Int. J. Inf. Secur. (2006) 5(4): 201–216
DOI 10.1007/s10207-006-0001-y

REGULAR CONTRIBUTION

Felix Brandt

How to obtain full privacy in auctions

Published online: 31 March 2006
© Springer-Verlag 2006

Abstract Privacy has become a factor of increasing importance in auction design. We propose general techniques for cryptographic first-price and $(M + 1)$st-price auction protocols that only yield the winners' identities and the selling price. Moreover, if desired, losing bidders learn no information at all, except that they lost. Our security model is merely based on computational intractability. In particular, our approach does not rely on trusted third parties, e.g., auctioneers. We present an efficient implementation of the proposed techniques based on El Gamal encryption whose security only relies on the intractability of the decisional Diffie–Hellman problem. The resulting protocols require just three rounds of bidder broadcasting in the random oracle model. Communication complexity is linear in the number of possible bids.

Keywords Auctions · Cryptographic protocols · Homomorphic encryption

1 Introduction

Auctions have become the major phenomenon of electronic commerce during the last years. They are not only widespread mechanisms to sell goods, but have also been shown applicable to task assignment, scheduling, or finding the shortest path in a network with selfish nodes. In recent times, the need for privacy has been a factor of increasing importance in auction design and various schemes to ensure the safe conduction of sealed-bid auctions have been proposed. The cryptographic protocols presented in this paper differ from existing work in that they do not require trusted third parties. They are merely based on computational intractability. Privacy is preserved to a maximal extent without compromising much of the round-efficiency that distinguishes sealed-bid auctions.

F. Brandt (✉)
Computer Science Department, University of Munich,
Oettingenstr. 67, 80538 Munich, Germany
E-mail: brandtf@tcs.informatik.uni-muenchen.de

Our setting consists of one seller and n bidders that intend to come to an agreement on the selling of a good.[1] The two major (sealed-bid) mechanisms that yield such an agreement are the *first-price* auction and the *second-price* auction (aka. Vickrey auction [50]). In both mechanisms, each bidder submits a sealed bid to the auctioneer expressing how much he is willing to pay. The auctioneer declares the bidder who submitted the highest bid as the winner of the auction. In the first-price auction, the winning bidder pays the amount that he bid, whereas in the second-price auction, he has to pay the amount of the *second*-highest bid. Both auction formats have their strengths and weaknesses. For example, the first-price auction yields more revenue when bidders are risk-averse. The second-price auction, on the other hand, is *strategy-proof*, which means that bidders are always best off bidding their true valuation of the good to be sold (when valuations are independent). This eliminates an agent's need to counterspeculate on other agents' valuations. Interestingly, the side-effects of this striking and celebrated advantage are said to contribute to the sparseness of the second-price auction in the real world for two reasons [42, 43, 45]:

- Bidders are reluctant to reveal their true valuations to the auctioneer.
- Bidders doubt the correctness of the result as they do not pay what they bid (unlike in the first-price auction).

An often-cited example of the latter problem is an auctioneer that creates a fake second-highest bid after having received all bids in order to increase his revenue (see e.g., [41]). Both above-mentioned issues are based on a lack of trust in the auctioneer. For this reason, it would be desirable to somehow "force" the auctioneer to always select the right outcome (*correctness*) and "prohibit" the propagation of private bid information (*privacy*). Various schemes for satisfying these desiderata (in first-price as well as second-price auctions) have been proposed in recent years (see Sect. 3). Virtually

[1] All the presented results also hold for similar auctions for other areas of application, in particular procurement auctions where there is one buyer and multiple sellers.

all of them rely on trusted third parties. The goal of this paper is to construct efficient multiparty protocols that allow *bidders* to jointly compute the auction outcome without revealing further information. In the rest of this paper, this will be called *emulation* of an auction mechanism. The two main objectives when designing our protocols were

- absolute privacy without assumptions in addition to intractability, and
- maximal round-efficiency.

We propose protocols for first-price and $(M+1)$st-price auctions. The latter are a generalization of second-price auctions. In an $(M+1)$st-price auction, the seller offers M indistinguishable units of the same item ($1 \leq M < n$) and each bidder desires to buy *one* of them (so-called "unit demand"). Vickrey has shown that it is a strategy-proof mechanism to sell those items to the M highest bidders for the uniform price given by the $(M+1)$st highest bid [50]. The second-price auction is just a special case of this mechanism for the selling of a single good ($M = 1$).

The remainder of this paper is structured as follows. In Sect. 2, we describe the general security model underlying this work. Recent related research on cryptographic auction protocols is reviewed and compared to our approach in Sect. 3. In Sect. 4, we give a detailed description of the concepts to be used in the concrete implementation presented in Sect. 5. Section 6 contains an analysis of the security and efficiency of the proposed protocols. The paper concludes with a brief overview of the results in Sect. 7.

2 Security model

Our primary goal is privacy that cannot be broken by any coalition of third parties or bidders. For this reason, we advocate a security model in which bidders themselves jointly compute the auction outcome so that any subset of bidders is incapable of revealing private information. Clearly, extensive interaction by bidders is undesirable in practice (but unavoidable given our objective). In order to minimize interaction, our secondary goal is to keep round complexity at a minimum (i.e., small constants) which requires the existence of a broadcast channel—a standard assumption in auction protocols (see Sect. 3). The main drawbacks implied by our setting are low resilience and relatively high computational and communication complexity. However, auctions that require such a high degree of privacy typically take place with few, well-known (i.e., non-anonymous) bidders, for instance when auctioning off radio spectrum licenses or former state-owned enterprises.

We consider cryptographic protocols for n bidders and one seller (to be called agents in the following). Each bidder i possesses a private input, his bid $bid_i \in B$. Agents engage in a multiparty protocol to jointly and securely compute the outcome function f. In our context, security consists of correctness (f is computed correctly) and privacy (no agent learns more information than what he can infer from the outcome and his private input). The classic results of secure multiparty computation (MPC) state that any such function f can be computed securely when

- at most $\lfloor \frac{n-1}{2} \rfloor$ agents share their information and trapdoor permutations exist [23], or
- at most $\lfloor \frac{n-1}{3} \rfloor$ agents share their information and a complete network of private channels exists [3, 11].

The first assumption is known as the *computational model* whereas the second one is called *unconditional model*. Neither assumption can be made in our context since it would enable subsets of bidders to manipulate the auction outcome and uncover other agents' bids. However, there are additional assumptions that allow for the secure computation of arbitrary functions in the computational model, and a restricted class of functions in the unconditional model, *without trusted thresholds of agents* (see, e.g., [22, 24, 38]). Privacy that only relies on the fact that less than *all* agents collude will be called *full privacy* in the following. In the computational model (on which we will focus in this paper), any function f can be computed fully privately when

- trapdoor permutations exist, and
- a designated agent does not quit or reveal information prematurely.[2]

In the auction protocols presented in this paper, the seller will take the role of the designated agent (see Sect. 6 for a justification). It is important to note that even when the seller quits or reveals information early, the worst thing that can happen is that a bidder learns the outcome and quits the protocol before the remaining agents were able to learn the outcome. Bid privacy is not affected by premature abort. Another common way to obtain fairness without a trusted majority is the incremental release of secrets (e.g., [20, 24, 51]).

Whenever a malicious bidder disrupts the protocol by sending faulty messages or failing to prove the correctness of his behavior in zero-knowledge, this bidder will be removed and the protocol will be restarted (termination is guaranteed after at most $n-M$ iterations). We presume that the "public" is observing the protocol and therefore a malicious bidder can undoubtedly be identified, independently of how many of the remaining agents are trustworthy. As malicious bidders can easily be fined and they do not gain any information, there should be no incentive to disrupt the auction and we henceforth assume that a single protocol run suffices.

Due to the inefficiency of existing *general* MPC schemes, it is inevitable to design special-purpose protocols for the computation of particular functions. Let $\mathbf{b} = (bid_1, bid_2, \ldots, bid_n)$ be the vector of all bids and $f : B^n \to O^{n+1}$ the outcome function where

$$f(\mathbf{b}) = (f_1(\mathbf{b}), f_2(\mathbf{b}), \ldots, f_n(\mathbf{b}), (f_1(\mathbf{b}), f_2(\mathbf{b}), \ldots, f_n(\mathbf{b})))$$

so that bidder i learns $f_i(\mathbf{b})$ and the seller learns $(f_1(\mathbf{b}), f_2(\mathbf{b}), \ldots, f_n(\mathbf{b}))$. If bidder i won the auction, $f_i(\mathbf{b})$

[2] This self-evident but very useful restriction to circumvent fairness problems was used in our previous work (e.g., [6, 7]). Independently, the security of such a model was recently analyzed in [25].

yields the selling price. Otherwise "useless" information is returned. This will be called the *private outcome* setting because the outcome (selling price and winners' identity) are only revealed to the concerned parties (seller and winners). For reasons of transparency and efficiency, we will also consider the computation of a *public outcome* function where all $f_i(\mathbf{b})$ are identical and yield the identity of the auction winner and the selling price.

In Sects. 4.1 and 4.2, outcome functions and corresponding computation protocols for first-price and $(M+1)$st-price sealed-bid auctions are proposed, respectively.

3 Related work

The interest in cryptographic protocols for auctions has been dramatically increasing. Starting with the work by Nurmi and Salomaa [37] and Franklin and Reiter [19], numerous secure sealed-bid auction schemes have been proposed in recent years, e.g., [1, 2, 27, 29, 32, 35, 36, 44]. Basically, all of the existing protocols have in common that privacy is obtained by distributing the computation of the outcome on a group of third parties. In *symmetric* protocols, there are multiple auctioneers that jointly determine the outcome by using threshold MPC (e.g., [27, 32, 44]). *Asymmetric* protocols introduce a second instance, for example an "auction issuer" or "auction authority," in addition to the auctioneer (e.g., [1, 2, 29, 35, 36]).

3.1 Cryptographic auction protocols

In the following, we briefly survey the security and efficiency of selected existing protocols.

Auction protocols based on Yao's garbled circuit technique [29, 36] are typical examples of asymmetric auction protocols that use two third parties to determine the auction outcome (one party constructs an obfuscated Boolean circuit and the other party obliviously evaluates the circuit). These protocols are very efficient both in terms of round complexity ($\mathcal{O}(1)$) and communication complexity ($\mathcal{O}(n \log k)$ where k is the number of possible bids). However, Yao's protocol was originally conceived for a model with passive adversaries. If you take into account malicious deviation by either one of the two parties, costly verification techniques such as cut-and-choose, consistency proofs, and the additional evaluation of a majority circuit need to be implemented [40]. Moreover, due to a lack of verifiability, a coalition of both third parties can not only reveal all private information but also claim an arbitrary auction outcome.

