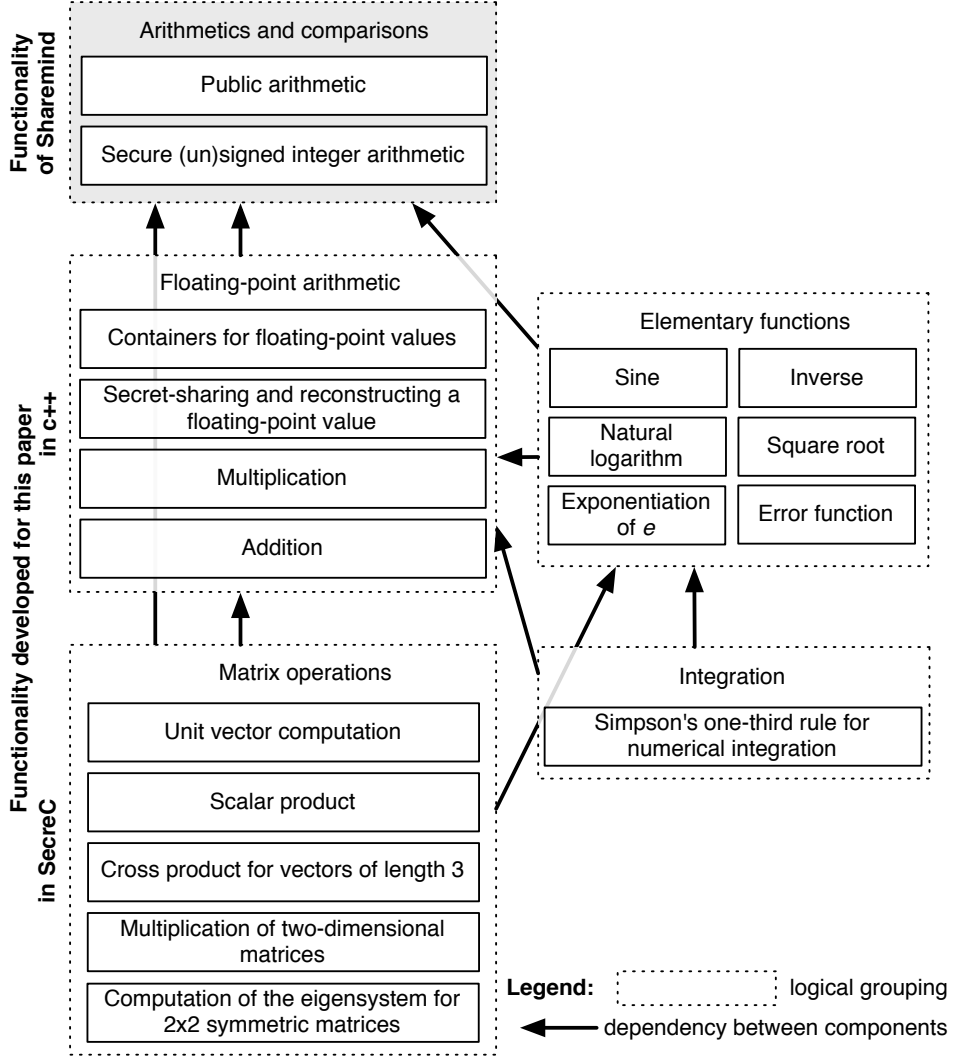


Figure 4.3: Overview of operations implemented for the privacy-preserving floating-point arithmetic and satellite collision analysis

computations and Chebyshev polynomials with more elements where possible. The algorithms essentially follow the same pattern as the algorithms of this work, but fixed-point computations are used when it is known where the decimal point lies. For all our elementary functions, we fix the approximation range and with this, the decimal point. For example, the inversion algorithm can utilise fixed point computations during the phase where the significand  $f \in [\frac{1}{2}, 1)$  is being inversed. Fixed-point arithmetic is faster, as many of the special cases that occur



in floating-point, can be ignored.

These so-called hybrid floating-point functions are a continuation of this work and are described in the article [53]. They offer significant speedup compared to the versions described in this work. The elementary functions implemented in C++, are available as function calls in the SECRES language.

Other members of our team have created another version of floating-point arithmetic that is IEEE 754 compliant [62]. They use garbled circuits to design and implement their operations on XOR-shared values in the secure multi-party setting. Their implementation is faster than ours when run on fewer than a 100 floating-point values. Unfortunately, the solution does not scale very well and, therefore, our solution outperforms theirs when more than 100 operations need to be done in parallel. However, we intend to integrate the two solutions to use the advantages of both.

As mentioned, the reusable vector and matrix algorithms are part of the standard library. The satellite collision analysis algorithm is used as a demo application. Because of the nature of the SHAREMIND platform and the SECRES language, the algorithms implemented in SECRES can use the newest and fastest elementary functions available without being reimplemented.

# CHAPTER 5

## A PRIVACY-PRESERVING STATISTICAL ANALYSIS SUITE

### 5.1 Motivation

In most areas of life, data are often gathered and stored for different reasons, be it for archival, for finding customer shopping patterns, for detecting tax fraud, or for better decision-making. Even though data are sometimes stored just because they are available, it is often wise to analyse them and make some conclusions that can be helpful for the business, for the government, or for the individuals whose information has been gathered.

There are several different tools available for simplifying the statistical analysis process—SAS, SPSS, R, to name a few. Using these, the statisticians do not have to implement all the necessary statistical tests every time they want to analyse new data. These tools, however, can only process unencrypted data. It is fairly simple to secret-share data and store them in a database. It is a different matter to analyse them feasibly and in a manner that is easily approachable to a data analyst who is used to working with unencrypted data where they can look at individual values.

Our aim was to develop such a general tool that allows simple data analysis for any kind of database. By this we mean a tool that allows the analyst to explore the database and find a basis for more specific studies or simply get an overview of the trends. We also wanted the tool to be intuitive and easy-to-use, e.g., similar to an existing statistical analysis tool like R. We chose R because its core is open-source, and hence it is easier to see, what kinds of algorithms are used within and how we could integrate our own algorithms as a package of R. We decided to first try to build the tool as a separate entity and later integrate it as an R package if this was needed and reasonable.

This kind of general tool allows us to set up studies and analyses more easily,

as some of the most often used tests and functions can be incorporated into the tool. This leaves only the tests and functions that are analysis-specific or more complex to be developed when need arises.

## 5.2 Background

Statistical analysis is a very wide and varied field. There are simple functions for getting an overview of data and very complicated algorithms for finding interesting details. We decided to focus on statistical tests and functions that are most commonly used in practice and that are not extremely specific to a certain field.

If data have not been gathered or merged for a certain study, but are collected and then analysed for interesting patterns, the analysts usually look at the values to formulate hypotheses based on the data distribution. This kind of access to the data is important for statistical analysts. Unfortunately, the privacy-preserving setting does not allow the user access to individual records. Therefore, we looked into descriptive statistics that allow us to give the analyst an overview of the data and their distribution without revealing individual values. For this, we chose the five-number summary that computes and returns the minimum, maximum, median, and 25% and 75% quantiles from a given vector. These statistics can be visualised using box-plots that give the analyst a nice visual description of the dataset or different groups within. We also decided to include an outlier detection and elimination function based on quantile calculation.

We chose the histogram and heatmap for showing the distribution of data in a dataset. A statistical tool also needs such components as mean, variance, standard deviation, and covariance. From statistical tests, we decided to reimplement the  $\chi^2$  test as it is one of the most commonly used tests in statistical analysis. In our first version of secure GWAS, we had to jump through hoops to perform this test as we did not have floating-point arithmetic. We implemented Student's t-test and we also chose the Wilcoxon rank sum test as its non-parametric counterpart. In addition, we implemented the paired t-test and, for completeness, the Wilcoxon signed rank test that also works on paired data.

Correction algorithms for multiple testing are designed to account for alternative hypotheses that can be falsely accepted, if multiple tests are run on the same data. Usually, the p-value cut-off is such that it is unlikely that the computed p-value is at least as extreme as the threshold if the null hypothesis holds. However, if several tests are run on the same data, it becomes more likely that one of these tests returns a false positive. Hence, genome-wide association studies use multiple testing when more than one SNP is being analysed at one time. We decided to implement Bonferroni correction and the Benjamini-Hochberg procedure for false discovery rate correction.

The most complex algorithms we chose to implement, were those of predictive modelling, mainly linear regression. For this, we looked at different algorithms used to solve sets of linear equations, namely, Gaussian elimination with back substitution, and LU decomposition. Additionally, we implemented the conjugate gradient method for quadratic forms.

## 5.3 Our solution

### 5.3.1 Descriptive statistics and statistical tests

Our solution for the statistical analysis methods is described in the articles [8, 9]. In this section, we give an overview of the solution and bring out the more interesting details of design and implementation of the methods in the privacy-preserving setting.

As mentioned, descriptive statistics can give an analyst an overview of the dataset without actually revealing individual records. For this purpose, we designed and implemented two versions of the five-number summary. The first uses the cut function and reveals the number of records corresponding to the filter. The other is oblivious and does not reveal the number of records, but this comes at the cost of increased running time. As mentioned previously, when data are filtered in the privacy-preserving setting, the number of records that are left in the dataset is not revealed. Computing the faster five-number summary on these records will reveal this number whereas the oblivious version will keep it a secret. This choice depends on the study as, in some cases, revealing the number of records is a desired side effect.

We implemented the computation of one- and two-dimensional frequency tables. The computation of mean, variance, standard deviation, covariance, and different versions of the t-test became straightforward using the division and square root functionality. We implemented all statistical tests according to the paradigm where a test statistic is computed in the privacy-preserving setting and privately compared to the critical test statistic that has been computed publicly, based on the significance threshold determined by the analyst. This paradigm is described by Option 2 of the private setting described in Chapter 2.

For the Wilcoxon rank sum and signed rank tests, the complex part was the design of the privacy-preserving ranking function. This function sorts the values and then gives ranks to each element based on their position in the sorted list. In addition, the ranks of equal values are averaged. For the privacy-preserving rank sum test, the ranking function works in the same way as in the standard setting, but much slower, as there are many comparisons and, for equal values, the average of the corresponding ranks has to be computed obliviously. We also designed a version of the ranking function that does not deal with the equal values. However,

this results in the test returning a stricter bound and some borderline cases might be falsely depicted as belonging to  $H_0$ .

The signed rank test uses a slightly different ranking function which does not give ranks to values that are equal to 0. Fortunately, the elements to be ranked are ordered based on absolute values, hence, the zero values are set at the beginning of the sorted vector. This allows us to count the number of zeros in a privacy-preserving manner and use this number as an offset for all the other ranks.

With the availability of division, the standard  $\chi^2$  test became straightforward to design and implement in the privacy-preserving setting. In addition, we now have the possibility to include as many groups and classifications as are required by the study, as opposed to the GWAS-specific solution. We did still implement the optimised version for two cases as this is a bit faster.

### 5.3.2 Linear regression

Linear regression is currently one of the favourite tools of statistical analysts, hence, we decided to design and implement some privacy-preserving algorithms for this purpose. As the linear regression task can be recast to an ordinary system of linear equations, we tried to find out what are the best general ways of solving sets of linear equations. For sets with up to four variables, the fastest and most accurate way is to simply invert the matrix by computing determinants. There are two types of linear models—those with and those without the constant term. The model without the constant term always predicts zero if all inputs are zero. The model with the constant term has a nonzero additive offset  $c$  and, thus, predicts  $c$  if all inputs are zero. The latter is more likely in a statistical setting and we account for this bias by inserting an extra variable into the linear equation to denote this offset.

We use simple inversion for sets of equations with up to four variables including the offset. To deal with more variables, we looked into Gaussian elimination with back substitution. We chose it because it is the most straightforward of the algorithms and, because with a few modifications, this method can compute the inverse matrix which can be useful in some future applications. As we had access to the necessary integer and floating-point operations (addition, multiplication and division), the algorithm was easily converted to the privacy-preserving setting.

The main difference between the standard and the privacy-preserving versions of the algorithm is in finding the pivot element. As we need to exchange rows based on the location of this element, this would be extremely slow if done obliviously. Publishing the location of the pivot element will make the algorithm significantly faster. The problem is that this leaks information about the location of the largest element in the original matrix. To counter this leakage, we start the algorithm by shuffling the linear equations as their order is not important. Now,

the location of the pivot element does not reveal its location in the input matrix any more.

We also designed a privacy-preserving version of LU decomposition, where we again used row shuffling for reasons similar to the Gaussian elimination case. We also needed to take out some optimisations that had been done for the standard setting, as these relied on the data values which are difficult to access for our algorithm. Namely, we rearrange a vector based on a given permutation instead of addressing the vector in the permuted order during execution.

