

A Practical Three-dimensional Privacy-preserving Approximate Convex Hulls Protocol

Dong Li, Liusheng Huang, Wei Yang, Youwen Zhu, Yonglong Luo, Zhili Chen, Lingjun Li, Yun ye

(National High Performance Computing Center at Hefei, Department of Computer Science and Technology, University of Science and Technology of China, Hefei, 230027, P. R. China)

(E-mail: lixido@mail.ustc.edu.cn {[lshuang](mailto:lshuang@ustc.edu.cn), [qubit](mailto:qubit@ustc.edu.cn)}@ustc.edu.cn zhuyw@mail.ustc.edu.cn, ylluo@ustc.edu.cn {[zlchen3](mailto:zlchen3@mail.ustc.edu.cn), [ljli](mailto:ljli@mail.ustc.edu.cn), [yeyunyy](mailto:yeyunyy@mail.ustc.edu.cn)}@mail.ustc.edu.cn)

Abstract

Convex Hulls Problem is a special case of Privacy-preserving Geometry Problem in the inquiry of Secure Multi-Party Computation. In the past, only in two-dimensional space privacy-preserving convex hulls have been investigated, and there is little focus in the three-dimensional space. However, three-dimensional privacy-preserving convex hulls can be applied in many fields, such as researching and exploration of the space, military, corporately finding the union range based on sensitive data from two parties. Approximate convex hulls have more advantages than conventional convex hulls in the theme of Secure Multi-Party Computation because it can hide the private points on the vertices. In this paper, we first present a practical privacy-preserving protocol to solve the three dimensional approximate convex hulls problem; we also discuss the correctness, security, and performance of our protocol.

1. Introduction

Secure Multi-party Computation (SMC) [1-2], designed to solve the problem that two or more parties want to jointly perform a computation, can provide useful solutions in such scenarios: each party needs to contribute its private inputs for the computation, but no party wants to disclose its private inputs to the other parties. SMC was introduced by A. C. Yao [1] in 1982. There are lots of theoretic results of SMC [3-4]. O. Goldreich pointed out that using the solutions derived by these theoretic results for special cases of SMC may be impractical [2]. As a result, special solutions should be developed for efficiency reasons in various applications.

Privacy-preserving Convex Hulls problem is an impor-

tant sub-issue of Privacy-Preserving Geometric Computation (PPGC) [3-4], in the inquiry of SMC. Convex hulls in three-dimensional space can be applied in many fields such as research and exploration of the space by different parties. For example, A has several detectors in the space, and B also has some detectors in the space. They can generate the privacy-preserving three-dimensional convex hull that will help them construct a detector-network if they want to cooperate in the space without disclose their stations' accurate location. Another example involves two parties finding the union range of two sensitive data sets. We assume that A and B are both conducting an expensive experiment. The result of the experiment having three properties, for example, length, weight and size, can be represented by three-dimensional values. By constructing the three-dimensional convex hull based on their result, they can get the union of the result space while keep their private result secret.

Atallah *et al.* [3] first proposed the Convex Hulls problem in two dimensions and Qi Wang [5] gives two solutions for two dimensional convex hulls. However, recently there is no privacy-preserving three dimensional convex hulls solution. In the following paper, we call the convex hulls defined in [3] as original convex hulls. The concept of original convex hulls can also extend to three-dimensional space.

There are a lot of algorithms proposed to construct convex hulls [6-7]. And the lower computational complexity bound of constructing a convex hull is $O(N \log N)$, where N is the number of points [7]. However, the original convex hulls they constructed in [6-7] will inevitably disclose the private points on the vertices of the convex hull. Figure 1 is an example of original convex hull. We can find that all the points on the vertices are exposed. Approximate convex hull has more advantage in privacy preserving computation because it can hide the private points exposed on the vertices of the original convex hull, also in most of the cases it can be constructed more efficiently. Jon Louis Bentley and Mark G. Faust [8] proposed an approximate solution for convex hulls in two and higher dimensions. The computational complexity is $O(n+k^2 \lg k)$, however, the approximate

This work was supported by the National Natural Science Foundation of China (No. 60773032 & No. 60703071), the Ph.D. Program Foundation of Ministry of Education of China (No. 20060358014), the Natural Science Foundation of Jiangsu Province of China (No. BK2007060), and the Anhui Provincial Natural Science Foundation (No. 070412043).

convex hull they constructed sometimes cannot contain all the points.

