The interactive Coin Offering is a method which aim to solve the certainty of participation / certainty of valuation dilemma.

It allows the buyers to set a maximum valuation for the tokens to be sold.

If at the end of the sale, the total amount raised is higher than some of the buyer maximum valuations, buyers (starting with the ones which put the lowest valuation) are refunded until the remaining amount raised is equal to the lowest maximum valuation of the non-refunded buyers.

You can look at this paper for more details:

people.cs.uchicago.edu

[

](https://people.cs.uchicago.edu/~teutsch/papers/ico.pdf)

ico.pdf

288.13 KB

We made a smart contract implementation of this token sale method:

github.com

kleros/openiico-contract/blob/master/contracts/IICO.sol

/ @title Interactive Coin Offering * @author Clément Lesaege - clement@lesaege.com * This smart contract has undertaken two audits and two bounty programs. However, keep in mind that smart contracts still rely on experimental technology. /*

pragma solidity ^0.4.23;

import "zeppelin-solidity/contracts/token/ERC20/ERC20.sol";

/** @title Interactive Coin Offering * This contract implements the Interactive Coin Offering token sale as described in this paper: * https://people.cs.uchicago.edu/~teutsch/papers/ico.pdf * Implementation details and modifications compared to the paper: * -A fixed amount of tokens is sold. This allows more flexibility for the distribution of the remaining tokens (rounds, team tokens which can be preallocated, non-initial sell of some cryptographic assets). * -The valuation pointer is only moved when the sale is over. This greatly reduces the amount of write operations and code complexity. However, at least one party must make one or multiple calls to finalize the sale. * -Buckets are not used as they are not required and increase code complexity. * -The bid submitter must provide the insertion spot. A search of the insertion spot is still done in the contract just in case the one provided was wrong or other bids were added between when the TX got signed and executed, but giving the search starting point greatly lowers gas consumption. * -Automatic withdrawals are only possible at the end of the sale. This decreases code complexity and possible interactions between different parts of the code. * -We put a full bonus, free withdrawal period at the beginning. This allows everyone to have a chance to place bids with full bonus and avoids clogging the network just after the sale starts. Note that at this moment, no information can be taken for granted as parties can withdraw freely. * -Calling the fallback function while sending ETH places a bid with an infinite maximum valuation. This allows buyers who want to buy no matter the price not need to use a specific interface and just send ETH. Without ETH, a call to the fallback function redeems the bids of the caller.

This file has been truncated. show original