We also wanted to try solving a linear regression task using an iterative method. We chose the conjugate gradient method as its initial convergence is rapid during a small number of iterations. As we had access to matrix operations from the satellite collision analysis application, this algorithm was also quite straightforward to design for the privacy-preserving setting.

Iterative minimisation methods form a core of many machine learning methods. Thus, our privacy-preserving conjugate gradient method has applications in machine learning as well. However, note that we implemented the conjugate gradient method only for quadratic forms. Whereas, in the future, we can apply our implementation in the context of support vector machines that reduce the classification problem to quadratic programming, our privacy-preserving conjugate gradient method cannot be used for a general purpose.

We put all of the implemented functions together in a tool called RMIND that resembles the R environment.

## 5.4 Impact of our work

Figure 5.1 depicts the privacy-preserving algorithms we created and implemented for these papers using SECREC. The figure is based on one from [8] but new functionality from [9] has been added.

We collected all of this functionality and the matrix operations, and put together the RMIND tool. This tool is similar to the R environment and should, therefore, be quite familiar to statistical analysts. The RMIND tool can be used to perform privacy-preserving statistical analyses in the future. Detailed algorithms for the statistical functions are available in the article [8]. From the most basic matrix operations, the only thing missing is eigenvalue calculation for matrices with arbitrary size. This can be implemented in the future using an iterative algorithm if it is needed for an analysis.

At the moment, the RMIND tool is being prepared for a privacy-preserving data analysis study that investigates whether the university drop-out rate in the ICT field is correlated to students working during their studies. We discuss this study in more detail in Chapter 6.

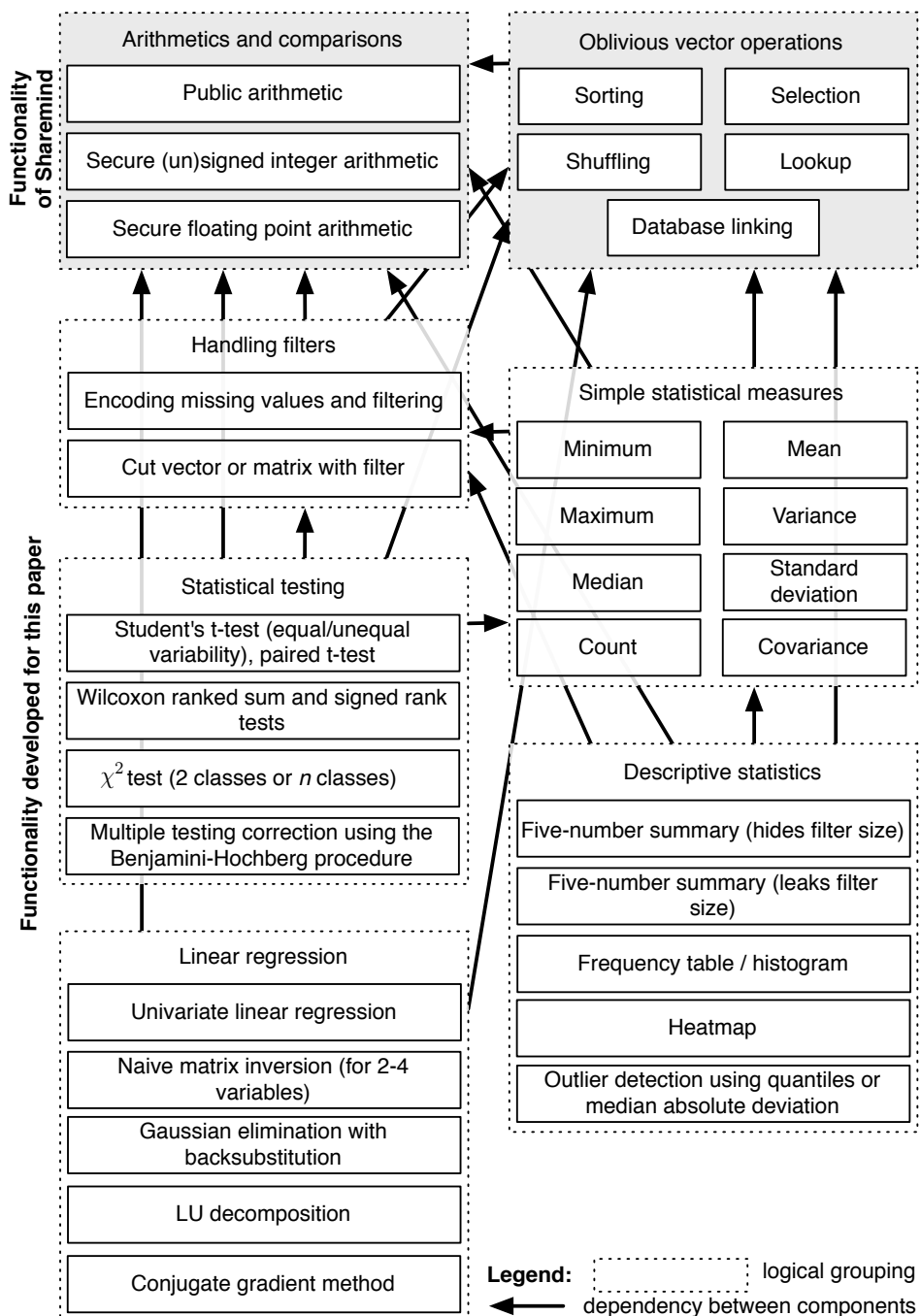


Figure 5.1: Overview of operations implemented for the privacy-preserving statistical analysis tool RMIND

# **CHAPTER 6**

## **USING THE STATISTICS SUITE IN A REAL-WORLD STUDY**

### **6.1 Motivation**

Having implemented the RMIND tool, the logical next step is to use it in practice. Even though we provide benchmark results with the tool, these are done in a laboratory setting, meaning that they only give an estimate of the performance. Using the SHAREMIND platform and the RMIND tool on real data will enable us to study the feasibility of privacy-preserving statistical studies in the real world.

First, the servers will be housed by three independent entities, with no local area network connections between them. This will give us the possibility to measure performance in a real-world setting where different issues can interfere with the work in progress. It also gives us insight into deploying the platform for different parties with different information system and infrastructure setups. However, the deployment issues will not be discussed in this dissertation, but in a separate upcoming work.

Second, using real-world data in a study introduces new interesting problems that we usually do not deal with when running tests. Real databases are seldom perfect and can contain data that are either faulty, missing, or encoded based on different rulesets. In the planned study, we will be working on the whole population, hence, the size of the cohorts will be determined by the number of actual records in the database. In addition, the different input parties will have to secret-share the data themselves and upload them to the servers as the sensitive data cannot leave the parties in the usual unencrypted format. Even though we will provide the input parties with a tool that does this for them, there are some issues that have to be dealt with. Namely, this requires that the data be described in detail beforehand in terms of data types, whether we are dealing with classifiers, whether values can be missing from a column, and most importantly, which



columns are secret-shared and which are not.

Access to real personal data also entails various legal aspects. We want to look into this as well, to see whether privacy-preserving statistical analysis using secure multi-party computation is a notion that can be adequately explained to parties without a thorough knowledge of cryptography. In addition, we want to investigate what the theoretical and practical legal arrangements are that we need to conduct such a study in the real world.

Third, the transformation of the data into an analysable format is something that we have not tried before. In a laboratory setting, the test data tables are generated in the required format. With real data, however, the format is determined by the source data. Moreover, we get data from different sources and we need to link them together. It is very difficult to abstract this extract, transform, load (ETL) process as it is very specific to the study and the data source. Therefore, a study on real-world data can help us look into this issue.

And fourth, we want to give the RMIND tool to statistical analysts and ask them to perform the analysis themselves. This will help us see what are the differences between the studies in the privacy-preserving and the standard setting based on where the analysts have trouble with their analysis. We will, of course, help them with queries and instruct them when they require assistance, but we ask them to work as they usually do on these kinds of studies.

All in all, we are not only solving an interesting statistical problem, we are also piloting the platform and statistics tool in a large-scale privacy-preserving statistical study on real data. We will help with the deployment of the platform where necessary, but the system administrators of the respective parties will be in charge of the actual process. During the study, we will also help with any questions that can arise, but as regards to data, we will only have the role of one of the computing parties and the technology provider. All other roles are filled by entities that are independent from us, most of them government institutions.

## 6.2 Problem statement

To test the platform and statistics tool in practice, we will be investigating information and communication technology (ICT) education in Estonia. ICT is a growing industry both in Estonia and the whole world. This kind of expansion brings along a surge in the number of companies and start-ups that would like to reap the benefit. For this, new personnel has to be employed. The Association of Information Technology and Telecommunications (ITL<sup>1</sup>) in Estonia has stated that they require 6000 ICT-specialists by year 2015 [32]. Estonia, however, is at a slight disadvantage considering this requirement. With our small population of

---

<sup>1</sup>ITL homepage. <http://www.itl.ee/>

around 1.3 million in January 2014 [59], and with our median wages being on the lower side in Europe [33], we have to rely on educating more and more of our students to fill this requirement, rather than expecting to make do with the existing workforce or hoping to recruit from abroad.

There are several universities in Estonia and many of them teach different specialisations of ICT. In recent years, there has been a lot of publicity surrounding this topic—universities are trying to recruit students by offering stipends and even free laptops to help with studies. Regardless, the quota is often not met and not all places are filled. What is worse, though, is the fact that a significant number of students that do enrol, quit their studies before completion. By December 2012, 43% of ICT students who started their studies in 2006-2012 had quit without graduating<sup>2</sup>.

For a small country trying to succeed in the ICT field, it is very important to have highly skilled employees. Hence, it is paramount to try to find out why so many students fail to graduate. Universities hypothesise that the high drop-out rate is connected to students being hired as early as their first year and that the students tend to favour their wages over a university degree. ITL consisting of universities and ICT companies, and the government would like to find an answer to whether the drop-out rate is higher for the students who work during their studies or whether it has to do with the higher acceptance rate.

## 6.3 Background

There are two major ways of studying this problem. On one hand, it is possible to make a survey and conduct interviews among the students who graduate and those who do not. On the other hand, Estonia has government databases that contain information about education and tax records. This allows the analysis to include the whole population and work on objective data rather than interview answers which can be somewhat biased as people might be ashamed or angry because their studies have ended prematurely.

Naturally, the education and tax records databases are not linked. In fact, the education database is managed by the Ministry of Education and Research, and the tax records are held by the Tax and Customs Board operating under the Ministry of Finance. To study the described problem, the two databases must be joined and the data from both used in the analysis. Education records give us the level and period of a person's studies, tax records tell us whether a person has been working in the ICT field during a given period of time. If these tables are joined, we can determine whether there is a relation between working and quitting

---

<sup>2</sup>Data from the Ministry of Education, 2014

studies. As we are not using a sample but the whole population, we can also use simpler statistical analyses.

The data are available in two institutions that are under the jurisdiction of the government. This should make it simple to carry out the study in the standard setting. However, these two institutions have to adhere to the same laws as companies do if a joint study is planned. The privacy issue, in this case, arises from the Personal Data Protection Act that regulates the use of education data [65, §4-§6], and the Taxation Act that regulates the use of tax data [64, §26-§30].

Needless to say that these data are useful in different analyses, but accessing them is not an easy process. While the Ministry of Education and Research can give data out for analysis under contracts with the analyst, the Tax and Customs Board cannot. The latter usually pre-aggregates data into groups that can later be used in analyses. This pre-aggregation is done in the following way. The analysts get data from the Ministry of Education and select the groups they want to analyse, such as all graduates from 2012 who graduated in nominal time. Other attributes can be added to define these groups but the more attributes there are, the fewer individuals will be included in the group. The Ministry of Education forwards the identifiers of the people in each group to the Tax and Customs Board who then outputs a list of desired attributes for each group.

This process requires there to be at least three people in one group. Unfortunately, when we are dealing with students in a country with the population of Estonia, the distinct groups are rather thin to begin with, especially in the PhD degree section. Hence, many students can be left out of the study due to this filtering process. Furthermore, the Tax Board also applies  $k$ -anonymity measures [70] to each group, decreasing the number of results even further.

Secure multi-party computation has never been employed in any statistical study involving the Tax Board and, therefore, the Tax Board has no previous position on its usage. The legal aspects of using SMC to hide tax secrets are discussed in Subsections 6.7.1 and 6.7.3.

## 6.4 The PRIST project

We started a project called PRIST (Private Statistics)<sup>3</sup> that deals with studying the described problem. Let us take a look at the parties in this analysis.

Figure 6.1 depicts the parties and the data flow in the privacy-preserving study process of the PRIST project. The Ministry of Education and Research and the Tax and Customs Board are the input parties in the study as they control the data

---

<sup>3</sup>Funded by the European Regional Development Fund from the sub-measure “Supporting the development of the R&D of information and communication technology” through the Archimedes Foundation.

