Exercise 5: An Auctioning Agent for the Pickup and Delivery Problem

Group 21: Martin Vold, Kristoffer Landsnes

November 25, 2018

1 Bidding strategy

Our bidding strategy is based on three aspects of running an auction against another adversary agent.

1. Take into account future distributions of tasks.
2. Estimate the adversary's cost and evaluate his bids to enhance the accuracy of our bids.
3. Bid low initially to obtain tasks and enable our agent to bid for future tasks with lower a cost.

To compute our bid, the agent considers the mean cost of three(N_{sample}) plans which have been computed with random tasks from the distribution of tasks in the topology and the marginal cost, assuming that we obtain the auctioned task. It's given that there will be minimum 10 tasks for every auction, thus we can bid with a deficit for some of these tasks to try obtaining a maximum amount of these tasks. Afterwards a Stochastic Local Search algorithm is used to reduce the deficit caused by under-bidding in the start of the auction.
To have a higher probability of obtaining the first task in a auction, we need to bid low to beat the competing agent. To get a sense of how low we can bid we need an estimation for the average cost of obtaining the first N_{task} tasks. The distribution of task in the topology will be created by adding dummy tasks from each city to every other city to a list of these possible tasks, and then choose randomly among these tasks. This will be done N_{sample} times to get a better estimate of the cost of a plan with N_{task} tasks. The formula for the low bid will the be:

$$B_{low} = mean(Cost(N_{task})) = \frac{1}{N_{sample}} \sum_{N_{sample}} Cost(P_{N_{task}})$$

Where $P_{N_{task}}$ is a plan containing N_{task} tasks and the cost of this plan is optimized with our Stochastic Local Search algorithm. Since we will use a function that approximately goes from 1 to 0 in the interval [1, 10] to assign weight to this low bid we will choose N_{task} to be 15. The reason behind this is that as our bid will be less and less effected by this low bid for each auction, we want to keep the effect of this low bid for as long as possible for the 10 first tasks. N_{sample} will be set to 3 so that we have time to calculate plans that are good enough to be used as an estimate for a plan with N_{tasks}.

The weight function used will be:

$$\lambda = \frac{1}{1 + e^{i-8}}$$

The i is the number of tasks that have been auctioned. This function will have a value very close to 1 when i is 1 and close to 0 when i is 10, as i continues to increase the value of the function will get closer

and closer to 0.
The bid will will end up biding is given as:

$$B_{actual} = \lambda B_{low} + (1 - \lambda) B_{mixed}$$

Since we know that the minimum number of task for an auction is 10, we don't want to bid B_{low} for all 10 tasks. The function above archives this by ensuring that we don't use B_{low} 10 times, but rather a mean between the two where B_{mixed} gets more and more dominant as i approaches 10.

To get our mixed bid, B_{mixed}, we consider our marginal cost assuming that we get the auctioned task and the marginal cost of the adversary optimized with our Stochastic Local Search algorithm assuming he gets it. The function for B_{mixed} is:

$$B_{mixed} = 0.5 * (Cost_{marginal}(P_{own}, T) - Cost_{marginal}(P_{adversary}, T)) + \beta$$

In this function $Cost_{marginal}$ is the marginal cost for adding a task T to a plan P, as described in the exercise description:

$$Cost_{marginal}(P, T)) = max(Cost(P \cup T) - Cost(P, 0)$$

We have added the max to make sure that we don't end up with a negative marginal bid. The β is added to let us try to compensate for the bids of the adversary. The β value will start at 0 so that when we don't know anything about the bids from previous auctions rounds. This value will then change for each task that has been auctioned.

- If we won the task, β will increase with α, a learning rate with value 0.2, multiplied with the difference between the bid that is closest to our bid and our bid. This change is going to be positive, letting us bid more as the adversary is bidding higher than us.
- If we didn't win the task, β will decrease with α multiplied the difference between the winning bid and our bid.

With this bidding strategy we combine both the results of previous auction rounds, to enhance the future bids of the agent, and the distribution of possible tasks in the topology, to enhance the early bids the agent will make.

2 Results

2.1 Experiment 1: Comparisons with dummy agents

2.1.1 Setting

For this experiment we will compare our agent against a dummy agent that will bid a random bid in the range of 500 to 2000. The settings that where used was: Seed = 12345, timeout-setup = 30000, timeout-plan = 20000, timeout-bid = 20000. We tested with both the auction.xml and auction2.xml, which contained 25 and 15 tasks respectively.

2.1.2 Observations

For all the auctions we can see that we are beating the random agent with a superior profit, results in Table 1 and 2. This is mostly because of the random nature of the random bidding, it is hard to recover after bidding very low on a task that has a high marginal bid, even if the planing algorithm is good. This will not happen for our agent as it will only make controlled low bids in the start of the auction. It is also confirming to observe we get many of the first 10 task, which is one of the goals for our bidding strategy.

*Auction*1	*First Tasks*	*Total Tasks*	*Profit*	*Auction nr.*2	*First Tasks*	*Total Tasks*	*Profit*
Main	5	13	3344	*Main*	6	13	2986
Random	5	12	2144	*Random*	4	12	−664

Table 1: Results: auction.xml

*Auction*1	*First Tasks*	*Total Tasks*	*Profit*	*Auction nr.*2	*First Tasks*	*Total Tasks*	*Profit*
Main	8	11	5178	*Main*	8	11	4618
Random	2	4	835	*Random*	2	4	550

Table 2: Results: auction2.xml

2.2 Experiment 2

2.2.1 Setting

To compare our agent with a slightly harder competitor, it was decided to use an agent that always bids its marginal cost. The same settings as 2.1.1 were used.

2.2.2 Observations

The main observation when conducting this experiment is the impact of our bidding strategy's adaption to the opponent. As a marginal bidder will perform better than a random bidder, our total profit will be directly affected. The marginal bidder will give lower bids and our bids will change accordingly, especially for the first 10 auctioned tasks. Therefore, when there are very few tasks, we got a slight negative profit due to our under-bidding with current parameters, even though we are outperforming the marginal agent(as in table 4). This is a consequence of our strategy. In light of these results, we tried to change the N_{task} to 10 to get a more accurate estimate of the first auctioned tasks and avoid negative profit for such low tasks, which yielded only positive profits. As we didn't take as many tasks into account for possible future plans, we followingly get a lower profit, but positive. N_{task} will still be 15, as we expect to not only get auctions with few tasks. Furthermore, when there are more tasks (i.e 25 tasks, as in table 3), we always outperform the marginal agent. The profit is marginally less than in Experiment 1, due to the adaption of the opponent. Regardless, our agent is performing good and implicitly causes the marginal agent to end with a negative profit.

*Auction*1	*First Tasks*	*Total Tasks*	*Profit*	*Auction nr.*2	*First Tasks*	*Total Tasks*	*Profit*
Main	6	10	1465	*Main*	3	13	1530
Marginal	4	15	−840	*Marginal*	7	12	−950

Table 3: Results: auction.xml

*Auction*1	*First Tasks*	*Total Tasks*	*Profit*	*Auction nr.*2	*First Tasks*	*Total Tasks*	*Profit*
Main	7	8	−2	*Main*	7	10	988
Marginal	2	7	−1064	*Marginal*	5	4	164

Table 4: Results: auction2.xml