The protocol by Baudron and Stern [2] is based on the joint evaluation of a special-purpose Boolean circuit with the help of a third party. The communication complexity is exponential in n ($\mathcal{O}(n(\log k)^{n-1})$) which makes the scheme only applicable to a very limited number of bidders (five to six, as stated by the authors). After the result is broadcasted, the winner is required to claim that he won (violating non-repudiation). This is a disadvantage because the winner is able to back out of the protocol if he is not satisfied with the selling price. When computing the outcome of a Vickrey auction, additional interaction is required to compute the second-highest bid (while also revealing the identity of the second-highest bidder which is undesirable). Bidders' actions are verifiable. However, it is not possible to verify whether the third party behaves correctly.

The protocol by Lipmaa et al. [35] requires a single semi-trusted third party, the auction authority, in addition to the seller. The auction authority receives all bids (blinded by the seller using homomorphic encryption), identifies the second-highest bid and proves the correctness of the outcome by applying a novel efficient zero-knowledge proof. Winning bidders are required to claim that they won (violating non-repudiation). The protocol scales well with respect to the number of bidders, but only provides limited privacy as the auction authority learns *all* bid amounts. The only information hidden from the authority is the connection between bidders and bids. Neither the seller nor the auction authority can manipulate the outcome without being detected.

Like our protocols, the protocol by Abe and Suzuki [1] is based on bid vectors as defined in Sect. 4. However, the position of the $(M+1)$st-highest bid is jointly determined by the auctioneer and an "authority" using a binary search subprotocol. More specifically, the auctioneer releases mixed vector components to the authority who decrypts them to detect if there are either more than M bidders or less than $M+1$ bidders willing to pay. The entire process takes $\log k$ rounds. The protocol is based on Jakobsson and Juels mix-and-match technique [28] and is publicly verifiable.

There are various other protocols (e.g., [27, 32, 44]) based on multiple auctioneers using threshold MPC. The round complexity of such protocols is generally not constant. Table 1 summarizes important features of all mentioned protocols.

3.2 Bidder-resolved auctions

Generally, the privacy of protocols listed in the previous section relies on the assumption that a collusion of *all* involved third parties is ruled out. In this paper, we extend our recent work on fully private "bidder-resolved" auction protocols [5, 6] in which bidders jointly emulate the auctioneer without relying on third parties.

When comparing the efficiency of our $(M+1)$st-price auction protocol to existing ones (Table 1), it stands out that communication complexity is linear in n and k while there are other constant-round protocols, namely Naor et al.'s [36] and Lipmaa et al.'s [35], whose complexity is at most logarithmic in one of these parameters. However, these asymmetric protocols, as well as those based on threshold MPC, cannot be modified to obtain fully private *bidder-resolved* protocols. Additionally, our protocols are the only ones to provide public verifiability, non-repudiation, and constant-round complexity at the same time. Finally, in

Table 1 Cryptographic auction protocols

Protocol	Price	Auctioneers	Rounds	Computation	Verifiability	Non-Repudiation
S00 [44]	1st	m	$\mathcal{O}(k)$	$\mathcal{O}(nk)$	✓	✓
BS01 [2]	1st	1	$\mathcal{O}(1)$	$\mathcal{O}(n(\log k)^{n-1})$	–	–
HTK98 [27]	2nd	m	$\mathcal{O}(\log k)$	$\mathcal{O}(n \log k)$	–	✓
LAN02 [35]	$(M+1)$st	2	$\mathcal{O}(1)$	$\mathcal{O}(k)$	✓	–
NPS99 [36] JS02 [29]	$(M+1)^{\text{st}}$	2	$\mathcal{O}(1)$	$\mathcal{O}(n \log k)$	–	✓
K01 [32]	$(M+1)$st	$m > k$	$\mathcal{O}(\log k)$	$\mathcal{O}(n \log k)$	–	✓
AS02 [1]	$(M+1)$st	2	$\mathcal{O}(\log k)$	$\mathcal{O}(k)$	✓	✓
Proposed	1st	0	$\mathcal{O}(1)$	$\mathcal{O}(k)$	✓	✓
Proposed	$(M+1)$st	0	$\mathcal{O}(1)$	$\mathcal{O}(nk)$	✓	✓

Notes. n: bidders; k: prices/possible bids. In order to enable a fair comparison, we assume that M is constant and each protocol's security parameter is greater than the number of bidders n. In protocol LAN02, complete bid statistics are revealed to one party.

contrast to all existing protocols, we also present private outcome protocols in which losing bidders learn no information at all. Due to conceptual differences, the high degree of privacy provided by bidder-resolved auction protocols naturally leads to weaker resilience when compared to protocols with multiple auctioneers. For example, if the highest bidder refuses to pay, the entire protocol needs to be restarted to determine the bidder who submitted the second-highest bid. However, as argued in Sect. 2, bidder-resolved auctions are aimed at high-value transactions involving few non-anonymous bidders who can be fined when aborting the protocol. All of the protocols mentioned in the previous section rely on the existence of a broadcast channel. In contrast to protocols with multiple auctioneers, bidder-resolved protocols additionally require that *bidders* can use the broadcast channel.

This paper exclusively deals with auction emulation in the *computational* model. In a related study, it has been shown that the fully private emulation of second-price auctions in the *unconditional* model is impossible whereas first-price auctions can only be emulated in a number of rounds that is exponential in the bid size [8]. As a tradeoff between both models, we previously proposed a second-price auction protocol that is unconditionally anonymous and computationally private [5]. Besides first-price and $(M+1)$st-price auctions, there are also cryptographic protocols for certain types of combinatorial auctions (e.g., [10, 47, 48]).

Parallel to our work on fully private auction and social choice protocols (e.g., [4, 5, 7–9]), there is an independent, yet quite similar, stream of research on self-tallying elections [26, 30, 31]. In both settings, agents jointly determine the outcome of a social choice function without relying on trusted third parties. What we call "full privacy" is termed "perfect ballot secrecy" in Kiayias et al.'s work. Similarly, the terms "self-tallying" and "dispute-free" [30] can be translated to "bidder-resolved" and "weakly robust" [4], respectively. In order to achieve fairness, both approaches assume a weakly trusted party (a "dummy voter" and the auction seller, respectively). Besides these similarities, Kiayias et al.'s approach mainly differs in the emphasis of non-interactiveness (once the random-generating preprocessing phase is finished) while computing simpler outcome functions (e.g., the sum of input values).

4 Conceptional protocol description

In this section, we propose an abstract description of our auction protocols. No matter whether an arithmetic MPC scheme is based on verifiable secret sharing, homomorphic encryption, or other techniques, the addition of secret values can usually be performed very efficiently whereas the multiplication of secrets requires a high amount of communication resources. For this reason, our protocols only require the computation of *linear combinations* of inputs (which can exclusively be based on addition) and *multiplications with jointly created random values* (for which we propose an efficient sub-protocol in Sect. 5.1). When computing on *vectors* of secrets, the computation of linear combinations enables the addition and subtraction of secret vectors, and the multiplication of vectors with predefined known matrices.

Let $\mathbf{p}$ be a vector of k possible prices (or valuations), $\mathbf{p} = (p_1, p_2, \ldots, p_k)$, and $bid_i \in \{1, 2, \ldots, k\}$ the bid of bidder i. The *bid vector* $\mathbf{b}_i$ of bidder i is defined so that $b_{i,bid_i} = 1$ (bidder i bids p_{bid_i}) and all other components are 0, i.e.,

$$\mathbf{b}_i = \begin{pmatrix} b_{ik} \\ \vdots \\ b_{i,bid_i+1} \\ b_{i,bid_i} \\ b_{i,bid_i-1} \\ \vdots \\ b_{i1} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

This representation allows for efficient proofs of the vector's correctness by showing $\forall j \in \{1, 2, \ldots, k\} : \; b_{ij} \in \{0, 1\}$ and $\sum_{j=1}^{k} b_{ij} = 1$ (see Sect. 5 for details). Yet, the main advantage of the vector representation is the possibility to

efficiently perform certain computations. For example, the "integrated" bid vector (a notion introduced by Abe and Suzuki [1])

$$\mathbf{b}'_i = \begin{pmatrix} b'_{ik} \\ \vdots \\ b'_{i,bid_i+1} \\ b'_{i,bid_i} \\ b'_{i,bid_i-1} \\ \vdots \\ b'_{i1} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}$$

can be derived by multiplying the bid vector with the $k \times k$ lower triangular matrix L_k, that is $\mathbf{b}'_i = \mathsf{L}_k \mathbf{b}_i$ where

$$\mathsf{L}_\ell = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ 1 & \cdots & \cdots & 1 \end{pmatrix}$$

and $\ell \in \{k, n\}$. Further $\ell \times \ell$ matrices that we will use in the following sections are the upper triangular matrix U_ℓ, the identity matrix I_ℓ, and random multiplication matrices R^*_ℓ:

$$\mathsf{U}_\ell = \begin{pmatrix} 1 & \cdots & \cdots & 1 \\ 0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 \end{pmatrix}, \quad \mathsf{I}_\ell = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix},$$

$$\mathsf{R}^*_\ell = \begin{pmatrix} * & 0 & \cdots & 0 \\ 0 & * & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & * \end{pmatrix}.$$

The components on the diagonal of R^*_ℓ are random numbers unknown to the agents. They are jointly created using a special sub-protocol to be proposed in Sect. 5.1. Multiplication with R^*_ℓ turns all vector components *that are not zero* into meaningless random numbers. For this reason, it is usually the final key step in our protocols. Please note that matrices are only used to facilitate the presentation. The special structure of all used matrices allows us to compute matrix–vector multiplications using only k multiplications.

The price we pay for round-efficiency enabled by this unary representation is communication and computation complexity that is linear in the number of different prices k. On the other hand, the unary representation allows us to easily adapt the given protocols to emulate *iterative* (e.g., ascending-price or descending-price) auctions in which bidders gradually express their willingness to pay for sequences of prices. In fact, there are common iterative correspondences for both auction types considered in this paper: The first-price auction is strategically equivalent to the Dutch (or descending-price) auction, and the second-price auction corresponds to the well-known English (or ascending-price) auction as used at eBay or Sotheby's. Sometimes, iterative auctions are preferred over sealed-bid auctions (see Sect. 7).