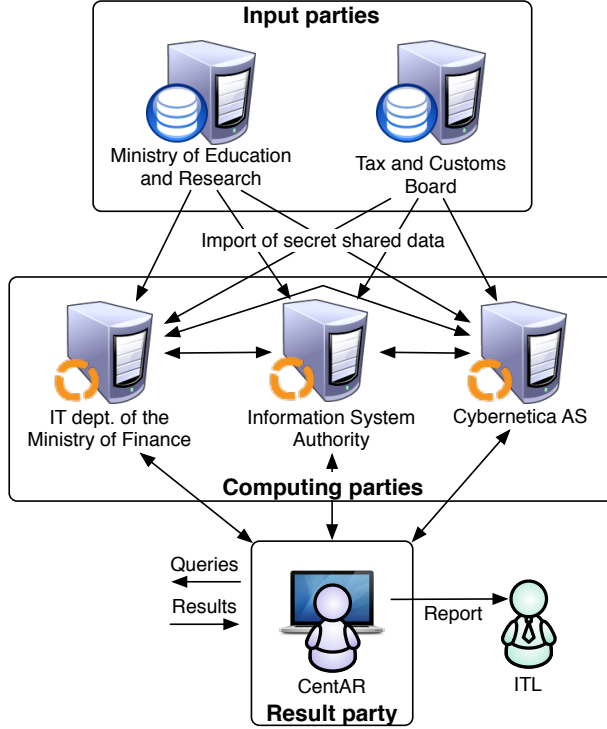


Figure 6.1: The PRIST project workflow

required for the analysis. The computing parties provide the servers that will house the platform and where the computations will be performed.

The three computing parties are The Information System Authority that coordinates the Estonian national information system, the IT department of the Ministry of Finance, and Cybernetica AS, a private company where the SHAREMIND platform is developed. The code for SHAREMIND, the data importer, and the statistical analysis tool RMIND will be reviewed by the IT department of the Ministry of Finance. The result party is the Estonian Center for Applied Research (CentAR), who will perform the statistical analysis on RMIND and publish the results as a report. This setting corresponds to the last deployment model in Table 2.2 from Section 2.2.

The Association of Information Technology and Telecommunications (ITL) is an organisation that unites Estonian information technology and telecommunications companies and universities, encouraging their co-operation and representing their interests. ITL is the customer in our study as their members receive complaints from the universities related to the early hiring of students. Their interest is to find out whether the hypotheses set by the universities have a basis to them

and whether they should start discouraging their members from hiring people still in their first years of university studies.

First, the input parties (Tax and Customs Board and Ministry of Education and Research) export the data agreed to be used in the study from their databases in comma separated value (CSV) format. Next, they insert these data into the CSV importer tool. This tool takes the data and a pre-defined data model, secret-shares the data into three parts and uploads each part into a separate remote database based on the data model. It is important to note, that the data are secret-shared at the input party side so the values do not leave these parties unencrypted. Each of the computing parties (IT department of the Ministry of Finance, the Informations System Authority, Cybernetica AS) only gets their share of each value.

Now that the data have been uploaded, the result party (CentAR) can start querying. With this type of directed study where we have already posed the study question that we ultimately want an answer to, the exploration of data is secondary and only needed for quality control. In this setting, the statistician does not need to see the data in order to choose analysis methods, as only the relevant attributes have been uploaded into the database. The statistical analyst uses the RMIND tool to make the necessary queries. The queries are sent to all the computing parties who then perform the analysis in a privacy-preserving manner. When they finish a task, test or algorithm, they get only the shares of the result, not the result itself. The shares are then returned to the result party where the RMIND tool declassifies and publishes the result to the analyst.

The statistical analyst puts together a report detailing the results received from the queries. As the RMIND tool can also return plots, the statistical analyst does not need to use another tool for visualising the results. When the report is compiled, the analyst can send it to the customer (ITL).

## **6.5 Data import and pre-processing**

Let us look at the statistical study part of the process in more detail. The data will be imported as two separate sets—the education information of students who finished or started their studies between 2006 and 2012, and their income from 2004 to 2013 to also evaluate the potential salary prior to studies. To get these two datasets into analysable format, they are extracted from their original structures, transformed into a format that is suitable for the study and loaded for analysis. This is known as the extract, transform, load (ETL) process. The main reason for carrying out this process is that real world data are rarely collected and stored in a format that is ideal or even suitable for analysis. Figure 6.2 shows the privacy-preserving ETL process for the PRIST project. The process prepares the input data from the Ministry of Education and the Tax Board as an analysis table that

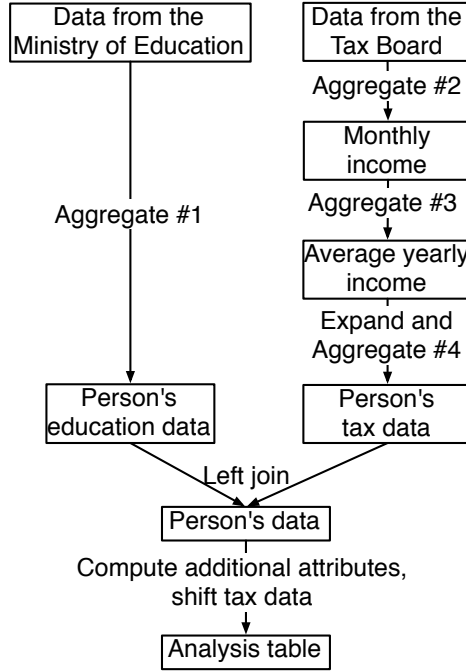


Figure 6.2: The PRIST privacy-preserving ETL process

can be used for computing the necessary statistics.

The ETL process can be roughly divided into three subtasks which will all be carried out in a privacy-preserving manner on secret-shared data. First, we want to extract the education data and transform them into a format where each row corresponds to one person's studies in one curriculum. Second, we want to extract the salary data and transform them into a format where each row corresponds to one person's salary information for all 10 years. And third, we want to join the two tables and transform the obtained table into the analysable format required by the statistical analysts. This analysis table has three types of attributes.

- Fixed attributes, such as ID or whether a person was working in an ICT company during their studies.
- Attributes ranging over years of study, such as whether a person was working during study year  $i$ .
- Attributes ranging over years after graduation, such as whether a person was working during year  $j$  after graduation.

The statistical analysts looked at the data descriptions of the two data sources and, based on these, provided us with the format for the table on which they will

perform their analyses. As they mean to use mainly filtering and summation as their tools, the privacy-preserving ETL process is expected to transform the data into a suitable format. The process will be carried out entirely in the RMIND tool using its query language. In this pilot study, we designed the ETL process ourselves based on the resulting dataset description given to us by the statistical analysts to see whether there were missing operations from the SHAREMIND system. We store the process as a set of RMIND queries that can later be run by the result party on real data.

Aggregation of the separate data tables can be performed in the public setting to speed up the process. However, this requires that the input parties transform the data into the format that the analysts need instead of outputting data in the format that their databases provide. We want to avoid giving additional tasks to the input parties as we want the process to be as close to the standard setting as possible. Another possibility for public aggregation is to have the importer tool perform these tasks. We do not view this as a good option either, as we want the importer tool to be universal for all input parties. Moreover, if these aggregations are done before data sharing and import, we cannot later query more detailed data without a new data import stage, should the study plan change. Therefore, we carry out the aggregation process in the secure multi-party setting.

### 6.5.1 Privacy-preserving aggregation

To transform the data tables into the necessary format, we need a privacy-preserving aggregation operation. Aggregation is a standard database operation that groups items based on a chosen attribute the values of which are equal in all rows within the group. Then an aggregation function is applied to the columns in each group and, as a result, we receive a dataset with one aggregated row for each group.

Consider the example in Figure 6.3. Here, the table is aggregated based on the attributes ID and Year and the rows in the resulting dataset are computed using the following aggregation functions on values of each group: random(ID), random(Year), avg(Salary), random(Education).

More formally, let  $\mathbf{A} = a_{s,t}$  be a dataset with  $m$  attributes and  $N$  records,  $s \in \{1, \dots, N\}$  and  $t \in \{1, \dots, m\}$ . We denote rows of this dataset as  $\mathbf{a}_s$ . Let  $C \subseteq \{1, \dots, m\}$  be the set containing the indices of the columns by which the rows in the table will be grouped. Then  $\{\mathbf{X}_1, \dots, \mathbf{X}_n\}$  is a set of  $n$  matrices ( $1 \leq n \leq N$ ) with  $m$  columns where each matrix  $\mathbf{X}$  is composed of rows  $\mathbf{a}_g$  such that for each  $c \in C$ , the elements of attribute  $c$  are equal in all rows.

Let  $j \in \{1, \dots, v\}$ ,  $1 \leq v$  and let  $\mathbf{b}$  be a tuple of indices of attributes, such that  $b_j \in \{1, \dots, m\}$ . In addition, let  $\mathbf{op}$  be a tuple of identifiers of aggregation operations so that  $op_j \in \{\text{first}, \text{max}, \text{min}, \text{sum}, \text{avg}, \text{count}\}$ . Let  $q_i$  be the number of rows in the grouped matrix  $\mathbf{X}_i$ ,  $i \in \{1, \dots, n\}$

ID	Year	Month	Salary	Education
1	2006	1	1000	4
1	2006	2	2000	4
1	2007	1	2000	4
2	2007	1	1000	3
2	2007	2	1000	3

↓  
group by ID and year  
↓

ID	Year	Salary	Education
1	2006	1500	4
1	2007	2000	4
2	2007	1000	3

Figure 6.3: Example of aggregation: Group by (ID, Year), select (random(ID), random(Year), avg(Salary), random(Education))

The aggregated dataset  $\mathbf{D} = d_{i,b_j}$  has  $v$  columns and  $n$  rows, so that

$$\mathbf{D} = \{d_{i,b_j} | op_j(y_j), y_j = (x_{g,b_j}), x_{g,b_j} \in \mathbf{X}_i, g \in \{1, \dots, q_i\}\}.$$

Note that the input identifiers of attributes  $b_1, \dots, b_v$  can have recurring elements, as the analyst might need to perform several operations on elements of one attribute. The possible aggregation operations  $op$  are the following:

- random - taking the first available element, as all elements are assumed to be equal (equality will not be checked),
- max - taking the maximal element (this also serves as the disjunction operation for Booleans),
- min - taking the minimal element (this also serves as the conjunction operation for Booleans),
- sum - summing the elements,
- avg - computing the average of elements,
- count - counting the available elements.

Algorithm 3 describes the privacy-preserving aggregation procedure. This algorithm works similarly to the join operation discussed in [54].



---

**Algorithm 3:** Privacy-preserving aggregation procedure

---

**Data:** Dataset  $\llbracket \mathbf{A} \rrbracket$  with  $m$  attributes and  $N$  records, indices of attributes  $C \subseteq \{1, \dots, m\}$  by which to group the dataset, tuple of indices  $(b_1, \dots, b_v), b_i \in \{1, \dots, m\}, (1 \leq v)$  of attributes that will be included in the resulting dataset, tuple of identifiers  $(op_1, \dots, op_v)$  of aggregation operations

**Result:** Dataset  $\llbracket \mathbf{D} \rrbracket$  with  $v$  attributes and  $n$  records ( $1 \leq n \leq N$ )

```
1 Obviously shuffle rows of  $\llbracket \mathbf{A} \rrbracket$ 
2 Combine the values of attributes  $c \in C$  into composite keys  $\llbracket k_1 \rrbracket, \dots, \llbracket k_N \rrbracket$ 
3 Use oblivious AES to encrypt the composite keys, denote them as
   $\llbracket k'_1 \rrbracket, \dots, \llbracket k'_N \rrbracket$ 
4  $(k'_1, \dots, k'_N) \leftarrow \text{declassify}(\llbracket k'_1 \rrbracket, \dots, \llbracket k'_N \rrbracket)$ 
5 Let  $n$  be the number of unique groups in  $(k'_1, \dots, k'_N)$ 
6 for  $i \in \{1, \dots, n\}$  do
7   for  $j \in \{1, \dots, v\}$  do
8     Obviously apply operation  $op_j$  to elements of attribute  $b_j$  within
     group  $i$ 
9   end
10  Write the result into  $\llbracket \mathbf{D} \rrbracket$ 
11 end
12 return  $\llbracket \mathbf{D} \rrbracket$ 
```

---

This algorithm leaks the number of groups and the number of elements in each group. The former is a desired result. As for the latter, we use shuffle at the beginning of the algorithm so the number of elements in each group cannot be linked to the original dataset. There is a possibility of performing aggregation without leaking the number of elements in each group. This can be done by obviously adding dummy elements into the set to compensate for the groups that have fewer elements. For details, see the size unification protocol from article [54] where the same idea is discussed for the join operation.

