TOWARDS PRIVACY PRESERVING OF FORENSIC DNA DATABASES

A Thesis

by

SANMIN LIU

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirement for the degree of

MASTER OF SCIENCE

December 2011

Major Subject: Computer Science

TOWARDS PRIVACY PRESERVING OF FORENSIC DNA DATABASES

A Thesis

by

SANMIN LIU

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirement for the degree of

MASTER OF SCIENCE

December 2011

Major Subject: Computer Science

ABSTRACT

Towards Privacy Preserving of Forensic DNA Databases. (December 2011)

Sanmin Liu, B.S., Huazhong University of Science & Technology;

M.E., Huazhong University of Science & Technology

Chair of Advisory Committee: Dr. Jyh-Charn Liu

Protecting privacy of individuals is critical for forensic genetics. In a kinship/identity testing, related DNA profiles between user's query and the DNA database need to be extracted. However, unrelated profiles cannot be revealed to each other. The challenge is today's DNA database usually contains millions of DNA profiles, which is too big to perform privacy-preserving query with current cryptosystem directly. In this thesis, we propose a scalable system to support privacy-preserving query in DNA Database. A two-phase strategy is designed: the first is a Short Tandem Repeat index tree for quick fetching candidate profiles from disk. It groups loci of DNA profiles by matching probability, so as to reduce I/O cost required to find a particular profile. The second is an Elliptic Curve Cryptosystem based privacy-preserving matching engine, which performs match between candidates and user's sample. In particular, a privacy-preserving DNA profile matching algorithm is designed, which achieves $O(n)$ computing time and communication cost. Experimental results show that our system performs well at query latency, query hit rate, and communication cost. For a database of one billion profiles, it takes 80 seconds to return results to the user.

To my parents

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to Dr. Jyh-Charn Liu for being my advisor and guiding me through this thesis. Interacting with him during this research work was a great learning experience and I thank him for making me part of the Real Time Distributed Systems Lab. I am thankful for his regular feedback and constant motivation which help me complete this thesis. I also thank Dr. Riccardo Bettati and Dr. Shuhua Yuan for being part of my thesis committee.

I would like to acknowledge the company of all members of the Real Time Distributed Systems Lab and in particular thank Pu Duan and Jonas Tan for their help during my research. I also thank my colleagues in Yuan & Dai's Lab and friends in College Station for keeping me in good spirits.

Last, but not least, I am indebted to my parents for their love and constant support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Nowadays privacy concerns have become one of the most important security issues in many Internet applications. One of them is in the area of forensic science, which has experienced significant progress recently. In 1984, DNA Fingerprint was used to describe the analysis of polymorphic regions of DNA. In the following year, at the request of the United Kingdom Home office, DNA profiling was successfully applied to resolve an immigration dispute. The use of genetics was quickly adopted by the forensic community and plays an important role worldwide in the investigation of crime [1].

1.1 DNA Profiling

A major research opportunity exists on how to develop a high efficient matching algorithm to support DNA database forensics, which will not suffer re-identification from malicious users. Here, we assume a query from a client is safe when only related DNA profiles between user and database could be learned by each other, without leaking whole content of unrelated profile to each other. Through partial value of a profile will be disclosed between client and server, it can still guarantee each individual’s privacy.

Forensic genetics is a branch of genetics that deals with the application of genetic knowledge to legal problems and legal proceedings. Forensic genetics is also a branch of forensic medicine that deals more broadly with the application of medical knowledge to

The journal model is *IEEE/ACM Transactions on Networking*.

legal matters [1]. Figure 1 describes the basic steps to identify criminal with forensic genetics.

Short Tandem Repeats (STRs) are currently the most common technique used for DNA forensics. STR is a pattern of two or more nucleotides repeated and the repeated sequences are directly adjacent to each other. Typically, the core unit is between 1-6 base pairs (bp), while the repeats range from 50-300bp. We call this repeat pattern *allele*, which is represented as a float number. In human genomes, there are totally more than 10,000 STR loci [Table III shown on page 41]. Due to a high polymorphism between individuals and strong stability across time, STR are used in many applications, including identification, parentage and kinship testing, genetic genealogy and answering historical questions, etc.

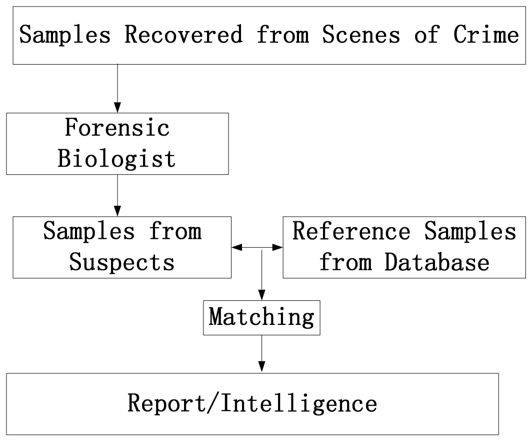

Samples Recovered from Scenes of Crime
Forensic Biologist
Samples from Suspects
Reference Samples from Database
Matching
Report/Intelligence

Fig. 1. Forensic genetics in identifying criminal

Since the 1990s, many countries have setup national DNA profile databases to assist in the identification of suspect of crimes. Among these, the Combined DNA Index System (CODIS) of the United States Federal Bureau of Investigation (FBI) contains over

6,300,000 offender profiles and 241,000 forensic profiles. CODIS employ 13 STR loci for identification [7]. Typically, a test is declared negative with exclusion at three or more loci [1].

While STR provides a reliable technique for identifying criminal, it can also be misused for re-identification, infer ethnic origin, and also for disease detection [5] [9] [10]. It has been found that these attacks in practice can often be facilitated by the unique background knowledge of human genomes, i.e., the availability of a reference population that resembles case individuals.

1.2 Kinship Testing

Kinship testing is used to match human remains from disaster scenes to declared parents. In many big disasters, we need to identify the victims so as to further deal with the results. In cases that involve hundreds of victims, remains need to be claimed by their families. Due to severe damage to the remains, potential families could be a large pool. Besides of this, it is very useful to identify suspects without accessibility to their DNA sample but their parents'. Kinship testing could be conducted in two situations: both parents appear, and just one parent appears.

(1) Kinship testing with one parent

In this case, we have access to the DNA profile of one potential parent. And we need to identify if a DNA sample from a crime scene is the potential child of this declared parent. Since a child inherits DNA from both parents evenly, there is a half matching between one parent and child. In reality, since sometimes mutations exist, it is a standard practice to require exclusion on three or more loci before a test is declared negative [1]. We use (X_{i1}, X_{i2}) and $(Y_{i1}, Y_{i2}), i = 1,2, \dots, n$ to represent the DNA profile of a child and a potential parent, respectively, then we have if

$$\sum_{i=1}^{N} [(X_{i1}, X_{i2}) \cap (Y_{i1}, Y_{i2}) \neq \emptyset] \geq N - 2,$$

then there is a significant probability that there is a kinship between the two given samples.

(2) Kinship testing with two parents

In this case, we have access to a DNA profile from crime scene and DNA profiles of both declared parents. And we would like to confirm the kinship between the criminal and two referred parents. Since we have two DNA reference, we can argue that the test is positive only when the DNA profile matches both the mother and father evenly. Then we have if

$$\sum_{i=1}^{N} \{[X_{i1} \in (Y_{i1}, Y_{i2})] \cap [X_{i2} \in (Z_{i1}, Z_{i2})]\} \geq N - 2,$$

then there is a significant probability that the sample is a child of the potential parents.

