Secure outsourcing of DNA sequences comparisons in a Grid environment

RACHEL AKIMANA, OLIVIER MARKOWITCH and YVES ROGGEMAN
Département d'Informatique
Université Libre de Bruxelles
Bd. du Triomphe – CP212, 1050 Bruxelles
BELGIUM

Abstract: Computing and data Grids are widely distributed computing systems usually used to resolve scientific or technical problems that require a large amount of computing power and/or storage resources. To be really attractive, Grids must provide secured environments (in terms of confidentiality, data integrity, entity identification, etc). In this paper, we consider the confidentiality aspects of Grid's applications related to string matching. We take as an example the area of genetic biology and, more precisely, the search of DNA similarities. Since DNA sequences comparisons need greedy and sensitive computations, we propose a model allowing to search DNA similarities in a public DNA database on the Grid. The model is related to private approximate string matching problem where neither the inputs nor the outputs of the comparisons are revealed. We analyze the performance of our proposed DNA disguising method by taking into account how the edit distances between the client's queries and their corresponding disguises are distributed along the DNA sequences. In order to outweigh the client's load of the initial proposed model, we propose also an extension of our model where the client's load is executed by a third untrusted server.

Key-Words: Grid systems, Secure outsourcing, Secure approximate matching

1 Introduction

Computing and data Grids are widely distributed computing systems usually used to resolve scientific or technical problems that require a large amount of computing power and/or storage resources. Since a lot of different users are using Grid's resources, the risks of eavesdropping of data and information that are stored or processed on Grid resources, or even that are traveling on the Grid's network, cannot be disregarded.

Large amount of data are stored on Grid's resources, and some of them may be related to individual private information (e.g. medical data, biological data, genetic data, etc.). In this case, confidentiality issues and protection of the users' privacy must be studied carefully. Moreover, confidentiality issues for sensitive data have to be adapted to Grid specificities. For example we have to take into account the fact that data may be stored or processed on a remote and possibly untrusted Grid node. In this work, the word *data* will be taken in a broad sense including data resulting from simulations and experiments that are organized in databases on the Grid as well as executables codes of jobs to be processed on the Grid. We will show that existing solutions for confidentiality issues in a Grid system as SSL for example ensure the confidentiality of data during their transport phase but do not guarantee the confidentiality to sensitive computation during their execution. We will focus our interest on the confidentiality aspects of genetic applications on the Grid; more precisely, in the search of DNA similarities on DNA sequences stored in Grid's databases. Such databases may be used in the elucidation of crimes, the establishment of DNA similarities for paternity test, the determination of genetic diseases ...

The DNA sequence comparisons are expensive computations since one DNA sequence may contain thousands to millions of nucleotides. Therefore such comparisons need powerful computing resources. Grids are of course an appropriate environment for such computations. A remote DNA sequence comparison mechanisms may be a sensitive computation in the sense that we may have to ensure that the DNA sequences are not subject to unauthorized tests whose outcome could have such serious consequences [3] (as jeopardizing an individual's insurability or employability, etc).

On the basis of these security and computing power requirements, we propose a disguise model allowing to search DNA similarities in a public DNA database on the Grid in such a way that neither the inputs nor the outputs of the comparisons are revealed to the computing node. This work is related to problems of Private Information Matching with a public database [7] where a client searches similarities to a given item in a public

database without revealing neither the client's item nor the output of the comparison. Solutions to the private information matching with private databases (PIM) are proposed in [7]. These solutions are also used to private information matching with a public database (PIMPD). In this paper, we propose a specific solution to a PIMPD problem that consider the fact that the database is public. The model exploits the opportunities offered by the data replication service in Grid system as the possibility of replication of a database on several servers. The fact that the database is public will be profitable in the sense that the client will download as many as possible DNA sequences. This avoids the use of a third untrusted entity as it is done in PIM solutions. This work provides an example of a practical grid enabled application that preserves in addition the confidentiality on the application.

