Agent Based Systems Coursework

u1925912

I. Second prize auctions

The auction provided is a second-prize auction. It consists of 200 rounds of auctions with from 2 up to 10 bots in a single room. After 200 rounds are played, the bot with the highest score is determined as the auction's winner. The aim is to program a bot that wins as many auctions as possible for different room configurations.
The second prize auction is designed so that the winner (highest bidder) of the round has to pay not their own bid, but the second-highest bid. This approach incentivizes truthful bidding. The approach taken by the designed bot is therefore to try and find the true value of the painting at a given time and bid that exact value. This approach would exploit over-bidding or under-bidding done by the other bots in a single round.

II. Determining the true value

We can establish the total budget as the sum of the budgets that all the players have at the start of the game.
We suppose that all the bidders will have spent all of their budgets by the time the 200 rounds of the auction come to an end.
We can change the list that contains the order of the artists, whose paintings are sold to the order of the paintings' values themselves. We can then sum up all the paintings' values.
The ratio of the total budget of all the bots and the total value of all the paintings is the price that is on average allocated to a single unit of painting value (Van Gogh has 12 units of painting value, Da Vinci has 7, Rembrandt has 3 and Picasso has 2).
If the budget is allocated uniformly to all the paintings, we can calculate the true value of the current painting by using the formula:

$$TV = cur\_paint\_val \cdot \frac{tot\_budget}{tot\_paint\_val} \tag{1}$$

The bid is therefore equal to the true value of the painting at all times. It does not change throughout the auction.

III. Identifying the poorly-performing bots

However, this bot requires some correction, as the approach would work best if the other bots also bid truthfully and more importantly, with the aim of spending their entire budget during the auction. The fact that this bot is going to compete with other bots that do not necessarily have the incentive to win the game or to spend all their budget makes a huge difference in the outcome of the auction. They might not win the game but they often skew the results in favor of another bot. It is especially challenging to distinguish between the bots that start bidding more money later and the bots that choose to bid more money later as part of their strategy. The more bots are in the room, the more difficult this task is.
Note that it is generally not a problem for my bot to win in a room with bots that want to spend too little money only. The difficulties occur when there is a mixture of different bots in the room, some of which may want to spend all their money at the start, some not at all, while the others will bid uniformly or use an even different strategy.

A. Adjusting for the remaining budget

Instead of identifying the true value for all rounds, the true value is determined in each round of auction. This is done by taking the ratio of the total remaining budget for all the bots combined to the total value of the remaining paintings. This makes sure the bids are updated in real-time, depending on whether the other bots have overbid or underbid. The following formula describes this section:

$$TV = cur\_paint\_val \cdot \frac{tot\_rem\_budget}{tot\_left\_paint\_val} \tag{2}$$

B. My budget

The bot provided works best if all the bots try to spend all their budget and do so rather uniformly throughout the whole auction. It fails when the total remaining budget is much bigger than expected because some bots decide not to spend their money. The bids of my bot become too big and it quickly runs out of money. To account for that, total_remaining_budget is replaced by the product of the number of bots in the room and my own budget. This would be equivalent to the total remaining budget if each of the bots had the same budget as my budget.

$$TV = cur\_paint\_val \cdot \frac{my\_budget \cdot num\_bots}{tot\_left\_paint\_val} \tag{3}$$

C. Budget and score ratio

The additional dictionary items for each bot were introduced to better determine the overall performance of each bot at every stage of the auction. Each bot has its budget ratio and score ratio. Those ratios combined will indicate the future involvement of the bot (the involvement of the particular bot in the rounds that are yet to be played)

1) Budget ratio: The budget ratio is the bot's remaining budget divided by the expected remaining budget at that stage. The expected remaining budget is the budget that a bot would still have if they were spending their money uniformly on each unit of painting value throughout the whole auction.
This is additionally adjusted by a threshold set at 3. This is done to account for the bots that bid too lowly and barely

spend any money. This ensures that the ratio does not reach very high figures as there are fewer and fewer rounds, while the bots still have not spent much money. Setting the threshold lower would later eliminate more passive bots, who will never buy any painting but would underestimate the bots which waited longer to place higher bids.