### 6.5.2 Education dataset

The Ministry of Education and Research imports values for the following attributes: person ID, gender, year of birth, year of observation, level of study (Bachelor's, Master's, PhD, professional higher education), curriculum, length of nominal period of curriculum, school, date of admission, status of studies (in progress, quit, graduated), date of graduation/termination.

This dataset is too detailed for our needs. We want the data to be in the format

where each row corresponds to one person's studies in one curriculum. However, the records originally have the following structure.

1. One record for each person's each study position for each year from 2006 to 2012.
2. One record for the enrolment of a person in a study position prior to 2006.

We need to aggregate the records based on unique people. We also need to keep separate records for a person's studies in different curricula, so the unique identifier in this case will be the person ID combined with the curriculum ID. This is depicted as Aggregation 1 in Figure 6.2. We use the aggregation procedure from Algorithm 3 to group the different records with the same unique identifier together. The year of observation becomes obsolete during this aggregation and will be left out of the resulting dataset.

The date of admission will be renamed as date of first admission to account for students who have started in the same curriculum several times. The minimum of the values will be selected as the date of first admission. The date of graduation/termination will similarly be taken as the maximum of the available values to obtain the latest of these dates. The status of studies is the trickiest of the attributes. There are four options—in progress, quit, graduated, in progress at the end of 2012—and the logic is the following.

1. The resulting status can never be “in progress”.
2. The resulting status can only be “quit”, if the person never graduated from the curriculum in question not even after several tries.
3. The resulting status is “graduated”, if the person graduated from the curriculum in question. A person is not allowed to reapply for a curriculum they have graduated from.
4. The resulting status is “in progress 2012” if the person's studies were still ongoing by the end of 2012.

If we give these options codes 1 for in progress, 2 for quit, 3 for graduated, and 4 for in progress 2012, we can take the maximum of these values as the result.

The original dataset does not have the fourth option. In fact we only need this for one special case in which a person has quit their studies in a curriculum, then re-enrolled, and is still studying at the end of the period under analysis (i.e., at the end of 2012). In such a case, the status after aggregation must be “in progress” instead of “quit”, which the maximum operation would return without the added fourth option. We add this option by obviously changing the values that are “in

progress” in 2012 to “in progress 2012”. Now taking the maximal element in a group will always return the right code.

Note that, during data import, codes are given to classifier values automatically, based on their order of appearance. Hence, an imported dataset might not immediately have the classifier options we require. For this purpose, we allow the users to obliviously recode the classifier values by providing the codes they wish the options to have. Hence, regardless of the codes that the classifier denoting the status of studies originally has, we can reorder them so that, during aggregation, the maximal element can be chosen.

All other values are the same within a group based on a person’s ID and the curriculum ID, hence, a random element is taken from the values of each of these attributes. Having done this, we have obtained our desired table format.

### 6.5.3 Tax dataset

The Tax and Customs Board inputs data with the following attributes: person ID, year, month, payment for which social security has been charged, dividend income, income from self-employment, whether the employer was from the ICT field, whether the employer was a member of ITL.

Similarly to the education dataset, the tax dataset is too detailed for our needs. Specifically, there is a record for each source of income per month per person. This means that if a person gets a salary from two companies for a year, there are 24 records in the table for that person that year. For our study, we only need information about the average salary per year and the number of months that a person worked during a year. In fact, our end-goal is to receive a table where each row corresponds to one person’s salary information for all 10 years.

As the first step, we will add some new attributes to the dataset based on the existing data. Namely, we add attributes whether the person received income from self-employment and whether the person received dividend income. These attributes will generalise some of the more detailed attributes in the original dataset.

Second, we want to combine a person’s salaries during one month for the cases where a person is holding multiple jobs (Aggregation 2 in Figure 6.2). For this, we group the data by person ID, year and month, and calculate the sum of the payment attributes within a group. During this operation, we leave out the detailed attributes (dividend income, income from self-employment) and take the maximum of the corresponding generalised attributes.

Next, we average the monthly income into average income per year (Aggregation 3 in Figure 6.2). We do this per month of employment, meaning that if a person worked for one month, their average yearly income will be that month’s salary. For this, we group the dataset by person ID and year. We compute the average income and count the records in each group to get the number of months

that the person worked during that year. In addition, we take the maximal element for the Boolean attributes (is ICT, is ITL, was self-employed, received dividend income) again.

Now, we expand the table adding an attribute for each year for all the attributes: income, number of months, is an ICT company, is not an ICT company, is an ITL company, received income from self-employment, received dividend income. Our table will have  $7 \cdot 10 = 70$  new columns. Let us look at an example record for person X for year 2006. This record will have all the attributes for that person during that year. After expansion, the record will have 70 new columns. To fill these columns we do the following.

1. We obviously build a mask vector for each record based on its year  $a \in \{2004, \dots, 2013\}$ . In our example the mask vector is built by comparing 2006 to all possible years, so the resulting mask vector is  $(0, 0, 1, 0, \dots, 0)$ .
2. For each of the attributes we expanded, we multiply the corresponding attribute with the mask vector and save the result as the corresponding expanded attribute. In our example this means that the original average salary  $s$  for person X is saved as  $(0, 0, s, 0, \dots, 0)$ .

The result is a fairly sparse table containing one record per person per year. The final step is to group by person ID (Aggregation 4 in Figure 6.2) to receive an expanded table that has one record per person which includes the data for all the years in question. Hence, we use sum as the operation for all of the grouped attributes. This works for the Boolean attributes as well, as within a group, each column has exactly one value. Finally, we have achieved the desired table format.

#### 6.5.4 Obtaining the analysis table

Now we combine our two tables—education and salary information—into one data table using left join, so that people with no salary information will also be retained in the resulting database. We apply the privacy-preserving join procedure described in [54].

For performing the statistical analysis, we require the salary information to be relative to the person's studies, meaning that we would like the salaries to correspond to the study years  $i$ . Hence, we will convert the data so that, for each student, there will be an attribute for work and salary information for each year  $i$  of their studies beginning with admission. As an example consider a person who started her studies in 2007. For this student, the salary information in the attribute  $\text{salary}_1$  will be information from 2007. For another student, starting in 2010, the same attribute  $\text{salary}_1$  will contain salary information from 2010. The resulting table will be relatively sparse.

For this, we need the possibility to shift vector elements. The shifting function is relatively study-specific. Oblivious vector shifting means that elements in the vector are shifted left by a number of spaces  $k$ , where  $k$  is a private value. The  $k$ -th element and all those to the left of it are copied to the end of the vector and are marked as not available in the corresponding availability vector.

Using this function, we add attributes for all the years  $i$  since admission for the following information:

- Whether the person was working during year  $i$ ;
- Whether the person was working and studying during year  $i$ ; whether they were working for at least 3, 6, and 9 months during year  $i$ ;
- Exactly how many months the person was working during year  $i$ ;
- Salary during studies during year  $i$ ; salary if they were working for at least 3, 6, and 9 months during studies in year  $i$ ;
- Whether the person was working in an ICT company during the nominal period during year  $i$ ;
- Whether the person was working in a non-ICT company during the nominal period during year  $i$ ;
- Whether the person was working in an ITL company during the nominal period during year  $i$ .

We also create another shift to reflect working during years  $j$  after graduation with the following attributes:

- Whether the person was working during year  $j$  after graduation, whether they were working for at least 3, 6, and 9 months during year  $j$ ;
- Salary during year  $j$  after graduation; salary if they were working for at least 3, 6, and 9 months during year  $j$  after graduation.

Unfortunately, oblivious shifting requires us to align the dataset with the person whose studies have been the longest, i.e., if the earliest admission date is from 1994, we will have to make columns for study years 1 through 20 for everyone. This will make the data matrix extremely sparse because most of the studies will be within 2 to 6 years. In addition, we do not have salary information before 2004, so the extreme cases can only be used to analyse graduation in expected time for different fields.

We create this sparse data matrix because later it will be easier to formulate the necessary analysis queries. Recall, that we have the salary data as a vector

of 10 values for years 2004-2013. We need to obviously shift this salary vector according to each person's individual year of admission. As there are 10 elements in the salary vector, but we might have more than 10 years of study, we also add a padding of zeros before the shifting process. The shift will essentially select the first element from the salary info of the year of admission and add all the previous salary data to the end of the vector, changing the corresponding last elements as not available.

To obtain the necessary attributes, the shift based on years since admission will range from one to  $2013 - \min(\text{year of admission}) + 1$ . For the shift reflecting working after graduation we need a shorter period of time, as we know that the earliest graduation information we have is from 2006. Hence, we shorten the salary vector to include only years 2007 to 2013 and shift this, instead, based on the year of the end of studies plus one. The index for this shift will range from one to seven.

We generalise some of the attributes that are more detailed in the joined data table. Namely, based on the year of graduation, we make attributes for dividend income before and after graduation, and do the same for income from self-employment. We also find out whether the person was working in an ICT company during their studies. The resulting analysis table will be used for performing the statistical queries.

## 6.6 Statistical analysis

When we have compiled the analysis database, the statistical analyst at CentAR can perform the necessary queries using the RMIND tool.

In Estonia, there are four major schools that teach ICT subjects: University of Tartu, Tallinn University of Technology, Tallinn University and Estonian Information Technology College. In the following, the statistics done across all schools refer to these four institutions. First, descriptive statistics for the dataset are computed using the following queries on the education database:

- Number of students starting ICT studies across all schools in different levels of study during the years 2006-2012 (1 query);
- Number of students graduating from ICT studies across all schools in different levels of study during the years 2006-2012 (1 query);
- Number of students quitting their ICT studies across all schools in different levels of study during the years 2006-2012 (1 query);
- Percentage of students graduating their Bachelor's studies in nominal time based on year of admission during the years 2006-2009 in *ICT and non-*

*ICT fields*. The same for professional higher education studies and Master's studies; the same queries for graduation of *ICT in nominal time* across different universities (6 queries).

To study general employment during studies, the following queries are performed on the analysis database:

- Percentage of working students based on year of admission during the three years of Bachelor's studies in *ICT and non-ICT fields*; the same for professional higher education studies and Master's studies; the same queries based on working during studies in *ICT across the schools* (6 queries);
- Number of months worked during a calendar year during nominal study time during the three years of Bachelor's studies in *ICT*; the same for professional higher education studies and Master's studies (3 queries).

To study employment in *ICT companies* and *ITL member companies* during studies, the following queries are performed on the analysis database:

- Percentage of students working in *ICT companies* based on year of admission during the years of Bachelor's studies in *ICT and non-ICT fields*; the same query for professional education studies and Master's studies; the same queries for *ITL member companies* (6 queries);
- Percentage of students working in *ICT companies* based on year of admission during the three years of Bachelor's studies in *ICT across three universities*; the same for professional higher education studies and Master's studies; the same queries for *ITL member companies* (6 queries).

To study employment after graduation or quitting studies, the following queries are performed on the analysis database:

- Rate of employment after graduation or quitting studies in *ICT and non-ICT fields* one to three years after graduating/quitting Bachelor's studies; the same for professional higher education studies and Master's studies; the same queries based on *ICT studies across the schools* (6 queries).

To study income during studies, the following queries are performed on the analysis database:

- Median monthly income of *ICT and non-ICT students* during Bachelor's studies based on year of admission and the fact of graduation/quitting; the same for professional higher education studies and Master's studies; the same queries for *ICT students across schools* (6 queries);

- Median monthly income of ICT students across schools during the *nominal time* Bachelor's studies based on year of study and the fact of graduation/quitting; the same for professional higher education studies and Master's studies (3 queries).

To study income after graduation or quitting studies, the following queries are performed on the analysis database:

- Median monthly income of *ICT and non-ICT students* after graduating or quitting Bachelor's studies one to three years after graduating/quitting; the same for professional higher education studies; the same queries for *ICT students across schools* (4 queries).

In the queries about median salary, boxplots can be used to also show variation of the salaries in each group. In addition, we can perform the Wilcoxon rank sum test to find out whether the difference in medians in both groups is significant or not.

To validate the results of the study, a study will be conducted on the data in the standard setting. CentAR has signed confidentiality agreements with the Ministry of Education and Research and the Tax Board. However, as tax data are very sensitive, the confidentiality agreement only allows access to anonymised and pre-aggregated data from the Tax Board. CentAR will perform the same queries on the non-encrypted data. As a side-result, we can get an idea of how many students the pre-aggregation process will leave out of the study due to their uniqueness and how much this influences the results of the study. This information can be valuable in justifying the use of secure computations.

## 6.7 Preparing for the study

### 6.7.1 Achieving compliance with data protection

The study is carried out by our project team. In this section, “we” refers to members of the team. The author of this work is responsible for creating the plan of the study in the privacy-preserving setting and designing the ETL process. Additionally, she has contributed to the management of the project.