Since there may be clients that do not have enough computing power to execute the computations needed in the solution, we propose an extension of the model in which these computations are executed by a third server. We will show that this extended version is simply the PIM model corresponding to the proposed PIMPD model in the context of DNA sequence comparisons.

The rest of this paper is organized as follows: after this introduction, in section 2, we examine confidentiality requirements for sensitive data during their transport in a Grid environment. In section 3, we consider the problem of secure outsourcing of sensitive computations. In section 4, we consider a particular case of secure outsourcing which is related to string matching framework and we propose a solution to a private string matching problem which is applied to DNA sequence comparison in a Grid system. In section5, in order to outweigh the computations done by the client while preserving the application's confidentiality , we extend the proposed model to its PIM version.

2 Confidentiality of transmitted data

When we deal with confidential information, we have to consider the fact that, among the entities involved in the transport of the corresponding data or in their processing, some may be authorized to read the data whereas others may not.

The "transport phase" concerns messages transmitted between Grid entities or jobs that are migrating on Grid nodes. The protocol SSL is integrated for confidentiality issues in the security layers of two well-known grid middlewares Globus and Unicore. In Legion middleware [2], it is up to users to choose which mechanisms they assume to be secure enough for their security requirements (identification, login, delegation, confidentiality). SSL is used to transmit messages between Grid entities in such a way that those only authorized to read the message can understand it. The protocol is initiated between two entities. Via a chain of delegation, the protocol may involve more than two entities that agree on peer-to-peer messages protection. In consequence, messages may be (symmetrically or asymmetrically) encrypted for their recipients and therefore may be securely handled by intermediary nodes. However, SSL fails to guarantee the confidentiality to sensitive computations toward the computing nodes. Indeed, during the execution of such computations, they have to be deciphered (if they were encrypted) before they are read, interpreted and executed on different Grid's nodes. Otherwise the execution of encrypted codes may lead to results from which it is hard to deduce the results of the original codes. The section 3 gives more details about this problem. "Secure outsourcing" methods are the mechanisms that may achieve confidentiality requirements for sensitive computations during their remote execution.

3 Secure outsourcing

Outsourcing is used by an entity that has to execute a task but does not have the appropriate hardware and/or software to realize the execution. Another entity, that has the appropriate resources, will execute the task and provide the results to the task's owner. Secure outsourcing refers to an outsourcing in which security requirements, as integrity, authentication and confidentiality, are involved.

We assume that the nodes involved in a computation are honest and execute the tasks correctly. However, since the tasks and/or the corresponding outputs may refer to private data, the tasks' owners may want to prevent the executing nodes to know the tasks related information. Therefore, secure outsourcing with confidentiality requirement may be such that the nodes involved in the computations never knows neither the task's inputs nor its outputs.

There are, at least, two ways to hide computation details to the agents that participate in this computation: the disguise and the encryption methods. A disguise

operation realizes a functional or mathematical transformation on the objects of the disguise (for example input of sensitive computations). Generally, the execution of the disguised problem leads to results from which it is possible to deduce, knowing some secret information about the disguise, the results of the original problem. The encryption is eventually another kind of disguise method based on the usage of secret keys. However, it seems harder and less efficient to recover the results of an original problem that has been encrypted before execution than when the problem has been disguised. Nevertheless, if disguise methods seem to be more convenient solutions for secure outsourcing, the disadvantage is that it seems that there is no generic disguise method that fits all possible problems. Since Grid systems are dedicated to a broad range of applications, this disadvantage becomes, in this framework, a serious problem.

4 Outsourcing of strings matching

Grid applications are currently mainly related to medicine, nuclear physics, climatology and genetics. Among Grid's applications, some may be sensitive in a secure point of view (e.g. medical and genetic applications). Therefore, before they are outsourced on Grid nodes, they may need to be disguised. In this work, we are interested in the outsourcing operations with confidentiality requirements. On that context, we will focus our interest on the disguise of string matching procedures that allow errors. This problem is also called "approximate string matching problems" [7].

