Towards Efficient Privacy-Preserving Similar Sequence Queries on Outsourced Genomic Databases

Thomas Schneider
TU Darmstadt
schneider@encrypto.cs.tu-darmstadt.de

Oleksandr Tkachenko
TU Darmstadt
tkachenko@encrypto.cs.tu-darmstadt.de

ABSTRACT

Nowadays, genomic sequencing has become affordable for many people. Since more people let analyze their genome, more genome data gets collected. The good side of this is that analyses on this data become possible. However, this raises privacy concerns because the genomic data uniquely identify their owner, contain sensitive information about his/her risk for getting diseases, and even sensitive information about his/her family members.

In this paper, we introduce a highly efficient privacy-preserving protocol for Similar Sequence Queries (SSQs), which can be used for finding genetically similar individuals in an outsourced genomic database aggregated from data of multiple institutions. Our SSQ protocol is based on the edit distance approximation by Asharov et al. (PETS'18), which we extend to the outsourcing scenario. We also improve their protocol by using more efficient building blocks and achieve a 5–6× run-time improvement compared to their work in the two-party scenario.

Recently, Cheng et al. (ASIACCS'18) introduced protocols for outsourced SSQs that rely on homomorphic encryption. Our approach outperforms theirs by more than factor 20 000× in terms of run-time in the outsourcing scenario.

CCS CONCEPTS

• **Security and privacy → Privacy-preserving protocols**; **Management and querying of encrypted data**; **Privacy protections**;

KEYWORDS

medical privacy; privacy-enhancing technologies; genomic research; edit distance; secure computation; outsourcing

ACM Reference Format:
Thomas Schneider and Oleksandr Tkachenko. 2018. Towards Efficient Privacy-Preserving Similar Sequence Queries on Outsourced Genomic Databases. In *2018 Workshop on Privacy in the Electronic Society (WPES'18), October 15, 2018, Toronto, ON, Canada.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3267323.3268956

WPES'18, October 15, 2018, Toronto, ON, Canada

ACM ISBN 978-1-4503-5989-4/18/10…$15.00
https://doi.org/10.1145/3267323.3268956

1 INTRODUCTION

The efforts by the research community, industries, and governments of different countries substantially reduced the costs of genome sequencing: the costs for sequencing a whole genome have fallen from 10 million USD to 1 000 USD in the last ten years [24]. This leads to more genome data being collected, and services that use genome data are becoming increasingly popular, e.g., 23andMe[1], MyHeritage[2], and ancestry[3]. Common use cases for genome data are: (i) Similar Sequence Queries (SSQs) for finding genome sequences that are similar to the sequence of the analyzed person, (ii) Genome-Wide Association Studies (GWAS) for finding associations between diseases and genetic variants, and (iii) genealogical tests for determining ancestral ethnicity of the analyzed person.

In this work, we focus on SSQs. They can be used for finding unknown relatives, and for making more correct diagnoses and prescribing the most promising treatments using the medical history of the people that are genetically similar to the patient [13]. However, a medical institution commonly has only a limited number of collected genome sequences which prevents a high-quality similar patient analysis. As a solution, we consider privacy-preserving aggregation of the databases of multiple institutions.

Despite the good uses of genomic data, their leakage causes severe privacy violations for the genomic data donors. This is due to the fact that genome data are unique for each individual and contain sensitive information about him/her and his/her relatives [7, 15], e.g., ethnicity information and predispositions to particular diseases. The possession of this information by third parties can give rise to genetic discrimination, e.g., if a health insurance company will increase the client's fee based on his/her predispositions to diseases, or if an employer will reject the candidate's application based on the aforementioned reasons. To address this, we employ Secure Multi-Party Computation (SMPC) techniques for constructing efficient privacy-preserving protocols for distributed SSQ. The current solutions are either inefficient or custom-tailored, i.e., it is far from trivial to extend these protocols to privacy-preserving aggregation of databases or thresholding distances to similar sequences.