As is the case with the real world, things rarely go as planned on paper months and even years in advance. When the project got its approval and financing, and was started on February 15, 2013, the first thing was to go to the Ministry of Education and Research and the Tax Board, to acquaint them with the notion of secure multi-party computation and its possibilities. Our team also met with the project managers from the IT department of the Ministry of Finance and the Estonian Information System Authority to talk about the possibility of housing the computing



servers with the secret-shared data. The computing parties expressed readiness to host the servers if we give them instructions for deployment. In September 2013, our team went to the input parties and asked if they were willing to share their data using our importer tool. The Ministry of Education and Research was prepared to export the data, but the Tax Board requested that we ask for an approval from the Data Protection Inspectorate to export the data even in privacy-preserving format.

We prepared a request for the Data Protection Inspectorate about the project and submitted it in November 2013. Members of our team also visited the inspectorate and explained what secure multi-party computation is and how the statistical analysis is conducted on secret-shared data. We explained that these data can be considered to be encrypted and that single values will not be accessible by any party. The Data Protection Inspectorate reviewed our proposal and, in January 2014, we received their decision that according to subsection 16 (1) of the Personal Data Protection Act, personal data can be used in scientific research only in coded form and, in such a case, no further permission is needed from the Data Protection Agency. Basically, they accepted secret-shared data as encrypted data and issued a statement that their permission need not be asked for this kind of study if we conduct it with the described security controls [26].

The response from the Data Protection Agency assured the Tax Board that the terms of the Data Protection Act will not be compromised. However, as mentioned before, access to tax information is also regulated by the tax secrecy clauses in the Taxation Act. Therefore, in April 2014, the Tax Board expressed interest in receiving the statistical analysis tool to see, what kind of analysis results the statistician will have access to.

Following this, we redesigned the study process so that before the real data are imported, the Compliance Department of the Tax Board can perform a test run using RMIND on generated data to evaluate the privacy-preservation capabilities of the platform and the statistical tool.

### **6.7.2 Improving the tool to handle real-world data**

In June 2014, we finished the statistics tool RMIND, adding the functionality necessary for the project. CentAR then went on to request different comparative plots and the possibility to paste together plots from analyses on different filters. We added these features in July. In August, we finished improving the logging system and prepared the code for review by the IT department of the Ministry of Finance. We also made the platform more stable for the case where one of the computing parties loses connection.

In August, we also got the sample data created by CentAR. We had received sample files from them before, but those had already been cleaned for the study. This new information left us with three new interesting obstacles.

First, it turned out that the datasets included dates. A usual database engine can accept dates without a problem. We can also use similar methods (e.g., Unix timestamps) to save dates, however, we had never needed these values in practice, so, naturally, this functionality had not been implemented. We decided on the format and implemented date importing as well.

Second, the classifier values (e.g., school, degree) were a combination of codes and names instead of simply a code. For secure multi-party computation, this is a problem. While we are technically capable of storing secret-share strings, it is significantly more time-consuming to perform comparison operations on them. Hence, we decided to change the importer so that it filters out all of the available values and prepares a classifier from them. The codes of this classifier will be stored in a secret-shared format, whereas the corresponding classifier values will be public. Making the classifier values public is not a problem, as they are freely available from the web page of Statistics Estonia. With this approach, we leak the information about which of the classifier values are present (and, therefore, missing) in the dataset.

This gives us practical information that is usually desired in a statistical study and, if it is not desired information, we have to give the importer tool the whole classifier set and ask it to encode the classifiers based on that. A worse option is that if only one value is present, we will know this value for all imported records. However, this is a problem with smaller data sizes, as with the whole population, such an attribute carries no real information if it is the same for every person in the population. This can also be countered by using the original classifier for data import.

This approach to encoding classifiers also leaks the order in which new elements appear. This, in turn, reveals, what the value is for the first record. This can be countered by the importer tool that shuffles either the input data or the classifiers. It is more feasible to shuffle the classifiers, as there is bound to be less of them.

Third, the recorded classifiers turn out, in fact, to be too detailed for our study. We wish to take all the available education levels and convert them into groups of applied science degree, bachelor's degree, master's degree, and PhD degree. There are three implicit ways of doing this—we can ask the input parties to group the classifier values for us, we can convert them on the fly in the importer tool, or we can import the detailed information and then perform the aggregation in the secret-shared database.

As we do not want the input parties to do extra work, we set aside the first option. Converting the values in the importer tool on the fly would require us to be familiar with all the possible input data values as we would already need to ship the conversion rules along with the importer. As with real-life databases, it

is often the case that there are mistakes and typos in them, and data have often been collected using different versions of classifiers, we would have to fix the importer, redeploy it and ask the input party to run the process again. This will again inconvenience the input parties as they might have to perform the same task several times.

Hence, we will choose the third option, importing the data as-is, and then converting the classifiers already in the secret-shared database. There are several advantages to this approach. First, if there are indeed typos or mistakes in the classifiers, we will have the entire list of available values and we can distribute these into the necessary groups using human intelligence where necessary. Second, we have the original classifier values and if the conversion requirements change, we can rerun the conversion query and have the values we need. All of this can be done without requiring direct action from the input parties, making it easier to carry out the changes as their deployment will not require a lot of resources. This kind of conversion can easily be carried out on secret-shared data. As we are dealing with millions of records, the process will take time. However, as this process needs to be carried out only once and the results will be saved, this can be done separately from the statistical analysis.

The ETL process was quite difficult to come up with in the privacy-preserving setting. We were given the attributes that the analysts expected to be working on. They also gave us their own ETL scripts, but as the data format is so different in the two settings, it would be highly infeasible to run the same scripts on secret shared data. With some thinking, though, we were able to come up with quite an elegant solution also for the privacy-preserving ETL process. However, at the moment, this solution is quite study-specific and it is difficult to generalise and apply it to other studies. Creating a set of ETL tools for future use is an interesting topic we want to look into. The aggregation procedure that was designed and implemented, as well as the vector element shifting are two components that will be available in RMIND should they be needed in future ETL processes.

### **6.7.3 Code review and acceptance testing by the Tax Board**

We delivered the source code of SHAREMIND and RMIND to the IT department of the Ministry of Finance in November 2014. They reviewed the source code and concluded that both the platform and the statistical tool are professional and skilfully built. However, they also admitted that they lack the detailed knowledge to verify all the security claims of the system.

In December 2014, we held a testing session with the specialists working in the Compliance Department of the Tax Board. We explained the cryptographic privacy guarantees given by SHAREMIND and described how the study plan is technologically enforced. Finally, we demonstrated the privacy-preservation tools

<b>Dataset</b>	<b>Education records</b>	<b>Tax records</b>
A	350 records	8200 records
B	831 000 records	16 million records

Figure 6.4: Data sizes in the test datasets

in practice by running various queries on generated data to show how RMIND ensures output privacy.

The demonstration was successful and in January 2015, the Tax Board agreed to provide data for the privacy-preserving study. All input parties, computing parties and result parties signed legal agreements to fix their obligations to each other within the security model of the SHAREMIND system. The parties agreed to host the secure multi-party computation system in good faith and without trying to tamper with its processing, hence, complying with the semi-honest security model of the underlying protocols.

#### 6.7.4 Performance measurements

We generated two sensible and coherent test datasets to test the accuracy and performance of our solution. Test set A is smaller and used for correctness testing, test set B is used for evaluating performance and its size is similar to that of real input data. Table 6.4 gives the data sizes in these two datasets. We asked our colleagues in CentAR to run queries on the generated data using STATA to see how long the ETL process and analysis take in the usual setting. We also asked them to provide us with results to the queries done on the smaller database so we could compare them to the results we got using the privacy-preserving tool.

We concluded our preparations for the study by measuring the performance of the privacy-preserving extract, transform, load process. We measured the performance of the RMIND scripts that perform data aggregations and transformations needed to construct the final analysis database. We ran the benchmark in a laboratory setting with three separate machines connected with 1 Gbit network links. Each server has two 2,93 GHz CPUs with 6 cores each and 8 GB of RAM per core to a total of 48 GB.

Table 6.5 contains the descriptions and running times of the five individual RMIND scripts. The work is divided between scripts based on the ETL process diagram given in Figure 6.2. The RMIND tool performs all operations in each script in succession. The performance was measured on the RMIND client side. In the table, Time A and Time B show the running times for the generated datasets A and B, respectively.

The total running time of about two and a half days for the whole extract,

<b>Script</b>	<b>Description</b>	<b>Time A</b>	<b>Time B</b>
1	Aggregation of education data	1 min 3 s	25 min
2	Aggregation of tax data (monthly income)	46 s	18 h 10 min
3	Aggregation of tax data (average yearly income)	4 min 34 s	1 h 55 min
4	Aggregation of tax data and joining the two datasets	32 s	32 min
5	Compiling the analysis table (shifting)	6 min 7 s	39 h 3 min
<b>Sum</b>		13 min 2 s	60 h 5 min

Figure 6.5: Running times of the privacy-preserving ETL scripts in test datasets A and B

transform, load step is not excessively high in this study, but may sound discouraging for wider applications. However, we stress that this is the first time a statistical study of this complexity has ever been performed on secure multi-party computation.

We expect that further research into the protocols and algorithms, as well as engineering efforts towards better virtual machines and compilers, will certainly reduce the running time of the study. Thus, we remain optimistic of the future of the technology and continue our efforts in increasing its efficiency.

## 6.8 Impact of our work

The work done within this dissertation has prepared the algorithms and tools needed for the described statistical study. The study itself is scheduled to be conducted in the first half of 2015.

This is the first study of this magnitude that will be carried out in a privacy-preserving setting based on secure multi-party computation. The data will be imported from existing real-world databases. It is also the first practical study where data from two different sources will be merged for joint data analysis, will be extracted, transformed and loaded into a unified analysis table in a privacy-preserving setting, and will be analysed in a privacy-preserving manner.

# CONCLUSION

Data are gathered in many fields of life. The purpose of this process is either data archival or, more often, data analysis to find interesting information or patterns to support decision making. However, the most interesting kind of data contains information about people or organisations. Analysing this sensitive information has restrictions, most of which are governed by national or international legislation.

These laws do not forbid data analysis if certain protective measures are taken. These include organisational measures, e.g., confidentiality agreements, or technical measures, e.g., pseudonymisation or pre-aggregation of data. These are meant to either prevent data analysts from identifying single subjects in a cohort in the case of technical measures, or simply having them promise not search for the information or disclose any confidential results in the case of organisational measures.

This dissertation investigates the possibility of using secure multi-party computation technology to perform statistical analyses in a privacy-preserving setting. We look at this problem from two perspectives. Firstly, we show that it is possible to implement statistical analysis algorithms in the privacy-preserving multi-party setting, and secondly, that the implemented solution is feasible enough for practical use.

The main result of this dissertation is the privacy-preserving statistical analysis tool RMIND that contains the most commonly used statistical operations, such as descriptive statistics, simple statistical measures, statistical tests for comparing two populations, and linear regression. To support the creation of this tool, we designed and implemented privacy-preserving floating-point arithmetic as part of this dissertation for the secure multi-party computation platform SHAREMIND and the arithmetic has been incorporated as a standard component of the platform.

RMIND provides an R-like environment that is based on the SHAREMIND platform and that can be used by statistical analysts who need not know the details of the cryptography that lies beneath the surface. This is a similar concept to how analysts do not have to know the details of the more complicated statistical tests they are running. The RMIND tool can be used with little instruction as we have aimed at making it very similar to the existing tools and all the secure multi-party

computation details are hidden from the user.

We will validate the usability of the RMIND tool by carrying out a nation-wide statistical study in a privacy-preserving manner about the influence of working during university studies. The data will be acquired from the Ministry of Education and the Tax Board of Estonia and the analyses will be performed using the privacy-preserving algorithms described in this dissertation. We show how the extract, transform, load process for such a study can be done in the privacy-preserving setting. This process turns out to be the most complicated part of privacy-preserving statistical analysis, as we cannot perform all queries in the same manner as in the standard setting. We offer a solution for the planned statistical study, but we leave a more generalised approach as future work.

All of the algorithms presented in this dissertation return results that are secret-shared. However, for a statistical analysis, results need to be declassified. Analysing how much data is leaked during this process and how to mitigate this leakage using query auditing and differential privacy results remains an interesting topic for future research.