4.1 Related works

Many works have been done in the framework of secure outsourcing [3, 4, 7, 10, 11, 12]. Unfortunately, it seems that no general and generic solution seems to exist. The different proposed secure outsourcing models are appropriate tools resolving specific situations. In [3], the authors propose a model for sequences comparison (in speech recognition, machine vision and molecular sequence comparisons) involving three entities: the client who needs the result of the comparison of two sequences and two other agents that participate in the comparison while ignoring the two sequences. In [4, 11], the problems of outsourcing and speeding up secret computations are evoked. Solutions to mathematical applications (matrix multiplications, quadratures, edge detections, solutions of differential equations, etc.) are discussed. The problems of secure outsourcing and speeding up fixed-based exponentiation and variable-based exponentiation computations are also evoked in [12]. S. Hohenberger et al. in [10] proposes two models of securely outsourcing cryptographic computations: the outsource of a modular exponentiation and the outsource-secure encryption using one trusted program.

The approximate string matching problem deals with the problem of finding all substrings of a text T that match a given pattern with the exception of at most m differences, for some given integer m. The differences being in the form of inserted, deleted or replaced characters [13]. The approximate string matching is the realistic version of the exact pattern matching where we have a database $x = x_1, \ldots, x_n$ and a user who has an item x_i and wants to verify whether its item x_i is in the database. If in addition to this exact pattern matching, the user wants his query to be kept confidential, we deal with the "private information retrieval" [5, 6]. In the approximate matching, the user has an item x_i and wants to find items in a database that are similar to x_i. The notion of distance can also be used, in the sense that the most similar element to a given item is the element which is at the minimum distance (compared to all the other elements in the database) to the item. Many algorithms to solve the problem of approximate string matching are proposed in [13, 14, 15, 16]. These solutions are applicable in matching fingerprint, voice, matching DNA sequence, image template matching, etc.

The private information matching (PIM) problem is an approximate matching problem where the confidentiality of client's queries and of the database content has to be preserved. Solutions to this problem are proposed in [7] with the metrics $\sum_{i=0}^{n}(a_i - b_i)^2$ and $\sum_{i=0}^{n}|a_i - b_i|$ where a and b are the sequences that are being compared. These solutions are also applicable to private information matching with public databases called in [7] "Public Information Matching with a Public Database" (PIMPD). In this work, we propose a specific solution to PIMPD. Considering encryptions and an additional computing server, we extend our PIMPD solution to its PIM version. The two proposed solutions are applied to DNA sequence comparisons.

4.2 DNA sequences comparisons

In this paper, as an example of possible string matching framework, we will consider the problem of private DNA sequence comparisons.

There are two types of nucleic acids: deoxyribonucleic acid (DNA) and ribonucleic acid (RNA). These molecules make it possible to living beings to reproduce their complex equipment from one generation to another. The RNA is used as intermediary in the genetic information flow of the DNA with proteins.

DNA is a polymer. The monomer units of DNA are nucleotides and the polymer is known as a polynucleotide. Each nucleotide consists of a 5-carbon sugar (deoxyribose), a nitrogen containing a base attached to the sugar and a phosphate group. There are four different types of nucleotides found in DNA, differing only in the nitrogen base. The four nucleotides are given one letter abbreviation as shorthand for the four bases: A for Adenine, G for Guanine, C for Cytosine and T for Thymine. A DNA is a normally double stranded macromolecule. Two polynucleotide chains are held together by a weak thermodynamic force. In the DNA helix, we have four different bonds $A-T$, $T-A$, $C-G$ and $G-C$ (by taking into account that one base is on the first polynucleotide chain and the other base is in the second chain). The ith character λ_i of the sequence may be one of the four base bounds $A-T$, $T-A$, $C-G$ and $G-C$.

