Blockchain-based Practical Multi-agent Secure Comparison and its Application in Auctions

Sankarshan Damle
sankarshan.damle@research.iiit.ac.in
Machine Learning Lab, IIIT, Hyderabad
Hyderabad, India

Boi Faltings
boi.faltings@epfl.ch
Artificial Intelligence Laboratory, EPFL
Lausanne, Switzerland

Sujit Gujar
sujit.gujar@iiit.ac.in
Machine Learning Lab, IIIT, Hyderabad
Hyderabad, India

ABSTRACT

AI applications find widespread use in a variety of domains. For further acceptance, mostly when multiple agents interact with the system, we must aim to preserve the privacy of participants information in such applications. Towards this, the Yao's Millionaires' problem (YMP), i.e., to determine the richer among two millionaires' privately, finds relevance. This work presents a novel, practical, and verifiable solution to YMP, namely, Secure Comparison Protocol (SCP). We show that SCP achieves this comparison in a constant number of rounds, without using encryption and not requiring the participants' continuous involvement. SCP uses semi-trusted third parties - which we refer to as privacy accountants - for the comparison, who do not learn any information about the values. That is, the probability of information leak is negligible in the problem size. In SCP, we also leverage the Ethereum network for pseudo-anonymous communication, unlike computationally expensive secure channels such as Tor. We present a Secure Truthful cOmbinatorial aUction Protocol (STOUP) for single-minded bidders to demonstrate SCP's significance. We show that STOUP, unlike previous works, preserves the privacies relevant to an auction even from the auctioneer. We demonstrate the practicality of STOUP through simulations.

KEYWORDS

Multi-agent System, Secure Combinatorial Auction, Yao's Millionaire Problem

ACM Reference Format:
Sankarshan Damle, Boi Faltings, and Sujit Gujar. 2021. Blockchain-based Practical Multi-agent Secure Comparison and its Application in Auctions. In *WI-IAT '21: IEEE/WIC/ACM International Conference on Web Intelligence, December 14–17, 2021, ESSENDON, VIC, Australia.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3486622.3493937

1 INTRODUCTION

Multi-agent based AI applications such as *distributed constraint optimization*, *e-commerce* and *e-voting* mechanisms have grown in popularity. Consequently, the need for privacy of the information exchange within these platforms has become imperative and is an area of active research [16, 20, 29, 38–40]. The participants (e.g., bidders), being *strategic agents*, prefer the preservation of their *private* information (e.g., bids) as well as (often) their public identities from other competitive agents. Such *anonymity* of information may also increase participation.

With *blockchain* gaining momentum, AI applications are now being conducted through computational logic over distributed platforms such as the *Ethereum* blockchain network [44]. Specifically, Ethereum allows for *smart contracts*, which are computer protocols intended to digitally facilitate, verify, or enforce the negotiation or performance of a contract [43]. Since these are on a publicly distributed ledger, they are open to any interested agent while making sensitive information (e.g., bids) and the execution of payments publicly verifiable, transparent, and pseudo-anonymous. Consequently, an agent's private information is publicly available for anyone to see and use. This further necessitates the need for secure (privacy-preserving) AI applications over a blockchain.

At the heart of several AI applications mentioned above is the comparison of two numeric values [9]. Therefore, to build a protocol that preserves each agent's private information, we require a method for comparing these values while preserving their privacy. In the literature, this challenge is referred to as *Yao's Millionaires' Problem* (YMP) [45] of securely determining the richer between two different agents and has been extensively studied.

YMP is as follows: Two agents (millionaires), *Alice* and *Bob* are interested in determining the richer among them – without revealing their actual wealth. Motivated by this, in this paper, we introduce a novel method for comparing two integers $x, y \in \mathbf{Z}$ securely, i.e., a practical solution to YMP designed for secure multi-agent applications. We refer to our method as *Secure Comparison Protocol* (SCP). In SCP, we assume that there are approved *cryptographic accountants* in the system, which assist the *central server* (CS) in determining whether $x \geq y$ or not. We show that the probability with which CS (or any other party) learns any information regarding x or y, in SCP, is negligible. Further, we show that the integer comparison in SCP is *verifiable*, by leveraging *zero-knowledge proof* (ZKP) techniques.

To securely deploy these AI applications over blockchain, we present SCP over the Ethereum network[1]. To the best of our knowledge, we are the first to introduce a dedicated and verifiable solution, in constant rounds, for YMP over the blockchain. We demonstrate

WI-IAT '21, December 14–17, 2021, ESSENDON, VIC, Australia

ACM ISBN 978-1-4503-9115-3/21/12…$15.00
https://doi.org/10.1145/3486622.3493937

[1] SCP may also be deployed using computationally expensive secure channels like Tor [10]. By coupling blockchain with an asymmetric encryption scheme, we ensure efficient pseudo-anonymous communication.

Paper	Homomorphic Encryption	Garbled Circuit	Continuous User Involvement	Dependent on Input size
[7, 11, 13, 18, 24]	✓	✗	✓	✓
[1]	✗	✗	✗	✗
[27]	✗	✓	✗	✓
[2, 6]	✗	✓	✓	✓
[5]	✓	✗	✗	✓
[46]	✓	✗	✗	✓
SCP	✗	✗	✗	✗

Table 1: Comparing existing YMP protocols with SCP. "✗" is desirable.

the significance of SCP by presenting a Secure, Truthful cOmbinatorial aUction Protocol (STOUP) for single-minded bidders.

With STOUP, we show how to sort and compare the bidding information of agents without revealing them, with the help of accountants. The accountants do not learn of any *bidding information*, i.e., bid values and the items which are bid for. Towards this, we assume that each agent's bundle size is ≥ 2. Otherwise, the items in a bidder's bundle may get revealed to the auctioneer in our protocol. Note that in STOUP, the accountant's role is only to assist the auctioneer in determining winners and their payments when the bid values and items are hidden.

Adversary Model. As standard in solutions for Yao's Millionaires' Problem (eg., [3, 14, 23, 24]), in this paper, we assume that CS is *semi-honest* or *honest-but-curious*. This implies that while CS can observe and cipher any information, it *will not* deviate from the defined protocol. However, unlike much of the previous works, we assume that all *other* agents, i.e., Alice, Bob, accountants, bidders, etc., are *strategic-but-curious*. That is, these agents do not deviate from the protocol but may potentially submit/send incorrect information to gain advantage or to increase their utility. Additionally, we also assume that agents do not collude. This assumption is standard in the literature for existing secure AI applications [16, 20, 38].

2 RELATED LITERATURE

Yao's Millionaires' Problem. Yao [45] introduces YMP along with its first solution. However, the presented solution is exponential in time and space. Thereafter, several protocols improve over the seminal solution [2, 6, 15].

