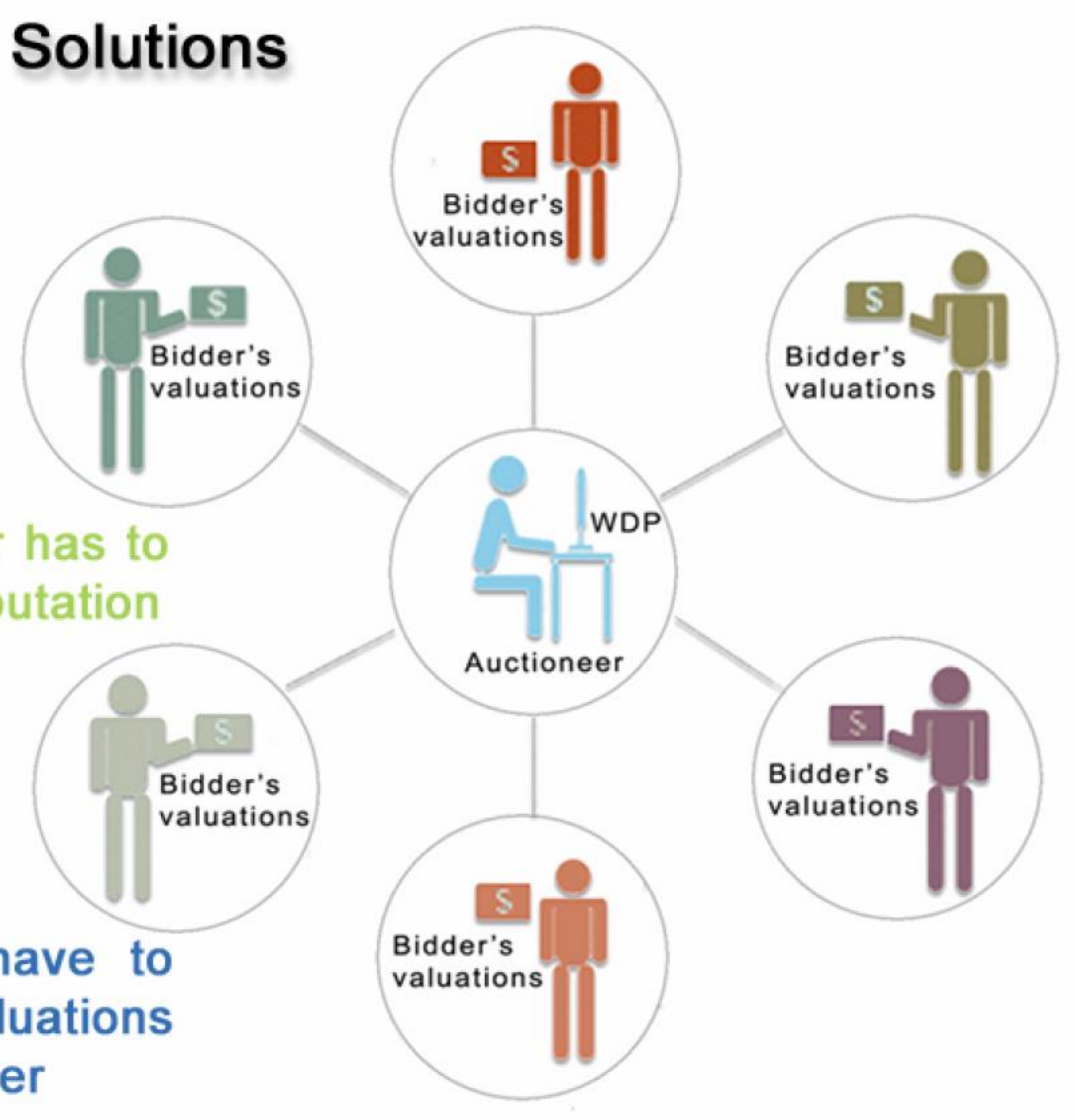
On Bidding Algorithms for a Distributed Combinatorial Auction

Prof. Benito Mendoza
Computer Engineering Technology

Note, this work was done as part of my PhD thesis at USC

COMBINATORIAL AUCTIONS are a great way to represent and solve distributed allocation problems. However, most of the existing winner determination algorithms for combinatorial auctions are centralized.