1.3 Identity Testing

Identity testing is used for matching a suspect to the samples from the scene of crime. In this case, we are able to obtain DNA sample from suspects directly. Some suspects may be innocent. If a suspect is convicted of the crime, his/her DNA profile will be loaded to the DNA database for future reference. Here, we use $(\mathrm{S}_{\mathrm{i}1}, \mathrm{S}_{\mathrm{i}2}), \mathrm{i} = 1,2, \dots, \mathrm{n}$, to denote the DNA profile of a suspect. Then, if

$$\sum_{\mathrm{i}=1}^{\mathrm{N}} [(\mathrm{X}_{\mathrm{i}1}, \mathrm{X}_{\mathrm{i}2}) = (\mathrm{S}_{\mathrm{i}1}, \mathrm{S}_{\mathrm{i}2})] \geq \mathrm{N} - 2,$$

for each locus, it is matched when two alleles are both equal to the reference. If totally at least *N - 2* loci are matched, we say the suspect is the criminal with a significant probability.

1.4 Protecting DNA Information

DNA information is sensitive for public usage. Abused usage can lead to the disclosure of an individual's characteristics, such as race and disease. There are two main problems for developing efficient and secure privacy-preserving DNA profile matching: (1) to protect the privacy of individuals, we want to extract related DNA profiles between user's query and database; unrelated information cannot be revealed to each other. F. Bruekers [3] proposed a privacy-preserving STR matching scheme based on the privacy-preserving set matching algorithm developed by M. Freedman [6]. Basically, they treated each DNA profile as a set with 13 pairs of allele values, and use Freedman's algorithm to compare these sets. Results can be deduced from the difference between two profiles. However, Freedman's algorithm also has high computing cost as the database grows big. Therefore, for database like CODIS, we need a better solution. (2) How to design efficient and secure protocols for DNA profile data matching? Suppose Alice sends a profile to Bob, they want to find the matching profiles without disclosing one's unmatched data to the other. This is the main technique for DNA based identification.

In this paper, we provide an efficient privacy-preserving DNA profile searching system, which could quickly detect all related entries in the database with the query. The major contributions of this paper are: (1) Profiles in database are indexed with a *matching probability* biased tree; (2) A two-step profile matching scheme is designed, with each step filtering out large amount of unrelated entries; (3) a high-performance ECC-based privacy preserving matching engine is developed for set matching.

CHAPTER II

RELATED WORK

2.1 Existing DNA Databases

In 1994, the US movement passed the DNA Identification Act which enables the establishment of the Combined DNA Index System. CODIS is funded by United States federal bureau of investigation (FBI). In essential, CODIS contains National DNA Index System (NDIS), State DNA Index System (SDIS) and Local DNA Index System (LDIS). Each sample in the database must include a laboratory identifier, a specimen identifier, and information to classify/review the integrity of the DNA record. The FBI selects 13 STR loci for developing database. Totally, there are four segments in CODIS:

- The forensic Index contains NDA profiles from crime scene samples
- The offender Index contains DNA profiles of individuals convicted of certain crime
- Unidentified human remains
- Relatives of missing persons

As of October 2009, the situation of CODIS is:

- Total number of offender profiles: 6,300,000
- Total number of forensic profiles: 241,000

Currently all 50 states have mandatory DNA collection from certain felony offenses like sexual assault and homicide, and 47 states collect DNA from all convicted felons. The

American Civil Liberties Union worries about the increasing use of collecting DNA from arrested suspects.

The UK National DNA database is established in 1995. NDNAD has mainly two segments:

- profiles generated from scenes of crime (263,000 by 2005)
- Profiles generated from individuals (3,450,000 by 2005)

With large amount of profiles in NDNAD, there is currently a 45% chance to match a sample from scenes to the database. More ethical concerns have been raised since 75% of young black males aged from 15 to 34 are collected while 22% of young white people are collected.

2.2 Performance vs. Privacy

2.2.1 Search Time

Search time is the time from starting a query to getting the results from DNA database server. According to [2], currently search performance of CODIS is approximately 5 seconds with a database of 100,000 profiles. This means that a big national database of one billion profiles, such as China and India, will require 5,000 seconds. This is inacceptable for a real-time query. To speedup CODIS, a hierarchical database is proposed in [2]. It gathers similar DNA profiles, and deploys them onto different machines. A query is divided into n communication parts, with each part handled by a machine separately. A parallel virtual machine (PVM) based computing architecture is designed to perform database search concurrently. However, there is still no performance evaluation for this idea.

2.2.2 Related Cryptographic Techniques

Most of the privacy-preserving genomic computation protocols rely on two cryptographic techniques: secure multi-party computation (SMC) and private set intersection (PSI). The idea of secure multi-party computation (SMC) was first proposed in [13] based on the solutions to the Millionaire's Problem. A general solution for the problem of secure two-party computation was later presented in [14]. The idea of SMC was to enable a set of untrusting parties to compute certain common function based on their own private inputs. After a successful secure multi-party computation each party only knows the output from the function and does not know anything about other's inputs.

The building block of SMC is Oblivious Transfer (OT) protocol [15]. It allows a receiver to choose one secret from any two secret generated by a sender without knowing the other one, while the sender cannot know which secret the receiver chose. The complexity to compute an OT for each bit of two parties' message (e.g., genomic data sequences) and transfer the whole garbled circuits makes SMC hard to implement for real-world security applications. While privacy-preserving computation of edit distance for measurement of similarity between gene sequences was achieved in [16] based on SMC, even the optimized SMC cannot handle genome computations of a realistic scale [17]. A PSI protocol aims to compute the intersection of elements provided by two parties (say Alice and Bob) without exposing other information. That is, if and only if Alice and Bob have matching elements, the equality of the matching elements should be computable by both parties.

In the process, outsiders cannot infer any of the matching results. Alice cannot know unmatched elements of Bob, and vice versa. Like SMC protocols, previous work on PSI designs had O(n^2) [6], where n is the number of elements of two user's sets. This is considered to be too high to be useable for real-world applications [18]. In addition, this is no existing implementation of using PSI protocols for complex string computations/comparisons like edit distance.

2.2.3 Existing Privacy-preserving Genomic Computation Protocols

[21] proposed the first protocol for privacy-preserving sequence comparisons such that one party reveals nothing about his private sequence to the other party. Their solution was extended to support privacy-preserving Smith-Waterman local nucleotide comparisons in [22]. Several more efficient secure genomic computation protocols were proposed in [16] that supported calculations of both edit distance and Smith-Waterman similarity scores between two sequences. The authors developed their protocols based on secure multi-party computation (SMC) [13][14] and evaluated their prototype by implementing it on sequences from the *Pfam* [23] database of protein families.

The most recent privacy-preserving genomic computation framework [17] was based on program specialization. In their framework genomic data was categorized as sensitive data and public data. The partition allowed a concrete biology computation on public data and a symbolic computation on sensitive data such that protection on sensitive genomic data was efficiently achieved. Another privacy-preserving matching protocol [3] was proposed to allow two parties to find their matching short tandem repeats (STRs) [24], which was the main technique for DNA based identification. Their solution was based on a private set intersection (PSI) [6] protocol so that two parties could find their matching STRs without disclosing the content s of their sequence to each other.

CHAPTER III

PRIVACY PRESERVING OF FORENSIC DNA DATABASES

3.1 Basics and Assumptions

Usually the DNA data gathered for genetic forensics can be conveniently accessed by authorized agents to benefit the social security. However, such data dissemination needs to be considered with the protection of participants' privacy. This becomes very important since unveiling of the gathered DNA data can cause serious consequence. For example, revealing the identity of a case individual usually relates him to the disease, which may cause denial of access to health/life insurance, education, and employment. Prior research shows that raw DNA data (genotypes) is often too risky to publish even after removal of explicit identifiers [25] (such as name, social security number, etc.) since recovery of a participant's identity can be achieved through examining his genetic markers related to his observable features [26][27]. So aggregate genome data, such as allele frequencies (i.e., the frequencies of different SNP values) are usually displayed because such data covers an individual's information with that of others and is thought to be of less sensitive. For example, the NHGRI/NIH used to make allele frequencies publicly available.