Our Contributions. We design, implement, and evaluate an efficient generic protocol for distributed privacy-preserving SSQ using approximated Edit Distance (ED) [3] that outperforms recent related work by orders of magnitude. In our protocol, the initialization phase (aggregating the database from multiple medical institutions) is a one-time expense, and the number of institutions affects only the initialization phase. Our protocol relies on a mix of

[1] https://www.23andme.com
[2] https://myheritage.com
[3] https://www.ancestry.com

the GMW [14] protocol, Yao's Garbled Circuits (GCs) [25], Arithmetic sharing [11], and Correlated Oblivious Transfer (C-OT) [4], which are used for computing the respective parts of the algorithms where they perform best. Furthermore, we enhance our algorithms to use Single Instruction Multiple Data (SIMD) operations which allows parallelization. Our implementation on commodity hardware yields more than a factor of 20 000× improvement over [8]. Even compared to the highly efficient two-party protocol of [3], our run-times are by factor 5–6× faster than theirs in the same two-party setting.

2 RELATED WORK

In this section, we compare our ED and SSQ algorithm with that in previous related work. We benchmark our SSQ protocol in the system model of the papers we compare to unless stated otherwise.

The authors of [5, 6] developed protocols for secure sequence comparison with $O(n^2)$ complexity, which is much larger than our complexity $O(n\omega)$, with a small constant ω.

Jha et al. [16] designed algorithms for privacy-preserving ED using Garbled Circuits (GCs). Their construction scales much worse than the recent solutions, e.g., our protocol runs more than 100 000× faster and requires at least by factor 900× less communication.

Wang et al. [23] proposed an extremely efficient approach for approximating ED, which however works only for datasets with less than 0.5 % variability between individuals.

Asharov et al. [3] introduced an approximation technique for ED that can handle high divergence data. Their contribution is twofold: (i) they construct an efficient and precise approximation for ED — which initially requires computation of a matrix with a quadratic size on the query/sequence length — and (ii) they use Look-Up Tables (LUTs) instead of direct computation of the ED, thus precomputing the most expensive parts of the computation in the clear. They split the genome sequences into blocks of small size (e.g., b=5) and pad them to a somewhat greater size (e.g., b'=16). Because of the much smaller size of the blocks compared to a full sequence, the overhead for computing ED is also much smaller. Also, they utilize the fact that genes in the blocks are naturally distributed highly non-uniformly. For all sequences available in the clear, this allows to compute cross-sequence LUTs block-wise for all observed block values. Their protocols work in a setting with two semi-honest parties, where the client inputs the query and the server inputs a database into a Secure Two-Party Computation (STPC) protocol. We show our improvements over their work in the same two-party setting in Section 5.

The authors of [1] designed two approximations for ED: a set intersection method based on [20] and a banded alignment-based algorithm that relies on GCs, but neither algorithm achieves good accuracy on long genome sequences.

Zhu et al. [26] design efficient algorithms for ED, which are, however, 108× slower and require 261× more communication on two 4 000-nucleotide sequences than the ED approximation of [3].

Mahdi et al. [19] securely computed the Hamming distance for SSQ, which is an error-prone metric for genome sequences.

The authors of [8] design an algorithm for privacy-preserving SSQ based on the ED approximation of [3] using homomorphic encryption in an outsourcing scenario with two non-colluding, semi-honest parties. Our algorithm significantly outperforms theirs in the same setting, which we extensively discuss in Section 5.

The privacy of other uses of genome information were also widely addressed in the literature (see, e.g., [10, 21, 22]).

3 PRELIMINARIES

In this section, we explain the basics that are needed later.