Centralized Solutions Drawbacks



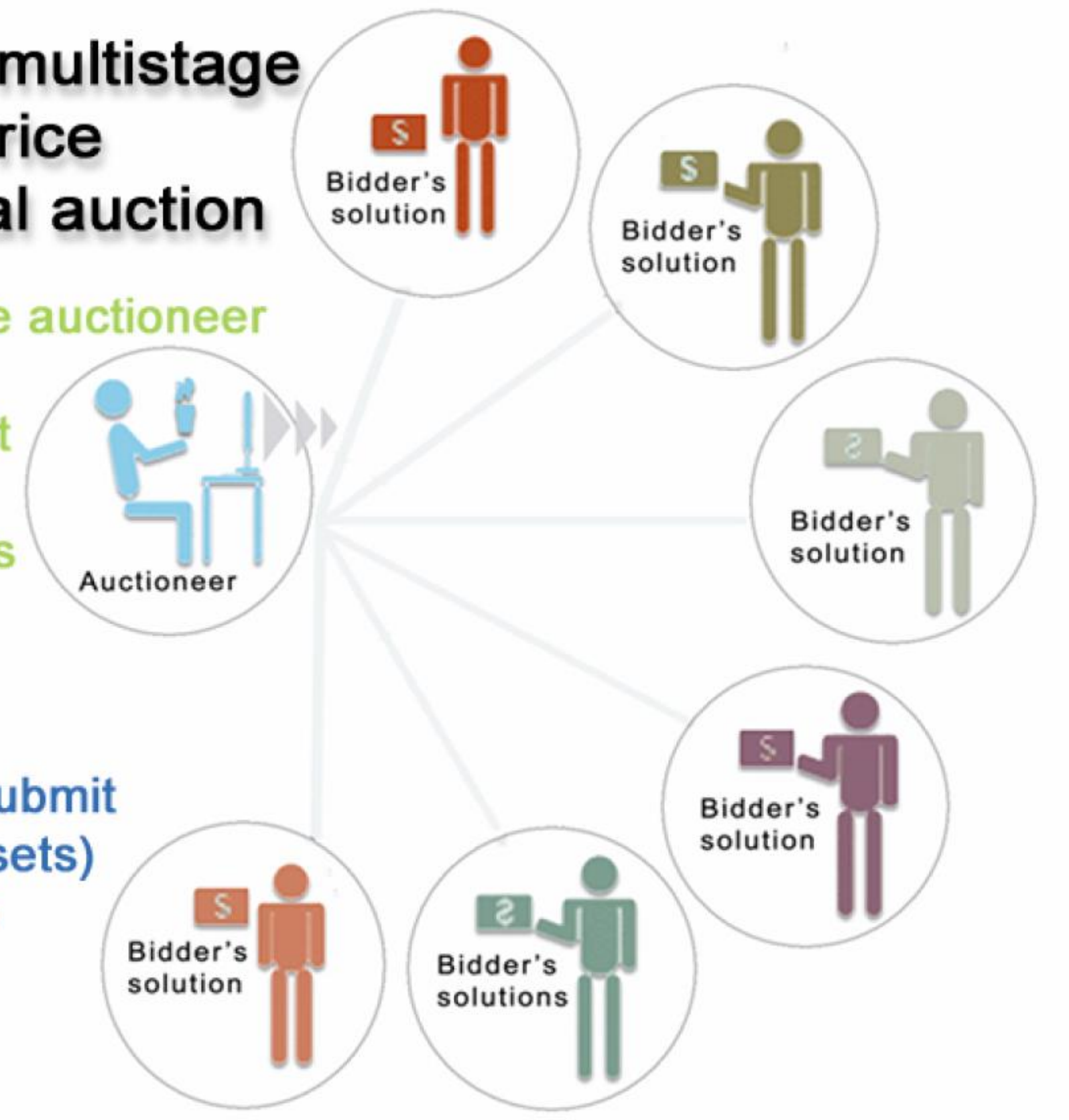
1 The auctioneer has to do all the computation

2 The bidders have to reveal their valuations to the auctioneer

We believe that distributed solutions to the winner determination problem (WDP) should be studied as they offer a better fit for some applications.