However, some recent work [28][29] show that even public allele frequencies and test statistics can incur effective attacks to participants' identities or SNP sequences. Homer et al. [29] found that the presence of an individual in a case group can be determined from allele frequencies, given the assumption that the victim's DNA profile has been acquired (from a single hair or a drop of blood). Another study [28] show that even the test statistics (e.g., *p*-value, *r*-squares) calculated from allele frequencies and published in HGS papers could facilitate attacks in some cases enough for identifying participants or recovering portions of their DNA sequences. Usually there are two types of common aggregated data, the allele frequencies for both individual SNPs and SNP pairs, and the test statistics derived from the frequencies. Such data is studied under two typical threats, identification attack and recovery attack. The former uses an individual's DNA profile to determine his relation with an aggregate data-set [8][28][29][30]. The latter re-constructs individuals' SNP sequences from test statistics. It has been found that these attacks in practice can often be facilitated by the unique background knowledge of human genomes, i.e., the availability of a reference population that resembles case individuals.

In general, protecting privacy of individual DNA when the profile is public to any potential attackers does not appear realistic. Developing solutions that can support kinship/identity testing without releasing holder's sensitive data (e.g., DNA profiles in CODIS and NDNAD) to others is perhaps one of the most important privacy challenges in forensic genetics. For this purpose researchers have developed some tools/solutions. Most of the solutions rely heavily on various cryptographic techniques. Despite their different degrees of success, it is well recognized that most existing solutions face challenges of poor system scalability, high computing costs, and inadequate security protections [18][31]. In summary, there are two main problems for developing efficient and secure privacy-preserving DNA profile matching protocols.

Problem 1: How to manage millions of DNA profiles to facilitate forensic query? Suppose Alice has ten million DNA profiles and wants to keep these data on a general computer she can afford. As we know, 10 million profiles will take up more than 1GB space. For a national DNA database, the number of profiles could be as big as one billion. Hence, instead of main memory, we have to put this data on hard disk, for which space is not a problem. To quickly locate a profile on the disk, we have to achieve the following goals:

G(1): How to organize all this DNA profiles on hard disk, so that we can operate (search/add/remove/update) a profile efficiently?

G(2): How to design an efficient algorithm to find all the related profiles to the query?

We will design a novel index tree to organize the massive DNA profiles based on the allele of each locus. Basically, since each locus has a different distribution of alleles, we propose a strategy to discriminate these loci wisely.

Problem 2: How to design efficient and secure protocols for basic DNA profile matching? Suppose Alice and Bob each has a set of DNA profiles and they want to find the related profiles without disclosing one’s unrelated profiles to each other. This is the main technique for DNA based identification. Here we view each allele as a single element (unit). Then the privacy-preserving element matching protocol should satisfy the following requirements:

R(1): If and only if Alice and Bob have matching elements, the equality of the elements must be computable by both parties.

R(2): Outsiders cannot infer any elements. Bob cannot know unmatched elements of Alice, and vice versa.

We propose a new privacy-preserving element matching protocol to solve these technical challenges based on ECDLP [32] and elliptic curve cryptosystem (ECC)-based homomorphism [33]. More specifically, ECDLP guarantees protection of a party's elements when they are associated with an ECC point (R(2)) and ECC-based homomorphism guarantees computability of encrypted elements (R(1)). For example, Alice can encrypt her elements by associating them with an elliptic point and send them to Bob. Based on ECC-based homomorphism, Bob can modify Alice's encrypted elements using his own elements. After Alice receives the modified elements, she can perform equality check on the returned modified elements to find out the intersection as the matching elements. Compared with the existing work [3][6], the proposed protocol has linear computational and communication costs. We also evaluate the protocol by implementing it for matching of simulated DNA profiles.

There are two types of attacks:

- Semi-honest attacks: Attackers follow appropriate computation protocols, but want to gain others' private information
- Malicious attacks: Attackers can do anything to obtain private information of the other parties, e.g., do not honor agreements, send false information, involve in collaborations with other attackers.

Our paper is based on the following two assumptions:

Assumption 1: to protect individual's privacy, only related profiles between users and database could be learned by each other.

Assumption 2: users are semi-honest. They will not break the rules in order to gain private information. Our protocol is designed to resist passive dictionary attacks launched by semi-honest adversary, but it is not designed to defend against active attacks, e.g., forging of attributes, traffic interception to block correlation.

3.2 Output Formats

There are generally two different kinds of results users want to get: (1) is my sample related to any of the profiles in DNA database? (2) If yes, what are the allele values of all the related profiles? For (1), the user has many samples from crime scene, and wants to know if which belongs to a potential criminal. They submit the samples to a reference database, e.g. CODIS, and just want to know *yes or no.* In doing so, right samples could be followed up. For example, after a disaster, we need to find the families of the victims. First, the government needs to obtain DNA samples for crime scene. Due to the severe destroy of disaster, many unrelated samples other than victims' are extracted. First we need to filter out these noises. To do this, a query is initiated to a reference DNA database where the victims should appear. A good reference database might be a national DNA database like NDNAD, which contains most citizens' samples. For the second output format, we would like to the extract content of related profiles, and use them for further analysis, e.g., compare them between each other to better identify the criminal.

CHAPTER IV

DESIGNING PRIVACY-PRESERVING SYSTEM

Population genetics is a study of factors affecting the allele and genotype frequencies of different genetic loci in a population. The Hardy-Weinberg law provides a simple mathematical representation of the relationship of genotype and allele frequencies within an ideal population. It states that within a randomly mating population the genotype frequencies at any single genetic locus remain constant [1]. To meet HW law, there are five preliminaries:

- the population is infinitely large;

  If the population is limited, then the frequency of alleles [4] will change through a genetic drift process. In a small population, the effect of genetic drift is more significant. However, most countries and races have a large population which genetic drift could be ignored.

- random mating occurs within the population;

  Since STR genotypes do not affect human's phenotype, such as height, intelligence, or strength, selection of STR through mating is unlikely.

- the population is free from the effects of migration;

  Human history is full of migration and this could affect allele frequency obviously. However, populations with different culture, language, religions are not likely to merge in a short term.

- there is no natural selection;

The loci used for forensic genetics are not located within functionally important regions. Hence, they are unlikely facing the impact of natural selection.

- no mutations occur.

The mutation of STR loci are relatively low at less than 0.2% per generation and do not have oblivious effect on allele frequency.

In [34], allele frequency of six groups of populations, including African Americans, U.S. Caucasians, Hispanics, Bahamians, Jamaicans, and Trinidadians are reported. The *TPOX* locus was the least discriminating locus for Caucasians and Hispanics, while *D12S317* is the least discriminating one for African-based populations. Also, there is little departure from HW laws in any of the six populations. To characterize the allele of each population, at least one hundred individuals need to be collected. The larger the database, the more representative of the population it will be.

From Table III [shown on page 41], we know that the allele distributions of 13 core loci are very diverse. For *FGA*, there are totally 16 different alleles, while 10 of them having a frequency larger than 1%. While for *TH01*, there are 6 alleles. And all of them appear bigger than 1%. To better represent this diversity, locus matching probability is proposed. We will show how to use MP to build our STR index tree for quick search.

4.1 Database Organization

The workflow of our system is as shown in Figure 2: (1) Firstly, we calculate allele distribution for each locus according to the entries in the database. To locate a profile efficiently, locus with lower matching probability is compared first. An STR index tree is build based on the matching probability of locus. The resulting STR indexing strategy is stored at the *coordinator* of server; (2) Secondly, when a client initiates a query, its coordinator will request indexing strategy from server. After receiving the indexing strategy, the client will re-order its STR into as $S_1, \ldots, S_{k,} \ldots, S_N$; (3) Then, we start a progressive plain-text matching procedure, from S_1, S_2, …, to S_k. For each round of matching, we record the total number of exclusion (number of different locus value), once the number exceed the threshold, we exclude all entries linked with S_k. (4) Finally, all candidates are further matched for the remaining loci, using privacy-preserving matching engine. Server's coordinator access all the entries meet the criteria from disk and send them to the client.