3.1 Genomic Primer

The cells of each living individual contain Deoxyribonucleic Acid (DNA), which encodes genome information. Based on DNA, individuals develop different phenotype traits — observable differences between individuals, e.g., hair or eye color. The basic components that form DNA are called nucleotides. The human DNA comprises $3.5 \cdot 10^9$ base pairs from which only 0.1 % vary among individuals [12]. The variations of single nucleotides in specific regions of a chromosome are called alleles, and the genes, each allele of which is represented to some minimal degree in the population (e.g., more than 1 %), are called Single-Nucleotide Polymorphisms (SNPs).

Similar Sequence Query (SSQ). The approach of Similar Sequence Queries is used for finding the sequences that are most similar to the analyzed query. This approach can be used, for example, for finding individuals that are genetically very similar to a patient in order to better analyze the health conditions of the patient. This leads to more precise medical diagnoses based on the additional information provided by genetically similar individuals. However, a precise SSQ algorithm requires the computation of the Edit Distance (ED) [18], which measures how different two sequences are by finding the minimum number of deletions, additions, and substitutions that are required to transform one DNA string into another.

This work focuses on the ED approximation of Asharov et al. [3]. It works as follows: first, the sequences in the database are aligned to a public reference genome and split into blocks of predefined size; then, the statistical distribution of the sequences in the database is used to construct a Look-Up Table (LUT) containing the most frequently observed block values and their distances to each other; afterwards, the value of the block i in the query is compared to each entry of the i-th row of the LUT (the comparison is performed only once for one database), and based on the comparison result pre-computed distances are either selected as output or set to 0; here, the sum of all outputs yields either the correct distance or 0 in the case of an error (this outcome is rare and influences the overall result only very slightly [3]). The last step is to sum up the distances of all blocks which results in the approximated distance between the query and sequence. After computing the approximated ED to the sequences in the database, we use them for finding the indices of the k most similar sequences.

3.2 Secure Multi-Party Computation

Secure Multi-Party Computation (SMPC) allows parties $P_1, \ldots, P_n$ to securely compute a function $f(x_1, \ldots, x_n)$ on their respective inputs without revealing the inputs to each other, i.e., one or more parties learn the result of f, but no intermediate values.

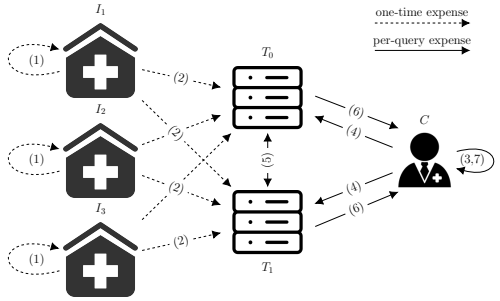

Figure 1: Privacy-preserving Similar Sequence Query system model with three medical institutions I_1, I_2, and I_3 that contribute their secret-shared genomic data to two Semi-Trusted Third Parties T_0 and T_1, and a client C who queries the secret-shared database. The communication links between all parties are protected with a secure channel, e.g., TLS. See Section 4.1 for more details.

We use protocols secure against passive adversaries who honestly follow the protocol specification but try to learn as much information as possible from the information they have.

ABY Framework. In this work, we use the ABY framework [11], which implements state-of-the-art optimizations for Secure Two-Party Computation (STPC) and is secure against passive adversaries. It enables privacy-preserving algorithms using three different STPC protocols called sharings: Arithmetic sharing (a generalization of the GMW protocol [14] to unsigned integers), Boolean sharing (the Boolean GMW protocol [14]), and Yao sharing (Yao's Garbled Circuits (GCs) [25]). It also enables mixing these protocols in order to use particular STPC protocols for the parts of the computed function where they perform best.

4 IMPLEMENTATION

Here, we describe our system model and show implementation details of our privacy-preserving Similar Sequence Query (SSQ) protocol.