The searching of specific sequences appears as a fundamental operation for problems such as looking for given features in DNA chains or determining how two genetic sequences are similar. For example a database of over 3 millions of DNA individual profiles has been constituted from 1995 in England and Wales [9]. One of the practical utilities of such database is the elucidation of crimes. Indeed, crimes are successfully solved when DNA is recovered from the crime scene and the DNA profiles are successfully loaded onto the DNA database. Since such databases are greedy in storage spaces, the Grid seems to be the right environment for their management.

We assume on the Grid, the existence of a DNA database that is replicated on different storage servers. The DNA database is public. The DNA's owners are either anonymous or their identities are stored on another secured storage server. The management of such identities is out of the scope of this work. We consider a client who has a DNA sequence $q = (\lambda_1, \ldots, \lambda_n)$ that he has recovered from either a crime scene or from a given living being. He wants to know whether his sequence exists already in the database or whether there is a sequence which is similar to his sequence. However he does not need to reveal neither his sequence nor the result of the comparison. Indeed, since the comparison is done on a remote entity, if the DNA is revealed, a dishonest remote entity may do some tests on one's DNA. Such tests may reveal private information on the DNA owner (as genetic diseases). Before going any further, we define the notion of distance in the context of DNA sequences.

Let consider two DNA sequences $q = (\lambda_1, \ldots, \lambda_n)$ and $q' = (\lambda'_1, \ldots, \lambda'_n)$ over a finite alphabet $\Sigma = A-T, T-A, C-G, G-C$. The distance between the two sequences is the minimum cost of the sequence of operations that transform q in q'. Such operations may be deletion, insertion or substitution of characters [3]. If the different operations have different costs or the cost depends on the characters involved, we speak of *the general edit distance*. Otherwise, if all the operations cost 1, we speak of *simple edit distance* [7]. Without lost of generality, we will consider the simple edit distance between two DNA sequences of the same length and we will deal only with the substitution operation.

4.3 The DNA Private matching model

In this section, we propose a model allowing to outsource DNA sequences comparisons on the Grid. We assume the existence of a DNA public database which is replicated on k replica servers. A client having a DNA sequence $q = (\lambda_1, \ldots, \lambda_n)$ will query the k servers to find in the database the most similar element to his item. This similarity searching is done is such way that neither the client's query nor the output of the comparisons are revealed to the replica servers. We assume that the servers do not collude against the client.

The client disguises his sequence $q = (\lambda_1, \ldots, \lambda_n)$ in the sequence $q' = (\lambda'_1, \ldots, \lambda'_n)$. This disguise operation is done by choosing random characters λ_j in q that we substitute by λ'_j. The elements λ'_j are taken from the DNA alphabet $\Sigma = A-T, T-A, C-G, G-C$. This means that at random positions i in q and q', we will have $\lambda_i = \lambda'_i$ whereas at other positions j we will have $\lambda_j \neq \lambda'_j$. The number k of all positions where $\lambda_j \neq \lambda'_j$ is the simple edit distance between q and q'. We assume that the client interacts with k database replica servers to find the similar sequence to his query. To the first server, the client sends the

disguised sequence $q^{'}$and the distance k. This server chooses in the database the elements $q^{''} = (\lambda_1^{''}, \ldots, \lambda_n^{''})$ such that $d(q^{'}, q^{''}) = min$; where min is the smallest distance between $q^{'}$ and the elements $q^{''}$ of length n in the database. The client can reduce the number of such elements by giving to the server in addition to $q^{'}$ and k, one of the elements λ_j such $\lambda_j \neq \lambda_j^{'}$; therefore, among the elements such that $d(q^{'}, q^{''}) = min$, the server will choose the elements $q^{''}$ such that $\lambda_j = \lambda_j^{''}$. Otherwise, if there is not any element such that $\lambda_j = \lambda_j^{''}$, the server has to retrieve all the elements such that $d(q^{'}, q^{''}) = min$.