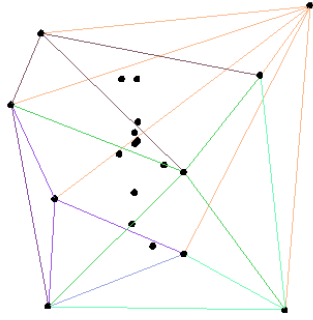


Figure 1. An example of original convex hull in three dimensions.

In this paper, we develop a new algorithm that can construct a privacy-preserving approximate convex hull in three-dimensional Euclidean space. It contains all the private points and more secure than original convex hulls. The result of our experiment shows that it is especially suitable to deal with a large quantity of points. The worst computational cost is $O(n^2)$, where n is the number of points. We also illustrate another two derivative protocols: one is more efficient and the other can construct the convex hull that has higher similarity measurement.

The paper is organized as follows. Section 2 discusses the preliminaries and basic definitions used in this paper. Section 3 introduces the privacy-preserving three-dimensional convex hull protocol we developed. Then section 4 discusses the protocol's efficiency and security by the result of our experiment. Finally section 5 concludes the paper.

2. Preliminaries and definitions

We introduce some preliminaries and definitions in this section.

2.1 Secure Two-party Computation

In this paper, only two parties, called Alice and Bob, conduct the computation. They are semi-honest parties [2]. It is a special case of SMC. Alice inputs her private data I_A and Bob inputs I_B respectively. Let O_A, O_B represent their outputs respectively. Then they jointly compute function f :

$$(O_A, O_B) = f(I_A, I_B)$$

Any protocol that satisfies the following two conditions is a secure protocol [9]:

- a) There exists infinite pairs (I'_A, O'_A) satisfies:

$$(O'_A, O_B) = f(I'_A, I_B)$$

- b) There exists infinite pairs (I'_B, O'_B) satisfies:

$$(O_A, O'_B) = f(I_A, I'_B)$$

2.2 Secure comparison protocols

We use this protocol to compare two private numbers. The secure comparison problem is first introduced by Yao [1], and referred to as Yao's Millionaire Problem: two parties want to determine who is richer without disclosing anything else about their wealth. Cachin proposed a solution [10] using an un-trusted third party. The communication complexity of Cachin's scheme is $O(l)$, in which l is the number of bits of each input number. M. Fischlin proposed a non-interactive crypto-computing protocol [11], the client perform $2n$ modular multiplications and the server needs $6n\lambda$ multiplications, where n is the size of input and λ is the error parameter. Ivan Damgard et al. proposed a more effective solution based on additive secret sharing and homomorphic encryption [12].

We use $SC - Max(\theta_1, \theta_2)$ to denote that we get the larger one from θ_1 and θ_2 without disclosing the smaller one by secure comparison protocol.

We use $SC - Min(\theta_1, \theta_2)$ to denote that we get the smaller one from θ_1 and θ_2 without disclosing the larger one by secure comparison protocol.

2.3 Three-dimensional Privacy-preserving Approximate Convex Hull

We first define the three-dimensional privacy-preserving approximate convex hull problem and then give a standard to evaluate the approximate convex hulls.

Definition 1: Three-dimensional Privacy-preserving Approximate Convex Hulls Problem:

Alice has M points in the three-dimensional space; Bob has N points in the space. They want to jointly find a convex polyhedron that contains all these $M+N$ points; however, neither Alice nor Bob wants to disclose any more information to the other party than what could be derived from the result.

Because there are many polyhedrons that satisfy the definition above, it is necessary for us to define a standard to measure the similarity of an approximate convex hull and the original convex hull. We denote the volume of the convex hull constructed by Alice and Bob's private points as V_1 and the volume of the approximate convex hull as V_2 . The similarity of the two convex hulls can be evaluated by the following definition.

Definition 2: similarity measurement of approximate convex hulls:

$$s = \sqrt[3]{V_1/V_2}$$

Obviously $0 \leq s \leq 1$. When s closes to 1, the approximate convex hull is more accurate, but some private points might be disclosed. When $s = 1$, it is the original convex hull. When s closes to 0, there is less probability the private points disclosed, however, more spaces that are not useful included. In the following, we can use this definition to evaluate the experiment result about the protocol.

3. Three-dimensional Privacy-preserving Approximate Convex Hulls Protocol (TPACHP)

We first give some notations that will be used in our protocol, and then we present a simple protocol that can construct the computational environment for the two parties. At the end we describe the detailed steps of our protocol.

3.1 Notations

We first define some notations, which will be used later in our protocol.

