Bit-Slice Auction Circuit *

Kaoru Kurosawa* and Wakaha Ogata**

* Ibaraki University, 4-12-1 Nakanarusawa, Hitachi, Ibaraki, 316-8511, Japan kurosawa@cis.ibaraki.ac.jp
** Tokyo Institute of Technology, 2–12–1 O-okayama, Meguro-ku, Tokyo 152–8552, Japan. wakaha@ss.titech.ac.jp

Abstract

In this paper, we introduce a *bit-slice* approach for auctions and present a more efficient circuit than the *normal* approach for the highest-price auction. Our circuit can be combined with any auction protocol based on general circuit evaluation. Especially, if we combine with the mix and match technique, then we can obtain a highest-price auction protocol which is at least seven times faster. A second-price auction protocol is also easily constructed from our circuit.

Keyword. auction, multiparty protocol, bit-slice, circuit, mix and match

1 Introduction

1.1 Sealed-Bid Auction

Auctions are getting very popular in the Internet. They are now a major area in the web electric commerce.

A sealed-bid auction consists of two phases, the bidding phase and the opening phase. In the bidding phase, all the bidders submit their bidding prices to the auctioneer. In the opening phase (of the highest-price auction), the auctioneer announces the highest price and the identities of the winners.

In the second-price auction (also known as Vickrey auction), the highest bidder wins, and the clearing price, the price that the winner has to pay, is equal to the second highest bid. (It is closer to the real life auction than the highest-price auction.)

Throughout the paper, we assume that:

- There are n servers.
- There are m bidders, denoted by $A_1, A_2, \ldots, A_m$.

*A preliminary version of this paper was presented at ESORICS 2002 and appeared in *Lecture Notes in Computer Science* **2502** (2002), 24–38.

- Each bidder A_i has a k-bit bid $B_i = (b_i^{(k-1)}, \ldots, b_i^{(0)})_2$.

In general, $X = (x^{(k-1)}, \ldots, x^{(0)})_2$ denotes a k-bit integer, where $x^{(k-1)}$ denotes the most significant bit and $x^{(0)}$ denotes the least significant bit.

1.2 Auction Protocols Based on Circuit Evaluation

In the trivial scheme which assumes a single trusted auctioneer, the auctioneer knows all the bidding prices. He may also tell a lie about the highest price and the winners.

Hence we need a cryptographically secure auction protocol which satisfies privacy and correctness. The correctness means that the announced highest price and the identities of the winners are guaranteed to be correct. The privacy means that no adversary can compute any other information.

In principle, it is known that any function can be computed securely by using general multiparty protocols [25, 16, 5, 9, 17]. (They are also known as general function evaluation protocols.) A problem is, however, that the cost of each bidder is very large in their general forms. Therefore, several schemes have been proposed to achieve efficient and secure auctions.

In the auction protocol of Naor, Pinkas and Sumner [22] which involves two servers, a proxy oblivious transfer protocol was introduced and it was combined with Yao's garbled circuit technique [25]. Jakobsson and Juels pointed out a flaw and it was fixed by Juels and Szydlo [21]. The fixed protocol is secure if the two servers do not collude. The cost of each server is $O(mkt)$, where t is a security parameter.

Jakobsson and Juels introduced a new general multiparty protocol called *mix and match* and showed its application to auctions [20]. The *mix and match* technique avoids the use of verifiable secret sharing schemes (VSS) which is intensively used in the other general multiparty protocols. In their auction protocol (JJ auction protocol), therefore, each bidder A_i has only to submit her encrypted bidding price without executing a VSS. The cost of each server is $O(mnk)$ exponentiations.

Cramer, Damgård and Nielsen introduced another general multiparty protocol which avoids VSS [10]. It can also be used for auctions. While the mix and match protocol is based on the DDH assumption alone, it is not known if this is possible for this protocol. On the other hand, the round complexity is $O(d)$ while it is $O(n + d)$ in the mix and match protocol, where d is the depth of the circuit of a given function. The message complexities are the same.

Baudron and Stern showed an auction protocol which assumes a semi-trusted auctioneer T [4]. In this protocol, T blindly and noninteractively evaluates a circuit whose output tells if A_i is a winner or not for each bidder A_i. A_i knows if he is a winner by decrypting the output ciphertext of this circuit. T learns no information if he does not collude with any bidder. The cost of T is $\Theta(mk^m)$. (This protocol is not based on *general* circuit evaluation. It uses some special predicates and makes use of its special properties.)

1.3 Our Contribution

In general, a multiparty protocol for computing a function $f(x_1, \ldots, x_n)$ is designed as follows. First we draw a Boolean circuit C_f which computes f. We next apply a general gate evaluation technique to each gate of C_f. Therefore, the smaller the circuit size is, the more efficient the protocol is.

The normal approach for circuit design for auctions is to compare the bidding prices one by one (in other words, to run a millionaire's problem protocol in order). To the authors' knowledge, the smallest size circuit in this approach for the highest-price auction requires $7mk$ logical gates and m $Select_k$ gates, where a logical gate has 2 input and 1 output bits, and a $Select_k$ gate has $2k+1$ input and k output bits. (We present such a circuit in Sec. 2.)

