DTB18-Assignment 1

Problem Statement

Develop a smart contract that will conduct a single-minded combinatorial auction for a set of $B = \{1, \ldots, n\}$ bidders and distinct $M = \{1, \ldots, m\}$ items.

The auction will consist of an auctioneer, bidders and a set of trusted third parties (notaries). Each of these entities will be represented by a distinct public address.

Preliminaries

The notations and definitions required for the implementation of the auction are as follows:

1. Each bidder i will provide as an input to the contract all the items in which it is interested in represented by the set S_i. In addition, i will also provide as an input the value $w_i = \vartheta_i/|S_i|$, where ϑ_i is its valuation of the items.

2. However, instead of directly providing their items and the value w as inputs, each bidder will input their *random representations*.

   A random representation of a number x is the pair (u, v) such that $x = (u + v) \pmod q$. For this, first randomly choose u and then pick $v = (x - u) \pmod q$. Here, q is a large prime.

3. For example, if a bidder A is interested in the items $\{1, 2\}$ and for it $w_A = 17$, it may input the following values: $\{(18, 2), (5, 16)\}$ and $(8, 9)$.

4. For implementation purpose, consider only integer values.

Auction Protocol

The smart contract must follow the following protocol:

1. **Step 1:** Auctioneer will invoke the smart contract. It will announce the value q and the set M.

2. **Step 2:** Each notaries must register for the auction through their public addresses on the smart contract. Each address must be distinct.

3. **Step 3:** Each bidder registers for the auction through its public address on the smart contract and provides its set of items and value w.

   Note, these are random representations as described earlier.

4. **Step 4:** Auctioneer randomly assigns each bidder *a distinct notary* as soon as they register for the auction.

   Note, all values given as an input by the bidder must be passed to its assigned notary.

5. **Step 5:** Auctioneer finds the winners and submits them on the smart contract.

Winner Determination

The auctioneer determines the set of winners (W) and their payments (σ) i.e., Step 5 of the protocol, through the following algorithm:

Algorithm 1: Algorithm

---

1. *Initialization:*

   - Sort the bidders according to the order : $\vartheta_1/\sqrt{|S_1|} \geq \vartheta_2/\sqrt{|S_2|} \geq \ldots \geq \vartheta_n/\sqrt{|S_n|}$
   - $W \leftarrow \emptyset$

2. *For* $i : 1 \rightarrow n$, if $S_i \cap (\cup_{j \in W} S_j) = \emptyset$ then $W \leftarrow W \cup \{i\}$

3. *Output:*

   - *Allocation:* The set of winners is W.
   - *Payments:* $\forall i \in W, \sigma_i = \vartheta_j/\sqrt{|S_j|/|S_i|}$ where j is the smallest index such that $S_i \cap S_j \neq \emptyset$, and for all $k < j$, $k \neq i$, $S_k \cap S_j \neq \emptyset$. If no such j exists then $\sigma_i = 0$.

To sort the values in initialization you may choose any comparison based sorting method. The comparison, however, must be done through Procedure 1. Note that, these communications in Procedure 1 are done as transactions on the smart contract.

In addition the comparison of the items in the sets S_i and S_j is also done through Procedure 1.

Procedure 1: To compare values $x = (u_i, v_i)$ and $y = (u_j, v_j)$ of bidder i and j

Output: $x \overset{?}{\geq} y$

Steps

Auctioneer asks the assigned notaries to exchange amongst each other the values in the following manner: n_i receives the value u_j from n_j and n_j receives the value v_i from n_i

n_i calculates $(u_i - u_j)$ as val_1 and n_j calculates $(v_i - v_j)$ as val_2.

n_i sends val_1 and n_j sends val_2 to the auctioneer.

Auctioneer then checks the following,
if $val_1 + val_2 = 0$ *return* **equal**
if $val_1 + val_2 < q/2$ *return* $>$
else return $<$

Notary Payments

The notaries will be paid based on the amount of work they do. Each notary will be paid some constant amount (specified in Step 1 of the auction protocol) times the number of times the notary has communicated with the auctioneer throughout the auction.

The payment will be done through a smart contract method invoked after the end of Step 5.

Testing

You are also required to design and write test cases in truffle for the auction. The test cases must be exhaustive i.e., all possible cases, including the boundary cases, must be covered.

Scoring

The scoring will be based on the following:

1. Security, privacy and cost efficiency of smart contracts.
2. Test cases covered.
3. Bonus:
    - Implementation in Vyper[1].
    - Using Oraclize[2] in the contacts.

[1] https://vyper.readthedocs.io/en/latest/index.html
[2] https://docs.oraclize.it

Other Details

- Programming Language : Solidity
- Testing Platform : Truffle
- Assignment Deadline : 24^{th} September, 8:29 AM
- Plagiarism: All the submissions will be checked for copy cases. If caught, 0 mark will be awarded to all the members.