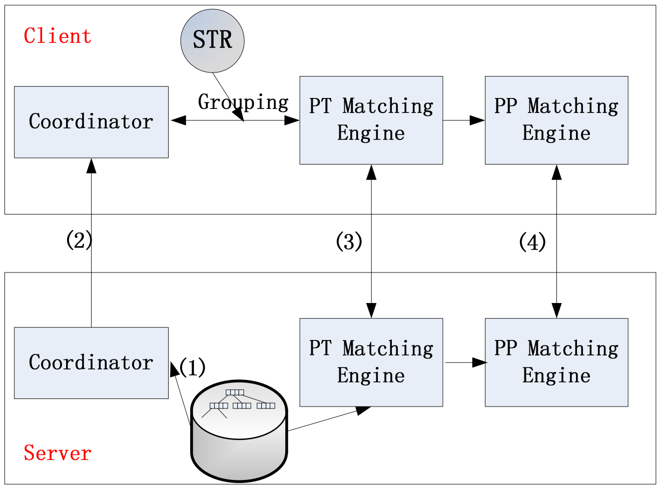


Fig. 2. System overview (PT represents Plain-Text, PP represents Privacy-Preserving)

4.1.1 Matching Probability for Multi-locus DNA Profiles

Sampling error, single bands and correlations within and between loci always exist. They bring uncertainty of matching probability between two random people in a population. Traditional method use product rule to calculate the matching probability for multi-loci DNA profiles. Aspects of the product rule have been criticized by many authors. Krane [35] points out that there is a higher matching probability in a sub-population than in a large population. Since a small sub-population infer a high relative frequency. When matching probability is small, the effect of relative frequency is significant.

Sampling error comes from experiment condition limit, or labor's error. It is inevitable in many cases. Table I shows how sampling error would affect our result. For a profile with 7 loci, if we ignore two alleles, then the matching probability increases from 10^{-7} to 1.45×10^{-6}.

Table I — Matching probabilities for 7 loci samples

Profile with 7 loci	**Matching probability**
original	10^{-7}
2 allele ignored	1.45×10^{-6}
3 allele ignored	5×10^{-6}

4.1.2 STR Index Tree

As illustrated in Section 3.1, a real DNA database could contains billions of profiles, which will requires hundreds of Gera Bytes to store. Thus, it is not suitable to store this data all in RAM. An alternative is to store them in external disk which is usually big enough. The disadvantage of external disk lies in it is relatively slow to access, 20 times lower than RAM. For a 7200 rpm ATA hard disk, it will take 200 seconds to read 10GB data. If we want to search for a profile on disk, we need to scan half of the data on average. To get rid of this pain, a better way is to organize the data with an index tree. Two basic requirements for the index tree are: (1) First, it could be put into memory for quick access. (2) Second, It indexes the location of each profile on disk, so as to find any profile quickly.

We use locus as the nodes of the index tree. Basically, it is a k-level tree, with each level representing the allele of one locus. The leaves point to the location of the disk block where the candidates are. The number of entries in the tree would be approximately p^k, here p is the expected fan-out of each node. For a 13-loci DNA database, $k = 6$, and $p = 10$. Hence the tree takes up 40MB space, which is totally fit to memory.

4.2 PTME: Plain-text Element Matching Engine

A DNA database usually contains millions of records. If we access every entry from the disk, it will induce a big overhead. Instead, we use an index tree to reduce the I/O cost. For a query,

$$[\ (X_{11}, X_{12}), (X_{21}, X_{22}), \ldots, (X_{N1}, X_{N2})\ ]$$

and a DNA database

$$\left\{\begin{matrix} [\ (Y_{11}, Y_{12}), (Y_{21}, Y_{22}), \ldots, (Y_{N1}, Y_{N2})\ ]_1 \\ [\ (Y_{11}, Y_{12}), (Y_{21}, Y_{22}), \ldots, (Y_{N1}, Y_{N2})\ ]_2 \\ . \\ . \\ . \\ [\ (Y_{11}, Y_{12}), (Y_{21}, Y_{22}), \ldots, (Y_{N1}, Y_{N2})\ ]_M \end{matrix}\right\}$$

Our goal is to find out all the entries which may be related to the query. This is measured by comparing two DNA profiles without leaking whole profiles between the client and the server. First, for each locus, we calculate matching probability based on the allele value frequency in the database.

$$MP = \sum_{i=1}^{n} P_i^4 + 4\sum_{j>i}^{n}(P_iP_j)^2 \quad (\,P_i\text{: probability of allele value equals to } i\,)$$

Particularly, for a N-loci database (N = 13 for CODIS), we first calculate MP_x, x = 1, …, N, and then sort them ascendant. Suppose the ordered MP_x is as follows: $(MP_i, MP_j, MP_{k,}, \ldots)$. Then we can generate the following index tree for all the entries as shown in Figure 3.

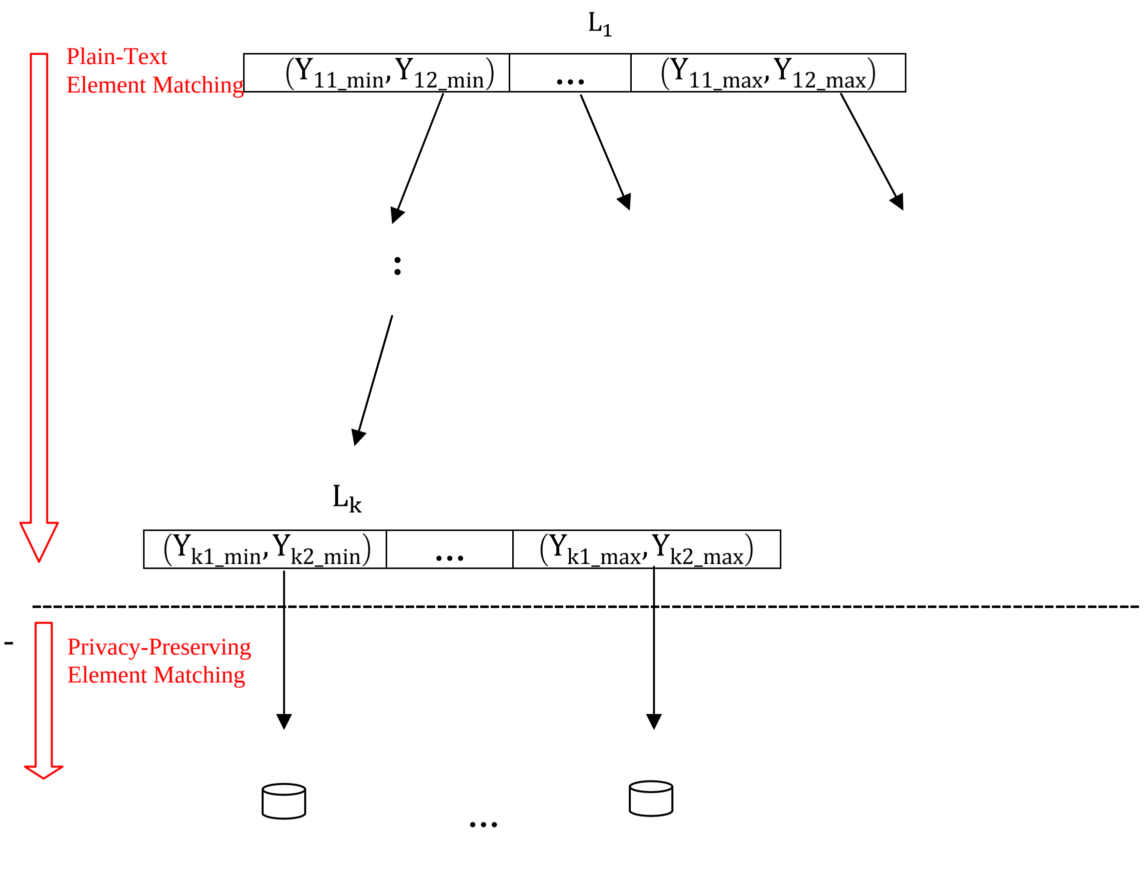


Fig. 3. Structure of STR index tree

As described in Figure 4, for nodes in the tree, the keys are alleles of locus. And for each allele, it has a pointer links to its sub-trees. Each node also has an augment hash-table. Basically, key is allele, and value is a pointer to its child node. With the hash-table, constant time can be achieved to search on each level of the tree.