In this paper, we introduce a *bit-slice* approach for auctions and present a more efficient circuit than the *normal* approach for the highest-price auction. The proposed circuit requires only $2mk$ logical gates while the normal approach circuit requires $7mk$ logical gates and m $Select_k$ gates as shown above.

Suppose that $B_{\max} = (b_{\max}^{(k-1)}, \ldots, b_{\max}^{(0)})_2$ is the highest bidding price, where $b_{\max}^{(k-1)}$ denotes the most significant bit and $b_{\max}^{(0)}$ denotes the least significant bit. Then the proposed approach first determines $b_{\max}^{(k-1)}$ by looking at the most significant bits of all the bids. It next determines $b_{\max}^{(k-2)}$ by looking at the second most significant bits of all the bids, and so on.

Our circuit can be combined with any auction protocol based on *general* circuit evaluation. Especially, if we combine our circuit with the mix and match technique, then we can further reduce the number of gates just to mk by using the homomorphic property of the encryption function. Hence we can obtain a protocol which is at least seven times faster than JJ auction protocol.

We also show that a second-price auction protocol (which is closer to the real-life auction than the highest-price auction) can be easily obtained from our bit-slice circuit for the highest-price auction.

1.4 Other Related Works

There are many auction protocols which do not use circuit evaluation. However, they have problems such as follows.

The first cryptographic auction scheme was proposed by Franklin and Reiter [14]. This scheme is not fully private, in the sense that it only ensures the confidentiality of bids until the end of the protocol.

In the scheme of Cachin [6] which involves two servers, a partial order of bids is leaked to one of the two servers. The cost of each bidder is $O(mkt)$ and the cost of each server is $O(m^2kt)$, where t is a security parameter.

In the scheme of Di Crescenzo [13] which involves a single server, a honest but curious server does not learn any information under the quadratic residuosity assumption. However, nothing is known about the security if the server is malicious. The cost of each bidder is $O(m^2k^2)$ and the cost of server is also $O(m^2k^2)$.

Some other works require $O(mn2^k)$ cost for each server [18, 23]. In the scheme of [3], the cost of a server is $O(m2^k)$ and the cost of each bidder is $O(2^k)$. Note that these costs are much larger than the cost of auction protocols based on circuit design such as [22, 20].

1.5 Organization of the Paper

In Sec. 2, we present the normal approach for circuit design for auctions. In Sec. 3, we propose a bit-slice circuit for the highest-price auction. In Sec. 4, we briefly describe the mix and match technique. In Sec. 5, we show a new highest-price auction protocol which is obtained by combining the bit slice circuit and the mix and match technique. In Sec. 6, we present our second-price auction protocol.

2 Normal Approach for Auction Circuit Design

The normal approach for circuit design for auctions is to compare the bidding prices one by one (in other words, to run a millionaire's problem protocol in order). In this section, we present such a circuit which seems to be the most efficient.

2.1 Primitive Gate

For two bits x and y, define $Bigger_1$ and EQ_1 by

$$\begin{aligned} Bigger_1(x,y) &= \begin{cases} 1 & \text{if } x > y \\ 0 & \text{otherwise} \end{cases} \\ EQ_1(x,y) &= \begin{cases} 1 & \text{if } x = y \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

We next define a gate $Select_\kappa$ which has $2\kappa+1$ input bits and κ output bits as follows.

$$Select_\kappa(b, x^{(\kappa-1)}, \ldots, x^{(0)}, y^{(\kappa-1)}, \ldots, y^{(0)}) = \begin{cases} (x^{(\kappa-1)}, \ldots, x^{(0)}) & \text{if } b = 1 \\ (y^{(\kappa-1)}, \ldots, y^{(0)}) & \text{if } b = 0 \end{cases}$$

2.2 Boolean Circuit for the Millionaire's Problem

For $X = (x^{(k-1)}, \ldots, x^{(0)})_2$ and $Y = (y^{(k-1)}, \ldots, y^{(0)})_2$, define

$$\begin{aligned} Bigger_k(X,Y) &= \begin{cases} 1 & \text{if } X > Y \\ 0 & \text{otherwise} \end{cases} \\ Max_k(X,Y) &= \begin{cases} X & \text{if } X > Y \\ Y & \text{otherwise} \end{cases} \\ EQ_k(X,Y) &= \begin{cases} 1 & \text{if } X = Y \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

We first show two circuits for the millionaire's problem, $Bigger_k$ and Max_k, which seem to be the most efficient.

Let $a_k = 1$. For $i = k-1$ to 1, do

$$a_i = a_{i+1} \wedge EQ_1(x^{(i)}, y^{(i)}).$$

Then

$$\begin{aligned} Bigger_k(X,Y) &= Bigger_1(x^{(k-1)}, y^{(k-1)}) \\ &\quad \vee (Bigger_1(x^{(k-2)}, y^{(k-2)}) \wedge a_{k-1}) \\ &\quad\quad \vdots \\ &\quad \vee (Bigger_1(x^{(0)}, y^{(0)}) \wedge a_1). \\ Max_k(X,Y) &= Select_k(Bigger_k(X,Y), X, Y). \end{aligned}$$

2.3 Boolean Circuit for Auction

We next present two circuits for the highest-price auction, *Highest* and *Winner*.