# Bibliography

- [1] Akella, M.R., Alfriend, K.T.: Probability of Collision Between Space Objects. *Journal of Guidance, Control and Dynamics* 23(5), 769–772 (2000)
- [2] Alfano, S.: A numerical implementation of spherical object collision probability. *Journal of the Astronautical Sciences* 53(1), 103 (2005)
- [3] Arfken, G.B., Weber, H.J.: *Mathematical Methods for Physicists*. Elsevier Academic Press, 6th edn. (2005)
- [4] Armitage, P.: Tests for Linear Trends in Proportions and Frequencies. *Biometrics* 11(2), 375–386 (Sep 1955)
- [5] Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: *Proceedings of the 20th annual ACM symposium on Theory of computing, STOC’88*. pp. 1–10 (1988)
- [6] Bogdanov, D.: Sharemind: programmable secure computations with practical applications. Ph.D. thesis, University of Tartu (2013), <http://hdl.handle.net/10062/29041>
- [7] Bogdanov, D., Kamm, L.: Constructing Privacy-Preserving Information Systems Using Secure Multiparty Computation. Tech. Rep. T-4-13, Cybernetica, <http://research.cyber.ee/>. (2011)
- [8] Bogdanov, D., Kamm, L., Laur, S., Pruulmann-Vengerfeldt, P., Talviste, R., Willemson, J.: Privacy-preserving statistical data analysis on federated databases. In: *Proceedings of the Annual Privacy Forum, APF’14*. LNCS, vol. 8450, pp. 30–55. Springer (2014)
- [9] Bogdanov, D., Kamm, L., Laur, S., Sokk, V.: Rmind: a tool for cryptographically secure statistical analysis. *Cryptology ePrint Archive*, Report 2014/512 (2014)



- [10] Bogdanov, D., Laud, P., Laur, S., Pullonen, P.: From Input Private to Universally Composable Secure Multi-Party Computation. In: Proceedings of the 27th IEEE Computer Security Foundations Symposium, CSF'14. IEEE Computer Society (2014)
- [11] Bogdanov, D., Laud, P., Randmets, J.: Domain-Polymorphic Programming of Privacy-Preserving Applications. In: Proceedings of the First ACM Workshop on Language Support for Privacy-enhancing Technologies, PETShop '13. pp. 23–26. ACM Digital Library, ACM (2013)
- [12] Bogdanov, D., Laur, S., Willemson, J.: Sharemind: A framework for fast privacy-preserving computations. In: Proceedings of the 13th European Symposium on Research in Computer Security, ESORICS'08. LNCS, vol. 5283, pp. 192–206. Springer (2008)
- [13] Bogdanov, D., Niitsoo, M., Toft, T., Willemson, J.: High-performance secure multi-party computation for data mining applications. *International Journal of Information Security* 11(6), 403–418 (2012)
- [14] Bogdanov, D., Talviste, R., Willemson, J.: Deploying secure multi-party computation for financial data analysis (short paper). In: Proceedings of the 16th International Conference on Financial Cryptography and Data Security, FC'12. pp. 57–64 (2012)
- [15] Bogetoft, P., Christensen, D.L., Damgård, I., Geisler, M., Jakobsen, T.P., Krøigaard, M., Nielsen, J.D., Nielsen, J.B., Nielsen, K., Pagter, J., Schwartzbach, M.I., Toft, T.: Secure Multiparty Computation Goes Live. In: 13th International Conference of Financial Cryptography and Data Security, FC'09. pp. 325–343 (2009)
- [16] Burkhart, M., Strasser, M., Many, D., Dimitropoulos, X.A.: SEPIA: Privacy-Preserving Aggregation of Multi-Domain Network Events and Statistics. In: Proceedings of the 19th USENIX Security Symposium 2010. pp. 223–240 (2010)
- [17] Canetti, R.: Security and composition of multiparty cryptographic protocols. *Journal of Cryptology* 13(1), 143–202 (2000)
- [18] Canetti, R.: Universally Composable Security: A New Paradigm for Cryptographic Protocols. In: Proceedings of the 42nd Annual Symposium on Foundations of Computer Science, FOCS'01. pp. 136–145 (2001)
- [19] Canetti, R., Ishai, Y., Kumar, R., Reiter, M.K., Rubinfeld, R., Wright, R.N.: Selective private function evaluation with applications to private statistics.

In: Proceedings of the 20th ACM Symposium on Principles of Distributed Computing, PODC'01. pp. 293–304. ACM (2001)

- [20] Catrina, O., De Hoogh, S.: Improved primitives for secure multiparty integer computation. In: Proceedings of the 7th International Conference on Security and Cryptography for Networks, SCN'10. pp. 182–199. Springer-Verlag (2010)
- [21] Chaum, D., Crépeau, C., Damgård, I.: Multiparty unconditionally secure protocols (extended abstract). In: Simon, J. (ed.) Proceedings of the 20th Annual ACM Symposium on Theory of Computing. STOC'88. pp. 11–19 (1988)
- [22] Chida, K., Morohashi, G., Fuji, H., Magata, F., Fujimura, A., Hamada, K., Ikarashi, D., Yamamoto, R.: Implementation and evaluation of an efficient secure computation system using 'R' for healthcare statistics. Journal of the American Medical Informatics Association 04 (2014)
- [23] Couzin, J.: Genetic privacy. Whole-genome data not anonymous, challenging assumptions. Science 321(5894), 1278 (Sep 2008)
- [24] Cramer, R., Damgård, I., Nielsen, J.B.: Secure Multiparty Computation and Secret Sharing - An Information Theoretic Approach (2012), <http://www.daimi.au.dk/~ivan/MPCbook.pdf>, book draft
- [25] Damgård, I., Pastro, V., Smart, N.P., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: Proceedings of the 32nd International Cryptology Conference, CRYPTO'12. LNCS, vol. 7417, pp. 643–662. Springer (2012)
- [26] Data Protection Inspectorate: The official response is available in Estonian. <http://adr.rik.ee/aki/dokument/3679385>. In Estonian. Last accessed 26.01.2015 (2014)
- [27] Du, W., Atallah, M.J.: Privacy-preserving cooperative statistical analysis. In: Proceedings of the 17th Annual Computer Security Applications Conference, ACSAC'01. pp. 102–110 (2001)
- [28] Du, W., Chen, S., Han, Y.S.: Privacy-preserving multivariate statistical analysis: Linear regression and classification. In: Proceedings of the Fourth SIAM International Conference on Data Mining, SDM'04. pp. 222–233 (2004)

- [29] Dwork, C.: Differential privacy. In: Proceedings of the 33rd International Colloquium on Automata, Languages and Programming, ICALP'06. LNCS, vol. 4052, pp. 1–12. Springer (2006)
- [30] El Emam, K., Samet, S., Hu, J., Peyton, L., Earle, C., Jayaraman, G.C., Wong, T., Kantarcioglu, M., Dankar, F., Essex, A.: A Protocol for the Secure Linking of Registries for HPV Surveillance. PLoS ONE 7(7), e39915 (July 2012)
- [31] Estonian Genome Center at the University of Tartu (EGCUT): Data access policy of EGCUT. <http://www.geenivaramu.ee/en/access-biobank/data-access>. Last accessed 26.01.2015
- [32] Estonian Public Broadcasting News Portal: Specialist: Misconceptions keep students away from studying IT. <http://uudised.err.ee/v/eesti/ea1e1f2a-f9d1-4b57-8608-ad9ff8c8eaf4>. In Estonian. Last accessed 26.01.2015 (2012)
- [33] Eurostat: Eurostat Yearbook. Eurostat, [http://epp.eurostat.ec.europa.eu/statistics\\_explained/index.php/Europe\\_in\\_figures\\_-\\_Eurostat\\_yearbook](http://epp.eurostat.ec.europa.eu/statistics_explained/index.php/Europe_in_figures_-_Eurostat_yearbook) (2014)
- [34] Feigenbaum, J., Pinkas, B., Ryger, R., Saint-Jean, F.: Secure computation of surveys. In: EU Workshop on Secure Multiparty Protocols (2004)
- [35] Franz, M., Katzenbeisser, S.: Processing encrypted floating point signals. In: Proceedings of the 13th ACM Multimedia Workshop on Multimedia and Security, MM&Sec '11. pp. 103–108. ACM (2011)
- [36] Geisler, M.: Cryptographic Protocols: Theory and Implementation. Ph.D. thesis, Aarhus University (February 2010)
- [37] Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the 41st ACM Symposium on Theory of Computing, STOC'09. pp. 169–178. ACM (2009)
- [38] Goethals, B., Laur, S., Lipmaa, H., Mielikäinen, T.: On private scalar product computation for privacy-preserving data mining. In: Proceedings of the 7th International Conference on Information Security and Cryptology, ICISC'04. LNCS, vol. 3506, pp. 104–120. Springer (2005)
- [39] Goldreich, O.: Foundations of Cryptography: Volume 2, Basic Applications, vol. 2. Cambridge University Press (2004)

- [40] Hamada, K., Kikuchi, R., Ikarashi, D., Chida, K., Takahashi, K.: Practically Efficient Multi-party Sorting Protocols from Comparison Sort Algorithms. In: Proceedings of the 15th International Conference on Information Security and Cryptology, ICISC'12, LNCS, vol. 7839, pp. 202–216. Springer (2013)
- [41] Hemenway, B., IV, W.W., Baiocchi, D.: Achieving Higher-Fidelity Conjunction Analyses Using Cryptography to Improve Information Sharing. Tech. rep., RAND Corporation (2014)
- [42] Henecka, W., Kögl, S., Sadeghi, A.R., Schneider, T., Wehrenberg, I.: TASTY: tool for automating secure two-party computations. In: Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS'10. pp. 451–462 (2010)
- [43] Homer, N., Szelling, S., Redman, M., Duggan, D., Tembe, W., Muehling, J., Pearson, J.V., Stephan, D.A., Nelson, S.F., Craig, D.W.: Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays. *PLoS Genet* 4(8), e1000167 (2008)
- [44] IEEE754: 754-2008 IEEE Standard for Floating-Point Arithmetic (2008)
- [45] International HapMap Consortium: The International HapMap Project. *Nature* 426(6968), 789–796 (Dec 2003)
- [46] Jäger, M., Kamm, L., Krushevskaja, D., Talvik, H.A., Veldemann, J., Vilgota, A., Vilo, J.: Flexible Database Platform for Biomedical Research with Multiple User Interfaces and a Universal Query Engine. In: DB&IS. Frontiers in Artificial Intelligence and Applications, vol. 187, pp. 301–310. IOS Press (2008)
- [47] Jawurek, M., Kerschbaum, F.: Fault-tolerant privacy-preserving statistics. In: Privacy Enhancing Technologies, LNCS, vol. 7384, pp. 221–238. Springer (2012)
- [48] Kamm, L., Bogdanov, D., Laur, S., Vilo, J.: A new way to protect privacy in large-scale genome-wide association studies. *Bioinformatics* 29(7), 886–893 (2013)
- [49] Kamm, L., Willemson, J.: Secure Floating-Point Arithmetic and Private Satellite Collision Analysis. *International Journal of Information Security* pp. 1–18 (2014)

- [50] Kiltz, E., Leander, G., Malone-Lee, J.: Secure computation of the mean and related statistics. In: Proceedings of the Second Theory of Cryptography Conference, TCC'05, LNCS, vol. 3378, pp. 283–302. Springer (2005)
- [51] Knuth, D.E.: The Art of Computer Programming, vol. 2. Addison-Wesley, 3rd edn. (1998)
- [52] Kreuter, B., Shelat, A., Mood, B., Butler, K.R.B.: PCF: A Portable Circuit Format for Scalable Two-Party Secure Computation. In: Proceedings of the 22nd USENIX Security Symposium 2013. pp. 321–336 (2013)
- [53] Krips, T., Willemson, J.: Hybrid Model of Fixed and Floating Point Numbers in Secure Multiparty Computations. In: Proceedings of the 17th International Conference on Information Security, ISC'14. LNCS, vol. 8783, pp. 179–197. Springer (2014)
- [54] Laur, S., Talviste, R., Willemson, J.: From Oblivious AES to Efficient and Secure Database Join in the Multiparty Setting. In: Proceedings of the 11th International Conference on Applied Cryptography and Network Security, ACNS'13. LNCS, vol. 7954, pp. 84–101. Springer (2013)
- [55] Li, N., Li, T., Venkatasubramanian, S.: Closeness: A New Privacy Measure for Data Publishing. IEEE Transactions on Knowledge and Data Engineering 22(7), 943–956 (July 2010)
- [56] Li, Q., Cao, G.: Efficient Privacy-Preserving Stream Aggregation in Mobile Sensing with Low Aggregation Error. In: Privacy Enhancing Technologies, LNCS, vol. 7981, pp. 60–81. Springer Berlin Heidelberg (2013)
- [57] Machanavajjhala, A., Kifer, D., Gehrke, J., Venkitasubramaniam, M.: *L*-diversity: Privacy beyond *k*-anonymity. IEEE Transactions on Knowledge Discovery from Data 1(1) (2007)
- [58] Malkhi, D., Nisan, N., Pinkas, B., Sella, Y.: Fairplay — a secure two-party computation system. In: Proceedings of the 13th USENIX Security Symposium 2004. pp. 287–302 (2004)
- [59] Põder, K., Sahk, K. (eds.): Eesti statistika aastaraamat. 2014. Statistical Yearbook of Estonia. Statistics Estonia (2014)
- [60] Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Proceedings of the 17th International Conference on the Theory and Application of Cryptographic Techniques, EUROCRYPT'99. pp. 223–238 (1999)