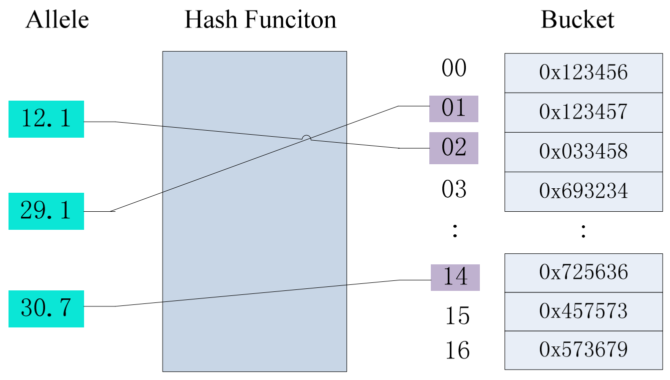
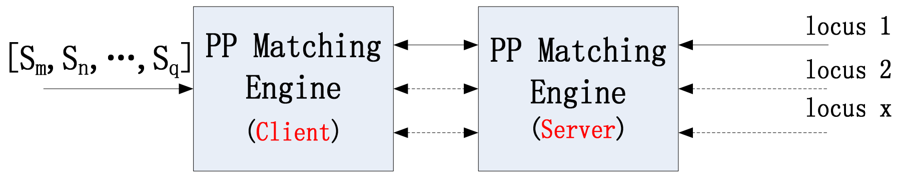


Fig. 4. Structure of nodes

4.3 PPME: Privacy-preserving Element Matching Engine

A privacy-preserving matching engine (PPME) is developed for the set comparison of two DNA profiles. PPME use elliptic curve discrete logarithm problem (ECDLP), which is immune from passive attacks (e.g., dictionary attack). Briefly, PPME is the first demonstration of crypto-based alert correlation which has the following key features: (1) linear running time, i.e., $O(n)$, where n is the number of input attributes, (2) proven privacy protection, and (3) secure against passive (dictionary) attacks. Figure 5 describes the procedure of our Privacy-Preserving matching engine. Firstly, the client loads its last $N - k$ alleles to its PPME module. At the same time, the PPME of server fetches the last $N - k$ alleles of all candidates. Secondly, for each candidate, a privacy-preserving set matching is performed for these alleles between the client and server. Based on the matching result, server will know which candidates meet the criteria to be related to the sample. Finally, server will send related profiles to user.



[S_m, S_n, ···, S_q]
PP Matching Engine (Client)
PP Matching Engine (Server)
locus 1
locus 2
locus x

Fig. 5. Privacy-preserving set matching procedure

Figure 6 presents the protocol of Privacy-Preserving Set Matching. First Alice uses ***E*** to generate the ciphertexts ***E***(A) = $[E(a_0), E(a_1), \ldots, E(a_n)]$ on the (hashed) input elements A, where each ciphertext $E(a_i)$ consists of two elliptic curve points as its "front" point and "back" point. The "front" points are produced by associating Alice's chosen random numbers with base point P. The "back" points are produced by associating both the random numbers and Alice's elements with her computed point Q_A, where Alice's elements cannot be deduced because of ECDLP. Alice then sends ***E***(A) to Bob. After receiving ***E***(A), Bob uses ***M*** and his own elements B to produce ***M***$[B,$ ***E***$(A)]$, i.e., *modification* of ***E***(A). The modification function ***M*** allows Bob to (1) associate both his own elements B and one chosen random number with the "front" points of ***E***(A), and (2) associate the same random number with the "back" points of ***E***(A).

Again, Bob's elements cannot be computed from the modified "back" points because of ECDLP. Then Bob sends his ***M***$[B,$ ***E***$(A)]$ back to Alice, who can use ***D*** to decrypt-and-then-evaluate the "front" and "back" points. After the decryption (i.e., eliminating the difference between the "front" points and "back" points caused by

Alice's own secret number and random number) if a pair of "front" point and "back" point are equal, Alice and Bob have an identical (matched) element. Otherwise, their elements are different. Bob has symmetric operations as Alice and therefore will not be further repeated here. Details of the protocol are discussed as follows [20].

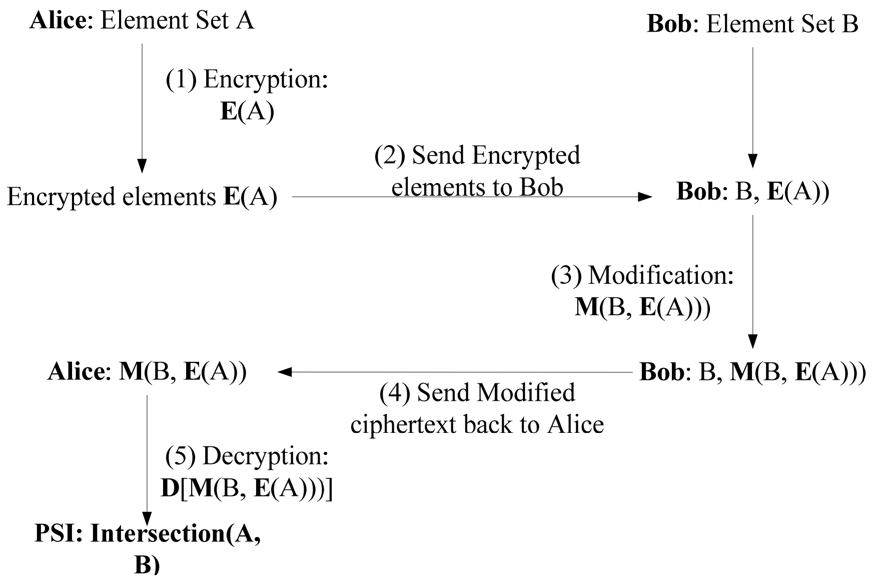

Alice: Element Set A
Bob: Element Set B
(1) Encryption: **E**(A)
Encrypted elements **E**(A)
(2) Send Encrypted elements to Bob
Bob: B, **E**(A))
(3) Modification: **M**(B, **E**(A)))
Bob: B, **M**(B, **E**(A)))
(4) Send Modified ciphertext back to Alice
Alice: **M**(B, **E**(A))
(5) Decryption: **D**[**M**(B, **E**(A)))]
PSI: Intersection(A, B)

Fig. 6. Flowchart of the privacy-preserving element matching protocol

Computation Complexity and Communication Complexity Analysis: We use the multiplications of large random numbers with ECC points to protect user's elements from dictionary attacks, i.e., $H_1(a_i)$ cannot be deduced from $E(a_i)_b = [r_{A1}H_1(a_i) + r_{A2}]Q_A$. In addition, I also use H_1 on $\boldsymbol{M}[A, \boldsymbol{E}(B)]$ and $\boldsymbol{M}[B, \boldsymbol{E}(A)]$ such that Alice (or Bob) cannot further manipulate the received ciphertexts as ECC points. Otherwise, inside adversaries, e.g., Alice, could launch dictionary attack to try every possible values for elements to get an equivalent equation between $H_1[k_A^{-1}r_{A2}^{-1}M(a_i)_b]$ and $M(b_j)_f$. If $n = m$, both Alice and Bob need $(4n + 6)$ (i.e., O(n)) point multiplications to compute the intersection: one multiplication to associate secret number with the base point, *(n + 2)* multiplications to generate the ciphertexts (step 1), *(2n + 2)* multiplications to execute ***M*** on received ciphertexts (step 3), *(n + 1)* multiplications to execute ***D*** on the modified ciphertexts (step 5) for decryption. Table II presents the comparison of computing costs of Protocol I and the PSI protocols used in [3][6]. Here n denote the number of elements of both Alice and Bob, and L is the bit length of each input element (assume all elements have the same length). Through Table II we can find that compared to [3][6] our protocol has linear computation costs based on same security model and different cryptography fundamentals.