The server chooses one of the elements $q^{''}$ and searches in the database the elements $p^{''}$ such that $\lambda_j = \lambda_j^{''}$ and $d(q^{''}, p^{''}) \leq k + min$. Among the elements $p^{''}$ there is the most similar element to q. Indeed, since $d(q, q^{'}) = k$ and $d(q^{'}, q^{''}) = min$ therefore $d(q, q^{''}) \leq k + min$ according to distance properties. The server returns to the client the elements $p^{''}$.

The client proceeds in the same way with the other $k-1$ replica servers. To each replica server, the client reveals a different character λ_j among the k characters where $\lambda_j \neq \lambda_j^{'}$. The client computes the intersection of all the elements $p^{''}$ returned by all the k replica servers. This intersection contains the sequence which is the most similar to sequence q.

4.4 Distances distribution and performance of the system

4.4.1 Assumption on pair bases distribution

In order to make an evaluation of the proposed model and for calculation facilities, we will make an assumption on the base bond distribution along a DNA sequence: we assume that each base bond occurs in a given 4-length subsequence with a probability of $0,25$. Two different 4-length DNA subsequences differ by the position of the four base bonds in each subsequence. This distribution constitutes the simplest and also the worst case toward the quality of disguise. Indeed, we keep in mind that there are other distributions where the occurrence of a given base bond may be random in a DNA subsequence. This might make more random the occurrence of a given subsequence and subsequently improves the quality of a disguise. If our model is secure (in the quality of disguise point of view) in a worst case, we will be ensured that this model will be more secure in other cases. It is important to note that this assumption about the base pairs distribution is incompatible with a certain distance distribution between two DNA subsequences. Indeed, if two 4-length DNA subsequences are at a distance 1 to each other, there is at least one base bond which occurs twice in one of the two subsequences. For example the subsequences $A-T, T-A, C-G, G-C$ and $A-T, T-A, C-G, T-A$ are at distance 1 to each other, but there is the base bond $T-A$ which occurs twice (this is incompatible with the equiprobable pair bases distribution). That's why in order to stay in the context of the assumption on base bonds distribution, we will consider in the rest of this paper that the distance between two 4-length DNA subsequences is > 1 and that $k > 1$.

4.4.2 Equiprobable distance distribution.

Here we assume that the distance between two DNA sequences is distributed equiprobably along the sequences. We are going to see that this distribution strengthens the security of our model while causing loss in performance. Indeed, in this section we will show that the quality of the disguise is good when the distance is equiprobably distributed. The disadvantage of this distribution is that the number of replica servers k is directly dependent of n (the DNA size). Since the DNA size is generally great, the number of servers becomes rapidly prohibitory.

Firstly, we show that the revelation of the distance k in the model we proposed does not affect the security of the disguised sequence q. Indeed, if two n-length sequences q and $q^{'}$ are distant of k, therefore two corresponding 4-length subsequences from q and $q^{'}$ are distant of $\frac{4k}{n}$ since the distance is distributed equiprobably. By corresponding subsequences, we mean subsequences that are at the same position in respective DNA sequences. Thus, given a 4-length subsequence of $q^{'}$, the number of corresponding 4-length subsequences from q that are distant of $\frac{4k}{n}$ is $\binom{4}{\frac{4k}{n}}$ with n dividing $4k$ and $\frac{4k}{n} > 1$.

Since in a n-length DNA sequence, there are $\frac{n}{4}$ 4-length DNA subsequences and each of such subsequences has $\binom{4}{\frac{4k}{n}}$ corresponding 4-length DNA sequences that are at $\frac{4k}{n}$ of distance; therefore, given a n-length DNA sequence $q^{'}$ (as the one sent to storage servers in the model), there are $S(n,k) = \binom{n}{4}\binom{4}{\frac{4k}{n}} = \frac{n^2 3!}{4k(\frac{4k}{n}-1)!(4-\frac{4k}{n})!}$ 4-length DNA subsequences

from which we can construct n-length different sequences q that are at a distance k to the given n-length sequence $q^{'}$. We can precisely construct $\binom{S(n,k)}{0,25n}$ n-length DNA sequences q that are at a distance k from $q^{'}$. This means that user's query q is disguised among $\binom{S(n,k)}{0,25n}$ elements that are at a distance k.