THE PAUSE AUCTION is one of a few efforts to release the auctioneer from having to do all the work. It naturally distributes the WDP amongst the bidders, and yet, it gives them an incentive to perform the calculation without revealing their true valuations unless necessary.

PAUSE is a multistage increasing price combinatorial auction



1 The task of the auctioneer is reduced to make sure that the bidders follow the rules

2 The bidders submit solutions (bidsets) that maximize their utility

A PAUSE AUCTION for m items has m **stages**. Stage 1 consists of having simultaneous ascending price open-cry auctions for each individual item. During this stage the bidders can only place individual bids on items. At the end of this stage we know what is the highest bid for each individual item and who placed that bid. In each successive stage $k = 2, 3, \ldots, m$ we hold an ascending price auction where the *bidders must submit sets of bids that cover all goods but each one of the bids must be for k goods or less*. The bidders are allowed to use bids that other agents have placed in previous rounds when placing their bid, thus allowing them to find better solutions. Also, any new bidset has to have a sum of bid prices which is bigger than the currently winning bidset. That is, revenue must increase monotonically.

Formally, let each **bid** b be composed of b^{items} which is the set of items the bid is over, b^{value} the value or price of the bid, and b^{agent} the agent that placed the bid. The agents maintain a set B of the **current best bids**, one for each set of items of size $\leq k$. At any point in the auction, after the first round, there will also be a set $W \subseteq B$ of **currently winning bids**. This is the set of bids that currently maximizes the revenue, where the **revenue** of W is given by

$$r(W) = \sum_{b \in W} b^{\text{value}}. \qquad (1)$$

Agent i's **value function** is given by $v_i(S) \in \Re$ where S is a subset of the items. Given an agent's value function and the current set of winning bids W we can calculate the agent's **utility** from W as

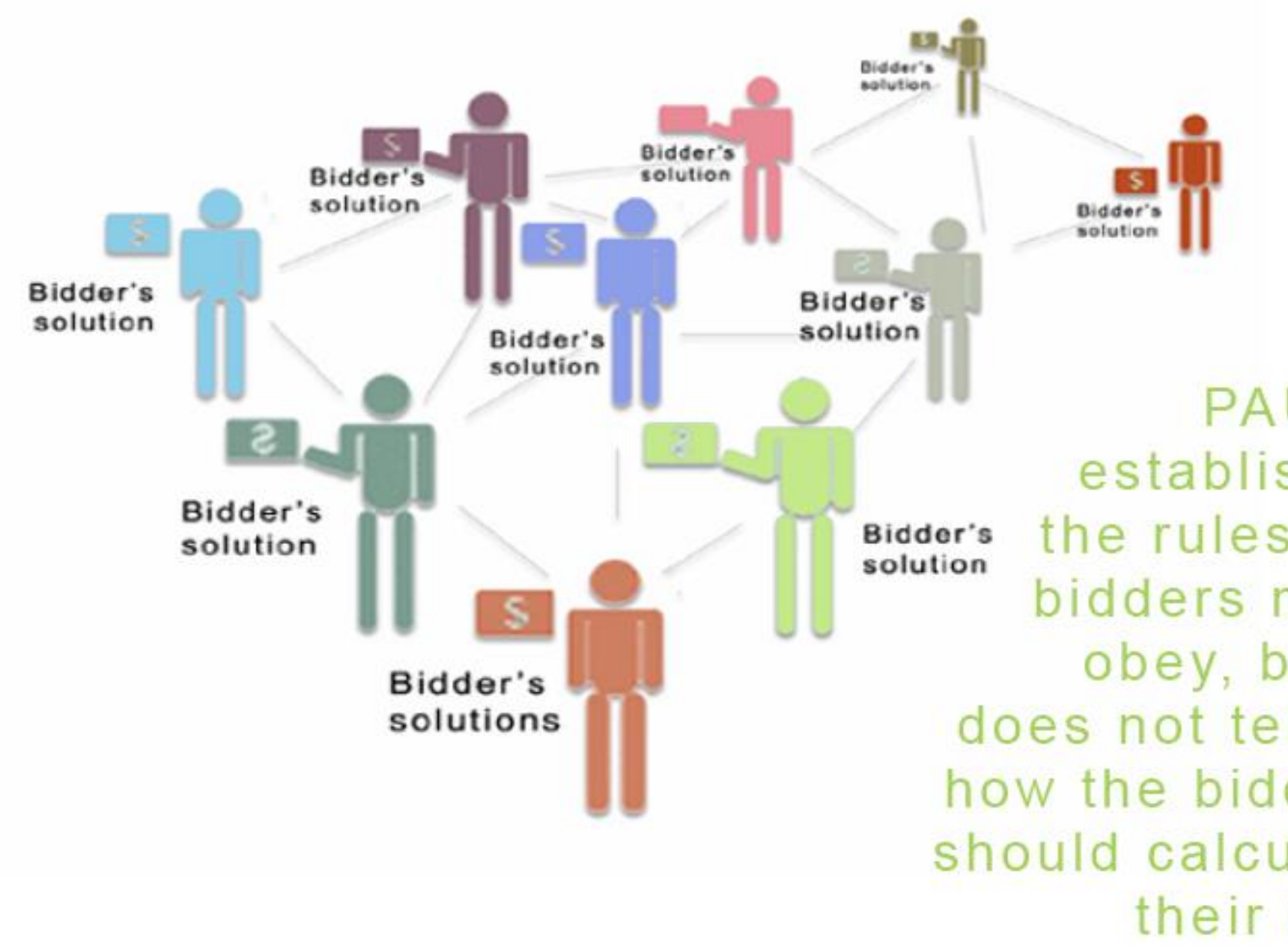
$$u_i(W) = \sum_{b \in W \,|\, b^{\text{agent}} = i} v_i(b^{\text{items}}) - b^{\text{value}}. \qquad (2)$$