Table II— Comparison between PPME and existing work

	Freedman's Method	**PPME**
Computation Cost	O(n^2)	O(n)
Communication Cost	O(n)	O(n)
Security Analysis	secure to passive attack	secure to passive attack

4.4 Security Analysis

Algorithm 1: Privacy-preserving Matching of DNA profiles

```
1. User A sends the first k alleles to database S
2. S use PTME to find candidates; and load them into memory
3. S initiates PPME with A to match the latter N - k loci
4. S sends all related profiles to A.
```

For the remaining *N* - *k* loci, we could not continue with the PTME strategy. Otherwise, whole content of unrelated profiles will be disclosed to each other. How to select an optimal *N* – *k*? It depends on two factors: first, the space should be big enough to avoid re-identification; Second, privacy-preserving matching on these *N* - *k* loci, should be finished in a reasonable time, e.g. several minute.

In this process,

- From the view point of *A*, it learned all related profiles in the database.
- From the view point of *B*, it definitely learns the first *k* loci of *A*. For the latter *N* - *k* loci, it can guess *A*'s allele from the matched ones. In total, *B* cannot learn *A* unless *A* is related to at less one sample in the database. In other words, if *A* is unrelated to any profile in *B*, then the unmatched allele in latter *N* - *k* loci could not be guessed by *B*. Hence, the whole procedure is secure.

CHAPTER V

EVALUATION

5.1 Database Simulation

5.1.1 Single-race, Multiple-races

To the best of our knowledge, due to security concern, there is no public human DNA database available. However, many researchers have summarized the characteristics of different DNA databases. One of these characteristics is allele distribution, which shows the stable allele frequency in each population. We decide to utilize the allele distributions for 13 STR loci published in [34] to generate our simulated database. Basically, our DNA database contains the characteristics of real population, i.e., allele distribution. And its size will be similar as real DNA database. As shown in [34], different populations/races have different allele distributions. For a multiple-population based database, we just need to get the size of each population, and then generate sub-databases for each population according to their allele distribution.

5.2 Index Tree Size vs. Disk Size

In this section, we perform simulations to calculate the space taken by index tree and the database. In these simulations, we set $k = 6$, and change the size of the database. Since all profiles are stored on disk, we can tell that the space of disk grows linearly as the number of DNA profiles increased. For index tree, since the level equals to k and the fan out is in the range of [6, 16], we can tell there is an upper bound for the space requirement. As shown in Figure 7, the maximum space taken by STR index tree is nearly 10MB. On the other hand, when the number of profiles in database rises to 1 billion, 100 GB will be needed to hold all the profiles.

5.3 Query Latency

Query latency is the time to get all the related profiles for a query. It shows the responsiveness of the matching system to the user, with lower latency indicating better responsiveness. In these experiments, we measure the average query latency of all the successful queries at a given *k*. Particularly, time cost of PTME and PPME are analyzed respectively.

To evaluate time complexity of PTME and PPME, we implemented a DNA profile generator to create one set of DNA profiles for database server on PC_1 (Intel Core Dual 2.00GHz, 2GB RAM) and a query from PC_2 (Intel Xeon 2.40GHz, 2GB RAM). We change the number of profiles in database from 10^4 to 10^5, …, 10^9, and perform database query respectively.

5.3.1 Phase I: PTME

As the first phase of our system, the performance of PTME is critical. In this section, we shows the time required to search in STR index tree in order to find all candidate profiles, which will be used for phase II. Since STR index tree is in memory and it is basically a tree structure. The effectiveness of matching probability is evaluated. Figure 8 compares PTME with MP-unbiased method using query latency measured in seconds. Observe that PTME has lower query latency than MP-unbiased index trees. One can get up to 1.8 times faster query responses in PTME when compared to MP-unbiased index tree. Figure 8 also tells that the time to locate candidates grows logarithmically as the size of database increase. This is because as the database grows, more candidates exist in the index trees. Therefore, more search time is needed totally.