The authors in [17] present a two-round protocol which is polynomial while [3, 23] provide a single-round solution which is linear in the order of the length of the integers to be compared. For their solutions, [17] uses complex *bitwise operators* while [3, 23] use *Paillier homomorphic encryption* and zero-knowledge proof. The computational cost per comparison in [3] is $(4b + 1)(\log p) + 6b$ and in [23] is $5b(\log p) + 4b - 6$, where b is the bit number and p modulus of the Paillier scheme. Recently, [24] proposes a single-round solution using Paillier encryption and *vectorization* method. However, the solution is of the order $2(s + 2)\log p$, where s is the vector dimension.

Because of space constraint, we place SCP with some of the plethoras of protocols available for YMP using Table 1. To the best of our knowledge, the existing protocols comprise one or more of (i) garbled circuits, (ii) partial-homomorphic encryption, and (iii) continuous involvement of the parties during execution. Note that, [1] is limited as it is only applicable for integers in the range of $\approx 2^{60}$; whereas SCP works for any range. Consequently, these protocols *can not* be adapted towards designing lightweight and secure AI applications. Aggravating this limitation is that the number of comparisons needed for such AI applications is significant – in the order of polynomials or more. One can not assign trusted third parties for the operations as it may reveal the owners' private information.

Combinatorial Auctions. A combinatorial auction, where the parties can bid for combination(s) of items, yields a higher revenue (lesser costs) than selling (buying) the items individually. E.g., wireless spectrum auctions [25] or allocating airport landing take-off slots [34]. Combinatorial auctions have an exponential number of possible valuations for each party and are NP-Complete [35]. Hence, we focus on a *single-minded* case. In this, the parties are interested in a specific bundle of items and obtain a particular value if they get the whole bundle (or any super-set) and zero otherwise. Unfortunately, even single-minded combinatorial auctions, being NP-Hard [36], are solved approximately. In particular, [22] proposes a polynomial-time algorithm for single-minded combinatorial auction, which gives $\sqrt{m}$-approximate winner and payment determination payment rule, which we refer to as ICA-SM (Incentive Compatible Approximate auctions for Single-minded bidders). Here, m denotes the number of items being auctioned.

Secure Auctions. Micali and Rabin [26] solve single-item and multi-unit auctions while preserving the privacy of the bids using Pedersen commitment, but reveal the bid information to the auctioneer after the end of the bidding phase. Similarly, [28, 31] present single and multi-unit auctions that reveal the bid-topology to third parties. The authors in [4] give a practical, multi-unit auction that does not reveal any private information to a third party, even after the auction closes. Parkes et al. [32] use *clock-proxy* auction to solve a privacy-preserving combinatorial auction, revealing private information to the auctioneer after the end of the clock phase. The protocol is linear in the size of the original computational time, from exponential. Suzuki and Yokoo [37] propose a privacy-preserving, secure combinatorial auction without revealing any bid information to a third party. The authors use dynamic programming, and [19] extends the work to add verifiability. The protocol, however, is exponential in the size of the number of bids. The protocol is thus impractical even for a small number of bids.

We leverage SCP to present a secure combinatorial auction protocol, namely STOUP. To the best of our knowledge, ours is the first work to present an efficient, secure combinatorial auction protocol that preserves the bidding information's privacy, even from the auctioneer. We omit proofs of the results presented because of space constraints. These are available in our extended version: [8].

3 SECURE COMPARISON PROTOCOL (SCP)

We first summarize the cryptographic background required for SCP.

3.1 Cryptographic Background

We leverage the following known cryptographic techniques to construct our solution for YMP. Let p and q denote large primes such that q divides $p - 1$, with G_q as the unique subgroup of $\mathbf{Z}_p^*$ of order q, and g as a generator of G_q.

Pedersen Commitment [33]. Let g and $h = g^a \pmod p$ be elements of G_q such that $log_g h$ is intractable, where $a \in \mathbf{Z}_q$ is the secret key. Then, a *Pedersen commitment scheme* is the commitment of a message $x \in \mathbf{Z}_q$, with a random help value $r \in \mathbf{Z}_q$, as, $C(x, r) = g^x h^r \pmod p$. We denote a_i as a party i's secret key, with $h_i = g^{a_i} \pmod p$.

Random Number Representation [26]. A *random number representation* of a number x, $R(x)$, is a representation of x as the pair (u, v) where $u, v \in \mathbf{Z}_q$ and $x = (u + v) \bmod q$, Note that, to find $R(x)$ of a number x, any party can randomly choose u and then pick $v = (x - u) \bmod q$.

Value Comparison [26]. For two integers $x, y < q/2$,

$$x - y \leq q/2 \iff x \geq y \text{ and } x - y > q/2 \iff x < y \quad (1)$$

Therefore, to compare x and y, we only need to check whether $x - y \leq q/2$.

Zero-knowledge Proof [12]. Zero-knowledge proof (ZKP) is a method by which a party, called a *Prover* ($\mathcal{P}$), is able to convince another party, called a *Verifier* ($\mathcal{V}$), that it knows some information ω, without revealing ω (or any other information related to ω). Formally, $\mathcal{P}$ must convince $\mathcal{V}$ that $\exists\omega : \mathcal{R}(l, \omega) = 1$ for a relation $\mathcal{R}$, an input l (from $\mathcal{V}$) and a witness ω from $\mathcal{P}$. A ZKP must satisfy:

- *Completeness:* If $\exists\omega : \mathcal{R}(l, \omega) = 1$, then an honest $\mathcal{P}$ convinces $\mathcal{V}$ except with negligible probability, i.e., with probability at-most $\ll 1/2$.
- *Soundness:* If $\nexists\omega : \mathcal{R}(l, \omega) = 1$, a dishonest $\mathcal{P}'$ convinces $\mathcal{V}$ with negligible probability, i.e., with probability at-most $\ll 1/2$.
- *Zero-knowledge.* If $\exists\omega : \mathcal{R}(l, \omega) = 1$, then $\mathcal{V}$ does not learn any information about ω except with negligible probability, i.e., with probability at-most $\ll 1/2$.

Notations. We utilize the following notations for SCP as well as throughout the paper.

- $C(R(x))$ represents the Pedersen commitment of x as $R(x) = (u, v)$, i.e., $C(R(x))$ denotes the pair of commitments $\big(C(u, r), C(v, r')\big)$.
- $A \xrightarrow{x} B$ denotes a party A submitting a value x to a smart contract, such that x is encrypted using B's public key.
- $\mathbb{H}(\cdot)$ denotes a *collision-resistant* hash function (e.g., [30]).
- $E_A(x)$ represents the ElGamal encryption [41] of x using party A's private key.

3.2 SCP: Procedure

We now describe SCP which securely compares two integers x and y owned by two agents, Alice and Bob. Towards this, let $||x||$ denote the number of bits required to represent the integer x. Now, in SCP, we assume that there exists a *central server* (CS) that coordinates the comparison. Note that, as shown later, the CS only aids the comparison and only learns additional information about the values of x and y with negligible probability. We assume that $x, y < \frac{q}{2 \cdot d_{max}}$, where $d_{max} \in (1, 2^{(||q||-1)})$. In SCP, we require Alice and Bob to privately select an *integer* $d_{Alice}, d_{Bob} \in (1, d_{max}]$, respectively. Let, $D = (d_{Alice} \oplus d_{Bob})$. Observe that, we have $||D|| < ||\frac{q}{2}||$. For readability, we also denote $||d_{max}||$ as $\mathbf{d}$.