4.1 First-price auctions

In this section, we propose a multiparty computation scheme for first-price auctions. With respect to the framework presented in Sect. 2, we specify function $f_a^1(\mathbf{b})$ to yield the selling price, i.e., the highest bid, if and only if bidder a is the winner of the auction.

We can obtain a vector in which all components referring to prices that are greater or equal than the maximum bid are zero by summing up all integrated bid vectors and shifting the result down by one position:

$$(\mathsf{L}_k - \mathsf{I}_k) \sum_{i=1}^{n} \mathbf{b}_i .$$

By adding vector $(\mathsf{U}_k - \mathsf{I}_k)\mathbf{b}_a$, a single component remains zero if and only if bidder a bid at least as much as the maximum bid, and thus qualifies as an auction winner. This component indicates the selling price. After multiplying with the secret random multiplication matrix, vector

$$\left((\mathsf{L}_k - \mathsf{I}_k) \sum_{i=1}^{n} \mathbf{b}_i + (\mathsf{U}_k - \mathsf{I}_k)\mathbf{b}_a \right) \mathsf{R}^*_k$$

only consists of random values if bidder a did not submit the highest bid. Otherwise, the position of the sole zero indicates the selling price (to the seller, since bidder a already knows his bid). Again, all remaining components are random values.

Example 1 Consider just two bidders and let the vector of possible prices be $\mathbf{p} = (10, 20, 30, 40, 50, 60)$. The first bid is 20 ($b_1 = 2$) and the second one is 50 ($b_2 = 5$). Asterisks denote arbitrary random numbers that have no meaning to bidders.

$$\begin{aligned} &\left((\mathsf{L}_6 - \mathsf{I}_6) \sum_{i=1}^{2} \mathbf{b}_i + (\mathsf{U}_6 - \mathsf{I}_6)\mathbf{b}_1 \right) \mathsf{R}^*_6 \\ &= \left(\begin{pmatrix} 0&0&0&0&0&0 \\ 1&0&0&0&0&0 \\ 1&1&0&0&0&0 \\ 1&1&1&0&0&0 \\ 1&1&1&1&0&0 \\ 1&1&1&1&1&0 \end{pmatrix} \left(\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \right) + \begin{pmatrix} 0&1&1&1&1&1 \\ 0&0&1&1&1&1 \\ 0&0&0&1&1&1 \\ 0&0&0&0&1&1 \\ 0&0&0&0&0&1 \\ 0&0&0&0&0&0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \right) \mathsf{R}^*_6 \end{aligned}$$

$$= \left(\begin{pmatrix} 0\\0\\1\\1\\1\\2 \end{pmatrix} + \begin{pmatrix} 1\\1\\1\\1\\0\\0 \end{pmatrix} \right) \mathsf{R}_6^* = \begin{pmatrix} 1\\1\\2\\2\\1\\2 \end{pmatrix} \mathsf{R}_6^* = \begin{pmatrix} *\\*\\*\\*\\*\\* \end{pmatrix}$$

$$\left((\mathsf{L}_6 - \mathsf{I}_6) \sum_{i=1}^{2} \mathbf{b}_i + (\mathsf{U}_6 - \mathsf{I}_6)\mathbf{b}_2 \right) \mathsf{R}_6^*$$

$$= \left(\begin{pmatrix} 0&0&0&0&0&0\\1&0&0&0&0&0\\1&1&0&0&0&0\\1&1&1&0&0&0\\1&1&1&1&0&0\\1&1&1&1&1&0 \end{pmatrix} \left(\begin{pmatrix} 0\\0\\0\\1\\0 \end{pmatrix} + \begin{pmatrix} 0\\1\\0\\0\\0 \end{pmatrix} \right) + \begin{pmatrix} 0&1&1&1&1&1\\0&0&1&1&1&1\\0&0&0&1&1&1\\0&0&0&0&1&1\\0&0&0&0&0&1\\0&0&0&0&0&0 \end{pmatrix} \begin{pmatrix} 0\\1\\0\\0\\0 \end{pmatrix} \right) \mathsf{R}_6^*$$

$$= \left(\begin{pmatrix} 0\\0\\1\\1\\1\\2 \end{pmatrix} + \begin{pmatrix} 1\\0\\0\\0\\0\\0 \end{pmatrix} \right) \mathsf{R}_6^* = \begin{pmatrix} 1\\0\\1\\1\\1\\2 \end{pmatrix} \mathsf{R}_6^* = \begin{pmatrix} *\\0\\*\\*\\*\\* \end{pmatrix}$$

Bidder 1 gains no information at all and bidder 2 only learns that he won. The seller receives both resulting vectors and thus learns the winner's identity (2) and the selling price (50).

4.1.1 Tie-breaking

So far, if there is more than one winning bid, the protocol reveals the identities of all winners (leaving it to the seller how to proceed in the case of ties). However, it is possible to modify the protocol to just yield the winner with the lowest index. This can be used in conjunction with a setup phase in which bidders who sign up for the auction are assigned consecutive numbers as indices, creating an incentive to sign up early. Alternatively, the indices of bidders could be permuted randomly (by letting each bidder commit to a permutation of indices) before the auction begins in order to obtain fairness. In either case, an additional vector, that hides zeros to all agents except the winning bidder *with the lowest index*, is added. Let

$$\mathbf{u}_j = \begin{pmatrix} b_{1j}\\b_{2j}\\\vdots\\b_{nj} \end{pmatrix} \quad \text{and}$$

$$\mathsf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n) = \begin{pmatrix} ((\mathsf{L}_n - \mathsf{I}_n)\mathbf{u}_k)^{\mathrm{T}}\\((\mathsf{L}_n - \mathsf{I}_n)\mathbf{u}_{k-1})^{\mathrm{T}}\\\vdots\\((\mathsf{L}_n - \mathsf{I}_n)\mathbf{u}_1)^{\mathrm{T}} \end{pmatrix}.$$

X is a $k \times n$ matrix consisting of n vectors $\mathbf{x}_i$. Finally, the complete first-price auction outcome function is defined as

$$f_a^1(\mathbf{b}) = \left((\mathsf{L}_k - \mathsf{I}_k) \sum_{i=1}^{n} \mathbf{b}_i + (\mathsf{U}_k - \mathsf{I}_k)\mathbf{b}_a + \mathbf{x}_a \right) \mathsf{R}_k^*. \quad (1)$$

Example 2 Consider three bidders and let the vector of possible prices be $\mathbf{p} = (10, 20, 30, 40, 50, 60)$ as above. The first bid is 20 ($b_1 = 2$), the second and the third are both 50 ($b_2 = b_3 = 5$).

$$\mathsf{X} = \begin{pmatrix} \left((\mathsf{L}_3 - \mathsf{I}_3) \begin{pmatrix} 0\\0\\0 \end{pmatrix} \right)^{\mathrm{T}}\\ \left((\mathsf{L}_3 - \mathsf{I}_3) \begin{pmatrix} 0\\1\\1 \end{pmatrix} \right)^{\mathrm{T}}\\ \left((\mathsf{L}_3 - \mathsf{I}_3) \begin{pmatrix} 0\\0\\0 \end{pmatrix} \right)^{\mathrm{T}}\\ \left((\mathsf{L}_3 - \mathsf{I}_3) \begin{pmatrix} 0\\0\\0 \end{pmatrix} \right)^{\mathrm{T}}\\ \left((\mathsf{L}_3 - \mathsf{I}_3) \begin{pmatrix} 1\\0\\0 \end{pmatrix} \right)^{\mathrm{T}}\\ \left((\mathsf{L}_3 - \mathsf{I}_3) \begin{pmatrix} 0\\0\\0 \end{pmatrix} \right)^{\mathrm{T}} \end{pmatrix} = \begin{pmatrix} 0&0&0\\0&0&1\\0&0&0\\0&0&0\\0&1&1\\0&0&0 \end{pmatrix}$$

Only bidder 2 learns that he won the auction.

$$f_1^1(\mathbf{b}) = \left(\begin{pmatrix} 0&0&0&0&0&0\\1&0&0&0&0&0\\1&1&0&0&0&0\\1&1&1&0&0&0\\1&1&1&1&0&0\\1&1&1&1&1&0 \end{pmatrix} \begin{pmatrix} 0\\2\\0\\0\\1\\0 \end{pmatrix} + \begin{pmatrix} 0&1&1&1&1&1\\0&0&1&1&1&1\\0&0&0&1&1&1\\0&0&0&0&1&1\\0&0&0&0&0&1\\0&0&0&0&0&0 \end{pmatrix} \begin{pmatrix} 0\\0\\0\\0\\1\\0 \end{pmatrix} + \begin{pmatrix} 0\\0\\0\\0\\0\\0 \end{pmatrix} \right) \mathsf{R}_6^* = \begin{pmatrix} *\\*\\*\\*\\*\\* \end{pmatrix}$$

$$f_2^1(\mathbf{b}) = \left(\begin{pmatrix} 0&0&0&0&0&0\\ 1&0&0&0&0&0\\ 1&1&0&0&0&0\\ 1&1&1&0&0&0\\ 1&1&1&1&0&0\\ 1&1&1&1&1&0 \end{pmatrix} \begin{pmatrix} 0\\2\\0\\0\\1\\0 \end{pmatrix} + \begin{pmatrix} 0&1&1&1&1&1\\ 0&0&1&1&1&1\\ 0&0&0&1&1&1\\ 0&0&0&0&1&1\\ 0&0&0&0&0&1\\ 0&0&0&0&0&0 \end{pmatrix} \begin{pmatrix} 0\\1\\0\\0\\0\\0 \end{pmatrix} + \begin{pmatrix} 0\\0\\0\\0\\1\\0 \end{pmatrix} \right) \mathsf{R}_6^* = \begin{pmatrix} *\\0\\*\\*\\*\\* \end{pmatrix}$$

$$f_3^1(\mathbf{b}) = \left(\begin{pmatrix} 0&0&0&0&0&0\\ 1&0&0&0&0&0\\ 1&1&0&0&0&0\\ 1&1&1&0&0&0\\ 1&1&1&1&0&0\\ 1&1&1&1&1&0 \end{pmatrix} \begin{pmatrix} 0\\2\\0\\0\\1\\0 \end{pmatrix} + \begin{pmatrix} 0&1&1&1&1&1\\ 0&0&1&1&1&1\\ 0&0&0&1&1&1\\ 0&0&0&0&1&1\\ 0&0&0&0&0&1\\ 0&0&0&0&0&0 \end{pmatrix} \begin{pmatrix} 0\\1\\0\\0\\0\\0 \end{pmatrix} + \begin{pmatrix} 0\\1\\0\\0\\1\\0 \end{pmatrix} \right) \mathsf{R}_6^* = \begin{pmatrix} *\\*\\*\\*\\*\\* \end{pmatrix}$$

4.1.2 Public outcome

For reasons of efficiency and transparency, it might be desirable to compute the auction outcome so that *all* bidders learn the selling price. In this case, there is just one outcome function (instead of n):

$$f^1(\mathbf{b}) = \left((\mathsf{L}_k - \mathsf{I}_k) \sum_{i=1}^{n} \mathbf{b}_i \right) \mathsf{R}_k^* + \sum_{i=1}^{n} 2^{i-1} \mathbf{b}_i . \tag{2}$$

Similar to the construction used in the previous section, the first component of $f^1(\mathbf{b})$ that is not equal to zero denotes the selling price. Bidder i qualifies as a winner of the auction if the $(i-1)$th bit of this component is set. Please note that our tie-breaking technique is not applicable in this setting because the identities of all winning bidders are squeezed into the same component which rules out the masking of identities as described in the previous section.