Given that W is the current set of winning bids, agent i i must find a g_i^* such that

$$g_i^* = \arg\max_{g \subseteq 2^B} u_i(g), \qquad (3)$$

where each g is a set of bids that covers all items and $r(g) \geq r(W) + \varepsilon$ and $\forall_{b \in g}$ $(b \in B)$ or $(b^{\text{agent}} = i$ and $b^{\text{value}} > B(b^{\text{items}})$ and $size(b^{\text{items}}) \leq k)$, and where $B(items)$ is the value of the bid in B for the set $items$ (if there is no bid for those items it returns zero). That is, each bid b in g must satisfy at least one of the two following conditions. 1) b is already in B, 2) b is a bid of size $\leq k$ in which the agent i bids higher than the price for the same items in B. The goal of our algorithms is to find this g_i^*.

We envision to completely eliminate the auctioneer by having every agent perform the auctioneer's task



PAUSE establishes the rules the bidders must obey, but it does not tell us how the bidders should calculate their bids

What is needed is an algorithm to bid on the PAUSE auction.

We have developed two myopic utility maximizing algorithms for bidding agents on the PAUSE auction

This is an example of how an agent uses PAUSEBID to calculate its next bidset

PAUSEBID performs a branch and bound search completely from scratch each time it is invoked. It implements other pruning techniques in order to further reduce the size of the earch tree.