Highest outputs the highest bidding price of m bids, $B_1, \ldots, B_m$. It is obtained by implementing the following algorithm by a circuit. Let $B_{\max} = 0$. For $i = 1, \ldots, m$, do

$$B_{\max} := Max_k(B_{\max}, B_i).$$

It is clear that the final $B_{\max}$ is the highest bidding price.

Winner is a circuit which outputs the winners. That is,

$$Winner(B_1, \ldots, B_m) = (w_1, \ldots, w_m),$$

where

$$w_i = \begin{cases} 1 & \text{if } B_i = B_{\max} \\ 0 & \text{otherwise} \end{cases}$$

Each w_i is obtained as follows.

$$w_i = EQ_k(B_i, B_{\max}).$$

The sizes of these circuits will be given in Sec. 3.3.

3 Bit-Slice Approach

In this section, we present a more efficient circuit than the *normal* approach by using a *bit-slice* approach for the highest-price auction. Suppose that $B_{\max} = (b_{\max}^{(k-1)}, \ldots, b_{\max}^{(0)})_2$ is the highest bidding price. Then the proposed circuit first determines $b_{\max}^{(k-1)}$ by looking at the most significant bits of all the bids. It next determines $b_{\max}^{(k-2)}$ by looking at the second most significant bits of all the bids, and so on.

For two m-dimensional binary vectors $\mathbf{X} = (x_1, \ldots, x_m)$ and $\mathbf{Y} = (y_1, \ldots, y_m)$, define

$$\mathbf{X} \wedge \mathbf{Y} = (x_1 \wedge y_1, \ldots, x_m \wedge y_m).$$

3.1 Proposed Circuit for Auction

Our idea is as follows. Let D_j be the highest price when considering the upper j bits of the bids. That is,

$$\begin{aligned} D_1 &= (b_{\max}^{(k-1)}, 0 \cdots, 0)_2, \\ D_2 &= (b_{\max}^{(k-1)}, b_{\max}^{(k-2)}, 0 \cdots, 0)_2, \end{aligned}$$

etc,

$$D_k = (b_{\max}^{(k-1)}, \ldots, b_{\max}^{(0)})_2 = B_{\max}.$$

In the first round, we find $b_{\max}^{(k-1)}$ and then eliminate all the bidders A_i such that $B_i < D_1$. In the second round, we find $b_{\max}^{(k-2)}$ and then eliminate all the bidders A_i such that $B_i < D_2$, and so on. At the end, the remained bidders are the winners. For that purpose, we update $\mathbf{W} = (w_1, \ldots, w_m)$ such that

$$w_i = \begin{cases} 1 & \text{if } B_i \geq D_j \\ 0 & \text{otherwise} \end{cases}$$

for $j = 1$ to k.

Our circuit is obtained by implementing the following algorithm. For given m bids, $B_1, \ldots, B_m$, define $\mathbf{V}_j$ as

$$\mathbf{V}_j = (b_1^{(j)}, \ldots, b_m^{(j)})$$

for $j = 0, \ldots, k-1$. That is, $\mathbf{V}_j$ is the vector consisting of the $j+1$th lowest bit of each bid.

Let $\mathbf{W} = (1, \ldots, 1)$. For $j = k-1$ to 0, do;

(Step 1) For $\mathbf{W} = (w_1, \ldots, w_m)$, let

$$\begin{aligned} \mathbf{S}_j &= \mathbf{W} \wedge \mathbf{V}_j \\ &= (w_1 \wedge b_1^{(j)}, \ldots, w_m \wedge b_m^{(j)}), && (1) \\ b_{\max}^{(j)} &= (w_1 \wedge b_1^{(j)}) \vee \cdots \vee (w_m \wedge b_m^{(j)}). && (2) \end{aligned}$$

(Step 2) If $b_{\max}^{(j)} = 1$, then let $\mathbf{W} = \mathbf{S}_j$.

Then the highest price is obtained as $B_{\max} = (b_{\max}^{(k-1)}, \ldots, b_{\max}^{(0)})_2$. Let the final $\mathbf{W}$ be $(w_1, \ldots, w_m)$. Then A_i is a winner if and only if $w_i = 1$.

We record this as the following theorem.

Theorem 3.1 *In the above algorithm,*

- $B_{\max}$ *is the highest bidding price.*
- *For the final* $\mathbf{W} = (w_1, \ldots, w_m)$*,* A_i *is a winner if and only if* $w_i = 1$*.*

The size of our circuit will be given in Sec. 3.3.