4.2 $(M+1)$st-price auctions

In this section, we will apply the techniques used so far to obtain the outcome of $(M+1)$st-price auctions. The construction of a vector (by using linear combinations of $\mathbf{b}_i$) in which only the $(M+1)$st-highest bid is marked by a zero turns out to be more complicated than the marking of the highest bid in the previous section. Let us for now assume that there is always a *single* $(M+1)$st-highest bid and let

$$\mathbf{e} = \begin{pmatrix} 1\\ \vdots\\ 1 \end{pmatrix}$$

be the k-dimensional unit vector.

$$(2\mathsf{L}_k - \mathsf{I}_k) \sum_{i=1}^{n} \mathbf{b}_i - (2M+1)\mathbf{e}$$

yields a vector in which all components except the one denoting the $(M+1)$st-highest bid are not zero. Computing $(2\mathsf{L}_k - \mathsf{I}_k) \sum_{i=1}^{n} \mathbf{b}_i$ is equivalent to adding all integrated bid vectors and down-shifted integrated bid vectors. This results in a strictly increasing sequence of components where 1 denotes the highest bid, 3 denotes the second-highest bid, 5 the third-highest, and so forth. Subtracting $(2M+1)\mathbf{e}$ yields a vector in which the component denoting the $(M+1)$st-highest bid is zero.

When masking this information to losing bidders, like in the previous section, each bidder's personal outcome vector is

$$price_a^{M+1}(\mathbf{b}) = \left((2\mathsf{L}_k - \mathsf{I}_k) \sum_{i=1}^{n} \mathbf{b}_i - (2M+1)\,\mathbf{e} + (2M+2)\mathsf{U}_k \mathbf{b}_a \right) \mathsf{R}_k^* .$$

Factor $(2M+2)$ ensures that no additional zeros turn up "accidently". If vector $price_a^{M+1}(\mathbf{b})$ contains a zero, then bidder a qualifies as a winner of the auction. The position of the zero indicates the selling price. All other components are random values.

Example 3 Consider a Vickrey auction ($M=1$) with the same price vector and bids as in the previous example ($\mathbf{p} = (10, 20, 30, 40, 50, 60)$, $b_1 = 2$, and $b_2 = 5$). All computations take place in the finite field $\mathbb{Z}_{11}$.

$$price_1^2(\mathbf{b}) = \left(\begin{pmatrix} 1&0&0&0&0&0\\ 2&1&0&0&0&0\\ 2&2&1&0&0&0\\ 2&2&2&1&0&0\\ 2&2&2&2&1&0\\ 2&2&2&2&2&1 \end{pmatrix} \left(\begin{pmatrix} 0\\0\\0\\0\\1\\0 \end{pmatrix} + \begin{pmatrix} 0\\1\\0\\0\\0\\0 \end{pmatrix} \right) - \begin{pmatrix} 3\\3\\3\\3\\3\\3 \end{pmatrix} + 4 \begin{pmatrix} 1&1&1&1&1&1\\ 0&1&1&1&1&1\\ 0&0&1&1&1&1\\ 0&0&0&1&1&1\\ 0&0&0&0&1&1\\ 0&0&0&0&0&1 \end{pmatrix} \begin{pmatrix} 0\\0\\0\\0\\1\\0 \end{pmatrix} \right) \mathsf{R}_6^*$$

$$= \left(\begin{pmatrix} 0\\1\\2\\2\\3\\4 \end{pmatrix} - \begin{pmatrix} 3\\3\\3\\3\\3\\3 \end{pmatrix} + \begin{pmatrix} 4\\4\\4\\4\\4\\0 \end{pmatrix} \right) \mathsf{R}_6^* = \begin{pmatrix} 1\\2\\3\\3\\4\\1 \end{pmatrix} \mathsf{R}_6^* = \begin{pmatrix} *\\*\\*\\*\\*\\* \end{pmatrix}$$

$$price_2^2(\mathbf{b}) = \left(\begin{pmatrix} 1&0&0&0&0&0\\2&1&0&0&0&0\\2&2&1&0&0&0\\2&2&2&1&0&0\\2&2&2&2&1&0\\2&2&2&2&2&1 \end{pmatrix} \left(\begin{pmatrix} 0\\0\\0\\0\\1\\0 \end{pmatrix} + \begin{pmatrix} 0\\1\\0\\0\\0\\0 \end{pmatrix} \right) - \begin{pmatrix} 3\\3\\3\\3\\3\\3 \end{pmatrix} + 4 \begin{pmatrix} 1&1&1&1&1&1\\0&1&1&1&1&1\\0&0&1&1&1&1\\0&0&0&1&1&1\\0&0&0&0&1&1\\0&0&0&0&0&1 \end{pmatrix} \begin{pmatrix} 0\\1\\0\\0\\0\\0 \end{pmatrix} \right) \mathsf{R}_6^*$$

$$= \left(\begin{pmatrix} 0\\1\\2\\2\\3\\4 \end{pmatrix} - \begin{pmatrix} 3\\3\\3\\3\\3\\3 \end{pmatrix} + \begin{pmatrix} 4\\4\\0\\0\\0\\0 \end{pmatrix} \right) \mathsf{R}_6^* = \begin{pmatrix} 1\\2\\10\\10\\0\\1 \end{pmatrix} \mathsf{R}_6^* = \begin{pmatrix} *\\*\\*\\*\\0\\* \end{pmatrix}$$

Bidder 1's outcome vector, $price_1^2(\mathbf{b})$, contains random numbers, whereas $price_2^2(\mathbf{b})$ indicates the selling price: 20.

When two or more bidders have the $(M+1)$st-highest bid in common, the technique described above does not work (no outcome vector contains a zero). For this reason, agents compute additional vectors for each bidder that yield the correct outcome in the case of these ties. The following method marks the $(M+1)$st-highest bid while not revealing any information about other ties. $\sum_{i=1}^n \mathbf{b}_i - t\mathbf{e}$ is a vector that contains zeros if there is a tie, i.e., t bidders share the same bid at the corresponding position ($1 < t \leq n$). We are only interested in ties involving the $(M+1)$st-highest bid because only these ties cause the techniques for computing $price_a^{M+1}(\mathbf{b})$ to fail. Other ties are masked by adding $(n+1)\big(\mathsf{L}_k \sum_{i=1}^n \mathbf{b}_i - (t+u)\mathbf{e}\big)$ where $u \in \{\max(0, M+1-t), \ldots, \min(M, n-t)\}$ for each t. The resulting vector contains a zero when t bids are equal and there are u bids higher than the tie. The preceding factor $(n+1)$ is large enough to ensure that both addends do not add up to zero. The term we have so far can be rewritten as follows:

$$\sum_{i=1}^n \mathbf{b}_i - t\mathbf{e} + (n+1)\left(\mathsf{L}_k \sum_{i=1}^n \mathbf{b}_i - (t+u)\mathbf{e}\right) = ((n+1)\mathsf{L}_k + \mathsf{I}_k) \sum_{i=1}^n \mathbf{b}_i - ((n+1)(t+u)+t)\mathbf{e}$$

Finally, the position of the tie (which is the selling price) has to be made invisible to losing bidders like before. This can be done by adding $(n+1)^2 \mathsf{U}_k \mathbf{b}_a$ or $(n+1)^2(\mathsf{U}_k - \mathsf{I}_k)\mathbf{b}_a$ depending on u. After all, agents need to compute the following additional outcome vectors:

$$pricetie_{atu}^{M+1}(\mathbf{b}) = \Big(((n+1)\mathsf{L}_k + \mathsf{I}_k) \sum_{i=1}^n \mathbf{b}_i - ((n+1)(t+u)+t)\mathbf{e} + (n+1)^2 \mathsf{U}_k \mathbf{b}_a\Big) \mathsf{R}_k^* \quad \text{if } u = M$$

$$pricetie_{atu}^{M+1}(\mathbf{b}) = \Big(((n+1)\mathsf{L}_k + \mathsf{I}_k) \sum_{i=1}^n \mathbf{b}_i - ((n+1)(t+u)+t)\mathbf{e} + (n+1)^2 (\mathsf{U}_k - \mathsf{I}_k)\mathbf{b}_a\Big) \mathsf{R}_k^* \quad \text{otherwise.}$$

Example 4 Suppose we have the following compilation of bids ($M = 1$, computation takes place in $\mathbb{Z}_{11}$ and $\mathbf{p} = (10, 20, 30, 40, 50, 60)$):

$$\mathbf{b}_1 = \begin{pmatrix} 0\\1\\0\\0\\0\\0 \end{pmatrix}, \quad \mathbf{b}_2 = \begin{pmatrix} 0\\1\\0\\0\\0\\0 \end{pmatrix}, \quad \mathbf{b}_3 = \begin{pmatrix} 0\\0\\0\\1\\0\\0 \end{pmatrix}, \quad \mathbf{b}_4 = \begin{pmatrix} 0\\0\\0\\1\\0\\0 \end{pmatrix}$$