Before describing SCP, we present the following claim. For this, let $R(x) = (u_1, v_1)$, $R(y) = (u_2, v_2)$, $val_1 = (u_1 - u_2) \bmod q$, and $val_2 = (v_1 - v_2) \bmod q$ with $x, y < \frac{q}{2 \cdot d_{max}}$. The claim is a simple rearrangement of the result presented in Eq. 1.

SCP Procedure.

Let, $(n^1_{Alice}, n^2_{Alice})$ and (n^1_{Bob}, n^2_{Bob}) be Alice and Bob's pair of *distinct* assigned accountants, respectively.

(1) Alice generates $R(x) = (u_1, v_1)$ and broadcasts $C(R(x))$ while Bob generates $R(y) = (u_2, v_2)$ and broadcasts $C(R(y))$; through SC.

(2)
$$\text{Alice} \xrightarrow{u_1, r_1, d_{Alice}} n^1_{Alice} \text{ and Alice} \xrightarrow{v_1, r'_1, d_{Alice}} n^2_{Alice}$$
$$\text{Bob} \xrightarrow{u_2, r_2, d_{Bob}} n^1_{Bob} \text{ and Bob} \xrightarrow{v_2, r'_2, d_{Bob}} n^2_{Bob}$$

(3)
$$n^1_{Alice} \xrightarrow{u_1} n^1_{Bob} \text{ and } n^2_{Alice} \xrightarrow{v_1} n^2_{Bob}$$
$$n^1_{Bob} \xrightarrow{d_{Bob}} n^1_{Alice} \text{ and } n^2_{Bob} \xrightarrow{d_{Bob}} n^2_{Alice}$$

(4)
$$n^1_{Alice} \xrightarrow{E_{CS}(d_{Alice} \oplus d_{Bob})} n^1_{Bob}$$
$$n^2_{Alice} \xrightarrow{E_{CS}(d_{Alice} \oplus d_{Bob})} n^2_{Bob}$$

(5)
$$n^1_{Bob} \xrightarrow{X = E_{CS}(d_{Alice} \oplus d_{Bob} \cdot (u_1 - u_2) \bmod q)} CS$$
$$n^2_{Bob} \xrightarrow{Y = E_{CS}(d_{Alice} \oplus d_{Bob} \cdot (v_1 - v_2) \bmod q)} CS$$

(6) CS then checks the following,
if $(X + Y) \bmod q = 0$ *return* "**equal**"
if $(X + Y) \bmod q < q/2$ *return* " $>$ "
else return " $<$ "

Figure 1: SCP Procedure

Claim 3.1. *(i)* $D \cdot (val_1 + val_2) \bmod q \leq q/2 \iff x \geq y$; *and (ii)* $D \cdot (val_1 + val_2) \bmod q > q/2 \iff x < y$.

3.2.1 Protocol. For SCP, we consider a smart contract SC which allows agents to post and get relevant information. Fig. 1 presents the procedure for SCP, while Fig. 2 presents the procedure for ZKP of SCP. Note that, for the ZKP, CS acts as $\mathcal{P}$. Trivially, SCP is independent of the length of the binary representation of x or y and hence is of constant order ($O(1)$) of computation rounds. We illustrate the protocol timeline of SCP with Fig. 3.

3.3 SCP: Security and Privacy Analysis

SCP (Fig. 1) preserves privacy of the values x and y from CS since CS only knows the values[2] $(D \cdot val_1) \bmod q$ and $(D \cdot val_2) \bmod q$. It is trivial to see that CS shall not be able to find anything about the values of (u_1, v_1) and (u_2, v_2). In addition, every accountant only has one component of the other party's (Alice or Bob) value,

[2]This follows as the values $(D \cdot val_i), \forall i \in \{1, 2\}$ are reduced over mod q. For the modular multiplication of $a \cdot b \pmod q$, where q is a prime, and no information of a is known, all possible values of b are equally likely.

ZKP for SCP Procedure.

(1) CS acts as $\mathcal{P}$. $\mathcal{V}$ submits β, $g^{\beta} \pmod p$ through SC, with $\beta \in \mathbf{Z}_q$ as the random challenge.

(2) The accountants, n^1_{Alice} and n^2_{Alice}, send the values $C(u_1, r_1)^D$, $C(u_2, r_2)^{-D}$, $C(v_1, r'_1)^D$, and $C(v_2, r'_2)^{-D}$, using SC. Note that the exponentiation is modular and over p. Here, $D = d_{Alice} \oplus d_{Bob}$.

(3) $\mathcal{P}$ computes, for some *private* $\alpha \in \mathbf{Z}_q$:

$$c_1 = C(u_1, \cdot)^D C(u_2, \cdot)^{-D} C(v_1, \cdot)^D C(v_2, \cdot)^{-D} \pmod p$$
$$c_2 = g^{(\alpha+\beta) \bmod q} \pmod p$$
$$c_3 = c_1 \cdot c_2 \pmod p$$

(4) $\mathcal{P}$ submits the following values using SC.

$$c_4 = \mathbb{H}(c_1, c_2, c_3)$$
$$c_5 = (\alpha + \beta + X + Y) \bmod q.$$

(5) $\mathcal{P}$, similar to X and Y, i.e., Figure 1, gets the values,

$$H_1 \bmod q = \big(D \cdot (r_1 + r'_1)\big) \bmod q$$
$$H_2 \bmod q = -\big(D \cdot (r_2 + r'_2)\big) \bmod q.$$

(6) $\mathcal{V}$ verifies:

$$\left.\begin{aligned} c_3 &\stackrel{?}{=} c_1 \cdot c_2 \pmod p \\ c_4 &\stackrel{?}{=} \mathbb{H}(c_1, c_2, c_3) \\ c_3 &\stackrel{?}{=} g^{c_5} \cdot h_{Alice}^{H_1 \bmod q} \cdot h_{Bob}^{H_2 \bmod q} \pmod p \end{aligned}\right\} \quad (2)$$

(7) $\mathcal{V}$ accepts that the comparison was correctly computed only if Eq. 2 holds.

Figure 2: ZKP for SCP Verification.

which implies that it can not either find out anything about the other party's value.

However, as $\mathbf{d}$ is publicly known, the value $X + Y$ leaks an *upper bound* of the value $x - y$. We now show that the probability of finding the actual value of $x - y$ from $X + Y$ is *negligible* through the following results.

Theorem 3.1. *In SCP, the probability of guessing the actual value of $x - y$ is $\frac{1}{2^{\mathbf{d}}}$, i.e., negligible in $\mathbf{d}$.*

For the proof, observe that in SCP, the value of D is never revealed. This implies that the probability of guessing the value of $x - y$ correctly is equal to the probability of guessing $D \in (1, d_{max}]$. Since d_{max} requires $\mathbf{d}$ bits to represent, the theorem follows. The following corollary also follows directly from Theorem 3.1.