Agent 0

S	$V_0(S)$
{0}	$198
{1}	$44
{2}	$62
{0,1}	$849

W - Best Bidset

b^{items}	b^{agent}	b^{value}
{0}	0	$131
{1,2}	1	$270

$k = 2$ r(W)=401 $u^* = u_0(W) = 67$ $r(W) + \varepsilon = 402$

B - Best bids

b^{items}	b^{agent}	b^{value}
{0}	0	$131
{1}	1	$45
{2}	0	$45
{1,2}	1	$270

Heuristics

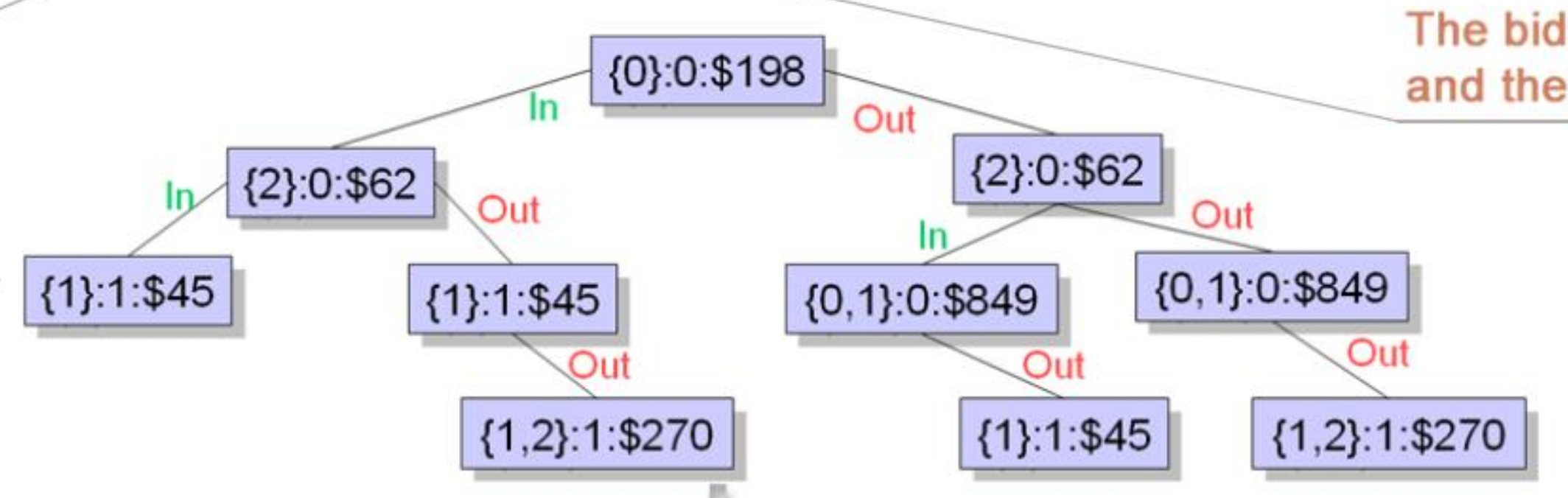
S	h(S)
{0}	$131
{1}	$135
{2}	$135

1.- PAUSEBID creates and ordered list of bids

bids= [{0}:0:$198, {2}:0:$62, {0,1}:0:$849, {1}:1:$45, {1,2}:1:$270

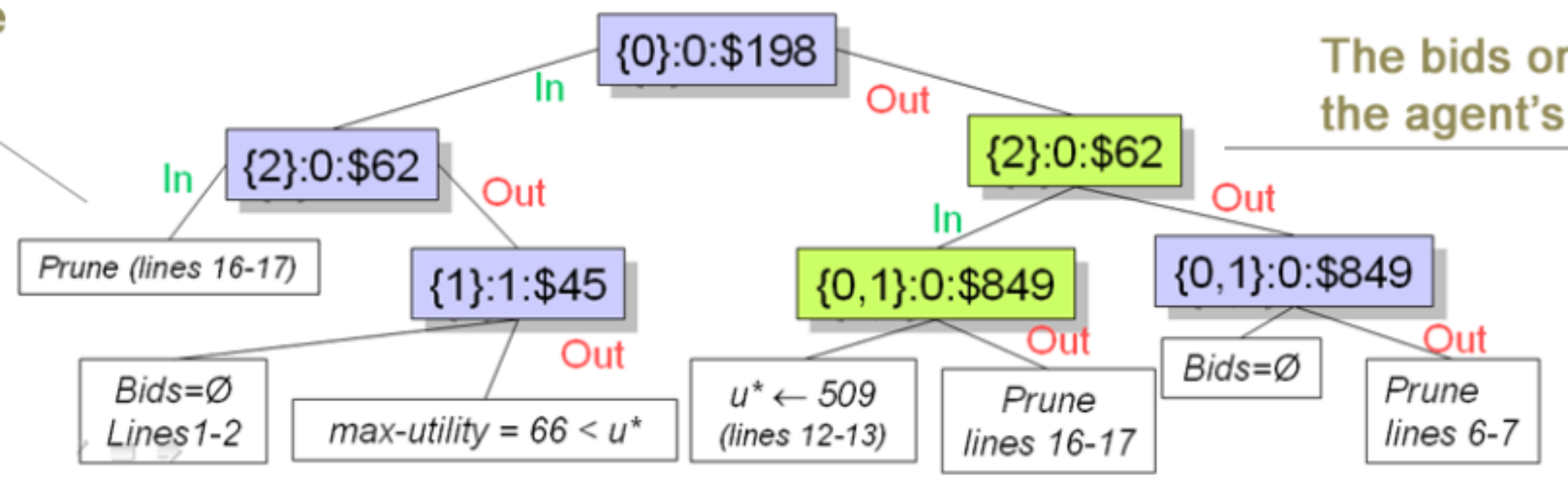
The bids of the agent are in the front and their value is the agent's valuation