- (1) Alice's private point set:

$$A = \{point\ a_i \mid i = 1..m\}$$

Bob's private point set:

$$B = \{point\ b_i \mid i = 1..n\};$$

Reference Point: In our protocol, we use reference point as auxiliary points to approximate Alice and Bob's convex hull. Here we only use two reference points R_1 and R_2 ;

Reference Line: In the following, we call the line through a reference point as reference line.

- (2) *Plane* can be denoted as three points that are not lies on the same line. The positive side of a plane is defined by the arrangement of the three points with the right-hand spiral law.
- (3) *Alice || Bob:* Alice and Bob respectively conduct step i on their private data;
- (4) *Alice ∨ Bob:* Alice and Bob conduct computation on the published data.
- (5) *Alice ∧ Bob:* Alice and Bob cooperate to implement a protocol or function using their private data as input.
- (6) *GENERATE:* construct a data object.
- (7) $\angle(\vec{v}_1, \vec{v}_2)$ denotes the angle between vectors \vec{v}_1 and \vec{v}_2 . $\angle(\vec{v}, Plane)$ denotes the angle between vector \vec{v} and the positive normal vector of *Plane*.

- (8) *List* is a data structure that all the data pushed into a List by the method *push_back* are linked in a sequence. Every data in List has the property *next* to identify the next date in the list.
- (9) Function *Intersect* test whether a ray and a plane intersect on one point. If not intersect, return false, else return the intersection point.

3.2 Constructing the Computing Environment

We use a simple and approximate method to construct the computing environment and get the two reference points. We assume their points are recorded by three coordinate values. All the computations on these points are conducted on the three values. Alice and Bob find a sphere whose center is O and radius is r . It contains all their private points. Then they select point O as the origin point of the new Cartesian coordinate system. The following computations are based on the coordinate system. The Construct Computing Environment Protocol (CCEP) can be described as follows:

Input: Alice's private points set A , Bob's private points set B .

Output: The approximate minimum circumscribed sphere, Reference Point R_1, R_2 .

Protocol CCEP:

- a) *Alice || Bob:* Alice use her private point to construct a new point A_0 :

$$A_0 = \frac{1}{m} \sum_{i=1}^m a_i$$

In the same way, Bob construct a new point B_0 .

- b) *Alice ∨ Bob :* They public A_0 and B_0 , get O :

$$O = \frac{1}{2}(A_0 + B_0)$$

Set point O as the origin of the new coordinate system. All the points are computed in the new system.

- c) *Alice || Bob:* Alice find:

$$d_A = \max\{|a_i O|, i = 1..m\}$$

Bob find d_B respectively. Then they jointly find $r = SC - \text{Max}(d_A, d_B)$. Then their points are all in sphere C whose center is O and radius is r .

- d) Alice and Bob agreed on a distance R that satisfies $R > r$. Then they select two reference point $R_1(R, 0, 0)$ and $R_2(-R, 0, 0)$.

Actually, in step d), we can choose any R_1 and R_2 that sa-

tisfy $|R_1| = |R_2| = R$ and $O = \frac{1}{2}(R_1 + R_2)$.

3.3 Three-dimensional Privacy-preserving Approximate Convex Hulls Protocol (TPACHP)

The main idea of our protocol is very intuitive. It can be described as follows. Alice and Bob use protocol CCEP introduced in section 3.2 to find two reference points: R_1 and R_2 . Then for reference point R_1 , Alice and Bob corporately use R_1 as the peak of a cone-shape object that can compactly wrap all their private points. Figure 2 illustrates the cone-shape object constructed by R_1 . They construct another one by R_2 . Then the intersection of the two objects is an approximate convex hull. We can find the vertices of the approximate convex hull by the following protocol.

Input: Alice's private points set A , Bob's private points set B .

Output: Point set S contains all the vertices of the approximate convex hull.