- [61] Pfitzmann, B., Waidner, M.: A Model for Asynchronous Reactive Systems and Its Application to Secure Message Transmission. In: Proceedings of the 2001 IEEE Symposium on Security and Privacy, SP '01. pp. 184–200. IEEE Computer Society (2001)
- [62] Pullonen, P., Siim, S.: Combining Secret Sharing and Garbled Circuits for Efficient Private IEEE 754 Floating-Point Computations. In: Proceedings of the 3rd Workshop on Encrypted Computing and Applied Homomorphic Cryptography, WAHC 2015 (2015)
- [63] Riigikogu: Human Genes Research Act of Estonia. <https://www.riigiteataja.ee/en/eli/518062014005/consolide> (Passed 2000)
- [64] Riigikogu: Taxation Act of Estonia. <https://www.riigiteataja.ee/en/eli/523012015008/consolide> (Passed 2002)
- [65] Riigikogu: Personal Data Protection Act of Estonia. <https://www.riigiteataja.ee/en/eli/ee/Riigikogu/act/509072014018/consolide> (Passed 2007)
- [66] Sasieni, P.D.: From genotypes to genes: doubling the sample size. *Biometrics* 53(4), 1253–1261 (1997)
- [67] Shi, E., Chan, T.H.H., Rieffel, E.G., Chow, R., Song, D.: Privacy-preserving aggregation of time-series data. In: Proceedings of the 18th Annual Network and Distributed System Security Symposium, NDSS'11 (2011)
- [68] Spielman, R.S., McGinnis, R.E., Ewens, W.J.: Transmission test for linkage disequilibrium: the insulin gene region and insulin-dependent diabetes mellitus (IDDM). *Am J Hum Genet* 52(3), 506–516 (1993)
- [69] Subramaniam, H., Wright, R.N., Yang, Z.: Experimental Analysis of Privacy-Preserving Statistics Computation. In: Proceedings of the Fourth SIAM International Conference on Data Mining, SDM'04, LNCS, vol. 3178, pp. 55–66. Springer (2004)
- [70] Sweeney, L.: k-anonymity: a model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10(5), 557–570 (2002)
- [71] Visscher, P.M., Hellard, S.L.: Simple method to analyze SNP-based association studies using DNA pools. *Genet Epidemiol* 24(4), 291–296 (2003)

- [72] Yang, Z., Wright, R.N., Subramaniam, H.: Experimental analysis of a privacy-preserving scalar product protocol. *Computer Systems Science & Engineering* 21(1) (2006)
- [73] Yao, A.C.C.: Protocols for Secure Computations (Extended Abstract). In: *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science, FOCS'82*. pp. 160–164. IEEE (1982)

# ACKNOWLEDGMENTS

One word after another.

That's the only way that novels get written and, short of elves coming in the night and turning your jumbled notes into Chapter Nine, it's the only way to do it.

So keep on keeping on. Write another word and then another.

(Neil Gaiman, *Pep talk for National Novel Writing Month*)

As my doctoral studies have been going on for some time, I have received support from numerous funds and projects. I have been supported by the European Regional Development Fund through the Estonian Centre of Excellence in Computer Science (EXCS), the Software Technology and Applications Competence Centre (STACC), and the Implementing Agency Archimedes Foundation for the Privacy-preserving statistical studies on linked databases (PRIST) project. I have also been supported by the European Social Fund through the Estonian Doctoral School in Information and Communication Technology (IKTDK), and by the Estonian Research Council through Research Grant IUT27-1. I have received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 284731 (UaESMC). I also acknowledge the support from the Estonian Information Technology Foundation through the Tiger University programme.

I have learned and grown a lot during the time I have spent in the PhD programme and, indeed, at the University of Tartu in general. I have had the luck of meeting different people who have shaped me into the person I am today. Even though some things have made me a bit cynical and too outspoken, I value these experiences as they have made me a stronger person. There is little left of my insecurity about my research choices, I have come to realise that what I do is important and useful. I could not have done this without the discussions and disputes that I have had with the researchers and professors at the University of Tartu.

For the past three and a half years, I have been working as a researcher at Cybernetica. I have been very lucky to find a company that not only supported my PhD studies but, in fact, encouraged them. My colleagues at Cybernetica



have helped and supported me immensely in my research endeavours, I would not have been able to finish this without their theoretical, practical and engineering knowledge. They are also simply cool people to be around, go orienteering with and talk about books, games or research politics.

This work would not be half of what it is without my supervisor Sven Laur. History has a tendency to repeat itself even over short periods and, luckily, Sven has been there to pick me up more than once when I have been out of ideas. Many of us know how busy he is and this is probably because he is an amazing researcher, a wonderful supervisor and a good friend.

Speaking of friends, one great thing about staying at the University for so long is the number of amazing people one meets during her time there. I have been lucky to find many great people before and during university with whom to discuss different topics, play board games, or just hang out. Going to university can be very stressful and I am really thankful to all of you for being there for me and helping me unwind and rant.

Finally, and maybe most importantly, I am eternally grateful to my family, who have stood by me in all my choices, both the successful and suboptimal ones, and have encouraged me to persevere even when I have been doubtful. I owe my sanity to my husband Dan who has been my friend since we started university together and has been beside me, always supporting and inspiring me in everything I do.

# **KOKKUVÕTE (SUMMARY IN ESTONIAN)**

## **TURVALISEL ÜHISARVUTUSEL PÕHINEV PRIVAATSUST SÄILITAV STATISTILINE ANALÜÜS**

Tundlike isikuandmete analüüs on probleem, mida ettevõtted ja valitsused päevast-päeva lahendada püüavad. Teadlased on uurinud ja kasutusele võtnud erinevaid statistilisi meetmeid, et vähendada isikuandmete leket. Veelgi levinum on konfidentsiaalsuslepete kasutamine. Selle asemel, et kasutada tõestatult turvalisi meetodeid, edastatakse andmed analüütikutele, kes viivad läbi statistilisi uuringuid tundlike andmete peal.

Turvaline ühisarvutus on vahend, mille abil on võimalik andmeid jagada ja töödelda privaatsust säilitavalt nii, et isegi andmeanalüütik ei näe üksikuid väärtusi. Need krüptograafilised protokollid on olnud teadlaste huviorbiidil alates sellest ajast, kui Yao tutvustas 1980ndatel oma ideid loogikaskeemide sogastamise kohta. Krüptograafid on loonud ka praktilisi rakendusi, mis kasutavad erinevaid turvalise ühisarvutuse protokolle. Need rakendused kasutavad Boole'i algebrat ja täisarvuaritmeetikat selleks, et pakkuda lahendusi erinevatele probleemidele, kaasaarvatud statistilisele analüüsile. Kahjuks kasutavad need eraldiseisvad funktsioonid väga erinevaid alusprotokolle ja on seetõttu piiratud käsitusala ja neid on raske koos kasutada.

Käesolev dissertatsioon kirjeldab protokollistikku, mille abil on võimalik viia läbi statistilisi uuringuid privaatsust säilitavas keskkonnas turvalise ühisarvutuse platvormi SHAREMIND [6] abil. Töö autor kavandas kõik töös kirjeldatud algoritmid ning lõi nende jaoks vajalikud eeltingimused kavandades ja realiseerides ujukomaaritmeetika. Samuti panustas autor töö käigus kavandatud statistikaalgoritmide realiseerimisse. Loodud algoritmid ei sõltu alusplatvormist ning neid on

seetõttu võimalik kasutada ka teistes turvalise arvutuse süsteemides, mis toetavad privaatsust säilitavat täisarvu- ja ujukomaaritmeetikat, sorteerimist, andmete juhuslikku ümberjärjestamist ning tabelite ühendamist.

Töö aluseks on järgmised kolm väidet. Esiteks väidame, et turvalist ühisarvutust kasutades on võimalik kavandada ja realiseerida enimkasutatud statistilise analüüsi algoritmid ilma eelnevalt teadmata, millistes uuringutes neid kasutama hakatakse. Teiseks väidame, et need algoritmid on võimalik kokku panna statistiliseks tööriistaks, mis sarnaneb olemasolevatele statistilistele pakettidele olles seega arusaadav analüütiku jaoks ning eeldades, et analüütik ei pea tingimata üksikasjalikult tundma allolevaid krüptograafilisi protokolle. Kolmandaks väidame, et need algoritmid on piisavalt kiired, et neid on võimalik kasutada isegi suuremahulistes uuringutes.

Töö koosneb kuuest peatükist. Esimene peatükk on sissejuhatav, kirjeldab probleemistikku, mille jaoks me lahendusi otsime ning räägib töö üldisest struktuurist. Teine peatükk võtab kokku teoreetilise tausta, millele me oma töös tugime.

Kolmandas peatükis vaatleme üht statistilise uuringu näidisrakendust, mis tegeleb privaatsust säilitavate ülegenoomsete assotsiatsiooniuuringutega. Peatüki juurde kuulavas artiklis [48] kirjeldame, kuidas kavandasime ja implementeersime neli neis uuringutes tihemini vajaminevat statistilist testi kasutades süsteemis SHAREMIND olemasolevat täisarvuaritmeetikat. Selle töö käigus jõuame järeldusele, et keerukamate statistiliste funktsioonide tarbeks on meil vaja privaatsust säilitavat ujukomaaritmeetikat. Samas näitame, et privaatsust säilitav statistiline analüüs on mõistlikus ajas teostatav ka andmebaaside peal, kus on miljoneid kirjeid.

Neljandas peatükis kirjeldame, kuidas me jätkame oma tööd kavandades ja implementeerides ujukomaarvud, nende liitmise, korrutamise ning võrdlemise raamistikul SHAREMIND [49]. Lisaks kavandame ja implementeerime privaatsust säilitavad elementaarfunktsioonid, mis arvutavad pöördarvu, ruutjuurt, siinust, naturaallogaritmi,  $e$  astmeid ning veafunktsiooni erf. Loodud aritmeetika ja funktsioonide testimiseks implementeerime algoritmi, mis arvutab satelliitide kokkupõrketõenäosust.

Loodud aritmeetikat kasutades saame luua ja implementeerida enimkasutatud statistilise analüüsi meetodid privaatsust säilitava keskkonna jaoks [8, 9]. Viiendas peatükis kirjeldamegi, kuidas ühendame loodud funktsionaalsuse tööriistaks RMIND, mis meenutab statistilise analüüsi keskkonda R. See tööriist on mõeldud statistikutele, et aidata neil läbi viia privaatsust säilitavaid uuringuid ilma, et nad peaksid detailiselt aru saama allolevast krüptograafiast. Loodud funktsionaalsuse hulgas on kirjeldava statistika meetodid (näiteks viie väärtusega kokkuvõte ja histogrammid), lihtsad statistilised mõõdikud (näiteks keskmine ja stan-

dardhälve), statistilised testid (näiteks t-test ja  $\chi^2$  test) ning lineaarregressioon.

Dissertatsiooni lõpetuseks kirjeldame kuuendas peatükis, kuidas me valmis-tame ette tööriista RMIND, et viia läbi suuremahulist privaatsust-säilitavat statisti-list uuringut tegelike andmete peal. Me kirjeldame probleeme, mis tekivad sellise uuringu praktilise läbiviimisega turvalise ühisarvutuse keskkonnas. Arutleme te-gelike andmetega töötamisega seotud ohtude ning puudujääkide üle, toome ära privaatsust säilitava andmete teisendamise protsessi detailid ning kirjeldame, kui-das tööriista RMIND antud uuringus kasutama hakatakse.