Corollary 3.1. *In SCP, the probability of finding the actual value of x (y) from $X + Y$, with the knowledge of the other value y (x) is $\frac{1}{2^{\mathbf{d}}}$, i.e., negligible in $\mathbf{d}$.*

Probability Threshold. Observe that the agents Alice and Bob know one of the terms in $D = d_{Alice} \oplus d_{Bob}$, but not the value itself. These agents also know one of the values, i.e., x or y. This implies,

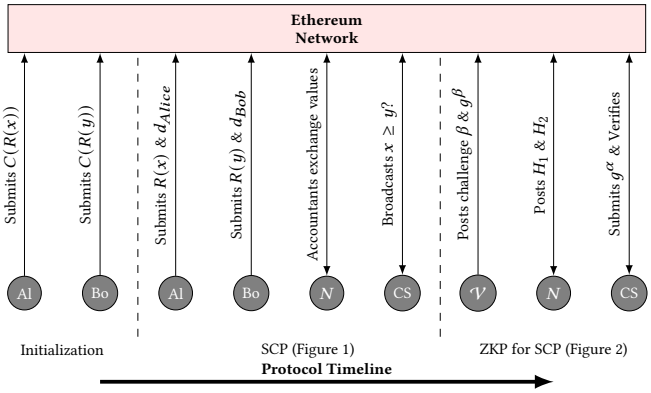

Figure 3: Illustration of the timeline in SCP. Here, Al: Alice, Bo: Bob, and N: set of assigned accountants.

from Corollary 3.1, that the probability with which these agents can guess the value of x or y from $X + Y$ is $\frac{1}{2^{\mathbf{d}}}$. Thus in SCP, the threshold of the probability with which a party can guess the value of x (y) is $\frac{1}{2^{\mathbf{d}}}$, which is negligible in $\mathbf{d}$.

ZKP in SCP. We now show that the ZKP described in Fig. 2 satisfies the three properties required for a ZKP, i.e.,

- *Completeness.* It is trivial to see that if Eq. 2 holds, then the comparison was correct. That is, a honest $\mathcal{P}$ will be able convince $\mathcal{V}$ that the comparison was correct.
- *Soundness.* If Eq. 2 does not hold, i.e., Alice and/or Bob misreported their values, then there can not be a case where $\mathcal{P}$ can find other values except for $(X + Y) \bmod q$, $(H_1) \bmod q$ and $(H_2) \bmod q$ for which Eq. 2 holds, with high probability. This is because Pedersen commitments are computationally binding[3].
- *Zero-knowledge.* Observe that, similar to the argument given for SCP $\mathcal{V}$ does not gain any knowledge of the committed values or the help values through the values $(X + Y) \bmod q$, $H_1 \bmod q$, and $H_2 \bmod q$. Moreover, the value $C(\cdot)^z \bmod p$ does not reveal any information about the value of z, at any stage of the procedure, because of the hardness of the discrete-log problem. □

We now use SCP introduced for secure comparison of two integers to present a novel, secure combinatorial auction for the single-minded case that preserves the privacy of each agent's bidding information even after the bidding phase is over, namely, STOUP.

4 STOUP

We first summarize our auction setting.

4.1 Auction Background

We are considering a situation where an auctioneer (AU), the seller itself, is interested in selling $M = \{1, \ldots, m\}$ indivisible items and there are $B = \{b_1, \ldots, b_{\hat{n}}\}$ ($|B| = \hat{n} \geq 2$) interested and strategic agents via a combinatorial auction. We assume there exists a set

[3]This property also makes the comparison *robust* to any misreporting done by the accountants. As we assume the accountants to be strategic-but-curious, they may strategically misreport information. However, Fig. 2 will allow any $\mathcal{V}$ to detect the misreporting. Thus, SCP (Fig.s. 1 and 2) is *robust* to any misreporting done by the accountants.

of privacy accountants N, that can assist AU in determining the winners and their payments. We denote the set consisting of every participating agent in this protocol as A i.e., $A = \{\{AU\} \cup B \cup N\}$.

Combinatorial auctions factor in the inter-dependency of the values to an agent with respect to the different combinations possible i.e., each agent has a different preference for different subsets. The valuation function ϑ_{b_i} describes these preferences $\forall b_i \in B$. In absence of payments, the agent b_i may boast about ϑ_{b_i}. We denote its payment as σ_{b_i}. Formally, for each possible subset $S \in 2^M$, ϑ_{b_i} is a real-valued function such that $\vartheta_{b_i}(S)$ is the value an agent b_i obtains if he wins the subset S. Also, if σ_{b_i} is the price paid by the agent for the subset, then its utility is given by $\psi_{b_i}(\cdot) = \vartheta_{b_i}(S) - \sigma_{b_i}(\cdot)$.

4.1.1 Cryptographic Properties in Auction. We now describe the required cryptographic properties of an auction protocol.

- *Non-repudiation.* This deals with the inability of an auctioneer or an agent to retract from their actions. Auction protocols must be able to commit an agent to its bid and prove the exclusion of any bid by the auctioneer.
- *Verifiability.* The public, including the agents, must be shown conclusive proof of the correctness of the auction protocol. The protocol must enforce correctness; an auctioneer should not present valid proofs for invalid winners or incorrect payments.
- *Privacy.* An auction protocol should hide the bidding information of an agent from the other participating agents. After the auction, only the information revealed from the winning agents should be known. The types of privacies relevant for an auction are defined below. For this, let W be the set of winning agents.

  Definition 4.1 (Agent Privacy). *No agent should be able to discover each others identity i.e., for an agent $a \in A$ during the auction and for an agent $a \in A \setminus W$ after the auction, no other agent $b \in A \setminus \{a, AU\}$ should know about a's participation in the auction.*

  Definition 4.2 (Bid Privacy). *No agent should be able to know any agent's bid valuation i.e., the probability with which an agent $a \in A \setminus \{b_i\}$ can guess agent b_i's bid valuation ϑ_{b_i} is $\ll 1/\vartheta_{b_i}$.*

  Definition 4.3 (Bid-Topology Privacy). *No agent should be able to know any other agent's bundle of items i.e., the probability with which an agent $a \in A \setminus \{b_i\}$ can guess the item bundle S_{b_i} of an agent $b_i \in B \setminus \{a\}$ during the auction and of an agent $b_i \in B \setminus \{\{a\} \cup W\}$ after the auction is negligible in the number of items being auctioned.*

Let us say that the allocation of the items is determined by an allocation rule $\mathbf{k}(\cdot)$, which takes $\vartheta = (\vartheta_{b_i}, \vartheta_{-b_i})$ as the input and outputs who gets which items, where ϑ_{-b_i} denotes the set of valuations of agents not including b_i. The payment rule is given by $\sigma = (\sigma_{b_1}(\cdot), \sigma_{b_2}(\cdot), \ldots, \sigma_{b_n}(\cdot))$. Thus, an auction is characterized by $(\mathbf{k}, \sigma)$, an allocation rule and the payment rule. Given an auction, we need the following game theoretic properties to be satisfied.