4.1 System Model

The main idea of our protocol is to secret-share the database aggregated from data of multiple medical institutions between two non-colluding Semi-Trusted Third Parties (STTPs). We depict our system model in Figure 1. The communication between all parties is performed over a secure channel (e.g., TLS). Note that our protocol alternatively can be run directly between a server and a client with approximately the same efficiency (the outsourcing scenario is beneficial for data aggregation, but has the same efficiency in the querying phase). In our protocol, we consider the following parties:

- Medical institutions $I_1, \ldots, I_\psi$ that securely contribute their genome sequences to the outsourced database in a secret-shared form.
- A client C who privately queries the database with a genome sequence for finding the most similar sequences in the database.
- Two non-colluding STTPs T_0 and T_1 who obliviously compute the Similar Sequence Query (SSQ) protocol.

$\langle distance \rangle^A \leftarrow \mathbf{ED}(seq, query)$

1: $row\_dist \leftarrow \emptyset$
2: **for** $i = 1$ **to** $seq.length$ **do**
3: $\quad \langle row\_sum \rangle^A \leftarrow \langle 0 \rangle^A$
4: $\quad$ **for** $j = 1$ **to** $seq.LUT.width$ **do**
5: $\quad\quad$ //For multiple **ED** computations, eq values are computed only once
6: $\quad\quad$ //Next 2 lines are equal to $e_{i\omega+j} == q_i \ ? \ d_{i\omega+j} \ : \ 0$
7: $\quad\quad \langle eq \rangle^B \leftarrow \langle seq[i].LUT[j].value \rangle^B == \langle query[i] \rangle^B$
8: $\quad\quad \langle dist \rangle^A \leftarrow \mathbf{OTM}(\langle eq \rangle^B, \langle seq[i].LUT[j].dist \rangle^A)$
9: $\quad\quad \langle row\_sum \rangle^A \leftarrow \langle row\_sum \rangle^A + \langle dist \rangle^A$
10: $\quad row\_dist.\mathbf{insert}(\langle row\_sum \rangle^A)$
11: **for** $i = 2$ **to** $seq.length$ **do**
12: $\quad \langle row\_dist[1] \rangle^A \leftarrow \langle row\_dist[1] \rangle^A + \langle row\_dist[j] \rangle^A$
13: **return** $\langle row\_dist[1] \rangle^A$

Algorithm 1: Privacy-preserving Edit Distance (ED) algorithm between a sequence *seq* containing a Look-Up Table with genomic variants and the corresponding distances, and a client's query containing genomic variants. OTM denotes Oblivious Transfer-Based Multiplication.

For running the two STTPs, one must choose two distinct organizations that have a high motivation to not collude, e.g., if they significantly loose in value if caught cheating. We can think of the following organizations: (i) health ministry, (ii) research institutes, or (iii) cloud service providers. This outsourcing model has been widely used in the literature, e.g., in [2, 9, 21].

Our protocol consists of the following steps:

Initialization

(1) Each medical institution I_i locally secret-shares its genomic data and Look-Up Table.
(2) I_i sends the corresponding shares to the STTPs T_0, T_1.

Querying

(3) Client C locally secret-shares its query Q.
(4) C sends the corresponding shares to T_0, T_1.
(5) Having the secret-shared database and client's query, T_0 and T_1 obliviously compute the SSQ protocol using STPC.
(6) T_0 and T_1 send the resulting shares containing secret-shared indices of the most similar sequences in the database to C.
(7) C, having both shares, locally reconstructs the result.

4.2 Privacy-Preserving Approximated Edit Distance

The algorithm for securely computing Edit Distance (ED) between a genome sequence stored in the outsourced database and a client's query is shown in Algorithm 1. Our protocol utilizes the idea of Asharov et al. [3] for improving the efficiency of computation by approximating ED using Look-Up Tables (LUTs) (see Section 3.1).

We carefully optimize the implementation using a mix of different sharing schemes and minimize costly operations, such as conversions between sharings and the operations that require interaction/heavy computations. The Similar Sequence Query (SSQ) algorithm is given in Algorithm 2.