The "regular" outcome function $price_a^{M+1}$ yields no outcome due to the tie at price 50. The first two ($t = 2, u \in \{0, 1\}$) additional outcome vectors look like this (before being masked for each bidder):

$$\left(\sum_{i=1}^4 \mathbf{b}_i - 2\mathbf{e} + 5\left(\mathsf{L}_6 \sum_{i=1}^4 \mathbf{b}_i - (2+0)\mathbf{e}\right)\right) \mathsf{R}_6^*$$

$$= \left(\begin{pmatrix} 0 \\ 2 \\ 0 \\ 2 \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{pmatrix} + 5 \left(\begin{pmatrix} 0 \\ 2 \\ 2 \\ 4 \\ 4 \\ 4 \end{pmatrix} - \begin{pmatrix} 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{pmatrix} \right) \right) \mathsf{R}_6^* = \begin{pmatrix} 10 \\ 0 \\ 9 \\ 10 \\ 8 \\ 8 \end{pmatrix} \mathsf{R}_6^* = \begin{pmatrix} * \\ 0 \\ * \\ * \\ * \\ * \end{pmatrix}$$

$$\left(\sum_{i=1}^{4} \mathbf{b}_i - 2\mathbf{e} + 5 \left(\mathsf{L}_6 \sum_{i=1}^{4} \mathbf{b}_i - (2+1)\mathbf{e} \right) \right) \mathsf{R}_6^*$$

$$= \left(\begin{pmatrix} 0 \\ 2 \\ 0 \\ 2 \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{pmatrix} + 5 \left(\begin{pmatrix} 0 \\ 2 \\ 2 \\ 4 \\ 4 \\ 4 \end{pmatrix} - \begin{pmatrix} 3 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3 \end{pmatrix} \right) \right) \mathsf{R}_6^* = \begin{pmatrix} 5 \\ 6 \\ 4 \\ 5 \\ 3 \\ 3 \end{pmatrix} \mathsf{R}_6^* = \begin{pmatrix} * \\ * \\ * \\ * \\ * \\ * \end{pmatrix}$$

For $t > 2$ the first difference contains no zeros, leading to random vectors. Nevertheless, all of these vectors need to be computed at the same time in order to minimize round complexity. After masking the outcome to losing bidders, only bidders 1 and 2 will be able to read the selling price (50).

Concluding, in order to obtain the outcome of an $(M+1)$st-price auction, agents jointly compute function

$$f_a^{M+1}(\mathbf{b}) = \left(price_a^{M+1}(\mathbf{b}), \left(pricetie_{atu}^{M+1}(\mathbf{b}) \right)_{\forall t,u} \right) \tag{3}$$

where $t = \{2, 3, \ldots, n\}$ and $u \in \{\max(0, M+1-t), \ldots, \min(M, n-t)\}$ for each t. Thus, a total amount of $(M+1)(n-(M+1))$ vectors of size k needs to be computed.

4.2.1 Tie-breaking

Even though ties involving the $(M+1)$st-highest bid cannot prevent the computation scheme from succeeding in determining the outcome, the scheme will still yield several winners when there are ties involving the *highest* bid (and thus also the second-highest bid). The tie-breaking method introduced in Sect. 4.1.1 can only be transferred to $(M+1)$st-price auctions when there is a single winner, i.e., $M = 1$ (the auction to be conducted is a Vickrey auction). Similar to the procedure in Sect. 4.1.1, additional outcome vector $pricetie_{at0}^2(\mathbf{b})$ can be modified to only yield the winner with the lowest index by adding vector $\mathbf{x}_a$. Thus,

$$pricetie_{at0}^2(\mathbf{b}) = \left(((n+1)\mathsf{L}_k + \mathsf{I}_k) \sum_{i=1}^{n} \mathbf{b}_i - \big((n+2)t\big)\mathbf{e} + (n+1)^2((\mathsf{U}_k - \mathsf{I}_k)\mathbf{b}_a + \mathbf{x}_a) \right) \mathsf{R}_k^*.$$

The remaining outcome vectors ($price_a^2(\mathbf{b})$ and $pricetie_{at1}^2(\mathbf{b})$) do not need to be masked.

4.2.2 Public price

Similar to Sect. 4.1.2, the protocol's efficiency can be improved by just computing public outcome vectors so that all agents, including losing bidders, learn the outcome. Like before, winning bidders can be identified by computing just two outcome vectors (for each t and u) so that one denotes the selling price and the other contains the winner's identities (as a base-2 number). Please note that, as in the first-price auction scheme, tie-breaking is not possible when using this technique. The public price outcome vectors are

$$price^{M+1}(\mathbf{b}) = \left((2\mathsf{L}_k - \mathsf{I}_k) \sum_{i=1}^{n} \mathbf{b}_i - (2M+1)\mathbf{e} \right) \mathsf{R}_k^*,$$

and

$$pricetie_{tu}^{M+1}(\mathbf{b}) = \left(((n+1)\mathsf{L}_k + \mathsf{I}_k) \sum_{i=1}^{n} \mathbf{b}_i - ((n+1)(t+u)+t)\mathbf{e} \right) \mathsf{R}_k^*$$

for the selling price, and

$$winner^{M+1}(\mathbf{b}) = \left((2\mathsf{L}_k - \mathsf{I}_k) \sum_{i=1}^{n} \mathbf{b}_i - (2M+1)\mathbf{e} \right) \mathsf{R}_k^* + (\mathsf{L}_k - \mathsf{I}_k) \sum_{i=1}^{n} 2^{i-1}\mathbf{b}_i$$

and

$$winnertie_{tu}^{M+1}(\mathbf{b}) = \left(((n+1)\mathsf{L}_k + \mathsf{I}_k) \sum_{i=1}^{n} \mathbf{b}_i - ((n+1)(t+u)+t)\mathbf{e} \right) \mathsf{R}_k^* + (\mathsf{L}_k - \mathsf{I}_k) \sum_{i=1}^{n} 2^{i-1}\mathbf{b}_i \quad \text{if } u = M$$

$$winnertie_{tu}^{M+1}(\mathbf{b}) = \Bigg(\Big((n+1)\mathsf{L}_k + \mathsf{l}_k\Big)\sum_{i=1}^{n}\mathbf{b}_i - \Big((n+1)(t+u)+t\Big)\mathbf{e}\Bigg)\mathsf{R}_k^* + \mathsf{L}_k\sum_{i=1}^{n}2^{i-1}\mathbf{b}_i \quad \text{otherwise}$$

for the winners' identities. Thus, the jointly computed outcome function is

$$f^{M+1}(\mathbf{b}) = \big(price^{M+1}(\mathbf{b}), winner^{M+1}(\mathbf{b}), (pricetie_{tu}^{M+1}(\mathbf{b}), winnertie_{tu}^{M+1}(\mathbf{b}))_{\forall t,u}\big). \tag{4}$$

5 Implementation using homomorphic encryption

Any homomorphic encryption scheme that besides the, say, additive homomorphic operation allows multiplication of encrypted values with a jointly generated random number can be used to implement the schemes described in the previous sections. It turns out that El Gamal encryption [17], even though it is multiplicative, is quite suitable because

- agents can easily create distributed keys, and
- encrypted values can be exponentiated with a shared random number in a single round.

As El Gamal cipher is a multiplicative homomorphic encryption scheme, the entire computation as described in the previous sections will be executed in the exponent of a generator. In other words, random exponentiation implements the random multiplication of the additive notation. As a consequence, the selling price is marked by ones instead of zeros in outcome vectors.

5.1 El Gamal encryption

El Gamal cipher [17] is a probabilistic and homomorphic public-key cryptosystem. Let p and q be large primes so that q divides $p-1$ and $\mathbb{G}_q$ denote $\mathbb{Z}_p^*$'s unique multiplicative subgroup of order q. We will focus on multiplicative subgroups of finite fields here, although El Gamal can also be based on other groups such as elliptic curve groups. All computations in the remainder of this paper are modulo p unless otherwise noted. The *private key* is $x \in \mathbb{Z}_q$, the *public key* $y = g^x$ ($g \in \mathbb{G}_q$ is an arbitrary, publicly known element). A message $m \in \mathbb{G}_q$ is *encrypted* by computing the ciphertext tuple

$$(\alpha, \beta) = (my^r, g^r)$$

where r is an arbitrary random number in $\mathbb{Z}_q$, chosen by the encrypter. A message is *decrypted* by computing

$$\frac{\alpha}{\beta^x} = \frac{my^r}{(g^r)^x} = m.$$

El Gamal is homomorphic as the component-wise product of two ciphertexts $(\alpha\alpha', \beta\beta') = (mm'y^{r+r'}, g^{r+r'})$ represents an encryption of the plaintexts' product mm'. It has been shown that El Gamal is semantically secure, i.e., it is computationally infeasible to distinguish between the encryptions of any two given messages, if the decisional Diffie–Hellman problem is intractable [49].

We will now describe how to apply the El Gamal cryptosystem as a fully private multiparty computation scheme. Please note that this multiparty scheme is limited in the sense that it does not allow the computation of *arbitrary* functions. If a value represents an additive share, this is denoted by a "+" in the index, whereas multiplicative shares are denoted by "×". Underlying zero-knowledge proofs will be presented in the next section.

Distributed key generation [39] Each participant chooses x_{+i} at random and publishes $y_{\times i} = g^{x_{+i}}$ along with a zero-knowledge proof of knowledge of $y_{\times i}$'s discrete logarithm. The public key is $y = \prod_{i=1}^{n} y_{\times i}$, the private key is $x = \sum_{i=1}^{n} x_{+i}$. This requires n multiplications, but the computational cost of multiplications is usually negligible in contrast to exponentiations. Broadcast round complexity and exponentiation complexity of the key generation are $\mathcal{O}(1)$.

Distributed decryption Given an encrypted message (α, β), each participant publishes $\beta_{\times i} = \beta^{x_{+i}}$ and proves its correctness by showing the equality of logarithms of $y_{\times i}$ and $\beta_{\times i}$. The plaintext can be derived by computing $\frac{\alpha}{\prod_{i=1}^{n}\beta_{\times i}}$. Like key generation, decryption can be performed in a constant number of rounds, requiring n multiplications and one exponentiation.