4.1.2 Game Theoretic Properties in Auction. The valuation of each agent is its private information, i.e., hidden from every other agent in the auction. This opens the door for any such agent to lie about their valuations for their benefit. Thus, we look for auctions that incentivize an agent to bid for its true valuation. In mechanism design theory, such truthful auctions are called *dominant strategy incentive compatible (DSIC)*. Further, an auction is *ex-post individually rational (IR)* if every agent b_i always gets non-negative *utility*, i.e., the difference in the items it wins with the price it pays for them is non-negative.

Algorithm 1: ICA-SM Algorithm

(1) *Initialization:*
- Sort the agents according to the order : $\vartheta^*_{b_1}/\sqrt{|S^*_{b_1}|} \geq \vartheta^*_{b_2}/\sqrt{|S^*_{b_2}|} \geq \cdots \geq \vartheta^*_{b_{\hat{n}}}/\sqrt{|S^*_{b_{\hat{n}}}|}$
- $W \leftarrow \emptyset$

(2) *For $i : 1 \rightarrow \hat{n}$,* if $S^*_{b_i} \cap (\cup_{b_j \in W} S^*_{b_j}) = \emptyset$ then $W \leftarrow W \cup \{b_i\}$

(3) *Output:*
- *Allocation:* The set of winners is W.
- *Payments:* $\forall b_i \in W, \sigma_{b_i} = \vartheta^*_{b_j}/\sqrt{|S^*_{b_j}|/|S^*_{b_i}|}$ where j is the smallest index such that $S^*_{b_i} \cap S^*_{b_j} \neq \emptyset$, and for all $k < j$, $b_k \neq b_i$, $S^*_{b_k} \cap S^*_{b_j} = \emptyset$. If no such j exists then $\sigma_{b_i} = 0$.

The allocation problem in this setting is *NP-Complete*. Also, it is challenging to represent and communicate the valuation functions of each agent (since these are exponential in size). Thus, we look for much simpler cases of auctions, such as the single-minded case.

4.1.3 The Single-Minded Case. These are auctions wherein agents are interested in a single specific bundle of items and get a scalar value if they get this whole bundle (or any super-set) and get zero value for any other bundle. Formally,

Definition 4.4. *A single-minded valuation function is a function in which there exists a bundle of items S^* and a value ϑ^* such that $\vartheta(S) = \vartheta^*$, $\forall S \supseteq S^*$ and $\vartheta(S) = 0$ for all other S. Here, a single-minded bid is the pair (S^*, ϑ^*).*

As the allocation problem, in this case, is *NP-Hard* [22], we look at algorithms that can solve this approximately.

4.1.4 Incentive Compatible approximation Algorithm (ICA-SM) [22]. Algorithm 1 describes ICA-SM, which is a *greedy* algorithm that solves the allocation problem for single-minded case with $\hat{n}$ agents, m items, ϑ_{b_i} and S_{b_i} as agent b_i's bid valuation and preferred bundle of items, with W as the set of winners approximately. ICA-SM is *computationally efficient*, *incentive compatible* and is $\sqrt{m}$-approximate [22].

Definition 4.5 (Trustworthy Implementation). *An auction protocol that provides non-repudiation and verifiability while preserving agent, bid, and bid-topology privacy and being dominant strategy incentive compatible and individually rational is a trustworthy implementation of an auction.*

4.2 STOUP: Protocol

In STOUP, A is a set of agents wherein AU is the seller itself. All arithmetic operations (except the payments) are modulo p for the commitments and modulo q for the values to be committed and the help values. Further, AU acts as the CS. As aforementioned, we assume that AU is honest-but-curious while the bidders and the set of accountants strategic-but-curious.

Item Bundle. In STOUP, an agent b_i submits its *item bundle* $\mathbb{S}_{b_i}$, consisting of commitments of its preferred items *at least* once as

Protocol 1: STOUP

procedure Authentication Phase

Each agent $a \in A \setminus \{AU\}$ gives its public id's to AU

AU assigns each agent a a secret identifier id_a

AU generates a random id for each item

AU randomly assigns $\left(n^1_{id_{b_i}}, n^2_{id_{b_i}}\right) \in N$ to each $id_{b_i} \in B$

end procedure

procedure Bidding Phase

Each bidder $id_{b_i} \in B$ submits $\mathbf{BT}_{id_{b_i}}$ to SC

Each bidder $id_{b_i} \in B$ sends $\left(u, r, d_{id_{b_i}}\right)$ to $n^1_{id_{b_i}}$ and $\left(v, r', d_{id_{b_i}}\right)$ to $n^2_{id_{b_i}}$ for $w_{id_{b_i}}$ and $\mathbb{S}_{id_{b_i}}$ with SC as described in Fig. 1

end procedure

procedure Winner Determination Phase

AU determines – in co-ordination with the assigned accountants – the set of the winning bidders W consisting of each winner's identifier, and calculates payments as defined in Algorithm 1

AU submits W and the payments with SC

end procedure

well as *different* commitments of some (or all) of their preferred items randomly such that $|\mathbb{S}_{b_i}| = m,\ \forall b_i \in B$.

Definition 4.6 (Item Bundle). *An agent b_i's item bundle is defined as $\mathbb{S}_{b_i} = \{C \cup D\}$ where $C = \{C(R(j)) \mid \forall j \in S_{b_i}\}$ and $D = \{C(R(k)) \mid \forall k \in S'_{b_i}\}$,*

where S'_{b_i} is the set of non-distinct items randomly chosen from S_{b_i} such that $|C| + |D| = m$.

Bid Tuple. With Definition 4.6, each bidder $b_i \in B$ participates in STOUP, by submitting the following bid tuple,

$$\mathbf{BT}_{b_i} = \left\langle C(\vartheta_{b_i}), C(|S_{b_i}|), C\big(R(\vartheta_{b_i}/\sqrt{|S_{b_i}|}), \mathbb{S}_{b_i}\right\rangle$$

where $w_{b_i} = \vartheta_{b_i}/\sqrt{|S_{b_i}|}$.

4.2.1 STOUP Protocol. For STOUP, similar to SCP, we consider a smart contract SC which allows agents to post and get relevant information. Protocol 1 illustrates STOUP with Figure 4 illustrating the protocol flow. As we use SCP for winner(s) and payment(s) determination, we require $w_{b_i} < \frac{q}{2 \cdot d_{max}}$, $\forall b_i$. We next describe how AU solves steps (1), (2) and (3) of Algorithm 1 in STOUP.

4.2.2 Bid Initialization. AU sorts the bids based on the values $w_{id_{b_i}}$ $\forall id_{b_i}$, using any comparison based sorting with the comparison done through SCP (Fig. 1). For the winner and payment determination phase, the highest agent's identifier is denoted as id_{b_1}, the second highest agent's as id_{b_2}, and so on. Let I consist of the set of identifiers $\{id_{b_1}, \ldots, id_{b_{\hat{n}}}\}$, $\mathbb{S}$ as the set of preferred item bundles of every agent $\{\mathbb{S}_{id_{b_1}}, \ldots, \mathbb{S}_{id_{b_{\hat{n}}}}\}$ and W as the set of winners initialized to $\emptyset$.