Protocol TPACHP:

a) Alice and Bob use the protocol in section 3.2. They get two reference points R_1 and R_2 .

b) For R_1 : GENERATE point list $LR_1 = \emptyset$

Alice||Bob:

Alice selects point P_0^A :

$$\begin{aligned}\theta_0^A &= \angle(\overline{R_1 P_0^A}, \overline{R_1 O}) \\ &= \max\{\angle(\overline{R_1 a_i}, \overline{R_1 O}) \mid i = 1 \dots m\}\end{aligned}$$

Bob selects point P_0^B :

$$\begin{aligned}\theta_0^B &= \angle(\overline{R_1 P_0^B}, \overline{R_1 O}) \\ &= \max\{\angle(\overline{R_1 b_i}, \overline{R_1 O}) \mid i = 1 \dots n\}\end{aligned}$$

Alice \wedge Bob:

$$\theta_0 = SC - \text{Max}(\theta_0^A, \theta_0^B)$$

$$\text{if } \theta_0 = \theta_0^A \text{ then } P_0 = P_0^A$$

$$\text{else } P_0 = P_0^B.$$

$LR_1.push_back(P_0)$

c) Alice \vee Bob: GENERATE Plane $Plane_0$ through line $P_1 R_1$ and vertical with Plane $P_1 O R_1$.

Alice||Bob:

Alice select point P_1^A :

$$\begin{aligned}\theta_1^A &= \angle(R_1 P_0 P_1^A, Plane_0) \\ &= \min\{\angle(R_1 P_0 a_i, Plane_0) \mid i = 1 \dots m\}\end{aligned}$$

Bob select point P_1^B :

$$\begin{aligned}\theta_1^B &= \angle(R_1 P_0 P_1^B, Plane_0) \\ &= \min\{\angle(R_1 P_0 b_i, Plane_0) \mid i = 1 \dots n\}\end{aligned}$$

Alice \wedge Bob: $\theta_1 = SC - \text{Min}(\theta_1^A, \theta_1^B)$

if $\theta_1 = \theta_1^A$ then $P_1 = P_1^A$ else $P_1 = P_1^B$.

d) Alice \wedge Bob: GENERATE Point P, Q ;

set $P = P_0, Q = P_1$.

GENERATE Plane $Plane_1 = R_1 P Q$;

do{

$LR_1.push_back(Q)$

set $P = Q$; // $R_1 Q$ is a reference line.

Alice||Bob:

Alice selects P_k^A :

$$\begin{aligned}\theta_k^A &= \angle(R_1 A_k Q, Plane_1) \\ &= \min\{\angle(R_1 a_i Q, Plane_1) \mid i = 1 \dots m\}\end{aligned}$$

Bob selects P_k^B :

$$\begin{aligned}\theta_k^B &= \angle(R_1 B_k Q, Plane_1) \\ &= \min\{\angle(R_1 b_i Q, Plane_1) \mid i = 1 \dots n\}\end{aligned}$$

Alice \wedge Bob:

$$\theta_k = SC - \text{Min}(\theta_k^A, \theta_k^B)$$

$$\text{if } \theta_k = \theta_k^A \text{ then } Q = P_k^A$$

$$\text{else } Q = P_k^B$$

$Plane_1 = R_1 Q P$

$k = k + 1$

} while ($Q \neq P_0$);

e) By step b), c) and d), Alice and Bob can construct a cone-shaped object. It can be illustrated in figure 1. Alice and Bob conduct step b), c) and d) for reference point R_2 . Get another point list LR_2 .

f) Alice \vee Bob:

GENERATE point set $S = \{R_1, R_2\}$.

For reference point R_1 :

For each point P in LR_1 , construct ray $\overline{R_1 P}$;

For each point Q in LR_2 , construct plane QR_2Q_{next} ;

if $\overline{R_1P}$ and QR_2Q_{next} intersect, compute the intersection point p

$$S = S \cup \{p\}$$

Conduct the same computation for reference point R_2 ;

Point set S is the point set of the approximate convex hull.

End of TPACHP

We can see the result more directly from Figure 3, which is the result of our protocol when there are 100 points and $R=1.5r$. According to the basic geometric knowledge, the correctness of TPACHP is self-evident. Here we only analyze the security of our protocol.

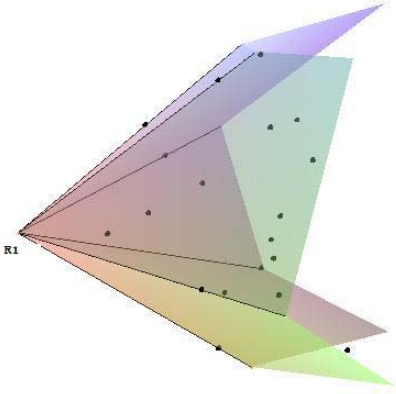


Figure 2. A cone-shape object constructed by R_1

Theorem 1. (Security) Neither Alice nor Bob will disclose any private points.