Random exponentiation A given encrypted value (α, β) can easily be raised to the power of an unknown random number $M = \sum_{i=1}^{n} m_{+i}$ whose addends can be freely chosen by the participants if each bidder publishes $(\alpha^{m_{+i}}, \beta^{m_{+i}})$ and proves the equality of logarithms. The product of published ciphertexts yields (α^M, β^M) in a single step. The computational cost is two exponentiations and $2n$ multiplications.

5.2 Zero-knowledge proofs

In order to obtain security against *malicious* or so-called *active* adversaries, bidders are required to prove the correctness of each protocol step. One of the objectives when designing the protocols presented in Sect. 4 was to enable *efficient* proofs of correctness for all protocol steps. In fact, the proposed protocols can be proven correct by only using so-called Σ-protocols which just need three rounds of interaction [14, 16]. Σ-protocols are not known to be zero-knowledge, but they satisfy the weaker property of *honest-verifier* zero-knowledge. This suffices for our purposes as we can use the Fiat–Shamir heuristic [18] to make these proofs non-interactive. As a consequence, the obtained proofs are indeed zero-knowledge *in the random oracle model* and only consist of a single message. It has become

common practice to use secure hash functions like MD5 or SHA-1 as random oracles for practical applications. We will make use of the following three non-malleable Σ-protocols.

5.2.1 Proof of knowledge of a discrete logarithm

This is a classic Σ-protocol by Schnorr [46]. Alice and Bob know v and g, but only Alice knows x, so that $v = g^x$. She can prove this fact, without revealing x, by executing the following protocol:

1. Alice chooses z at random and sends $a = g^z$ to Bob.
2. Bob chooses a challenge c at random and sends it to Alice.
3. Alice sends $r = (z + cx) \mod q$ to Bob.
4. Bob checks that $g^r = av^c$.

Alice needs to send $\log p + \log q$ bits.

5.2.2 Proof of equality of two discrete logarithms

When executing the previous protocol in parallel, the equality of two discrete logarithms can be proven [12]. Alice and Bob know v, w, g_1, and g_2, but only Alice knows x, so that $v = g_1^x$ and $w = g_2^x$.

1. Alice chooses z at random and sends $a = g_1^z$ and $b = g_2^z$ to Bob.
2. Bob chooses a challenge c at random and sends it to Alice.
3. Alice sends $r = (z + cx) \mod q$ to Bob.
4. Bob checks that $g_1^r = av^c$ and that $g_2^r = bw^c$.

Alice needs to send $2\log p + \log q$ bits. It is possible to show the equality of any polynomial number of discrete logarithms in parallel. Thus, for showing that the discrete logarithms of n values are equal, Alice only sends $n \log p + \log q$ bits.

5.2.3 Proof that an encrypted value is one out of two values

The following protocol was proposed by Cramer et al. [15]. Alice proves that an El Gamal encrypted value $(\alpha, \beta) = (my^r, g^r)$ either decrypts to 1 or to a fixed value $z \in \mathbb{G}_q$ without revealing which is the case, in other words, it is shown that $m = \{1, z\}$.

1. If $m = 1$, Alice chooses r_1, d_1, w at random and sends (α, β), $a_1 = g^{r_1}\beta^{d_1}$, $b_1 = y^{r_1}(\frac{\alpha}{z})^{d_1}$, and $a_2 = g^w$, $b_2 = y^w$ to Bob.
   If $m = z$, Alice chooses r_2, d_2, w at random and sends (α, β), $a_1 = g^w$, $b_1 = y^w$, $a_2 = g^{r_2}\beta^{d_2}$, and $b_2 = y^{r_2}\alpha^{d_2}$ to Bob.
2. Bob chooses a challenge c at random and sends it to Alice.
3. If $m = 1$, Alice sends d_1, $d_2 = c - d_1 \mod q$, r_1, and $r_2 = w - rd_2 \mod q$ to Bob.
   If $m = z$, Alice sends $d_1 = c - d_2 \mod q$, d_2, $r_1 = w - rd_1 \mod q$, and r_2 to Bob.
4. Bob checks that $c = d_1 + d_2 \mod q$, $a_1 = g^{r_1}\beta^{d_1}$, $b_1 = y^{r_1}(\frac{\alpha}{z})^{d_1}$, $a_2 = g^{r_2}\beta^{d_2}$, and $b_2 = y^{r_2}\alpha^{d_2}$.

The total amount of bits Alice sends to Bob is $4\log p + 4\log q$.

5.3 Protocol implementation

We are now ready to give a detailed protocol description. Let Y be an arbitrary value in $\mathbb{G}_q \backslash \{1\}$ that is known to all bidders, e.g., g. The computation schemes described in Sect. 4 can be executed in the exponent of Y. Please note that generating "unbiased" parameters p, q, and g requires no extra communication in the random oracle model.

Figure 1 shows the first-price auction protocol rules for an arbitrary bidder a. First, all bidders jointly generate a public El Gamal key as described in Sect. 5.1. Then, each bidder publishes his encrypted bid vector (see Sect. 4) and proves its correctness by showing that (i) each component is either Y or 1, and (ii) the product of all components is Y. In the second round of the protocol, bidders compute the first-price outcome function $f_a^1(\mathbf{b})$ as defined in Eq. (1) in Sect. 4.1.1. Each inner parentheses of $\gamma_{ij}^{\times a}$ and $\delta_{ij}^{\times a}$ in Fig. 1 relates to each addend in Eq. (1). Finally, the nk outcome values (k for each bidder) are jointly decrypted so that bidder a only learns whether he won the auction. Once the keys are generated, the protocol only requires three rounds of bidder broadcasting.[3] The implementation of an $(M+1)$st-price auction protocol works similarly (see [6]).

Figure 2 illustrates the modus operandi of the proposed protocols in a more general sense. The seller broadcasts the type of good to be sold, the number of units M, a deadline, and the price vector $\mathbf{p}$. Interested bidders then have the chance to publish their identities, accompanied by a randomly generated share of the public key to be used for the auction, before the deadline expires. In the following, each bidder broadcasts two encrypted messages and sends shares of the decrypted outcome to the seller who broadcasts these messages after he received all of them. Steps 3–5 correspond to the rounds explained in Fig. 1.

6 Analysis

In this section, we analyze security and efficiency of the auction protocols. Throughout this analysis, we assume that there is a reliable broadcast channel, i.e., the adversary has no control of communication,

Proposition 1 *The proposed protocols are*

- correct *with negligible error probability,*
- fully private *if the decisional Diffie–Hellman problem is intractable, and*

[3] As explained in Sect. 2, we do not consider the additional overhead caused by bidders aborting the protocol.

Prologue: Generate public key

- Choose $x_{+a} \in \mathbb{Z}_q$ and $m_{ij}^{+a}, r_{aj} \in \mathbb{Z}_q$ for each i and j at random.
- Publish $y_{\times a} = g^{x_{+a}}$ along with a zero-knowledge proof of knowledge of $y_{\times a}$'s discrete logarithm (Section 5.2.1).
- Compute $y = \prod_{i=1}^{n} y_{\times i}$.

Round 1: Encrypt bid

- Set $b_{aj} = \begin{cases} Y & \text{if } j = b_a \\ 1 & \text{else} \end{cases}$ and publish $\alpha_{aj} = b_{aj} y^{r_{aj}}$ and $\beta_{aj} = g^{r_{aj}}$ for each j.
- Prove that $\forall j : \log_g(\beta_{aj})$ equals $\log_y(\alpha_{aj})$ or $\log_y\left(\frac{\alpha_{aj}}{Y}\right)$ (Section 5.2.3), and

$$\log_y\left(\frac{\prod_{j=1}^{k} \alpha_{aj}}{Y}\right) = \log_g\left(\prod_{j=1}^{k} \beta_{aj}\right) \text{ (Section 5.2.2).}$$

Round 2: Compute outcome

- Compute and publish for each i and j:

$$\gamma_{ij}^{\times a} = \left(\left(\prod_{h=1}^{n}\prod_{d=j+1}^{k} \alpha_{hd}\right) \cdot \left(\prod_{d=1}^{j-1} \alpha_{id}\right) \cdot \left(\prod_{h=1}^{i-1} \alpha_{hj}\right)\right)^{m_{ij}^{+a}} \quad \text{and}$$

$$\delta_{ij}^{\times a} = \left(\left(\prod_{h=1}^{n}\prod_{d=j+1}^{k} \beta_{hd}\right) \cdot \left(\prod_{d=1}^{j-1} \beta_{id}\right) \cdot \left(\prod_{h=1}^{i-1} \beta_{hj}\right)\right)^{m_{ij}^{+a}}$$

  with a proof of correctness (Section 5.2.2).

Round 3: Decrypt outcome

- Send $\varphi_{ij}^{\times a} = \left(\prod_{h=1}^{n} \delta_{ij}^{\times h}\right)^{x_{+a}}$ for each i and j with a proof of correctness (Section 5.2.2) to the seller who publishes all $\varphi_{ij}^{\times h}$ and the corresponding proofs of correctness for each i, j, and $h \neq i$ after having received all of them.

Epilogue: Outcome determination

- Compute $v_{aj} = \dfrac{\prod_{i=1}^{n} \gamma_{aj}^{\times i}}{\prod_{i=1}^{n} \varphi_{aj}^{\times i}}$ for each j.
- If $v_{aw} = 1$ for any w, then bidder a is the winner of the auction. p_w is the selling price.

Notes: $i, h \in \{1, 2, \ldots, n\}$, $j, b_a \in \{1, 2, \ldots, k\}$, $Y \in \mathbb{G}_q \backslash \{1\}$.

Fig. 1 Fully private first-price auction protocol

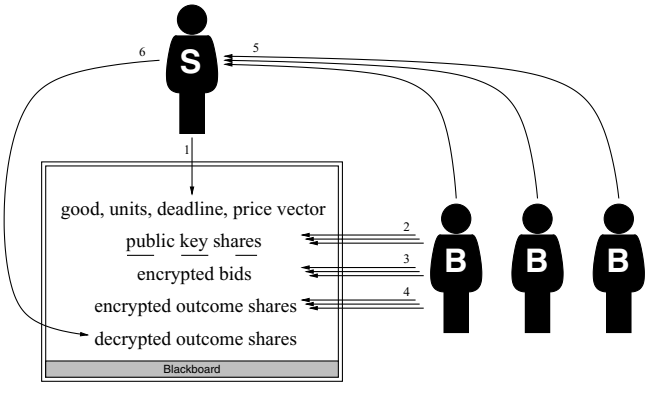

Fig. 2 High-level protocol visualization

– fair, *i.e., either all bidders or no bidder learns the outcome, if the seller does not quit or reveal information prematurely.*

Proof (sketch)
Correctness The protocols only fail when the random exponentiation for any outcome vector "accidently" yields a one, i.e., $\sum_{h=1}^{n} m_{ij}^{+h} = 0 \mod q$ for any i and j. Due to the exponential size of $\mathbb{G}_q$ and the polynomial number of outcome components, the probability of this event is negligible. The malleability of El Gamal encryption does not pose a problem because bidders prove that they know each plaintext using non-malleable zero-knowledge proofs.