4.2.3 Winner Determination. AU carries out winner determination (Algorithm 1), in co-ordination with accountants. In this, the highest agent is automatically selected, and its *identifier* is added to W. To determine the other winners, AU compares every pair of item, $\forall\ id_{b_j} \in I \setminus \{id_{b_1}\}$ with every id_{b_k} currently in W, using Fig. 1.

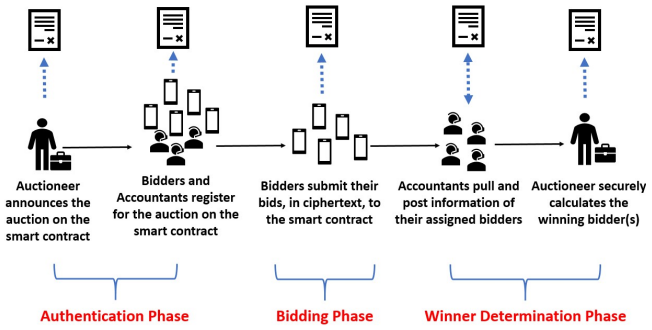

Figure 4: Overview of STOUP

If AU does not find any identical pair of items for an agent id_{b_j} for every id_{b_k} currently in W i.e., $\mathbb{S}_{id_{b_j}} \cap (\cup_{k \in W} \mathbb{S}_{id_{b_k}}) = \emptyset$, it adds id_{b_j} to W. Otherwise, it discards that agent and continues with the next highest agent.

Note. As the set of items M is finite, i.e., there are only $\binom{m}{2}$ distinct combinations possible, AU can deterministically get the items, x and y, being compared from the value $x - y$. By using SCP however, the AU will get the value $X + Y$. With this, if $x \neq y$, i.e., $X + Y \neq 0$, all possible $\binom{m}{2}$ combinations will be equally likely.

4.2.4 Payment Determination. The payments for every winner $id_{b_i} \in W$ are as described in Algorithm 1. AU can find out an agent id_{b_j}, $\forall\ id_{b_i} \in W$, where j is the smallest index such that $\mathbb{S}_{id_{b_i}} \cap \mathbb{S}_{id_{b_j}} \neq \emptyset$, and an agent id_{b_k} for $k < j$, $id_{b_k} \neq id_{b_i}$ such that $\mathbb{S}_{id_{b_k}} \cap \mathbb{S}_{id_{b_j}} = \emptyset$, similar to the procedure to the winner determination described in Section 4.2.3. If such id_{b_j} and id_{b_k} exists, then AU asks the assigned accountant $n^1_{id_{b_j}}$ of id_{b_j} to calculate the payment $\sigma_{id_{b_i}} = \vartheta_{id_{b_j}}/\sqrt{|S_{id_{b_j}}|/|S_{id_{b_i}}|}$. The agent id_{b_j} opens its commitment $C(R(w_{id_{b_j}}))$ for $n^1_{id_{b_j}}$, securely. AU asks id_{b_i} to open its commitment for $C(|S_{id_{b_i}}|)$, and sends the value to $n^1_{id_{b_j}}$, which calculates $\sigma_{id_{b_i}}$ and sends it to AU. If no such id_{b_j} or id_{b_k} exist, then $\sigma_{id_{b_i}} = 0$.

5 STOUP: SECURITY AND PRIVACY ANALYSIS

STOUP preserves non-repudiation since all the relevant information is submitted on the blockchain, an append-only ledger. We now look at verifiability and the nature of the privacy guarantees as provided by STOUP. In this section, we denote the identifier $id_{b_i} \in B$ as b_i for simplicity of notation.

Verifiability. A prover $\mathcal{P}$ (AU) proves to a verifier $\mathcal{V}$ the correctness of the order $w_{b_1} \geq \cdots \geq w_{b_{\hat{n}}}$ and the correctness of the comparisons for $\mathbb{S}_{b_i} \cap \mathbb{S}_{b_j} = \emptyset$, for each $b_i, b_j \in B$. As all values *as well as* item comparisons in STOUP, are done using SCP, the ZKP for the comparisons follows the same as described in Fig. 2[4].

[4] As Pedersen commitments are computationally binding, $\mathcal{V}$ does not require multiple proofs for different commitments of the same values. This significantly reduces the computational time as compared to [26, 31].

Privacy Analysis. STOUP provides the following privacy guarantees.

Proposition 5.1. *STOUP preserves agent privacy.*

Proposition 5.2. *STOUP preserves each agent's bid privacy.*

Proposition 5.3. *STOUP preserves bid and bid-topology privacy from the accountants.*

The proofs of the propositions follow simply by observing the information exchange in STOUP.

Lemma 5.1. *In STOUP, the probability with which AU can know at least one item in agent b_j's bid-topology is $1/s_{b_i}$. The probability with which AU can know the complete bid-topology of an agent b_j is,*

$$P_{b_j}(s_{b_i}) = \frac{1}{2^m - 2^{m-s_{b_i}}} \tag{3}$$

$\forall b_j \in B \setminus W$, such that $b_i \in W$ is that agent for which $S_{b_j} \cap S_{b_i} \neq \emptyset$ in Step 2 of Algorithm 1, $s_{b_i} = |S_{b_i}|$ and m is the number of items.

The above lemma follows by observing that *AU* through the bidding topology of the winners and its knowledge about which agents have at least one item in common, can infer some information about the bid-topology of an agent $b_j \in B \setminus W$. We then get the probability defined in Eq. 3 by eliminating the subsets which do not comprise the shared item.

From Eq. 3, STOUP preserves bid-topology privacy with high probability when $s_{b_i} \geq 2, \forall b_i \in W$. For the analysis of the result, observe that Eq. 3 can be written as,

$$P_{b_j}(s_{b_i}) = \frac{1}{2^m - 2^{m-s_{b_i}}} = \frac{2^{s_{b_i}}}{2^{s_{b_i}} - 1}\left(\frac{1}{2^m}\right).$$

Thus, the increase in the probability with which *AU* can determine the complete bid-topology of an agent with respect to randomly guessing the complete bid-topology is by a *constant factor*, i.e., by $\frac{2^{s_{b_i}}}{2^{s_{b_i}}-1}$. Assuming that each agent's bundle size is ≥ 2, the worst case follows when $s_{b_i} = 2$. The probability that *AU* can know the complete bid-topology of an agent b_j in this case is, $P_{b_j}(s_{b_i}) = \frac{4}{3}\left(\frac{1}{2^m}\right)$, which is an increase by a factor $\frac{4}{3}$ or an increase by 33.33% of $O(\frac{1}{2^m})$ which is negligible in m.