We extend the protocol of [3] for the outsourcing scenario. We improve the protocol of [3] by using more lightweight GMW [14] instead of GCs [25] for comparisons, Correlated Oblivious Transfers (C-OTs) [4] instead of general Oblivious Transfers (OTs), and a more efficient k-NN algorithm in Yao sharing. Although we use a more efficient flavor of OT, we require two OTs instead of one, which is due to the fact that Semi-Trusted Third Parties (STTPs) do not

$ids \leftarrow$ **SSQ**$(sequences, query, k)$

1: $dists \leftarrow \emptyset$
2: $ids \leftarrow \langle 1 \rangle^Y, \ldots, \langle sequences.size \rangle^Y$
3: **for** $i = 1$ **to** $sequences.size$ **do**
4: $\quad \langle dist \rangle^A \leftarrow$ **ED**$(sequences[i], query))$
5: $\quad dists$.**append**(**A2Y**$(\langle dist \rangle^A))$
6: $ids \leftarrow$ **k-NN**$(dists, ids, k)$
7: **return** ids

Algorithm 2: Privacy-preserving Similar Sequence Query (SSQ) algorithm between a query and sequences of genomic data in the outsourced database. k-NN denotes the k-Nearest Neighbors algorithm.

Table 1: Comparison of our SSQ algorithm with that of Asharov et al. [3] with sequence length n=3 470, number of nearest sequences k=5, block length b=4, padded block length b'=16, and one medical institution performed on localhost for different parameters N (number of sequences) and ω (Look-Up Table width).

N	ω	Run-time (localhost / LAN) in s			Communication in MB		
		[3]	Ours	Improvement	[3]	Ours	Improvement
1 000	25	6.03 / -	1.07 / 1.89	5.6× / -	180	130	1.3×
2 000	30	14.11 / -	2.20 / 4.07	6.4× / -	340	268	1.2×
4 000	35	31.60 / -	4.83 / 9.02	6.5× / -	660	571	1.1×

Table 2: Run-time comparison of our Edit Distance algorithm with that of Cheng et al. [8] with sequence length n, Look-Up Table width ω=20, and block size b=2.

n	Run-time (LAN)		Run-time
	[8]	Ours	improvement
10	1.2 s	2.1 ms	571×
20	2.2 s	3.0 ms	733×
30	3.4 s	3.1 ms	1 096×
40	4.7 s	3.1 ms	1 516×
50	6.0 s	3.2 ms	1 875×

possess the LUTs in cleartext. However, the cost for OTs remain approximately the same as in the protocol of [3] for large databases (less than 1 % overhead for a database with 10 000 sequences). In contrast to [8], we compute most of the functionalities using generic protocols which enables arbitrary extensions of our protocol.

4.3 Privacy-Preserving Similar Sequence Queries

In this work, we consider a system model where the client wants to find k genome sequences that are most similar to its query among the sequences stored in the outsourced database. For this purpose, we proceed as follows: the distances to single sequences are first calculated using the Similar Sequence Query (SSQ) algorithm, and afterwards the distances are used along with the corresponding ids for finding the k closest distances using the k-NN algorithm (see Algorithm 2).

4.4 Security Analysis

Our protocol is based on the outsourcing protocol of Kamara and Raykova [17], which gives a generic construction and a security proof for turning an N-party SMPC protocol into a secure outsourcing scheme where data is oursourced to N non-colluding STTPs. We instantiate SMPC with the N = 2-party ABY framework. Our protocols implement the algorithm for SSQs of [3]. Hence, our protocols are secure against malicious medical institutes and clients (who all send a single message in the protocol), and secure against semi-honest STTPs.

Table 3: Comparison of our SSQ algorithm with that of Cheng et al. [8] with sequence length n=500, Look-Up Table width ω=20, block size b=5, number of blocks t=20, number of most similar sequences k=10, and number of sequences N.