3.2 Example

Suppose that $m = 4$, $k = 5$ and each bid is

$$\begin{aligned} B_1 &= 20 = (1,0,1,0,0)_2, \\ B_2 &= 17 = (0,1,1,1,1)_2, \\ B_3 &= 18 = (1,0,0,1,0)_2, \\ B_4 &= 29 = (1,1,1,0,1)_2. \end{aligned}$$

Then $\mathbf{V}_4 = (1,0,1,1)$, $\mathbf{V}_3 = (0,1,0,1)$ and etc. Let $\mathbf{W} = (1,1,1,1)$. Now

1. $\mathbf{S}_4 = \mathbf{W} \wedge \mathbf{V}_4 = (1,0,1,1)$, $b_{\max}^{(4)} = 1$ and $\mathbf{W} := \mathbf{S}_4 = (1,0,1,1)$.
2. $\mathbf{S}_3 = \mathbf{W} \wedge \mathbf{V}_3 = (0,0,0,1)$, $b_{\max}^{(3)} = 1$ and $\mathbf{W} := \mathbf{S}_3 = (0,0,0,1)$.
3. $\mathbf{S}_2 = \mathbf{W} \wedge \mathbf{V}_2 = (0,0,0,1)$, $b_{\max}^{(2)} = 1$ and $\mathbf{W} := \mathbf{S}_2 = (0,0,0,1)$.
4. $\mathbf{S}_1 = \mathbf{W} \wedge \mathbf{V}_1 = (0,0,0,0)$, $b_{\max}^{(1)} = 0$.
5. $\mathbf{S}_0 = \mathbf{W} \wedge \mathbf{V}_0 = (0,0,0,1)$, $b_{\max}^{(0)} = 1$ and $\mathbf{W} := \mathbf{S}_0 = (0,0,0,1)$.

Therefore, we obtain that the highest bidding price is $(b_{\max}^{(4)}, \ldots, b_{\max}^{(0)})_2 = (1,1,1,0,1)_2 = 29$ and A_4 is the winner.

3.3 Comparison of Circuit Size

In this subsection, we compare the size of the normal circuit shown in Sec. 2 and that of our bit-slice circuit for the highest-price auction. See Table 1.

First the size of the normal circuit is given as follows. The circuit $Bigger_k$ requires k $Bigger_1$ gates, $2(k-1)$ AND gates, $k-1$ OR gates and $k-1$ EQ_1 gates. The circuit Max_k requires one $Select_k$ gate and one $Bigger_k$ circuit. The circuit *Highest* is obtained by implementing m Max_k circuits. In addition, the circuit *Winner* requires m EQ_k gates, where an EQ_k gate is implemented by k EQ_1 gates and $(k-1)$ AND gates.

Therefore, the normal circuits for the highest-price auction, *Highest* and *Winner*, require mk $Bigger_1$ gates, $3m(k-1)$ AND gates, $m(k-1)$ OR gates, $m(2k-1)$ EQ_1 gates and m $Select_k$ gates in total.

Next our bit slice circuit is given by implementing Eq.(1) and Eq.(2) k times. Therefore, it requires mk AND gates and $(m-1)k$ OR gates.

Hence roughly speaking, the proposed circuit requires only $2mk$ logical gates while the normal circuit requires $7mk$ logical gates and m $Select_k$ gates.

4 MIX and Match Protocol

4.1 Overview

In this section, we briefly describe the mix and match technique introduced by Jakobsson and Juels [20]. It is a general multiparty protocol which does not use

Table 1: Comparison of circuit sizes

	AND	OR	$Bigger_1$	EQ_1	$Select_k$
Normal circuit	$3m(k-1)$	$m(k-1)$	mk	$m(2k-1)$	m
Bit-slice circuit	mk	$(m-1)k$	0	0	0

VSS. Instead, it uses a homomorphic encryption scheme (for example, ElGamal) and a MIX net [8, 1, 2].

This model involves n players, denoted by $P_1, P_2, \ldots, P_n$ and assumes that there exists a public board. We consider an adversary who may corrupt up to t players, where $n \geq 2t+1$.

The players agree in advance on a representation of the target function f as a circuit C_f. Suppose that C_f consists of N gates, $G_1, \ldots, G_N$. Let the input of P_i be a k-bit integer B_i. The aim of the protocol is for players to compute $f(B_1, \ldots, B_n)$ without revealing any additional information. It goes as follows.

Input stage: Each P_i computes ciphertexts of the bits of B_i and broadcasts them. She proves that each ciphertext represents 0 or 1 in zero-knowledge by using the technique of [11].

Mix and Match stage: The players blindly evaluates each gate G_j in order.

Output stage: After evaluating the last gate G_N, the players obtain o_N, a ciphertext encrypting $f(B_1, \ldots, B_n)$. They jointly decrypt this ciphertext value to reveal the output of the function f.

This protocol meets the security requirements formalized by Canetti [7] for secure multiparty protocols [20, page 171]. The cost of each player is $O(nN)$ exponentiations and the overall message complexity is also $O(nN)$ (see [20, page 172]).

The details of the protocol are described in the following subsections.

4.2 Requirements for the Encryption Function

Let E be a public-key probabilistic encryption function. We denote by $E(m)$ the set of encryptions for a plaintext m and by $e \in E(m)$ a particular encryption of m. We say that e is a standard encryption if it is encrypted with no randomness.

E must satisfy the following properties.

- **homomorphic property**

  There exists a polynomial time computable operations, $^{-1}$ and $\otimes$, as follows for a large prime q.

  1. If $e \in E(m)$, then $e^{-1} \in E(-m \bmod q)$.
  2. If $e_1 \in E(m_1)$ and $e_2 \in E(m_2)$, then $e_1 \otimes e_2 \in E(m_1 + m_2 \bmod q)$.