In the model, when the server chooses in the database the elements $q^{''} = (\lambda_1^{''}, \ldots, \lambda_n^{''})$ such that $d(q^{'}, q^{''}) = min$. The number of such elements is $\binom{S(n,min)}{0,25n}$. We can have an idea of how the number of such elements is reduced when the client reveals one of the elements λ_j such that $\lambda_j \neq \lambda_j^{'}$ and that the server chooses the elements $q^{''}$ such that $\lambda_j = \lambda_j^{''}$. Indeed, $\lambda_j^{''} \in$ in a 4-length subsequence from $q^{''}$ which is at $\frac{4min}{n}$ to the corresponding 4-length subsequence from $q^{'}$.

Normally there are $\binom{4}{\frac{4min}{n}}$ such elements. However if $\lambda_j^{''}$ is fixed, this number is reduced to $\binom{3}{\frac{4min}{n}}$. This means that there are $\binom{3}{(\frac{4min}{n})-1}$ elements that are not considered (since $\binom{4}{\frac{4min}{n}}$=$\binom{3}{\frac{4min}{n}}$+$\binom{3}{(\frac{4min}{n})-1}$).

The number of elements $p^{''}$ such $d(q^{''}, p^{''}) \leq k + min$ is $\sum_{i=2}^{k+min} \binom{S(n,i)}{0,25n}$ when there is not any element $\lambda_j = \lambda_j^{''}$; otherwise, this number is reduced since at each iteration, there are $\binom{3}{(\frac{4i}{n})-1}$ 4-length subsequences that are not considered.

After we have analyzed the quality of the disguise, we are going in this paragraph to analyze the performance of the model. Given two different 4-length DNA subsequences, the distance between the two DNA subsequences varies from 2 to 4 (according to what has been stated in 4.4.1). When we consider our model, this means that $\frac{4k}{n}$ varies from 2 to 4 and k varies from $0,5n$ to n. For example with $n = 200$ and k distributed with a ration 2 in each 4-length DNA subsequence, we will have $k = 100$. In the model, k is also the number of servers with which the user interacts. This number of servers depends on the DNA sequence size. Such amount of servers is prohibitory. We are going to see in subsection 4.4.3 that there is a way to distribute the distance along the sequences which reduces this amount of servers.

4.4.3 Distance distributed randomly along the DNA sequences

In this subsection, we are going to consider the case where the distance k is distributed randomly in the n-length DNA sequences q and $q^{'}$. By random distribution, we mean that in despite of distributing the distance k in all 4-length DNA subsequences, the client can distribute the distance k in random chosen 4-length subsequences. For example, the client can distribute the distance k in $\frac{k}{2}$ random 4-length subsequences with the ratio 2. In this case, the distance k (and implicitly the number of servers) does not depend on the DNA size n.

We assume that the distance k is distributed in random $\frac{k}{2}$ 4-length DNA subsequences, each 4-length DNA sequence from q (where the distance k is distributed) being at a distance 2 to the corresponding 4-length DNA subsequence from $q^{'}$. We will analyze how the quality of the disguise and the performance of the model are affected by this distribution.

Given two corresponding 4-length DNA subsequences distant of 2 each other, this means that they differ by one of the positions of two base bonds; for example the two subsequences $A-T, C-G, T-A, G-C$ and $A-T, C-G, G-C, T-A$. In our model, given a 4-length DNA subsequence from $q^{'}$ where the distance is distributed, there are $\binom{4}{2}$=6 4-length subsequences from q that are at a distance 2. Since the distance k is distributed in $\frac{k}{2}$ 4-length DNA subsequences, we have $\frac{k}{2} * \binom{4}{2} = 3 * k$ such 4-length subsequences from which we can construct the sequences q that are at a distance k to the sequence $q^{'}$. Precisely we can construct $\binom{3*k}{\frac{k}{2}}$ n-length DNA sequences q that are at a distance k to the DNA sequence $q^{'}$.