N	Run-time (LAN)		Run-time	Communication		Communication
	[8]	Ours	improvement	[8]	Ours	improvement
100	25 min	62 ms	24 193×	50 MB	3 MB	16×
200	50 min	96 ms	31 250×	100 MB	4 MB	25×
300	80 min	132 ms	36 363×	150 MB	5 MB	30×
400	105 min	171 ms	36 842×	200 MB	6 MB	33×
500	135 min	207 ms	39 130×	250 MB	7 MB	35×

5 EVALUATION

Our two Semi-Trusted Third Parties (STTPs) are each implemented by a standard PC equipped with an Intel Core i7-4770K 3.5 GHz processor and 32 GB RAM. They are connected via a 1 Gbit/s network with an average latency of 0.1 ms. All protocols are instantiated with a symmetric security parameter of 128 bit. We intentionally exclude the evaluation part for the preprocessing (extensively discussed in [3]) and aggregation of the databases (extensively discussed in [21]).

In the following, we compare our implementation with recent related works.

Asharov et al. [3]. In Table 1, we compare our algorithms in the benchmark setting of [3, Table 3], where both parties, client and server, are run on one machine (our protocol can trivially be applied for direct STPC between the client and server). In addition, we benchmark our algorithm in a local network (LAN).

Please note that the authors do not detail their hardware setting, whereas we benchmark our algorithm on commodity hardware. Our algorithm outperforms that of Asharov et al. by factor 5-6× in run-time which is due to more light-weight building blocks. Furthermore, our algorithm is still by factor 3× faster even on a LAN, compared to the localhost benchmarks of [3]. The communication of our algorithm is slightly lower.

Cheng et al. [8]. The most recent related work that covers privacy-preserving ED computations in the outsourcing scenario is [8]. Unfortunately, the authors give the exact run-times only for the ED computation between two sequences. We compare our ED run-times in the system setting of [8], i.e., the outsourcing scenario, with the run-times of their most efficient protocol in Table 2.

We achieve a significant run-time improvement over [8] of 500× to 1 800×. We can increase this even further when many sequences are computed in parallel. For this, we (optimistically for [8]) approximate the benchmarking results of [8, Figure 4 (a)] in Table 3. As a result of the comparison, our algorithm outperforms that of Cheng et al. by more than factor 20 000× in run-time and by more than factor 15× in communication.

Acknowledgements. This work has been co-funded by the DFG as part of project E4 within the CRC 1119 CROSSING, and by the German Federal Ministry of Education and Research (BMBF) and the Hessen State Ministry for Higher Education, Research and the Arts (HMWK) within CRISP.

REFERENCES

[1] Md Momin Al Aziz, Dima Alhadidi, and Noman Mohammed. 2017. Secure approximation of edit distance on genomic data. In *BMC medical genomics*.

[2] Gilad Asharov, Daniel Demmler, Michael Schapira, Thomas Schneider, Gil Segev, Scott Shenker, and Michael Zohner. 2017. Privacy-preserving interdomain routing at Internet scale. In *Privacy Enhancing Technologies Symposium (PETS)*.

[3] Gilad Asharov, Shai Halevi, Yehuda Lindell, and Tal Rabin. 2018. Privacy-preserving search of similar patients in genomic data. In *Privacy Enhancing Technologies Symposium (PETS)*.

[4] Gilad Asharov, Yehuda Lindell, Thomas Schneider, and Michael Zohner. 2013. More efficient oblivious transfer and extensions for faster secure computation. In *ACM SIGSAG Conference on Computer and Communications Security (CCS)*.

[5] Mikhail J. Atallah, Florian Kerschbaum, and Wenliang Du. 2003. Secure and private sequence comparisons. In *ACM Workshop on Privacy in the Electronic Society (WPES)*.

[6] Mikhail J. Atallah and Jiangtao Li. 2005. Secure outsourcing of sequence comparisons. In *International Journal of Information Security*.

[7] Erman Ayday. 2016. Cryptographic solutions for genomic privacy. In *Financial Cryptography and Data Security (FC)*.