The probability result follows from the fact that at no point during the auction or post-auction and $\forall b_j \in B \setminus W$, the cardinality of the preferred bundle of items of an agent b_j, i.e., s_{b_j}, is revealed to *AU* in STOUP. Note that Eq. 3 does not hold for an auction protocol that leaks the cardinality of S_{b_j} of an agent b_j. For instance, if *AU* knew that for an agent b_j, $s_{b_j} = m$, the probability with which agent b_j's bid-topology is leaked to *AU* would be 1. Lastly, combining these privacy guarantees implies the following theorem.

Theorem 5.1. *STOUP is a trustworthy implementation of ICA-SM.*

6 STOUP: IMPLEMENTATION

To avoid any floating-point number, *AU* can announce at the start of the auction, that w_{b_i} for every party b_i, will have *x-precision* i.e., each value w_{b_i} will be significant up to x decimal places.

Simulation Analysis. We generate all auction instances as a CATS file using the SATS command-line tool [42]. We calculate the optimal social welfare by solving the winner determination problem for the general single-minded case through FRODO 2.0 [21]. For this calculation, the generated CATS file is parsed through the in-built FRODO 2.0 parser to convert it to XCSP. The XCSP file is then solved using optimal algorithms (such as DPOP, P-DPOP, etc.) provided in FRODO 2.0 (through GUI or command line). Further, the primes p and q are of size 1024 bits. We use a quad-core Intel i5-4210U CPU with a 1.70GHz processor and 8GB RAM for the simulations. We also assume no latency in inter-party communication. Consequently, the computational bottleneck of STOUP corresponds to the verification of *every* value and item comparison, i.e., Fig. 2. Table 2 presents the results. Note that, for large $\hat{n}$ it is difficult to calculate the optimal welfare[5], as the problem is NP-Hard.

$\hat{n}$	m	Upper Bound	$\frac{\text{Optimal Welfare}}{\text{Approximate Welfare}}$	Time Taken (mins)
25	9	3	1.11905993576	2.1826
25	12	3.4641	1.1313692063	5.21355
25	15	3.8729	1.05711039103	11.103467
100	9	3	-	11.59642
100	12	3.4641	-	19.72178
100	15	3.8729	-	54.084380

Table 2: STOUP bound for 25 random auction instances

As stated, the mean time taken for STOUP in Table 2 includes the verification of every value and item comparison done throughout the execution of STOUP. However, the time consumed for verification of the value and item comparisons is significantly less than other secure auction protocols such as [31]. For comparison, a 100 bid *single-item* auction (i.e., $\hat{n} = 100$ and $m = 1$) takes approximately 2.51 hours in [31] (see [31, Table 2]), while a 100 bid *single minded* combinatorial auction (i.e., $\hat{n} = 100$) even with $m = 15$, only takes approximately 0.91 hours in STOUP. This decrease in the run-time shows the practicality of SCP.

Gas Consumption. A smart contract is compiled as bytecode and executed on the Ethereum Virtual Machine (EVM). EVM charges a fee per computational step executed in a contract or transaction to prevent deliberate attacks and abuse on the Ethereum network. This fee is measured in terms of *gas* units.

The estimate depends on the post operations described in STOUP. E.g., each bidder submitting its bid tuple and other required information. Then, *AU* and the accountants further exchange information on-chain. As the EVM uses 256-bit as default, changing p does not affect the estimate. Smaller p's will result in greater gas consumption, as the EVM "downscales" the values. Typically, the gas associated with a post-operation for the *uint256* variable in *Solidity* is ≈ 62664. With this in STOUP, per comparison, *AU* will consume 313320 while each accountant consumes 501312. Each participating bidder will at-worst consume $(3 + m) \cdot 62664$ gas units.

7 CONCLUSION

This paper observed that Yao's Millionaires' Problem (YMP) is fundamental to designing secure AI applications. Towards this, we presented a practical, and verifiable solution to YMP, namely, SCP

[5]Game-theoretically, for an auction, *social welfare* is the summation of all the winning bidders' valuations.

(Figures 1 and 2). SCP uses third-party agents to securely compare two integers that do not learn any information (Theorem 3.1). Significantly, SCP achieves the comparison in constant time and one execution of Figure 1.

To demonstrate the effectiveness of SCP, we use it to design a Secure, Truthful cOmbinatorial aUction Protocol (STOUP) for single-minded bidders (Protocol 1). STOUP preserves an agent's bid valuation as well as bid-topology at any time during the auction and post-auction, even to the auctioneer, unlike prior works. The bid-topology is preserved with high probability when every agent's bundle size is ≥ 2, which is a fair assumption in practice for combinatorial auctions (Lemma 5.1). We further believe that SCP will find an application for other secure AI applications, including different auctions, voting, distributed optimization, etc.

REFERENCES

[1] Mark Abspoel, Niek J Bouman, Berry Schoenmakers, and Niels de Vreede. 2019. Fast secure comparison for medium-sized integers and its application in binarized neural networks. In *Cryptographers' Track at the RSA Conference*. Springer, 453–472.

[2] Donald Beaver and Shaft Goldwasser. 1989. Multiparty computation with faulty majority. In *Conference on the Theory and Application of Cryptology*. Springer, 589–590.

[3] Ian F Blake and Vladimir Kolesnikov. 2004. Strong conditional oblivious transfer and computing on intervals. In *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 515–529.

[4] Felix Brandt and Tuomas Sandholm. 2005. Efficient privacy-preserving protocols for multi-unit auctions. In *International Conference on Financial Cryptography and Data Security*. Springer, 298–312.

[5] Christian Cachin. 1999. Efficient private bidding and auctions with an oblivious third party. In *Proceedings of the 6th ACM conference on Computer and communications security*. 120–127.

[6] David Chaum, Ivan B Damgård, and Jeroen Van de Graaf. 1987. Multiparty computations ensuring privacy of each party's input and correctness of the result. In *Conference on the Theory and Application of Cryptographic Techniques*. Springer, 87–119.

[7] Geoffroy Couteau. 2016. Efficient Secure Comparison Protocols. *IACR Cryptology ePrint Archive* 2016 (2016), 544.

[8] Sankarshan Damle, Boi Faltings, and Sujit Gujar. 2019. A Practical Solution to Yao's Millionaires' Problem and Its Application in Designing Secure Combinatorial Auction. *CoRR* abs/1906.06567 (2019). arXiv:1906.06567

[9] Sankarshan Damle, Boi Faltings, and Sujit Gujar. 2019. A Truthful, Privacy-Preserving, Approximately Efficient Combinatorial Auction For Single-minded Bidders.. In *AAMAS*. 1916–1918.

[10] Roger Dingledine, Nick Mathewson, and Paul Syverson. 2004. *Tor: The second-generation onion router*. Technical Report. Naval Research Lab Washington DC.

[11] Marc Fischlin. 2001. A cost-effective pay-per-multiplication comparison method for millionaires. In *Cryptographers' Track at the RSA Conference*. Springer, 457–471.

[12] Oded Goldreich and Yair Oren. 1994. Definitions and properties of zero-knowledge proof systems. *Journal of Cryptology* 7, 1 (1994), 1–32.