Proof: In our protocol, we only used Yao's Millionaire Protocol for the secure comparison of Alice's and Bob's Distances and angles. The public data are derived from the result of the computation. Based on the security of Yao's Millionaire Protocol, neither Alice nor Bob can get any extra information about the private points.

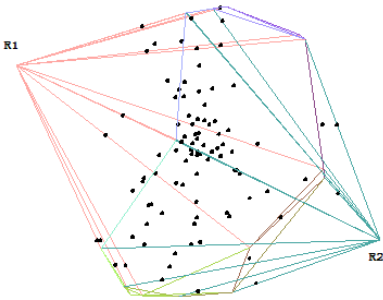


Figure 3. An example of TPACHP's result

4. Evaluations

We will analyze the efficiency and then discuss the similarity of the convex hull constructed by our protocol. We also will propose two protocols derived from TPACHP in each section.

4.1 Efficiency Evaluation

In this section, we will analyze the computational complexity of our protocol, and assess it by the result of our experiment. Then we will introduce a simplified version of our protocol which is more efficient than TPACHP.

Table 1 The relations of n , R/r and V

$n \backslash t^*$	1.25	1.50	1.75	2.00	2.25
10	12	12	12	12	12
20	14	14	14	14	14
40	16	16	17	17	17
60	17	18	18	19	19
80	18	19	20	20	20
100	18	20	20	21	22
200	20	23	23	25	25
400	23	26	27	28	29
600	24	27	30	31	32
800	25	29	31	33	34
1000	27	30	33	35	36

* $t = R/r$, $r = 100$

The worst case analyze: We analyze the efficiency of our protocol by n and V , the total number of points and the number of vertices of the approximate convex hull. In TPACHP, we use secure comparison $V-1$ times. For Bob and Alice, in the worst case, the total local computational cost is the sum of all the steps: step a) is $O(n)$, step b) and c) both $O(n)$. The cost of step d) and e) is $O(n \cdot (V-3))$, step f) will cost $O(V^2)$. The total computational cost is $O(nV + V^2)$. In the worst case $V=n$, so the worst case computational cost is $O(n^2)$. However, on average, the result is much better, table 1 is the average result of our protocol. It describes the relations of n , R/r and V .

If we can classify all the points, then we can easily exclude all the points that impossible to construct the minimum or maximum angle and it will be much easier to construct the approximate convex hull.

In our protocol, Yao's Millionaire Protocol is used when we compare distances or angles from Alice and Bob. In TPACHP, we use this protocol $V-1$ times. This is the main computational cost of our protocol. In step a), it is used to

compare two distances from one point. However, if we just public two values, the other party can only know that there is a point on the surface of a sphere. In step b), c) and d), we use this protocol to find the maximum or minimum angle of two vectors or planes. However, if Alice and Bob public their angles, the other side can only know that there is a point on a plane. Every point in three-dimensional Euclidean space can only be identified by three coordinate values. Without Yao's Millionaire protocol, only one will be disclosed. It is still impractical for the other party to identify the point. Without the secure comparison protocols, our scheme will be more efficient with very little compromise to the security. It will be more practical when there are a large amount of points to compute, or the data is not very sensitive.

4.2 Similarity Evaluation

There are two main factors that affect the result of our protocol: the number of Alice and Bob's points and the distance of reference points R . We use the experiment result to evaluate the similarity and efficiency of the approximate convex hull. In our experiment, we randomly generate n points in a sphere and test the volume and vertices of the approximate convex hulls we constructed with different reference points.

Table 2 is the average similarity measurement of one hundred times test. If Alice and Bob have 800 points and want to generate an approximate convex hull whose similarity measurement is about 0.9, then they can choose $R=1.75r$ according the result of table 2. Also, we can conclude in table 2 that it is easier to achieve high similarity measurement when Alice and Bob have a large quantity of points.

Table 2 The similarity measurement of TPACHP

$n \backslash t^*$	1.25	1.50	1.75	2.00	2.25
10	0.683	0.665	0.633	0.604	0.565
20	0.792	0.760	0.722	0.697	0.650
40	0.843	0.815	0.795	0.768	0.719
60	0.867	0.852	0.823	0.797	0.751
80	0.875	0.861	0.837	0.811	0.789
100	0.878	0.867	0.846	0.828	0.805
200	0.886	0.891	0.873	0.853	0.829
400	0.882	0.898	0.891	0.872	0.851
600	0.878	0.903	0.895	0.879	0.859
800	0.876	0.905	0.900	0.884	0.863
1000	0.872	0.906	0.902	0.886	0.867

* $t=R/r, r=100$

To raise the similarity measurement, we can cut away the two reference points. In the protocol, the two reference points are selected as the vertices of the approximate convex hull, however, both parties know that these two points are redundant points. To achieve higher similarity measurement, we can cut the two redundancy cones easily. Alice and Bob just jointly find point p nearest to R_1 , and then construct a plane on point p and vertical with array $\overline{pR_1}$. They compute the intersection points of the plane and the reference lines passing through R_1 . Delete R_1 from point S and add these intersection points to S . Then conduct the same computation to R_2 . Figure 4 is the compares the similarity measurement when $R/r = 1.5$.