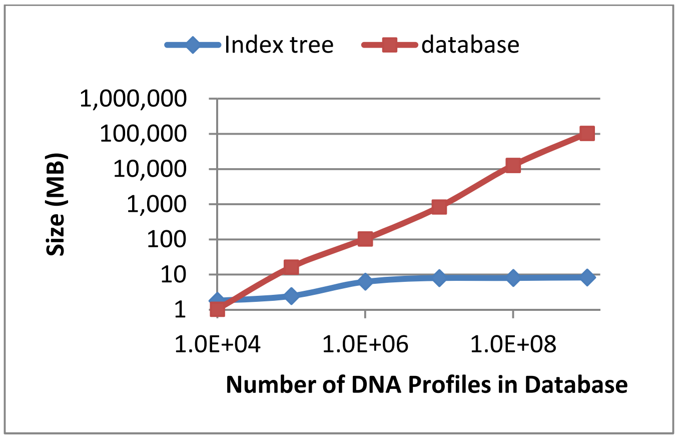


Fig. 7. Space cost of index tree

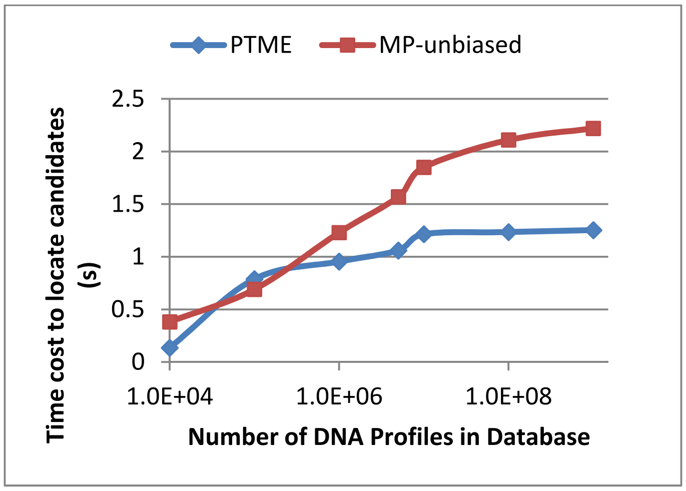


Fig. 8. Time cost of plain-text element matching engine

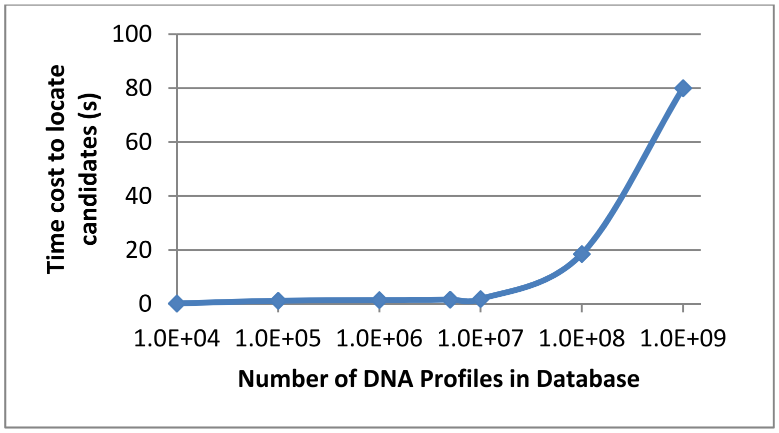


Fig. 9. Time cost of privacy-preserving element matching engine

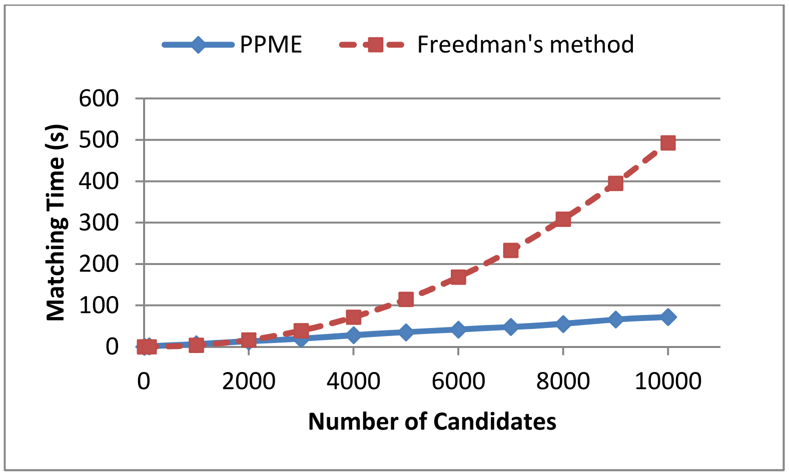


Fig. 10. The relationship between candidate number and PPME time cost

5.3.2 Phase II: PPME

After we load all candidates into memory, we are able to evaluate the time cost for privacy-preserving matching between user's samples and these candidates.

We have implemented PPME with the public crypto library of MIRACL [11]. I chose G1 as an additive group of points on E: $y^2 = x^3 + x$, with the prime order $q = 2^{159} + 2^{17} + 1$, and SHA-1 [12] is selected as H_1.. Each experiment is repeated for ten times to get the average computing costs, and the results are presented in Figure 9 and Figure 10. The computing cost grows linearly with the number of DNA profiles: 1 second is needed for matching of 100 profiles, while 72 seconds for matching of 10,000 profiles.

5.4 Query Hit Rate

A query is said to be successful if it results in at least one related profile hit. Total number of query-hits returned for a query started in the system can also be used as an end-to-end metric to compare different DNA databases. Figure 11 shows the comparison using this metric. As the size of the database grows, the system achieves higher number of query-hits, which climb up to 95% as at the size of 10^9, while a 10^7-scale database only guarantees a 8% query-hit rate. This infers that more related profiles can be discovered in a bigger database.

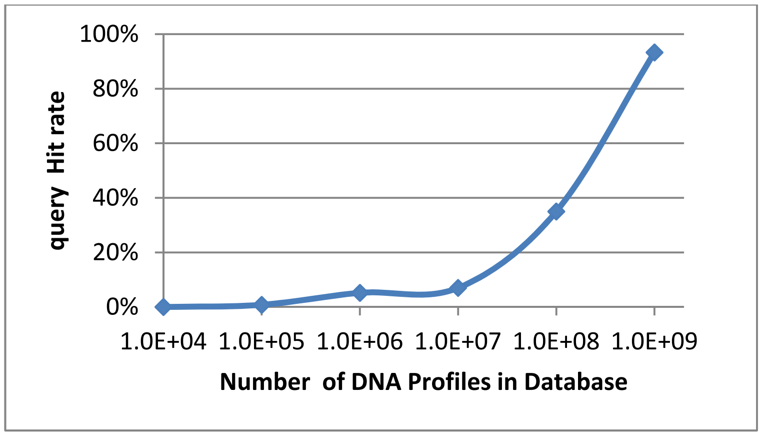


Fig. 11. Query hit rate

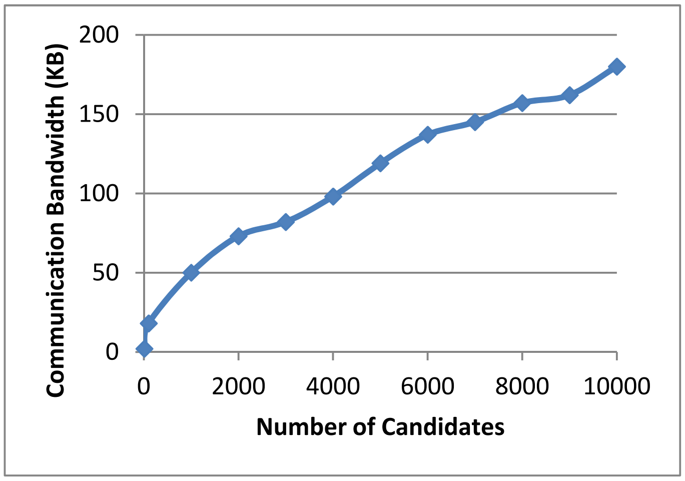


Fig. 12. Communication cost between user and database

5.5 Communication Cost

We also measure the average communication cost of privacy-preserving profile matching between user and database, i.e., network bandwidth, at the database's side in the system. A database's bandwidth cost limits the number of incoming queries it can process concurrently. When a database is overloaded, the query of a user is delayed and added to an output queue of infinite length. For equal query rates, the database achieves smaller bandwidth cost would have less queue length at its users' side. Figure 12 compares the average bandwidth cost for different size of databases. It shows that the bandwidth requirement grows linearly with the number of candidates, based on the measurements made by Wireshark [19]. Observe that at the size of 10^9, the average bandwidth is less than 200 KB. It means 625 queries can be performed concurrently with a 1Gb Network Interface Card.

CHAPTER VI

CONCLUSION

This thesis has proved a high-performance system for privacy-preserving DNA database forensics. We provide two key techniques related to DNA database query. The STR index tree aims at indexing the disk location of each DNA profile in the database. Locus matching probability is used to build a light-weight index tree. It groups DNA profiles based on allele. Moreover, we develop a privacy-preserving matching engine, which is used to find related profiles without leaking unrelated profiles between user and the database. The simulations have shown that PTME has faster query latency than MP-unbiased method. PPME performs much better than previous methods in a million-scale database. The results will help the community to develop new techniques to cope with data security while trying to find related entries in the database, at large scale.

REFERENCES

[1] W. Goodwin, A. Linacre, and S. Hadi, *An Introduction to Forensic Genetics*. Hoboken, NJ: John Wiley and Sons, 2007.

[2] J. D. Birdwell, R. D. Horn, D. J. Icove, T. W. Wang, P. Yadav, and S. Niezgoda, “A hierarchical database design and search method for codis,” in *Proc. HIC*, Sep.1999, pp. 2-3.

[3] F. Bruekers, S. Katzenbeisser, K. Kursawe, and P. Tuyls, “Privacy-preserving matching of dna profiles,” ACR Cryptology ePrint Archive, 2008, Tech. Rep. 2008-203.

[4] “EHSTRAFD—Earth Human STR Allele Frequencies Database. c2009-2010,” [online]. Available: http://www.ehstrafd.org, Aug. 2011.

[5] B. Malin, “An evaluation of the current state of genomic data privacy protection technology and a roadmap for the future,” *J. of the Amer. Med. Inform. Assoc.*, vol. 12. no. 1, pp. 28–34, 2005.

[6] M. Freedman, K. Nissim, and B. Pinkas, “Efficient private matching and set intersection,” in *Proc. EUROCRYPT*, May 2004, vol. 3027, pp. 1–19.

[7] “CODIS—combined DNA index system,” [online]. Available: http://www.fbi.gov/ hq/lab/codis, Aug. 2011.

[8] A. L. Lowe, A. Urquhart, L. A. Foreman, and I. W. Evett, “Inferring ethnic origin by means of an STR profile,” *Forensic Sci. Int.*, vol. 119, pp. 17–22, 2001.