For a positive integer a, define

$$a \cdot e = \underbrace{e \otimes e \otimes \cdots \otimes e}_{a}.$$

- **random re-encryptability**

  Given $e \in E(m)$, there is a probabilistic re-encryption algorithm that outputs $e' \in E(m)$, where e' is uniformly distributed over $E(m)$.

- **threshold decryption**

  For a given ciphertext $e \in E(m)$, any t out of n players can decrypt e along with a zero-knowledge proof of the correctness. However, any $t-1$ out of n players cannot decrypt e.

Such $E(\cdot)$ can be obtained by slightly modifying ElGamal encryption scheme over a group G of order $|G| = q$, where q is a large prime. G can be constructed as a subgroup of Z_p^*, where p is a prime such that $q \mid p-1$. It can also be obtained from elliptic curves.

Let g be a generator of G, i.e. $G = \langle g \rangle$. The secret key x is randomly chosen from Z_q and the public key is $y = g^x$. An encryption of m is given by

$$(g^r, g^m y^r) \in E(m),$$

where $r \in Z_q$ is a random element. For ciphertexts, define $^{-1}$ and $\otimes$ as

$$(u, v)^{-1} = (u^{-1}, v^{-1}).$$

$$(u_1, v_1) \otimes (u_2, v_2) = (u_1 u_2, v_1 v_2).$$

Then it is easy to see that the homomorphic property is satisfied. A re-encryption of $(u, v) \in E(m)$ is given by $(u', v') = (g^{r'} u, y^{r'} v)$ for a random element $r' \in Z_q$.

For threshold decryption, each player obtains a private share x_i of x in Shamir's $(t+1, n)$-threshold secret-sharing scheme [24]. Each g^{x_i} is published. For details, see [12, 15]. Each player needs to broadcast $O(1)$ messages and compute $O(n)$ exponentiations in threshold decryption.

4.3 MIX Protocol

A MIX protocol takes a list of ciphertexts $(\xi_1, \ldots, \xi_L)$ and outputs a permuted and re-encrypted list of the ciphertexts $(\xi'_1, \ldots, \xi'_L)$ without revealing the relationship between $(\xi_1, \ldots, \xi_L)$ and $(\xi'_1, \ldots, \xi'_L)$, where ξ_i or ξ'_i can be a single ciphertext e, or a list of l ciphertexts, $(e_1, \ldots, e_l)$, for some $l > 1$. We further require that anybody (even an outsider) can verify the validity of $(\xi'_1, \ldots, \xi'_L)$ (public verifiability).

For small L, a MIX protocol is efficiently implemented as shown in [1, 2, 19]. In this protocol, each player needs to compute $O(nlL \log L)$ exponentiations and broadcast $O(nlL \log L)$ messages.

Table 2: Logical Table of AND

x_1	x_2	$x_1 \wedge x_2$
$a_1 \in E(0)$	$b_1 \in E(0)$	$c_1 \in E(0)$
$a_2 \in E(0)$	$b_2 \in E(1)$	$c_2 \in E(0)$
$a_3 \in E(1)$	$b_3 \in E(0)$	$c_3 \in E(0)$
$a_4 \in E(1)$	$b_4 \in E(1)$	$c_4 \in E(1)$

4.4 Plaintext Equality Test

Given two ciphertexts $e_1 \in E(m_1)$ and $e_2 \in E(m_2)$, this protocol checks if $m_1 = m_2$. Let $e_0 = e_1 \otimes e_2^{-1}$.

(Step 1) For each player P_i (where $i = 1, \ldots, n$):

P_i chooses a random element $a_i \in Z_q$ and computes $z_i = a_i \cdot e_0$. He broadcasts z_i and proves the validity of z_i in zero-knowledge.

(Step 2) Let $z = z_1 \otimes z_2 \otimes \cdots \otimes z_n$. The players jointly decrypt z by threshold verifiable decryption and obtain the plaintext m.

Then it holds that

$$m = \begin{cases} 0 & \text{if } m_1 = m_2, \\ random & \text{otherwise} \end{cases} \tag{3}$$

This protocol is minimal knowledge under the DDH assumption (see [20, page 169]). The cost of each player is $O(n)$ exponentiations.

4.5 Mix and Match Stage

For each logical gate $G(x_1, x_2)$ of a given circuit, n players jointly computes $E(G(x_1, x_2))$ from $e_1 \in E(x_1)$ and $e_2 \in E(x_2)$ keeping x_1 and x_2 secret. For simplicity, we show a mix and match stage for AND.

1. n players first consider the standard encryption of each entry of Table 2.
2. By applying a MIX protocol to the four rows of Table 2, n players jointly compute blinded and permuted rows of Table 2. Let the ith row be (a'_i, b'_i, c'_i) for $i = 1, \ldots, 4$.
3. n players next jointly find the row i such that the plaintext of e_1 is equal to that of a'_i and the plaintext of e_2 is equal to that of b'_i by using the plaintext equality test protocol.
4. For this i, it holds that $c'_i \in E(x_1 \wedge x_2)$.

5 Bit-Slice Circuit and Mix-Match Protocol

5.1 Overview

We can obtain a highest-price auction protocol by combining the proposed circuit of Sec. 3.1 with any general multiparty protocol. Then a more efficient protocol is obtained because the bit-slice circuit is more efficient than the normal circuit as shown in Table 1.