This is the search tree created from BIDS.



2.- PBSEARCH prunes the tree and finds the solution

The bids on the solution found still have the agent's valuation as value



Agent 0

S	$V_0(S)$
{0}	$198
{1}	$44
{2}	$62
{0,1}	$849

$W=g^*_i$

b^{items}	b^{agent}	b^{value}
{2}	0	$45
{0,1}	0	$357

$k = 2$ r(W)=402 $u^* = u_0(W) = 509$ $r(W) + \varepsilon = 403$

B - Best bids

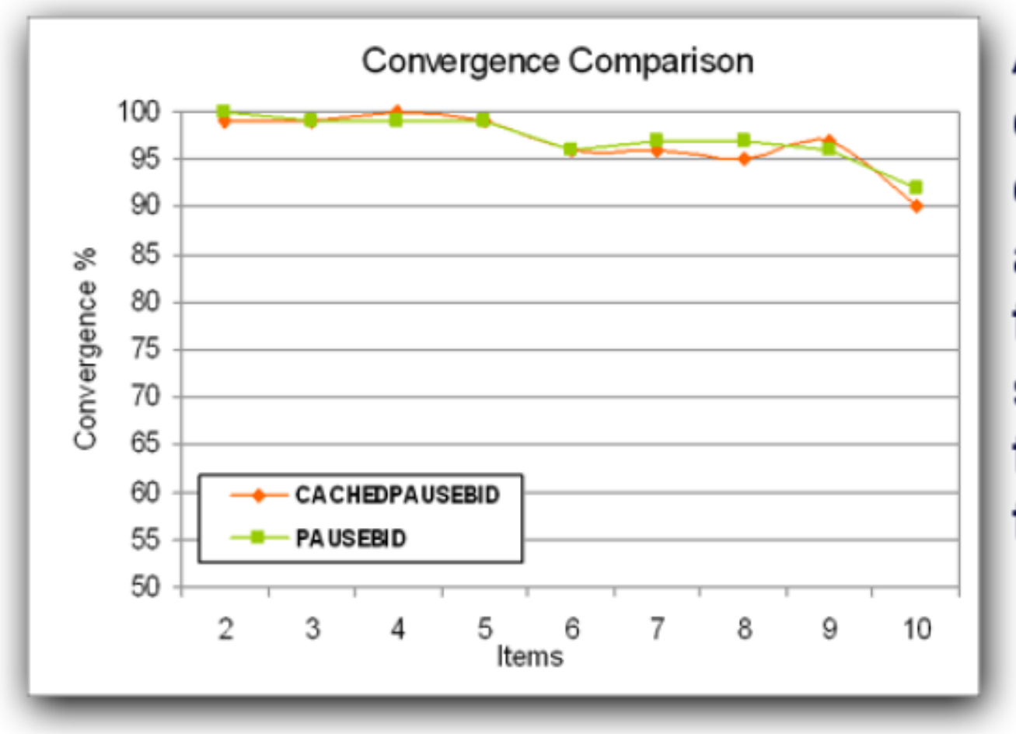
b^{items}	b^{agent}	b^{value}
{0}	0	$131
{1}	1	$45
{2}	0	$45
{1,2}	1	$270
{0,1}	0	$357