[8] Ke Cheng, Yantian Hou, and Liangmin Wang. 2018. Secure similar sequence query on outsourced genomic data. In *ACM Asia Conference on Computer and Communications Security (ASIACCS)*.

[9] Marco Chiesa, Daniel Demmler, Marco Canini, Michael Schapira, and Thomas Schneider. 2017. SIXPACK: securing internet exchange points against curious onlookers. In *International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*.

[10] Daniel Demmler, Kay Hamacher, Thomas Schneider, and Sebastian Stammler. 2017. Privacy-preserving whole-genome variant queries. In *Cryptology and Network Security (CANS)*.

[11] Daniel Demmler, Thomas Schneider, and Michael Zohner. 2015. ABY - a framework for efficient mixed-protocol secure two-party computation. In *Network and Distributed System Security Symposium (NDSS)*.

[12] European Bioinformatics Institute. 2017. Genome Reference Consortium Human Build 38, Ensembl release 91. http://dec2017.archive.ensembl.org/Homo_sapiens/Info/Annotation. (2017). Accessed on 2018-03-27.

[13] Roger Allan Ford and W. Nicholson Price II. 2016. Privacy and accountability in black-box medicine. *Michigan Telecommunications and Technology Law Review* (2016).

[14] Oded Goldreich, Silvio Micali, and Avi Wigderson. 1987. How to play any mental game. In *ACM Symposium on Theory of Computing (STOC)*.

[15] Mathias Humbert, Erman Ayday, Jean-Pierre Hubaux, and Amalio Telenti. 2015. On non-cooperative genomic privacy. In *Financial Cryptography and Data Security (FC)*.

[16] Somesh Jha, Louis Kruger, and Vitaly Shmatikov. 2008. Towards practical privacy for genomic computation. In *IEEE Symposium on Security and Privacy (S&P)*.

[17] Seny Kamara and Marina Raykova. 2011. Secure outsourced computation in a multi-tenant cloud. In *IBM Workshop on Cryptography and Security in Clouds*.

[18] Vladimir I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*.

[19] Md Safiur Rahman Mahdi, Mohammad Zahidul Hasan, and Noman Mohammed. 2017. Secure sequence similarity search on encrypted genomic data. In *IEEE/ACM Conference on Connected Health: Applications, Systems and Engineering Technologies*.

[20] Benny Pinkas, Thomas Schneider, Gil Segev, and Michael Zohner. 2015. Phasing: private set intersection using permutation-based hashing. In *USENIX Security Symposium*.

[21] Oleksandr Tkachenko, Christian Weinert, Thomas Schneider, and Kay Hamacher. 2018. Large-scale privacy-preserving statistical computations for distributed genome-wide association studies. In *ACM Asia Conference on Computer and Communications Security (ASIACCS)*.

[22] Bing Wang, Wei Song, Wenjing Lou, and Y. Thomas Hou. 2017. Privacy-preserving pattern matching over encrypted genetic data in cloud computing. In *IEEE Conference on Computer Communications (INFOCOM)*.

[23] Xiao Shaun Wang, Yan Huang, Yongan Zhao, Haixu Tang, XiaoFeng Wang, and Diyue Bu. 2015. Efficient genome-wide, privacy-preserving similar patient query based on private edit distance. In *ACM SIGSAG Conference on Computer and Communications Security (CCS)*.

[24] Kris A. Wetterstrand. 2017. DNA sequencing costs: Data from the NHGRI Genome Sequencing Program (GSP). (2017). http://www.genome.gov/sequencingcostsdata. Accessed on 2018-03-23.

[25] Andrew Yao. 1986. How to generate and exchange secrets. In *Foundations of Computer Science (FOCS)*.

[26] Ruiyu Zhu and Yan Huang. 2017. Efficient privacy-preserving general edit distance and beyond. Cryptology ePrint Archive, Report 2017/683. (2017). https://ia.cr/2017/683.