# ORIGINAL PUBLICATIONS

Publication	Pages
Liina Kamm, Dan Bogdanov, Sven Laur, Jaak Vilo “A new way to protect privacy in large-scale genome-wide association studies”	97 – 104
Liina Kamm, Jan Willemson “Secure Floating-Point Arithmetic and Private Satellite Collision Analysis”	107 – 124
Dan Bogdanov, Liina Kamm, Sven Laur, Pille Pruulmann-Vengerfeldt, Riivo Talviste, Jan Willemson “Privacy-preserving statistical analysis on federated databases”	127 – 152
Dan Bogdanov, Liina Kamm, Sven Laur, Ville Sökk “Rmind: a tool for cryptographically secure statistical analysis”	155 – 194

# CURRICULUM VITAE

## Personal data

Name	Liina Kamm
Birth	February 15th, 1983 Tallinn, Estonia
Citizenship	Estonian
Marital Status	Married
Languages	Estonian, English, French, Russian
Address	Pärna 5, Nõo 61601 Tartumaa Estonia
Contact	+372 51 40 443 liina@cyber.ee

## Education

2007–	University of Tartu, Ph.D. candidate in Computer Science
2005–2007	University of Tartu, M.Sc. in Computer Science
2001–2005	University of Tartu, B.Sc. in Computer Science
1990–2001	Tallinn English College

## Employment

2011–	Cybernetica AS, researcher
2009–2011	Software Technology and Applications Competence Centre, researcher
2006–2009	OÜ Quretec, systems analyst
2005–2006	AS EGeen, systems analyst

# ELULOOKIRJELDUS

## Isikuandmed

Nimi	Liina Kamm
Sünniaeg ja -koht	15. veebruar 1983 Tallinn, Eesti
Kodakondsus	eestlane
Perekonnaseis	abielus
Keelteoskus	eesti, inglise, prantsuse, vene
Aadress	Pärna 5, Nõo 61601 Tartumaa Eesti
Kontaktandmed	+372 51 40 443 liina@cyber.ee

## Haridustee

2007–	Tartu Ülikool, informaatika doktorant
2005–2007	Tartu Ülikool, MSc informaatikas
2001–2005	Tartu Ülikool, BSc informaatikas
1990–2001	Tallinna Inglise Kolledž

## Teenistuskäik

2011–	Cybernetica AS, teadur
2009–2011	Tarkvara Tehnoloogia Arenduskeskus
2006–2009	OÜ Quretec, analüütik
2005–2006	AS EGeen, analüütik

## DISSERTATIONES MATHEMATICAE UNIVERSITATIS TARTUENSIS

1. **Mati Heinloo.** The design of nonhomogeneous spherical vessels, cylindrical tubes and circular discs. Tartu, 1991, 23 p.
2. **Boris Komrakov.** Primitive actions and the Sophus Lie problem. Tartu, 1991, 14 p.
3. **Jaak Heinloo.** Phenomenological (continuum) theory of turbulence. Tartu, 1992, 47 p.
4. **Ants Tauts.** Infinite formulae in intuitionistic logic of higher order. Tartu, 1992, 15 p.
5. **Tarmo Soomere.** Kinetic theory of Rossby waves. Tartu, 1992, 32 p.
6. **Jüri Majak.** Optimization of plastic axisymmetric plates and shells in the case of Von Mises yield condition. Tartu, 1992, 32 p.
7. **Ants Aasma.** Matrix transformations of summability and absolute summability fields of matrix methods. Tartu, 1993, 32 p.
8. **Helle Hein.** Optimization of plastic axisymmetric plates and shells with piece-wise constant thickness. Tartu, 1993, 28 p.
9. **Toomas Kiho.** Study of optimality of iterated Lavrentiev method and its generalizations. Tartu, 1994, 23 p.
10. **Arne Kokk.** Joint spectral theory and extension of non-trivial multiplicative linear functionals. Tartu, 1995, 165 p.
11. **Toomas Lepikult.** Automated calculation of dynamically loaded rigid-plastic structures. Tartu, 1995, 93 p, (in Russian).
12. **Sander Hannus.** Parametrical optimization of the plastic cylindrical shells by taking into account geometrical and physical nonlinearities. Tartu, 1995, 74 p, (in Russian).
13. **Sergei Tupailo.** Hilbert's epsilon-symbol in predicative subsystems of analysis. Tartu, 1996, 134 p.
14. **Enno Saks.** Analysis and optimization of elastic-plastic shafts in torsion. Tartu, 1996, 96 p.
15. **Valdis Laan.** Pullbacks and flatness properties of acts. Tartu, 1999, 90 p.
16. **Märt Pöldvere.** Subspaces of Banach spaces having Phelps' uniqueness property. Tartu, 1999, 74 p.
17. **Jelena Ausekle.** Compactness of operators in Lorentz and Orlicz sequence spaces. Tartu, 1999, 72 p.
18. **Krista Fischer.** Structural mean models for analyzing the effect of compliance in clinical trials. Tartu, 1999, 124 p.



19. **Helger Lipmaa.** Secure and efficient time-stamping systems. Tartu, 1999, 56 p.
20. **Jüri Lember.** Consistency of empirical k-centres. Tartu, 1999, 148 p.
21. **Ella Puman.** Optimization of plastic conical shells. Tartu, 2000, 102 p.
22. **Kaili Müürisep.** Eesti keele arvutigrammatika: süntaks. Tartu, 2000, 107 lk.
23. **Varmo Vene.** Categorical programming with inductive and coinductive types. Tartu, 2000, 116 p.
24. **Olga Sokratova.**  $\Omega$ -rings, their flat and projective acts with some applications. Tartu, 2000, 120 p.
25. **Maria Zeltser.** Investigation of double sequence spaces by soft and hard analytical methods. Tartu, 2001, 154 p.
26. **Ernst Tungel.** Optimization of plastic spherical shells. Tartu, 2001, 90 p.
27. **Tiina Puolakainen.** Eesti keele arvutigrammatika: morfoloogiline ühestamine. Tartu, 2001, 138 p.
28. **Rainis Haller.**  $M(r,s)$ -inequalities. Tartu, 2002, 78 p.
29. **Jan Villemson.** Size-efficient interval time stamps. Tartu, 2002, 82 p.
30. **Eno Tõnisson.** Solving of expression manipulation exercises in computer algebra systems. Tartu, 2002, 92 p.
31. **Mart Abel.** Structure of Gelfand-Mazur algebras. Tartu, 2003. 94 p.
32. **Vladimir Kuchmei.** Affine completeness of some ockham algebras. Tartu, 2003. 100 p.
33. **Olga Dunajeva.** Asymptotic matrix methods in statistical inference problems. Tartu 2003. 78 p.
34. **Mare Tarang.** Stability of the spline collocation method for volterra integro-differential equations. Tartu 2004. 90 p.
35. **Tatjana Nahtman.** Permutation invariance and reparameterizations in linear models. Tartu 2004. 91 p.
36. **Märt Möls.** Linear mixed models with equivalent predictors. Tartu 2004. 70 p.
37. **Kristiina Hakk.** Approximation methods for weakly singular integral equations with discontinuous coefficients. Tartu 2004, 137 p.
38. **Meelis Käärrik.** Fitting sets to probability distributions. Tartu 2005, 90 p.
39. **Inga Parts.** Piecewise polynomial collocation methods for solving weakly singular integro-differential equations. Tartu 2005, 140 p.
40. **Natalia Saecalle.** Convergence and summability with speed of functional series. Tartu 2005, 91 p.
41. **Tanel Kaart.** The reliability of linear mixed models in genetic studies. Tartu 2006, 124 p.
42. **Kadre Torn.** Shear and bending response of inelastic structures to dynamic load. Tartu 2006, 142 p.

43. **Kristel Mikkor.** Uniform factorisation for compact subsets of Banach spaces of operators. Tartu 2006, 72 p.
44. **Darja Saveljeva.** Quadratic and cubic spline collocation for Volterra integral equations. Tartu 2006, 117 p.
45. **Kristo Heero.** Path planning and learning strategies for mobile robots in dynamic partially unknown environments. Tartu 2006, 123 p.
46. **Annely Mürk.** Optimization of inelastic plates with cracks. Tartu 2006. 137 p.
47. **Annemai Raidjõe.** Sequence spaces defined by modulus functions and superposition operators. Tartu 2006, 97 p.
48. **Olga Panova.** Real Gelfand-Mazur algebras. Tartu 2006, 82 p.
49. **Härmel Nestra.** Iteratively defined transfinite trace semantics and program slicing with respect to them. Tartu 2006, 116 p.
50. **Margus Pihlak.** Approximation of multivariate distribution functions. Tartu 2007, 82 p.
51. **Ene Käärrik.** Handling dropouts in repeated measurements using copulas. Tartu 2007, 99 p.
52. **Artur Sepp.** Affine models in mathematical finance: an analytical approach. Tartu 2007, 147 p.
53. **Marina Issakova.** Solving of linear equations, linear inequalities and systems of linear equations in interactive learning environment. Tartu 2007, 170 p.
54. **Kaja Sõstra.** Restriction estimator for domains. Tartu 2007, 104 p.
55. **Kaarel Kaljurand.** Attempto controlled English as a Semantic Web language. Tartu 2007, 162 p.
56. **Mart Anton.** Mechanical modeling of IPMC actuators at large deformations. Tartu 2008, 123 p.
57. **Evely Leetma.** Solution of smoothing problems with obstacles. Tartu 2009, 81 p.
58. **Ants Kaasik.** Estimating ruin probabilities in the Cramér-Lundberg model with heavy-tailed claims. Tartu 2009, 139 p.
59. **Reimo Palm.** Numerical Comparison of Regularization Algorithms for Solving Ill-Posed Problems. Tartu 2010, 105 p.
60. **Indrek Zolk.** The commuting bounded approximation property of Banach spaces. Tartu 2010, 107 p.
61. **Jüri Reimand.** Functional analysis of gene lists, networks and regulatory systems. Tartu 2010, 153 p.
62. **Ahti Peder.** Superpositional Graphs and Finding the Description of Structure by Counting Method. Tartu 2010, 87 p.
63. **Marek Kolk.** Piecewise Polynomial Collocation for Volterra Integral Equations with Singularities. Tartu 2010, 134 p.

64. **Vesal Vojdani.** Static Data Race Analysis of Heap-Manipulating C Programs. Tartu 2010, 137 p.
65. **Larissa Roots.** Free vibrations of stepped cylindrical shells containing cracks. Tartu 2010, 94 p.
66. **Mark Fišel.** Optimizing Statistical Machine Translation via Input Modification. Tartu 2011, 104 p.
67. **Margus Niitsoo.** Black-box Oracle Separation Techniques with Applications in Time-stamping. Tartu 2011, 174 p.
68. **Olga Liivapuu.** Graded  $q$ -differential algebras and algebraic models in noncommutative geometry. Tartu 2011, 112 p.
69. **Aleksei Lissitsin.** Convex approximation properties of Banach spaces. Tartu 2011, 107 p.
70. **Lauri Tart.** Morita equivalence of partially ordered semigroups. Tartu 2011, 101 p.
71. **Siim Karus.** Maintainability of XML Transformations. Tartu 2011, 142 p.
72. **Margus Treumuth.** A Framework for Asynchronous Dialogue Systems: Concepts, Issues and Design Aspects. Tartu 2011, 95 p.
73. **Dmitri Lepp.** Solving simplification problems in the domain of exponents, monomials and polynomials in interactive learning environment T-algebra. Tartu 2011, 202 p.
74. **Meelis Kull.** Statistical enrichment analysis in algorithms for studying gene regulation. Tartu 2011, 151 p.
75. **Nadežda Bazunova.** Differential calculus  $d^3 = 0$  on binary and ternary associative algebras. Tartu 2011, 99 p.
76. **Natalja Lepik.** Estimation of domains under restrictions built upon generalized regression and synthetic estimators. Tartu 2011, 133 p.
77. **Bingsheng Zhang.** Efficient cryptographic protocols for secure and private remote databases. Tartu 2011, 206 p.
78. **Reina Uba.** Merging business process models. Tartu 2011, 166 p.
79. **Uuno Puus.** Structural performance as a success factor in software development projects – Estonian experience. Tartu 2012, 106 p.
80. **Marje Johanson.**  $M(r, s)$ -ideals of compact operators. Tartu 2012, 103 p.
81. **Georg Singer.** Web search engines and complex information needs. Tartu 2012, 218 p.
82. **Vitali Retšnoi.** Vector fields and Lie group representations. Tartu 2012, 108 p.
83. **Dan Bogdanov.** Sharemind: programmable secure computations with practical applications. Tartu 2013, 191 p.
84. **Jevgeni Kabanov.** Towards a more productive Java EE ecosystem. Tartu 2013, 151 p.
85. **Erge Ideon.** Rational spline collocation for boundary value problems. Tartu, 2013, 111 p.
86. **Esta Kägo.** Natural vibrations of elastic stepped plates with cracks. Tartu, 2013, 114 p.

87. **Margus Freudenthal.** Simpl: A toolkit for Domain-Specific Language development in enterprise information systems. Tartu, 2013, 151 p.
88. **Boriss Vlassov.** Optimization of stepped plates in the case of smooth yield surfaces. Tartu, 2013, 104 p.
89. **Elina Safiulina.** Parallel and semiparallel space-like submanifolds of low dimension in pseudo-Euclidean space. Tartu, 2013, 85 p.
90. **Raivo Kolde.** Methods for re-using public gene expression data. Tartu, 2014, 121 p.
91. **Vladimir Šor.** Statistical Approach for Memory Leak Detection in Java Applications. Tartu, 2014, 155 p.
92. **Naved Ahmed.** Deriving Security Requirements from Business Process Models. Tartu, 2014, 171 p.
93. **Kerli Orav-Puurand.** Central Part Interpolation Schemes for Weakly Singular Integral Equations. Tartu, 2014, 109 p.