Privacy The security of El Gamal cipher as well as the applied zero-knowledge proofs can be based on the intractability of the decisional Diffie–Hellman assumption [49]. The security of *distributed* El Gamal cipher, in particular Pedersen's straightforward key generation [39] which might result in non-uniformly distributed keys, follows from a recent argumentation by Gennaro et al. [21]. Since encryption keys are essentially distributed by using n-out-of-n secret sharing, privacy can not be breached (unless *all* bidders collude). In the absence of ties, only the

Table 2 Communication complexity (number of bits each bidder broadcasts) of the protocol proposed in Fig. 1

	Body	Zero-knowledge proofs
Prologue	P	$P+Q$
Round 1	$2kP$	$4k(P+Q)+2P+Q$
Round 2	$2nkP$	$nk(2P+Q)$
Round 3	nkP	$(nk+1)P+Q$
Σ	$(k(3n+2)+1)P$	$(k(3n+4)+4)P+(2k(n+2)+3)Q$
Σ Body+ZK	$(6k(n+1)+5)P+(2k(n+2)+3)Q$	

Notes. $P = \lceil \log p \rceil$; $Q = \lceil \log q \rceil$.

Table 3 Protocol complexity (computation/communication per bidder)

Auction type	Outcome	Automatic tie-breaking	Rounds	Exponentiations/communication
First-price	Private	Yes	$\mathcal{O}(1)$	$\mathcal{O}(nk)$
First-price	Public	No	$\mathcal{O}(1)$	$\mathcal{O}(k)$
Second-price	Private	Yes	$\mathcal{O}(1)$	$\mathcal{O}(n^2k)$
Second-price	Public	No	$\mathcal{O}(1)$	$\mathcal{O}(nk)$
$(M+1)$st-price	Private	No	$\mathcal{O}(1)$	$\mathcal{O}(n(n-M)kM)$
$(M+1)$st-price	Public	No	$\mathcal{O}(1)$	$\mathcal{O}((n-M)kM)$

Notes. n: bidders; k: prices/possible bids; M: units to be sold.

auction outcome is revealed. Some protocols (see Table 3) are able to automatically break ties whereas others reveal the identities of all tied winning bidders. The outcome function for $(M+1)$st-price auctions reveals the following information if bidders tie for the $(M+1)$st-highest bid: the number of tied bidders (t) and the number of bidders with higher bids (u). Since the protocol is restarted *from scratch* when a malicious bidder disrupts, privacy is also preserved under sequential composition.

Fairness The seller is the first party to learn the auction outcome. If he does not quit or give away information (which would allow bidders to quit after having learned the outcome), all bidders are simultaneously informed about the outcome without any possibility to refuse further interaction. □

A simple game-theoretic observation shows why it is quite reasonable to assume that the seller does not quit prematurely. Since the seller can set a reserve price by also appearing as a pseudo-bidder, he can never face an outcome that decreases his utility. In game-theoretic terms, the auction mechanism is *ex post* individually rational. If the seller learns that no bid exceeds the reserve price, he might decide to quit the protocol prematurely because he wants to hide the reserve price in a public outcome protocol. However, as this is the only case in which a *rational* seller quits, bidders could derive the auction outcome (i.e., that the reserve price exceeds all bids) from the fact that the seller leaves the protocol. Obviously, a game-theoretic analysis always has to make assumptions on the players' utility functions. A malicious seller might still quit an auction early, simply because he is just interested in learning the auction outcome but not in selling anything. Nevertheless, he is only able to learn the outcome, not particular bids. A higher degree of robustness can be obtained by fining agents that abort the protocol prematurely or introducing a fairness-providing third party. Please note that such a third party would not learn any private information about bids.

In the following, we analyze the computation complexity (number of exponentiations and multiplications) and communication complexity (number of bits to be broadcasted) of the protocol proposed in Fig. 1. Typically, the computational cost of performing multiplications is negligible. Exponentiation and communication complexity are identical in all proposed protocols. All applied zero-knowledge proofs are non-interactive and have low constant overhead. The complexity of these proofs is given in Table 2. In the prologue of the protocol, each bidder broadcasts his share of the public key (one exponentiation). When all n key shares have been published, each bidder computes the public key by multiplying them. In Round 1, each bidder broadcasts k El Gamal ciphertexts ($2k$ exponentiations and one multiplication). In the second round, the shares for each bidder's private outcome are computed. This results in $2nk$ exponentiations. The number of multiplications required to compute $\gamma_{ij}^{\times a}$ and $\delta_{ij}^{\times a}$ is $8nk$. Round 3 consists of nk exponentiations and n^2k multiplications. Finally, outcome determination requires no exponentiations (and further communication) and $2nk$ multiplications. The overall complexity of the protocol is $3nk+2k+1$ exponentiations and $(3nk+2k+1)\lceil \log p \rceil$ bits each bidder needs to publish. Table 2 summarizes the communication complexity of each protocol step and also gives the complexity of the accompanying zero-knowledge proofs. As described in Sect. 4.2, the protocols for $(M+1)$st-price auctions require the computation of $(M+1)(n-(M+1))$ outcome vectors, either for each bidder (private outcome) or just once (public outcome). As a consequence, in order to compute the complexity of the private outcome $(M+1)$st-price auction protocol, the numbers in the rows for Round 2 and 3 in Table 2 need to be multiplied with the aforementioned factor.

The total number of bits each bidder needs to communicate in the private outcome first-price protocol given in Fig. 1 is $(6k(n+1)+5)\lceil \log p \rceil + (2k(n+2)+3)\lceil \log q \rceil$. To achieve an appropriate level of security today, 1024 bits for p and 768 bits for q are reasonable settings. Then, in an auction with 10 bidders and 500 prices,[4] a bidder broadcasts about 5.1 MB. This is quite an amount of data but certainly manageable in today's networks. Complexity can be significantly reduced by running the public outcome protocol in which approximately 1.0 MB is communicated by each bidder.

7 Conclusion

We presented cryptographic constant-round protocols for all common types of sealed-bid auctions. The security of the proposed protocols is merely based on computational intractability and does not rely on third parties. Table 3 summarizes the protocols' properties. We considered the *private* outcome setting, in which only the winning bidders and the seller learn the auction's result, and the *public* outcome setting, in which all agents learn the result. Private outcome protocols for first-price and second-price auctions have the ability to "automatically" break ties whereas other protocols yield the identities of all tied winning bidders. In this case, ties could be broken by a subsequent distributed coin-tossing protocol. The computational complexity for $(M+1)$st-price auction protocols (including $M=1$) is n times as much as for first-price auctions due to the possibility of ties involving the $(M+1)$st-highest bid. In practice, these additional outcome vectors could just be computed on demand when the "regular" protocol fails. This increases round complexity and the risk of premature abort but reduces the computational (and communication) burden on average.

It would be desirable to construct protocols whose computational complexity is only logarithmic in k. However, it seems that this will very likely raise round complexity. Besides, experimental results indicate that the computational amount and message sizes are manageable in many realistic settings, despite its linearity in k [13]. Auctions that require such a high degree of privacy typically involve few bidders, for instance when auctioning off radio spectrum licenses or former state-owned enterprises. Furthermore, the unary representation allows us to easily adapt the given protocols to emulate *iterative* (e.g., ascending-price or descending-price) auctions in which bidders gradually express their willingness to pay for sequences of prices. Iterative auctions are sometimes preferred over sealed-bid auctions because bidders are not required to exhaustively determine their valuations and because they can lead to higher revenue if valuations are interdependent (see e.g., [33, 34]).

[4] Often, the number of possible bids k can be set to a much lower value than one would expect, e.g., Lipmaa et al. argue that $k \leq 500$ is sufficient for most auctions in practice [35].

We believe that techniques underlying the proposed protocols can be useful to construct efficient multiparty protocols for further tasks such as voting or other types of auctions.

Acknowledgements Thanks to Jens Groth for some valuable comments on an earlier version of this. Further thanks to Alina Oprea, Ke Yang, and the anonymous referees for helpful feedback.

References

1. Abe, M., Suzuki, K.: M+1-st price auction using homomorphic encryption. In: Proceedings of the 5th International Conference on Public Key Cryptography (PKC). Lecture Notes in Computer Science (LNCS), vol. 2274, pp. 115–224. Springer, Berlin Heidelberg New York (2002)
2. Baudron, O., Stern, J.: Non-interactive private auctions. In: Proceedings of the 5th Annual Conference on Financial Cryptography (FC). Lecture Notes in Computer Science (LNCS), vol. 2339, pp. 300–313. Springer, Berlin Heidelberg New York (2001)
3. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: Proceedings of the 20th Annual ACM Symposium on the Theory of Computing (STOC), pp. 1–10. ACM, New York (1988)
4. Brandt, F.: Secure and private auctions without auctioneers. Tech. Rep. FKI-245-02, Department for Computer Science, Technical University of Munich, Munich (2002). ISSN 0941-6358
5. Brandt, F.: A verifiable, bidder-resolved auction protocol. In: Falcone, R., Barber, S., Korba, L., Singh, M. (eds.) Proceedings of the 5th AAMAS Workshop on Deception, Fraud and Trust in Agent Societies (Special Track on Privacy and Protection with Multi-Agent Systems), pp. 18–25 (2002)
6. Brandt, F.: Fully private auctions in a constant number of rounds. In: Wright, R.N. (ed.) Proceedings of the 7th Annual Conference on Financial Cryptography (FC). Lecture Notes in Computer Science (LNCS), vol. 2742, pp. 223–238. Springer-Verlag, Berlin Heidelberg New York (2003)
7. Brandt, F.: Social choice and preference protection—Towards fully private mechanism design. In: Nisan, N. (ed.) Proceedings of the 4th ACM Conference on Electronic Commerce, pp. 220–221. ACM, New York (2003)
8. Brandt, F., Sandholm, T.: (Im)possibility of unconditionally privacy-preserving auctions. In: Sierra, C., Sonenberg, L. (eds.) Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), pp. 810–817. ACM, New York (2004)
9. Brandt, F., Sandholm, T.: On correctness and privacy in distributed mechanisms. In: Faratin, P., Rodriguez-Aguilar, J.A. (eds.) Selected and revised papers from the 6th AAMAS Workshop on Agent-Mediated Electronic Commerce (AMEC). Lecture Notes in Artificial Intelligence (LNAI), vol. 3435 (2004)
10. Brandt, F., Sandholm, T.: Efficient privacy-preserving protocols for multi-unit auctions. In: Patrick, A., Yung, M. (eds.) Proceedings of the 9th International Conference on Financial Cryptography and Data Security (FC). Lecture Notes in Computer Science (LNCS), vol. 3570, pp. 298–312. Springer, Berlin Heidelberg New York (2005)
11. Chaum, D., Crépeau, C., Damgård, I.: Multi-party unconditionally secure protocols. In: Proceedings of the 20th Annual ACM Symposium on the Theory of Computing (STOC), pp. 11–19. ACM, New York (1988)