Heuristics

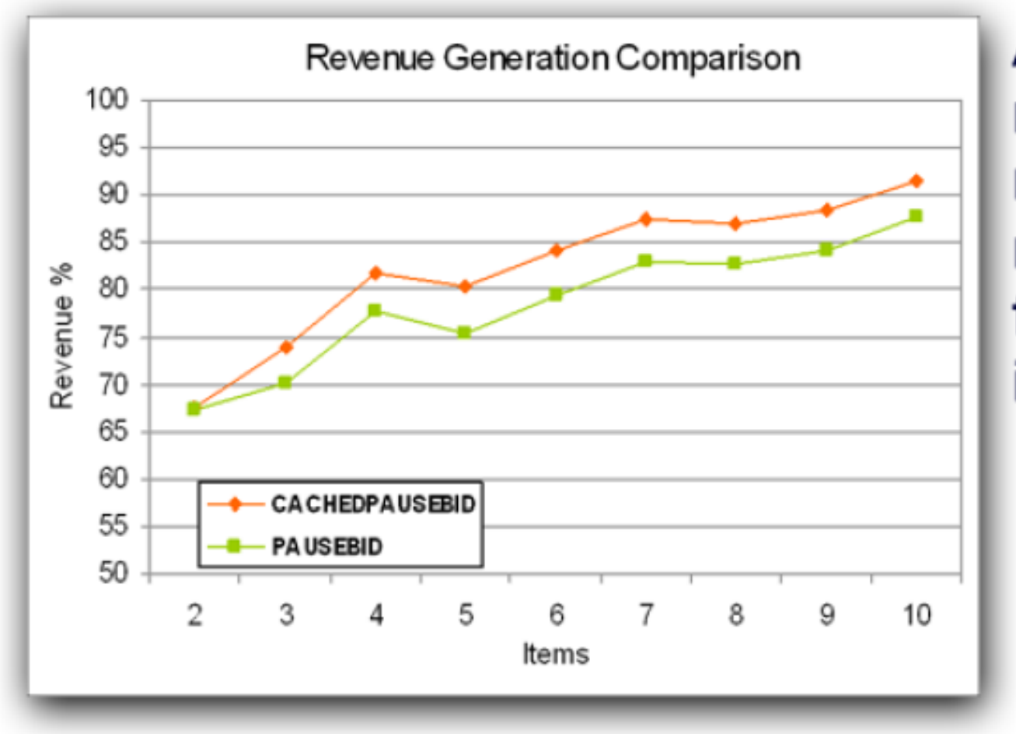
S	h(S)
{0}	$178.5
{1}	$178.5
{2}	$135

We have chosen to distribute the payments in proportion to the agent's true valuation for each set of items

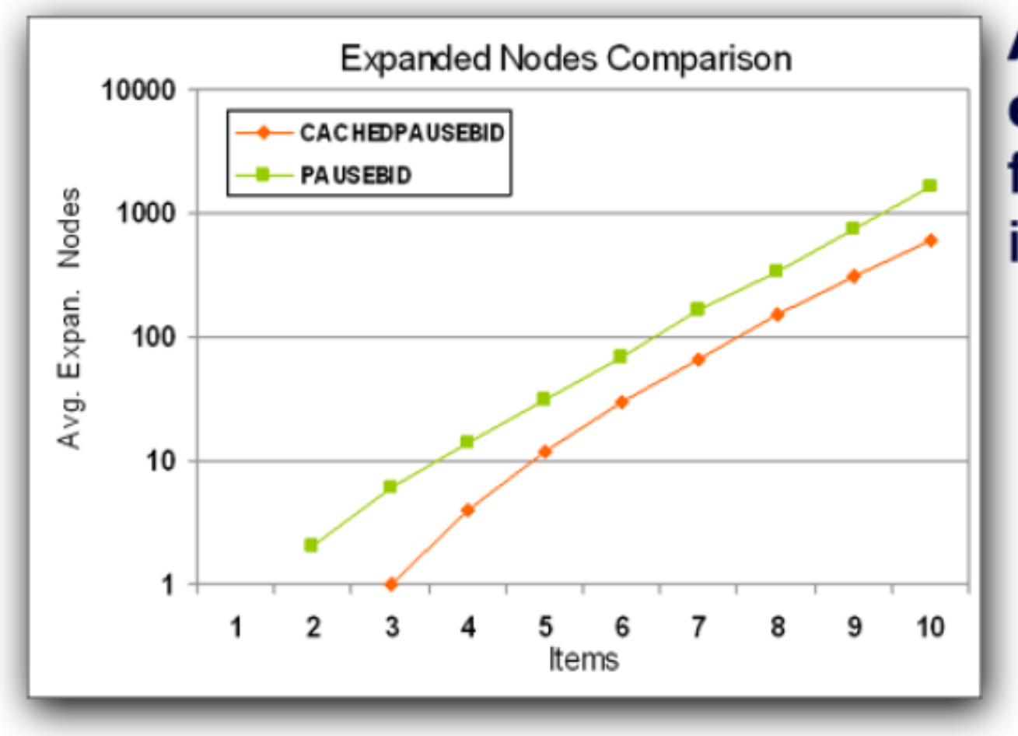
CACHEDPAUSEBID is similar to PAUSEBID but it caches some solutions and performs a branch and bound search only on the few portions affected by the changes on the bids between consecutive times