In this section, we show a combination with the mix and match technique, as an example. In this case, we can further improve the efficiency. That is, we can remove all the OR computations of Eq.(2) by using the homomorphic property of the encryption function as follows. Let

$$h_j = (x_1 \wedge b_1^{(j)}) + \cdots + (x_m \wedge b_m^{(j)}).$$

Then it is easy to see that $h_j = 0$ if and only if $b_{\max}^{(j)} = 0$. Therefore, n servers have only to execute a plaintext equality test protocol for checking if $h_j = 0$ to decide if $b_{\max}^{(j)} = 0$. Note that each server can compute summation locally by using the homomorphic property of the encryption scheme. Hence we can replace $(m-1)k$ OR computations with only k plaintext equality tests.

5.2 Bidding Phase

Each bidder A_i computes a ciphertext of her bidding price B_i as

$$ENC_i = (e_{i,k-1}, \ldots, e_{i,0}),$$

where $e_{i,j} \in E(b_i^{(j)})$, and submits ENC_i. She also proves in zero-knowledge that $b_i^{(j)} = 0$ or 1 by using the technique of [11]. (This submission may be also digitally signed by A_i.)

5.3 Opening Phase

Suppose that $e_1 \in E(b_i)$ and $e_2 \in E(b_2)$, where b_1 and b_2 are binary. Let $Mul(e_1, e_2)$ denote a protocol which outputs

$$e \in E(b_1 \wedge b_2)$$

by applying a mix and match protocol for AND.

Let $\widetilde{\mathbf{W}} = (\tilde{w}_1, \ldots, \tilde{w}_m)$, where each $\tilde{w}_j \in E(1)$ is the standard encryption.

(Step 1) For $j = k-1$ to 0, do:

(Step 1-a) For $\widetilde{\mathbf{W}} = (\tilde{w}_1, \ldots, \tilde{w}_m)$, n servers jointly compute

$$\widetilde{\mathbf{S}}_j = (Mul(\tilde{w}_1, e_{1,j}), \ldots, Mul(\tilde{w}_m, e_{m,j})).$$

Table 3: Comparison of the highest-price auction protocols

	AND	OR	$Bigger_1$	EQ_1	$Select_k$
JJ [20]	$3m(k-1)$	$m(k-1)$	mk	$m(2k-1)$	m
Proposed	mk	0	0	0	0

(Step 1-b) Each server locally computes

$$h_j = Mul(\tilde{w}_1, e_{1,j}) \otimes \cdots \otimes Mul(\tilde{w}_m, e_{m,j}).$$

(Step 1-c) n servers jointly check if $h_j \in E(0)$ by using a plaintext equality test. Let

$$b_{\max}^{(j)} = \begin{cases} 0 & \text{if } h_j \in E(0) \\ 1 & \text{otherwise} \end{cases}$$

(Step 1-d) If $b_{\max}^{(j)} = 1$, then let $\widetilde{\mathbf{W}} = \widetilde{\mathbf{S}}_j$.

(Step 2) For the final $\widetilde{\mathbf{W}} = (\tilde{w}_1, \ldots, \tilde{w}_m)$, n servers jointly decrypt each $\tilde{w}_i$ by threshold verifiable decryption and obtain the plaintext w_i.

The highest price is obtained as $B_{\max} = (b_{\max}^{(k-1)}, \ldots, b_{\max}^{(0)})_2$. A_i is a winner if and only if $w_i = 1$.

5.4 Comparison

If we combine the normal circuit with the mix and match technique, then JJ auction protocol is obtained [20]. Table 3 shows the comparison between JJ auction protocol and proposed protocol.

In our protocol, we can replace $(m-1)k$ OR computations of Table 1 with only k plaintext equality tests as shown in Sec. 5.1. Therefore, our protocol requires mk AND computations and k plaintext equality tests.

We omit the cost of k plaintext equality tests in this table because it is much smaller than the cost for each logical gate. For example, an AND computation requires 8 plaintext equality tests and a MIX protocol of four items if we implement the protocol of Sec. 4.5. Hence mk AND computations requires $8mk$ plaintext equality tests and mk MIX protocols of four items. This is much larger than the cost of k plaintext equality tests.

Now roughly speaking, our protocol requires mk logical gates while JJ auction protocol requires $7mk$ logical gates and m $Select_k$ circuit. Therefore, our protocol is at least seven times faster than JJ auction protocol.

5.5 Discussion

We can slightly improve the opening phase by letting the initial value of $\widetilde{\mathbf{W}}$ be

$$\widetilde{\mathbf{W}} = (e_{1,k-1}, \cdots, e_{m,k-1}).$$

The efficiency is further improved if each bid is only $k' < k$ bits long. (Of course, this fact is not known in advance.) The smaller k' is, the faster the modified version is. For example, if $B_i = (0, \ldots, 0, b_i^{(0)})$ for all i, then no mix and match for AND is required. If $B_i = (0, \ldots, 0, b_i^{(1)}, b_i^{(0)})$ for all i, then the mix and match for AND is executed only once, and so on.

Such speed-up is impossible in JJ auction protocol.