12. Chaum, D., Pedersen, T.P.: Wallet databases with observers. In: Advances in Cryptology—Proceedings of the 12th Annual International Cryptology Conference (CRYPTO). Lecture Notes in Computer Science (LNCS), vol. 740, pp. 3.1–3.6. Springer, Berlin Heidelberg New York (1992)
13. Chen, W.: Kryptographische Auktionsprotokolle—Implementierung und Analyse (2002). Systementwicklungsprojekt, Department for Computer Science, Technical University of Munich. http://www.chenwilly.info
14. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Advances in Cryptology—Proceedings of the 14th Annual International Cryptology Conference (CRYPTO). Lecture Notes in Computer Science (LNCS), vol. 893, pp. 174–187. Springer, Berlin Heidelberg New York (1994)
15. Cramer, R., Gennaro, R., Schoenmakers, B.: A secure and optimally efficient multi-authority election scheme. In: Advances in Cryptology—Proceedings of the 14th Eurocrypt Conference. Lecture Notes in Computer Science (LNCS), vol. 1233, pp. 103–118. Springer, Berlin Heidelberg New York (1997)
16. Damgård, I.: On Σ-protocols. Lecture Notes, University of Aarhus, Department for Computer Science (2002)
17. El Gamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. Inf. Theory **31**, 469–472 (1985)
18. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Advances in Cryptology—Proceedings of the 12th Annual International Cryptology Conference (CRYPTO). Lecture Notes in Computer Science (LNCS), pp. 186–194. Springer, Berlin Heidelberg New York (1987)
19. Franklin, M.K., Reiter, M.K.: The design and implementation of a secure auction service. IEEE Trans. Softw. Eng. **22**(5), 302–312 (1996)
20. Garay, J., MacKenzie, P., Yang, K.: Efficient and secure multiparty computation with faulty majority and complete fairness. Cryptology ePrint Archive, Report 2004/009 (2004)
21. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Applications of Pedersen's distributed key generation protocol. In: Proceedings of the Cryptographers' Track at the 12th RSA Conference. Lecture Notes in Computer Science (LNCS), vol. 2612, pp. 373–390. Springer, Berlin Heidelberg New York (2003)
22. Goldreich, O.: Foundations of Cryptography, vol. 2. Basic Applications. Cambridge University Press, Cambridge (2004)
23. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or a completeness theorem for protocols with honest majority. In: Proceedings of the 19th Annual ACM Symposium on the Theory of Computing (STOC), pp. 218–229. ACM, New York (1987)
24. Goldwasser, S., Levin, L.: Fair computation of general functions in presence of immoral majority. In: Advances in Cryptology—Proceedings of the 10th Annual International Cryptology Conference (CRYPTO). Lecture Notes in Computer Science (LNCS), vol. 537, pp. 77–93. Springer, Berlin Heidelberg New York (1990)
25. Goldwasser, S., Lindell, Y.: Secure computation without agreement. In: Proceedings of the 16th International Symposium on Distributed Computing (DISC). Lecture Notes in Computer Science (LNCS), vol. 2508, pp. 17–32. Springer, Berlin Heidelberg New York (2002)
26. Groth, J.: Efficient maximal privacy in boardroom voting and anonymous broadcast. In: Proceedings of the 8th Annual Conference on Financial Cryptography (FC). Lecture Notes in Computer Science (LNCS), vol. 3110, pp. 90–104. Springer, Berlin Heidelberg New York (2004)
27. Harkavy, M., Tygar, J.D., Kikuchi, H.: Electronic auctions with private bids. In: Proceedings of the 3rd USENIX Workshop on Electronic Commerce, pp. 61–74 (1998)
28. Jakobsson, M., Juels, A.: Mix and match: Secure function evaluation via ciphertexts. In: Proceedings of the 6th Asiacrypt Conference. Lecture Notes in Computer Science (LNCS), vol. 1976, pp. 162–177. Springer, Berlin Heidelberg New York (2000)
29. Juels, A., Szydlo, M.: A two-server, sealed-bid auction protocol. In: Blaze, M. (ed.) Proceedings of the 6th Annual Conference on Financial Cryptography (FC). Lecture Notes in Computer Science (LNCS), vol. 2357, pp. 72–86. Springer, Berlin Heidelberg New York (2002)
30. Kiayias, A., Yung, M.: Self-tallying elections and perfect ballot secrecy. In: Proceedings of the 5th International Workshop on Practice and Theory in Public Key Cryptography (PKC), no. 2274. Lecture Notes in Computer Science (LNCS), pp. 141–158. Springer, Berlin Heidelberg New York (2002)
31. Kiayias, A., Yung, M.: Non-interactive zero-sharing with applications to private distributed decision making. In: Proceedings of the 7th Annual Conference on Financial Cryptography (FC). Lecture Notes in Computer Science (LNCS), vol. 2742, pp. 303–320. Springer, Berlin Heidelberg New York (2003)
32. Kikuchi, H.: (M+1)st-price auction protocol. In: Proceedings of the 5th Annual Conference on Financial Cryptography (FC). Lecture Notes in Computer Science (LNCS), vol. 2339, pp. 351–363. Springer, Berlin Heidelberg New York (2001)
33. Klemperer, P.: Auction theory: A guide to the literature. J. Econ. Surv. **13**(3), 227–286 (1999)
34. Krishna, V.: Auction Theory. Academic, New York (2002)
35. Lipmaa, H., Asokan, N., Niemi, V.: Secure Vickrey auctions without threshold trust. In: Blaze, M. (ed.) Proceedings of the 6th Annual Conference on Financial Cryptography (FC). Lecture Notes in Computer Science (LNCS), vol. 2357, pp. 87–101. Springer, Berlin Heidelberg New York (2002)
36. Naor, M., Pinkas, B., Sumner, R.: Privacy preserving auctions and mechanism design. In: Proceedings of the 1st ACM Conference on Electronic Commerce (ACM-EC), pp. 129–139. ACM, New York (1999)
37. Nurmi, H., Salomaa, A.: Cryptographic protocols for Vickrey auctions. Group Decis. Negot. **2**, 363–373 (1993)
38. Pass, R.: Bounded-concurrent secure multiparty computation with a dishonest majority. In: Proceedings of the 36th Annual ACM Symposium on the Theory of Computing (STOC), pp. 232–241. ACM, New York (2004)
39. Pedersen, T.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Feigenbaum, J. (ed.) Advances in Cryptology—Proceedings of the 11th Annual International Cryptology Conference (CRYPTO). Lecture Notes in Computer Science (LNCS), vol. 576, pp. 129–140. Springer, Berlin Heidelberg New York (1991)
40. Pinkas, B.: Fair secure two-party computation. In: Proceedings of the 20th Eurocrypt Conference. Lecture Notes in Computer Science (LNCS), vol. 2656, pp. 87–105. Springer, Berlin Heidelberg New York (2003)
41. Porter, R., Shoham, Y.: On cheating in sealed-bid auctions. In: Proceedings of the 4th ACM Conference on Electronic Commerce (ACM-EC), pp. 76–84. ACM, New York (2003)
42. Rothkopf, M.H., Harstad, R.M.: Two models of bid-taker cheating in Vickrey auctions. J. Business **68**(2), 257–267 (1995)
43. Rothkopf, M.H., Teisberg, T.J., Kahn, E.P.: Why are Vickrey auctions rare? J. Pol. Econ. **98**(1), 94–109 (1990)
44. Sako, K.: An auction protocol which hides bids of losers. In: Proceedings of the 3rd International Conference on Public Key Cryptography (PKC). Lecture Notes in Computer Science (LNCS), vol. 1751, pp. 422–432. Springer, Berlin Heidelberg New York (2000)
45. Sandholm, T.: Issues in computational Vickrey auctions. International Journal of Electronic Commerce, Special Issue Intell. Agents Electron. Commer. **4**(3), 107–129 (2000)
46. Schnorr, C.P.: Efficient signature generation by smart cards. J. Cryptol. **4**(3), 161–174 (1991)
47. Suzuki, K., Yokoo, M.: Secure combinatorial auctions by dynamic programming with polynomial secret sharing. In: Proceedings of the 6th Annual Conference on Financial Cryptography (FC). Lecture Notes in Computer Science (LNCS), vol. 2357. Springer, Berlin Heidelberg New York (2002)

48. Suzuki, K., Yokoo, M.: Secure generalized Vickrey auction using homomorphic encryption. In: Proceedings of the 7th Annual Conference on Financial Cryptography (FC). Lecture Notes in Computer Science (LNCS), vol. 2742, pp. 239–249. Springer, Berlin Heidelberg New York (2003)
49. Tsiounis, Y., Yung, M.: On the security of El Gamal-based encryption. In: Proceedings of the 1st International Workshop on Practice and Theory in Public Key Cryptography (PKC). Lecture Notes in Computer Science (LNCS), vol. 1431, pp. 117–134. Springer, Berlin Heidelberg New York (1998)
50. Vickrey, W.: Counter speculation, auctions, and competitive sealed tenders. J. Finance **16**(1), 8–37 (1961)
51. Yao, A.C.: How to generate and exchange secrets. In: Proceedings of the 27th Symposium on Foundations of Computer Science (FOCS), pp. 162–167. IEEE Comput. Soc. Press (1986)