[9] P. Bohannon, M. Jakobsson, and S. Srikwan, “Cryptographic approaches to privacy in forensic DNA databases,” in *Proc. Intern. Workshop on Prac. and Theory in Public Key Crypt.*, Jan. 2000, vol. 1751, pp. 373–390.

[10] D. E. Riley and J. N. Krieger, “Short tandem repeat(STR) replacements in UTRs and introns suggest an important role for certain STRs in gene expression and disease,” *Gene*, vol. 344, pp. 203-211, Jan. 2005.

[11] “MIRACL Library,” [online]. Available: http://indigo.ie/~mscott/, Aug. 2011.

[12] “US Secure Hash Algorithm 1 (SHA-1),” [online]. Available: http://tools.ietf.org/html/rfc3174, Aug. 2011.

[13] A. C. Yao, “Protocols for secure computations,” in *Proc. FOCS*, Nov. 1982, pp. 160-164.

[14] A. C. Yao, “How to generate and exchange secrets,” in *Proc. FOCS*, Oct. 1986, pp. 162–167.

[15] M. Bellare, and S. Micali, “Non-interactive oblivious transfer and applications,” in *Proc. CRYPTO*, Aug. 1989, pp. 547-557.

[16] S. Jha, L. Kruger, and V. Shmatikov, “Towards practical privacy for genomic computation,” In *Proc. IEEE S&P*, May 2008, pp. 216-230.

[17] R. Wang, X. Wang, Z. Li, H. Tang, M. Reiter, and Z. Dong, “Privacy-preserving genomic computation through program specialization,” in *Proc. ACM CCS*, Nov. 2009, pp. 338-347.

[18] H. Ringberg, B. Applebaum, M. J. Freedman, M. Caesar, and J. Rexford, “Collaborative, Privacy-Preserving Data Aggregation at Scale,” [online]. Available: http://eprint.iacr.org/2009/180.pdf, May 2009.

[19] “Wireshark,” [online]. Available: http://www.wireshark.org/, Aug. 2011.

[20] P. Duan, S. Liu, W. Ma, G. Gu and J. C. Liu, “Privacy-Preserving Matching Protocols for Attributes and Strings,” [online]. Available: http://eprint.iacr.org/2010/061.pdf, Aug. 2011.

[21] M. Atallah, F. Kerschbaum, and W. Du, “Secure and private sequence comparisons,” *in Proc. WPES*, Oct. 2003, pp. 39-44.

[22] E. Szajda, M. Pohl, J. Owen, and B. Lawson, “Toward a practical data privacy scheme for a distributed implementation of the smith-waterman genome sequence comparison algorithm,” in *Proc. NDSS*, Feb. 2006, pp. 253-265.

[23] A. Bateman, E. Birney, L. Cerruti, R. Durbin, L. Etwiller, S. R. Eddy, S. Jones, K. L. Howe, M. Marshall, and E. L. Sonnhammer, “The Pfam protein families database,” *Nucleic Acids Res.*, vol. 30, no. 1, pp. 276–280, Jan. 2002.

[24] “Short Tandem Repeat (STR),” [online]. Available: http://en.wikipedia.org/wiki/Short_tandem_repeat, Aug. 2011.

[25] B. Malin, “Protecting dna sequence anonymity with generalization lattices.” CMU, 2007, Tech. Rep. CMU-ISRI-04-134.

[26] “Genomic Privacy Project,” [online]. Available: http://privacy.cs.cmu.edu/dataprivacy/projects/genetic/, Aug. 2011.

[27] "Policy for sharing of data obtained in NIH supported or conducted genome-wide association studies (gwas)," [online]. Available: http://grants.nih.gov/grants/guide/notice-files/not-od-07-088.html, Aug. 2010.

[28] R. Wang, Y. F. Li, X. Wang, H. Tang, and X. Zhou, "Learning your identity and disease from research papers: information leaks in genome wide association study," in *Proc. ACM CCS*, Nov. 2009, pp. 534–544.

[29] N. Homer, S. Szelinger, M. Redman, D. Duggan, W. Tembe, J. Muehling, J. V. Pearson, D. A. Stephan, S. F. Nelso, and D. W. Craig, "Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays," *PLoS. Genet.*, vol. 4, no. 8, pp. 1000167, doi: 10.1371/journal.pgen.1000167, 2008.

[30] S. Sankararaman, G. Obozinski, M. I. Jordan, and E. Halperin, "Genomic privacy and limits of individual detection in a pool," *Nat. Genet.*, vol. 41, no. 9, pp. 965–967, 2009.

[31] W. Du and M. J. Atallah, "Secure multi-party computation problems and their applications: A review and open problems," in *Proc. NSPW*, Sep. 2001, pp. 11-20.

[32] L. C. Washington, *Elliptic Curves: Number Theory and Cryptography*. London, England: Chapman & Hall/CRC, 2003.

[33] H. Cohen, G. Frey, and R. Avanzi, *Handbook of Elliptic Curve and Hyperelliptic Curve Cryptography*. Boca Raton, FL: CRC Press, 2006.

[34] B. Budowle, T. R. Moretti, A. L. Baumstark, D. A. Defenbaugh, and K. M. Keys, "Population data on the thirteen CODIS core short tandem repeat loci in African

Americans, U.S. Caucasians, Hispanics, Bahamians, Jamaicans, and Trinidadians," *J. Forensic. Sc.i*, vol. 44, no.6, pp. 1277–1286, 1999.

[35] D. E. Krane, R. W. Allen, S. A. Sawyer, D. A. Petrov, and D.L. Hartl, "Genetic differences at four DNA typing loci in Finnish, Italian and mixed Caucasian populations," *Proc. Natl. Acad. Sci*, vol. 89, pp. 10583-10587, 1992.

APPENDIX

Table III— Observed allele distributions (as %) for 13 STR loci in African American

Alelles	D3S1358	VWA	FGA	D8S1179	D21S11	D18S51	D5S818	D13S317	D7S820	CSF1PO	TPOX	TH01	D16S539
6				0.3							8.6	11.0	
7							0.3		0.7	4.3	2.2	44.1	
8							5.0	3.6	17.4	8.6	36.8	18.6	3.6
9				0.6			1.4	2.8	15.7	3.3	18.2	14.5	19.9
9.3												10.5	11.0
10	0.5		0.3	2.5		0.6	6.4	5.0	32.4	27.1	9.3	1.4	29.4
11		0.3		3.6		0.6	26.1	23.7	22.4	20.5	22.5		18.7
12	0.2			10.8		5.8	35.6	48.3	9.1	30	2.4		16.5
13	1.2	0.6		22.2		5.6	24.4	12.6	1.9	5.5			1.0
14	12.1	6.7		33.3		6.4	0.6	3.6	0.5	0.7			
15	29.1	23.6		21.4		16.7		0.3					
16	30.7	26.9		4.4		18.9							
17	20	18.3		0.8		16.4							
18	5.5	13.6	0.8			13.1							
18.2			0.8										
19	0.5	7.2	5.3			7.8							
19.2			0.3										
20	0.2	2.8	7.2			5.6	0.3						
21			12.5			1.1							
22			22.5			0.6							
23			12.5										
24			18.6		0.3								
25			10			0.6							
26			3.6		0.3								
27			2.2		6.2								
28			1.7		21.5								
29			0.6		19.0								
29.2					0.3								
30			0.3		17.9								
>30			0.3		34.6								

VITA

Sanmin Liu received his Bachelor of Science and Master of Engineering in computer science from Huazhong University of Science & Technology, China, in 2006 and 2008 respectively. He began pursuing his Master of Science in computer science at Texas A&M University in August 2008 and graduated in December 2011.

His research interests include network & computer security. He may be contacted at:

Sanmin Liu

Department of Computer Science and Engineering

Texas A&M University

College Station

Texas - 77843-3128

The typist for this thesis was Sanmin Liu.