6 Second-Price Auction

In the second-price auction (also known as Vickrey auction), the highest bidder wins, and the clearing price, the price that the winner has to pay, is equal to the second highest bid. Therefore, Vickrey auctions are much closer to the real life auction than the highest-price auctions. A cryptographically secure second-price auction scheme should reveal only the identities of the winners (the highest bidders) and the second-highest bid.

In this section, we show that a second-price auction protocol is easily obtained from our bit-slice circuit for the highest-price auction.

6.1 Normal Approach Circuit

If we use the normal approach, we can obtain a second-price auction circuit by keeping track of the two highest bids found so far, B_{first} and B_{second}, as follows.

Let $B_{\text{first}} = B_{\text{second}} = 0$. For $i = 1, \ldots, m$, do

$$\begin{aligned} X &:= \mathit{Select}_k(\mathit{Bigger}_k(B_{\text{second}}, B_i), B_{\text{second}}, B_i). \\ B_{\text{second}} &:= \mathit{Select}_k(\mathit{Bigger}_k(B_{\text{first}}, X), X, B_{\text{first}}). \\ B_{\text{first}} &:= \mathit{Select}_k(\mathit{Bigger}_k(B_{\text{first}}, X), B_{\text{first}}, X). \end{aligned}$$

It is easy to see that the final B_{first} is the highest bidding price and B_{second} is the second-highest price. The identities of the winners are obtained similarly to Sec. 2.3.

6.2 Bit-Slice-Type Second-Price Auction

We define two types of highest-price auction schemes, a winner-only scheme and a price-only scheme. A winner-only scheme reveals only the identities of the winners, but not the highest price. A price-only scheme reveals only the highest bid, but not the identities of the winners.

Now suppose that there is a winner-only scheme Q_1 and a price-only scheme Q_2. Then we can obtain a second-price auction scheme as follows:

Step 1. Run Q_1.

Step 2. Delete the winners of Q_1 from the set of bidders.

Step 3. Run Q_2 for the rest of the bidders.

Indeed, the above scheme reveals only the identities of the winners of Q_1, and the highest bidding price of Q_2 which is the second highest price among the bidders.

Such protocols Q_1 and Q_2 are easily obtained from our bit-slice circuit for the highest-price auction as follows.

- Apply any multiparty protocol to the circuit of Sec. 3.1 and decrypt only $b_{\max}^{(k-1)}, \ldots, b_{\max}^{(0)}$ (but not $\mathbf{W}$) in the output stage. Then we obtain a price-only scheme.

- In the circuit of Sec. 3.1, replace Step 2 with

$$\mathbf{W} = \mathit{Select}_m(b_{\max}^{(j)}, \mathbf{S}_j, \mathbf{W}). \tag{4}$$

  Then apply any multiparty protocol to the above circuit and decrypt only $\mathbf{W}$ (but not $b_{\max}^{(k-1)}, \ldots, b_{\max}^{(0)}$) in the output stage. We now obtain a winner-only scheme.

6.3 Comparison

Table 4 shows the comparison of the sizes of the normal second-price auction circuit and the bit-slice circuit.

The size of the normal circuit is given as follows (see Sec. 6.1). The circuit which computes the second-highest bid, shown in Sec. 6.1, requires $3m$ Select_k gates and $2m$ Bigger_k circuits, where the circuit Bigger_k requires k Bigger_1 gates, $2(k-1)$ AND gates, $k-1$ OR gates and $k-1$ EQ_1 gates as mentioned in section 3.3. The circuit which computes *Winner*, as shown in Sec. 2.3, requires mk EQ_1 gates and $m(k-1)$ AND gates. Therefore, the normal circuits for the second-price auction require $2mk$ Bigger_1 gates, $5m(k-1)$ AND gates, $2m(k-1)$ OR gates, $m(3k-2)$ EQ_1 gates and $3m$ Select_k gates in total.

Next our bit slice circuit consists of the winner-only scheme Q_1 and the price-only scheme Q_2. Q_1 is obtained by implementing Eq.(1), (2) and (4) k times. Q_2 is obtained by implementing Eq.(1) and (2) k times for $m-1$ bidders. Therefore, our second-price auction circuit requires $(2m-1)k$ AND gates, $(2m-3)k$ OR gates and k Select_m gates.

Table 4: Comparison of the second-price auction circuits

	AND	OR	Bigger_1	EQ_1	Select_k	Select_m
Normal	$5m(k-1)$	$2m(k-1)$	$2mk$	$m(3k-2)$	$3m$	0
Bit slice	$(2m-1)k$	$(2m-3)k$	0	0	0	k

Next, suppose that we use the mix and match technique as a general multi-party protocol. In this case, the price-only scheme is obtained by deleting Step 2 of Sec. 5.3. Then roughly speaking, our second-price auction protocol requires

$3mk$ logical gates and k $Select_m$ gates. On the other hand, the second-price auction protocol obtained by combining the normal circuit and the mix and match technique requires $11mk$ logical gates and $3m$ $Select_k$ gates. We show a comparison in Table 5.