We note that this amount of n-length DNA sequences q that are at a distance k to the DNA sequence $q^{'}$ with a random distribution of the distance k is small in comparison of the amount of the n-length DNA sequences q we can construct with an equiprobable distribution of the same distance ($\binom{S(n,k)}{0,25n}$). This allow us to state that the quality of the disguise is better with an equiprobable distribution of the distance than with a random distribution. Nevertheless, we estimate that the quality of the disguise with random distance distribution still is efficient to hide the client's query q. For example with the distance $k = 6$ distributed with the ratio 2 in random 4-length subsequence, there are $\binom{18}{3}$=816 n-length DNA sequences q that are at a distance k from the n-length sequence $q^{'}$.

This loss in quality of disguise is compensated by a profit in performance especially in the number of servers which is reduced with a random distance distribution. Indeed, with the random distribution of the distance, the number of servers k is equal to the dis-

tance between the two n sequences. The client still is able to adapt the distance k to the number of servers which is available. This is not possible with the distance k which is equiprobably distributed along the DNA sequence since the distance k depends on the DNA size as it is explained in section 4.4.2. With a large DNA sequence, the number of servers becomes prohibitory.

Considering together the quality of disguise and the performance of the model, we see that it is advantageous to the client to distribute randomly the distance k when he is disguising his query.

5 Outsourcing of intersections computation

Previously, we assumed that each storage server sends to the client a certain amount of DNA sequences. This suppose that the client host has sufficient storage and computing resources to evaluate the intersection of all sequences sent by the k servers. However this is not always the case and we propose, in consequence, a model in which the client's computations are executed by a untrusted server, named s_{k+1}.

The communication from the client to the k servers remains as in the proposed model (the PIMPD model), this means that the client sends to each storage server the sequence $q^{'}$, the distance k and the element λ_j. Each server computes the elements $q^{''}$ such that $d(q^{'}, q^{''}) = min$ and all $p^{''}$ such that $d(q^{''}, p^{''}) \leq k + min$.

Instead of sending the different $p^{''}$ to the client in clear, they are asymmetrically encrypted with the client public key and sent to the server s_{k+1}: $s_i \longrightarrow s_{k+1} : E_{K_C}(p^{''})$ for all $i \in [1, k]$, where K_C is the client's public key associated to a private key $K_C^{'}$ securely saved by the client and s_i is one of the k storage servers.

The server s_{k+1} computes the intersection between all the encrypted sequences sent by the storage servers and sent it to the client. Since the client has the private key $K_C^{'}$, he can decipher the encrypted sequences that appear in the intersection and retrieve the most similar element to his item.

Here we make the assumption that the used asymmetric encryption scheme has to be such that a given item, whatever its position in the plaintext, when encrypted with a same encryption key, leads to an identical encrypted item. The classical RSA encryption scheme, for example, meets our assumption. Moreover, the size of an encrypted block has to be the same as the size of each DNA sequence.

The server s_{k+1} is said to be untrusted because we require, for confidentiality reasons, that it cannot access to the client's item q, neither to the database, nor to the output of the comparison.

The server s_{k+1} does not have access to the client's item q since this latter is disguised in the item $q^{'}$.

However, s_{k+1} is able to reach the public databases considered previously in this paper. Therefore, s_{k+1} is theoretically able to encrypt each sequence in the databases with the client public key, and to compare the encrypted elements of the computed intersection with the encrypted elements of the databases. Consequently, s_{k+1} may learn the values in clear of the elements of the intersection.

To avoid to reveal in this way the result of the comparisons, the database servers have to implement an access control system (to prevent s_{k+1} from accessing to the content of the databases). In other words, the database has to be private. As consequence, we are now considering a private information matching with a private database (PIM), rather than a private information matching with a public database (PIMPD).

We assume also that the server s_{k+1} does not collude with the storage servers that host the databases. Otherwise, since each storage server can access to the DNA sequences in clear, s_{k+1} would also be able to easily know the values in clear of the intersection.