From figure 4 we can find the similarity measurement increased. There is more similarity increase when n is small while less when n is large. The computational complexity just increased by $O(V)$. Weather cutting away the reference points depends on the practical application's requirement.

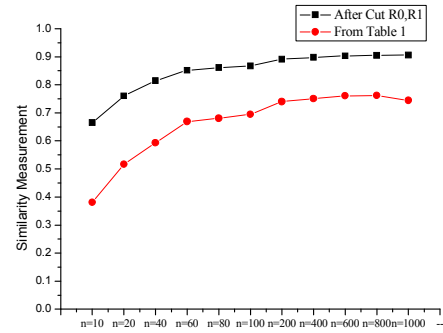


Figure 4. Similarity measurement comparison when $R/r=1.5$

5. Conclusion and future work

Three-dimensional Convex Hulls problem plays a significant role in the field of secure multi-party computational geometry. In this paper, we have proposed a practical protocol that can construct approximate convex hulls in three dimensions and can conceal more private information than the original convex hulls. We also introduced another two changes of our protocol.

Though it is very suitable for SMC, there are still some spaces needed to be investigated. In some cases, the values of different dimensions are not comparable, we need to add weight for each dimensional value and adjust it for practical applications. Another problem is that the approximate convex hulls in higher dimensions will be more complex. And In our future work, we will strive to design better ways to solve these problems.

6. References

tions” Volume 4586 , Springer Berlin / Heidelberg
ISSN 0302-9743 (Print) 1611-3349 (Online)

- [1] A. C. Yao. “Protocols for secure computations”. In: *Proc. 23rd Annual IEEE Symposium on Foundations of Computer Science*. Los Alamitos: IEEE Computer Society Press, 1982. 160-164.
- [2] O. Goldreich. “Secure multi-Party Computation”, 1998.
- [3] M. J. Atallah, W. L. Du, Secure multi-party computational geometry, *Lecture Notes in Computer Science*, Vol. 2125, Springer Verlag. *Proc. of 7th International Workshop on Algorithms and Data Structures (WADS)*, pp.165-179, 2001.
- [4] W. L. Du, M. J. Atallah, “Secure multi-party computation problems and their applications: A review and open problems”, *Proc. of New Security Paradigms Workshop*, Cloudcroft, New Mexico, USA, pp.11-20, 2001.
- [5] WANG Qi, LUO Yong-Long, HUANG Liu-Sheng, “Privacy-preserving Protocol for Finding the Convex Hulls”, *The Third International Conference on Availability, Reliability and Security*, 2008.
- [6] B. Barber, D. Dobkin, and H. Huhdanpaa. The quick-hull algorithm for convex hull. Technical Report GCG53, The Geometry Center, MN, 1993.
- [7] Preparata, F.P. and Hong, S.J. Convex Hulls of finite sets in two and three dimensions. *Comm. ACM* 20, 2, (Feb, 1977),87-93.
- [8] Jon Louis Bentley, Franco P. Preparata, Mark G. Faust, “Approximation algorithms for convex hulls”, *Communications of the ACM*, v.25 n.1, p.64-68, Jan. 1982.
- [9] LUO yong-Long, HUANG Liu-Sheng, *et al.* “Privacy-preserving Distance Measurement and Its Application”, *Chinese Journal of Computers*, Vol.30. No.2. Feb.2007
- [10] C. Cachin, “Efficient private bidding and auctions with an oblivious third party”, *Proc. of the 6th ACM conference on Computer and Communications Security*, Singapore, pp.120-127, 1999.
- [11] M. Fischlin. “A cost-effective pay-per-multiplication comparison method for millionaires”. *In CT-RSA 2001: The Cryptographers’ Track at RSA Conference*, pages 457–472, 2001.
- [12] Ivan Damgard, Martin Geisler, and Mikkel Kroigaard “Efficient and Secure Comparison for On-Line Auc-