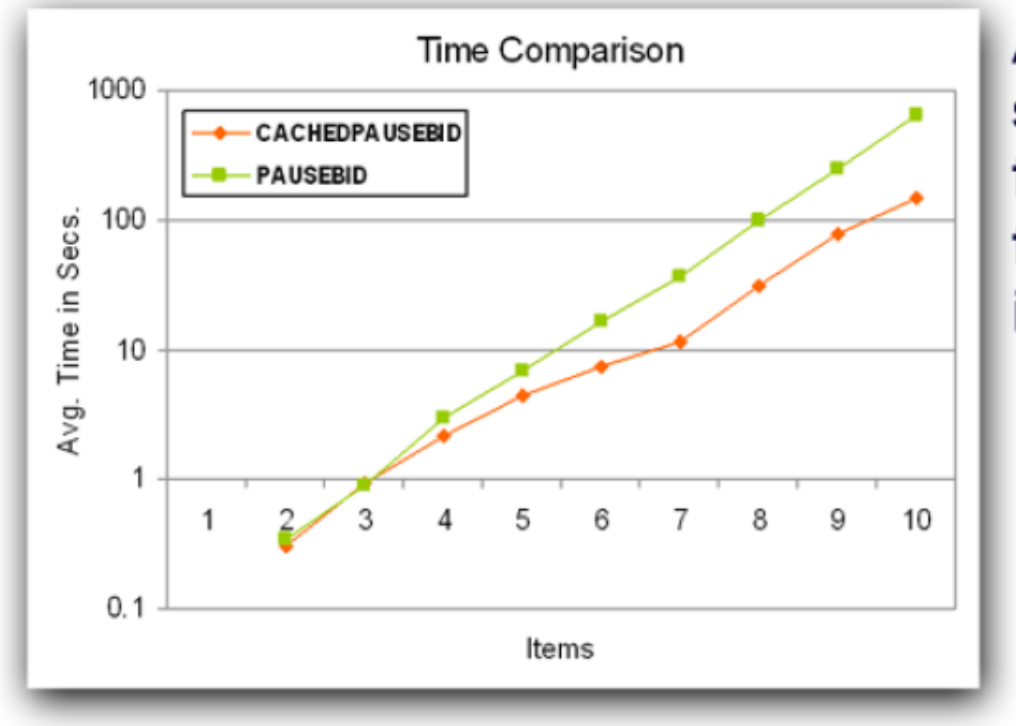
Average percentage of convergence is the percentage of times that our algorithms converge to the revenue-maximizing solution, as a function of the number of items in the auction



Average percentage of revenue from our algorithms relative to maximum revenue as function of the number of items in the auction



Average number of expanded nodes as a function of the number of items in the auction



Average time in seconds that it takes to finish an auction as a function of the number of items in the auction

Conclusions

Distributed solutions to the winner determination problem should be studied.

PAUSE does not tell us how the bidders should calculate their bids.

Our algorithms implement a myopic utility maximizing strategy that guarantees to find the bidset that maximizes the agent's utility given the set of outstanding best bids at any given time, without considering possible future bids.

Our algorithms find, most of the time, the same distribution of items as the revenue-maximizing solution

We envision to completely eliminate the auctioneer (having every agent perform the auctioneer's task).