[13] L. Gong, S. Li, C. Wu, and D. Wang. 2018. Secure "Ratio" Computation and Efficient Protocol for General Secure Two-Party Comparison. *IEEE Access* 6 (2018), 25532–25542.

[14] Dima Grigoriev, Laszlo B Kish, and Vladimir Shpilrain. 2017. Yao's millionaires' problem and public-key encryption without computational assumptions. *International Journal of Foundations of Computer Science* 28, 04 (2017), 379–389.

[15] Dima Grigoriev and Vladimir Shpilrain. 2014. YAO'S MILLIONAIRES'PROBLEM AND DECOY-BASED PUBLIC KEY ENCRYPTION BY CLASSICAL PHYSICS. *International Journal of Foundations of Computer Science* 25, 04 (2014), 409–417.

[16] Tal Grinshpoun and Tamir Tassa. 2014. A privacy-preserving algorithm for distributed constraint optimization. In *AAMAS*. International Foundation for Autonomous Agents and Multiagent Systems, 909–916.

[17] Ioannis Ioannidis and Ananth Grama. 2003. An efficient protocol for Yao's millionaires' problem. In *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*. IEEE, 6–pp.

[18] P. Kaghazgaran and B. Sadeghyan. 2011. Secure two party comparison over encrypted data. In *2011 World Congress on Information and Communication Technologies*. 1123–1126.

[19] Maya Larson, Chunqiang Hu, Ruinian Li, Wei Li, and Xiuzhen Cheng. 2015. Secure auctions without an auctioneer via verifiable secret sharing. In *Proceedings of the 2015 Workshop on Privacy-Aware Mobile Computing*. ACM, 1–6.

[20] Thomas Léauté. 2011. *Distributed Constraint Optimization: Privacy Guarantees and Stochastic Uncertainty*. PhD Thesis. Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland. http://thomas.leaute.name/main/DCOP_privacy_uncertainty_thesis.html

[21] Thomas Léauté, Brammert Ottens, and Radoslaw Szymanek. 2009. FRODO 2.0: An open-source framework for distributed constraint optimization. In *Proceedings of the IJCAI" 09 Distributed Constraint Reasoning Workshop (DCR" 09)*. 160–164.

[22] Daniel Lehmann, Liadan Ita Oćallaghan, and Yoav Shoham. 2002. Truth revelation in approximately efficient combinatorial auctions. *Journal of the ACM (JACM)* 49, 5 (2002), 577–602.

[23] Hsiao-Ying Lin and Wen-Guey Tzeng. 2005. An efficient solution to the millionaires' problem based on homomorphic encryption. In *International Conference on Applied Cryptography and Network Security*. Springer, 456–466.

[24] Xin Liu, Shundong Li, XiuBo Chen, Gang Xu, Xiaolin Zhang, and Yong Zhou. 2017. Efficient solutions to two-party and multiparty millionaires' problem. *Security and Communication Networks* 2017 (2017).

[25] John McMillan. 1994. Selling spectrum rights. *Journal of Economic Perspectives* 8, 3 (1994), 145–162.

[26] Silvio Micali and Michael O Rabin. 2014. Cryptography miracles, secure auctions, matching problem verification. *Commun. ACM* 57, 2 (2014), 85–93.

[27] Hiraku Morita, Nuttapong Attrapadung, Tadanori Teruya, Satsuya Ohata, Koji Nuida, and Goichiro Hanaoka. 2018. Constant-round client-aided secure comparison protocol. In *European Symposium on Research in Computer Security*. Springer, 395–415.

[28] Moni Naor, Benny Pinkas, and Reuban Sumner. 1999. Privacy preserving auctions and mechanism design. In *Proceedings of the 1st ACM conference on Electronic commerce*. ACM, 129–139.

[29] Raz Nissim and Ronen Brafman. 2014. Distributed heuristic forward search for multi-agent planning. *Journal of Artificial Intelligence Research* 51 (2014), 293–332.

[30] U.S. Department of Commerce, National Institute of Standards, and Technology. 2012. *Secure Hash Standard - SHS: Federal Information Processing Standards Publication 180-4*. CreateSpace Independent Publishing Platform, North Charleston, SC, USA.

[31] David C Parkes, Michael O Rabin, Stuart M Shieber, and Christopher Thorpe. 2008. Practical secrecy-preserving, verifiably correct and trustworthy auctions. *Electronic Commerce Research and Applications* 7, 3 (2008), 294–312.

[32] David C Parkes, Michael O Rabin, and Christopher Thorpe. 2009. Cryptographic combinatorial clock-proxy auctions. In *International Conference on Financial Cryptography and Data Security*. Springer, 305–324.

[33] Torben Pryds Pedersen. 1991. Non-interactive and information-theoretic secure verifiable secret sharing. In *Annual International Cryptology Conference*. Springer, 129–140.

[34] Stephen J Rassenti, Vernon L Smith, and Robert L Bulfin. 1982. A combinatorial auction mechanism for airport time slot allocation. *The Bell Journal of Economics* (1982), 402–417.

[35] Michael H Rothkopf, Aleksandar Pekeč, and Ronald M Harstad. 1998. Computationally manageable combinational auctions. *Management science* 44, 8 (1998), 1131–1147.

[36] Tuomas Sandholm. 1999. An algorithm for optimal winner determination in combinatorial auctions. (1999).

[37] Koutarou Suzuki and Makoto Yokoo. 2002. Secure combinatorial auctions by dynamic programming with polynomial secret sharing. In *International Conference on Financial Cryptography*. Springer, 44–56.

[38] Tamir Tassa, Tal Grinshpoun, and Roie Zivan. 2017. Privacy preserving implementation of the Max-Sum algorithm and its variants. *Journal of Artificial Intelligence Research* 59 (2017), 311–349.

[39] Tamir Tassa, Roie Zivan, and Tal Grinshpoun. 2015. Max-Sum Goes Private.. In *IJCAI*, Vol. 1360. 425–431.

[40] Tamir Tassa, Roie Zivan, and Tal Grinshpoun. 2016. Preserving Privacy in Region Optimal DCOP Algorithms.. In *IJCAI*. 496–502.

[41] Yiannis Tsiounis and Moti Yung. 1998. On the security of ElGamal based encryption. In *International Workshop on Public Key Cryptography*. Springer, 117–134.

[42] Michael Weiss, Benjamin Lubin, and Sven Seuken. 2017. Sats: A universal spectrum auction test suite. In *AAMAS*. 51–59.

[43] Wikipedia contributors. 2018. Smart contract — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Smart_contract.

[44] Gavin Wood et al. 2014. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper* 151, 2014 (2014), 1–32.

[45] Andrew C Yao. 1982. Protocols for secure computations. In *Foundations of Computer Science, 1982. SFCS'08. 23rd Annual Symposium on*. IEEE, 160–164.

[46] Y. Yao, J. Wei, J. Liu, and R. Zhang. 2016. Efficiently secure multiparty computation based on homomorphic encryption. In *2016 4th International Conference on Cloud Computing and Intelligence Systems (CCIS)*. 343–349.