Table 5: Comparison of the second-price auction protocols

	AND	OR	$Bigger_1$	EQ_1	$Select_k$	$Select_m$
Normal	$5m(k-1)$	$2m(k-1)$	$2mk$	$m(3k-2)$	$3m$	0
Bit slice	$(2m-1)k$	$(m-2)k$	0	0	0	k

7 Conclusion

In this paper, we introduced a *bit-slice* approach for auctions and presented a more efficient circuit than the *normal* approach for the highest-price auction. Especially, if we combine our circuit with the mix and match technique, then we can obtain an auction protocol which is at least seven times faster than JJ auction protocol.

We also showed that a second-price auction protocol (which is closer to the real-life auction than the highest-price auction) can be easily obtained from our bit-slice circuit for the highest-price auction.

Suppose that there are w winners such that $w \geq 2$. Then our circuits reveals the indetities of all the winners. However, there are some cases such that we must select just one winner randomly. It will be a further work to design an efficient circuit or a protocol which implements such an auction.

Acknowledgement

The authors thank Martin Hirt for providing many useful comments.

References

[1] M. Abe, "Mix-Networks on permutation networks," Proc. of Asiacrypt '99, LNCS Vol. 1716, pp. 258–273 (1999).

[2] M. Abe and F. Hoshino, "Remarks on Mix-Network Based on Permutation Networks," Proc. of PKC 2001, pp. 317–324 (2001).

[3] M. Abe and K. Suzuki, "M1-st Price Auction Using Homomorphic Encryption," Proc. of PKC2002, pp. 115–124 (2002).

[4] O. Baudron and J. Stern, "Non-Interactive Private Auctions," Proc. of Financial Cryptography 2001 (2001).

[5] M. Ben-Or, S. Goldwasser, and A. Wigderson. "Completeness theorems for non-cryptographic fault-tolerant distributed computation," Proceedings of the twentieth annual ACM Symp. Theory of Computing, STOC, pp. 1–10, May 2–4 (1988).

[6] C. Cachin, "Efficient private bidding and auctions with an oblivious third party," ACM CSS '99, pp. 120–127, ACM (1999).

[7] R. Canetti, "Security and composition of multiparty cryptographic protocols," Journal of Cryptology, Vol. 13, No. 1, pp. 143–202 (2000).

[8] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," Communications of the ACM, Vol. 24, pp. 84–88 (1981).

[9] D. Chaum, C. Crépeau, and I. Damgård. "Multiparty unconditionally secure protocols," Proc. of the twentieth annual ACM Symp. Theory of Computing, STOC, pp. 11–19, May 2–4 (1988).

[10] R. Cramer, I. Damgård, and J. B. Nielsen, "Multiparty Computation from Threshold Homomorphic Encryption," Proc. of Eurocrypt'01, LNCS Vol. 2045, pp. 280–300 (2001).

[11] R. Cramer, I. Damgård, and B. Schoenmakers, "Proofs of partial knowledge and simplified design of witness hiding protocols," Proc. of CRYPTO '94. LNCS Vol. 839, pp. 174–187 (1994).

[12] Y. Desmedt and Y. Frankel, "Threshold cryptosystems," In Proc. of Crypto '89, pp. 307–315.

[13] G.Di Crescenzo, "Private selective payment protocols," In Proc. of Financial Cryptography '00, LNCS Vol. 1962, pp. 72–89 (2000).

[14] M. Franklin and M. Reiter, "The Design and Implementation of a Secure Auction Service," IEEE Trans. on Software Engineering, Vol. 22, No. 5 (1996).

[15] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Robust threshold DSS signatures," In Proc of Eurocrypt '96, LNCS Vol. 1070, pp. 354–371 (1996).

[16] O. Goldreich, S. Micali, and A. Wigderson. "How to play any mental game," In Proceedings of the nineteenth annual ACM Symp. Theory of Computing, STOC, pp. 218–229, May 25–27 (1987).

[17] M. Hirt, U. Maurer, and B. Przydatek, "Efficient secure multiparty computation," Proc. of Asiacrypt'2000, LNCS Vol. 1976, pp. 143–161 (2000).

[18] M. Harkavy, J.D. Tygar, and H. Kikuchi, "Electronic auction with private bids," In Third USENIX Workshop on Electronic Commerce Proceedings, pp. 61–74 (1998).

[19] M. Jakobsson and A. Juels, "Millimix: Mixing in small batches," DIMACS Technical report 99-33 (June 1999).

[20] M. Jakobsson and A. Juels, "Mix and Match: Secure Function Evaluation via Ciphertexts," In Proc. of Asiacrypt 2000, LNCS Vol. 1976, pp. 162–177 (2000).

[21] A. Juels and M. Szydlo, "An Two-Server Auction Protocol," In Proc. of Financial Cryptography 2002, (2002).

[22] M. Naor, B. Pinkas, and R. Sumner, "Privacy preserving auctions and mechanism design," In 1st ACM Conference on Electronic Commerce, pp. 129–140, ACM (1999).

[23] K. Sako, "An auction protocol which hides bids of losers," Proc. of PKC'2000, LNCS Vol. 1751, pp. 422–432 (2000).

[24] A. Shamir, "How to Share a Secret," Communications of the ACM, Vol. 22, pp. 612-613 (1979).

[25] A. Yao, "Protocols for secure computations (extended abstract)," Proc. of FOCS'82, pp. 160–164, IEEE Computer Society (1982).