It is important to note that this PIM solution can be used in order to solve a problem of approximate matching with a public database in case of DNA sequences comparisons. This meets what has been stated in [7] that PIM solutions can also be used on behalf of a PIMPD solution.

However we can see that a specific PIMPD solution (as the one proposed in subsection 4.3) is more efficient (no bulk encryption) and less complex than a PIM solution when we deal with public databases. Of course, the advantage of the PIM solution is that the client is released of his computations which are executed by the $k + 1$th server.

6 Conclusion

We investigated the problem of private information matching with a public database (PIMPD) in a Grid environment. We proposed a PIMPD solution in the case of DNA sequences comparisons on Grid's replicated databases. We analyzed the performance and the

security of the model in taking into account the distribution of the distance between clients's queries and their corresponding disguises. We compared an equiprobable distance distribution to a random distance distribution and we showed that the random distance distribution is more efficient than the equiprobable distribution. For clients's facilities, we have also proposed an extension of the PIMPD solution to its corresponding PIM solution.

References

[1] S. Naqvi, M. Riguidel *Threat model for Grid Security Services* Lecture Notes in Computer Science, Springer, Vol 3470, Advances in Grid Computing - EGC 2005, pp.1048-1055.

[2] J. Chapin, *A new Model of security for Metasystems* Future Generation Computer Systems, Vol. 15, No. 5, pp.713-722, 1999.

[3] M.J.Atallah, J. Jiangtao *Secure Outsourcing of Sequence Comparisons* International Journal of Information Security, Vol 4, pp.277-287, 2005.

[4] M.J. Atallah, J.R. Rice *Secure Outsourcing of Scientific Computations* Technical Report 98-15, Purdue University.

[5] B. Chor, N. Gilboa *Computationally Private Information Retrieval* Proceedings of the ACM symposium on Theory of computing, pp.304-313, 1997.

[6] B. Chor, O. Goldreich, E. Kushilevitz, M. Sudan *Private Information Retrieval* In proceedings of the IEEE Symposium on foundations of Computer science, 1995.

[7] W. Du, M. Atallah. *Protocols for Secure Remote Database Access with Approximate Matching* 7th ACM Conference of Computer and Communications Security, 25 pages, 2000.

[8] G. Navarro *A Guided Tour to Approximate String Matching* ACM Computing Surveys, 33(1):pp.31-88, 2001.

[9] The national DNA database Parliamentary Office of Science and Technology, Postnote, Number 258, 2006.

[10] S. Hohenberger, A. Lysyanskaya *How To Securely Outsource Cryptographic Computations* Theory of Cryptography Conference, Lecture Notes in Computer Science, Springer, Vol 3378, pp.264-282, 2005.

[11] T. Matsumoto, K. Kato, H. Imai *Speeding Up Secret Computations with Insecure Auxiliary Devices* Proceedings of Crypto, Lecture Notes in Computer Science, Springer, Vol 403, pp.497-506, 1988.

[12] M. Dijk, D. Clarke, B. Gassend G. Suh, S. Devadas *Speeding up Exponentiation using an Untrusted Computational Resource* Designs, Codes and Cryptography, Vol 39, Issue 2, pp.253-273, 2006.

[13] S. Sahinalp, U. Vishkin *Efficient Approximate and Dynamic Matching of Patterns Using a Labeling Paradigm* IEEE Symposium on Foundations of Computer Science, pp. 320-328, 1996.

[14] G. Landau, U. Vishkin *Efficient String Matching in the presence of errors* IEEE Symposium on Foundations of Computer Science, pp.126-136, 1985.

[15] G. Landau, U. Vishkin *Fast String Matching with k differences* Journal of Computer and System Sciences, 37(1), pp.63-78, 1988.

[16] G. Landau, U. Vishkin *Fast Parallel and Serial Approximate String Matching* Journal of Algorithms, 10(2), pp.157-169, 1989.