2) Score ratio: The score ratio takes the score accumulated by the bot so far and divides that by the money spent on accumulating the scores. The higher this number is, the more efficiently has the bot's money been spent. There are also cases, particularly with the low-spending bots, in which a few items have been bought but they have been bought so late that the total score at the end of the game is not likely to exceed a certain value. This value is obtained by supposing that all money left in the game will be uniformly distributed across all the units of painting values.
In the case, when there is a bot that has not spent anything at all, the score ratio will be determined by dividing the total expected remaining budget by the total remaining budget. If this value is greater than 1 it means the other bots have over-bid and therefore the bot was rightly better off, not spending any money. If it is lower than 1, it means other bots have under-bid, and therefore the bot has missed out on getting low-price paintings.

3) Identifying the early over-bidders: The lower a bot's budget ratio is, the less money they have for the remainder of the auction. This means that their involvement in future rounds is highly likely to be low because proportionally they no longer have a big budget at their disposal.

4) Identifying the passive bots: The lower the score ratio is the worse the expected performance of a particular bot. If the score ratio is significantly lower than others, this should indicate that there is a high chance this is a passive bot, which bids too low to win any of the rounds. The score usually significantly decreases only after many rounds have been played.

5) Combining budget and score ratios: To identify the passive bots quicker, the score ratio has been raised to the power of 3. It was then multiplied by the budget ratio. If this product was greater than 1, its 4th root was taken (the idea here is that we should not overestimate the importance of the very good bots, although they are efficient, the fact that they are good should not make my bot overbid). If the product was not greater than 1, the amount was left as it was. For all bots other than mine, the numbers were summed up and the result was stored in the scores_budget_ratio variable. The variable is equivalent to the number of other bots, expected to contribute to the remaining auction. Note that this number is not necessarily an integer.

D. Applying the adjustments

The final bid was by adjusting equation (3), specifically, by changing the number of bots to the score_budget ratio plus 1 (coming from my bot):

$$TV = cur\_paint\_val \cdot \frac{my\_budg \cdot (score\_budget\_ratio + 1)}{tot\_left\_paint\_val} \tag{4}$$

E. Not spending most money at the early stages

To ensure that we gain some information on the strategies of other bots before spending most of the money, we implement a threshold that forces us to bid less whenever we want to exceed it. This approach forces the bot to spend money in a more distributed way throughout the entire auction.
We calculate the bid that the bot would place if the current painting was Van Gogh and if the budget after spending this money to win the bid would be less than a certain threshold, the bid itself would be adjusted to be lower. The reason why we calculate the bid as if the current painting was Van Gogh is that we do not want to favour the paintings with less value and only adjust the bids with the most expensive value. More importance is placed on not exceeding the thresholds earlier in the auction because this is when we want to get the most information about the nature of the room (other bots). The following thresholds have been set with the bids adjusted accordingly:

1) for rounds 1 to 50 the threshold has been set to 0.9 of the expected budget left at that stage. If the bid exceeds this value, set the bid to 0.7 of the original value.
2) for rounds 51 to 100 the threshold has been set to 0.8 of the expected budget left at that stage. If the bid exceeds this value, set the bid to 0.8 of the original value.
3) for rounds 101 to 200 the threshold has been set to 0.7 of the expected budget left at that stage. If the bid exceeds this value, set the bid to 0.95 of the original value.

This approach also ensures that when there is severe under-bidding from the other bots in the earlier stages of the game, my bot would still win the round.

IV. Testing

My bot was tested in rooms including the following bots:

- random_bot: bids a random number within the bot's budget
- flat_bot_20: bids 20 on everything
- flat_bot_5: bids 50 on everything
- flat_bot_100: bids 100 on everything
- exact_value_2 bids 2 times the painting's exact value
- exact_value_3 bids 3 times the painting's exact value
- exact_value_4 bids 4 times the painting's exact value
- random_choice_bot: bids (at random) 2,3 or 4 times the painting's exact value
- true_value: bids the true value of the painting (from equation 1)
- true_value_late: bids the true value of the painting multiplied by the current round value divided by 100.

The testing was then done in the following rooms and the following win percentage of bot u1925912 was reported.

- The random room: [u1925912, random_choice_bot, random_bot, random_bot]: 20 out of 20 wins, 100%
- true value 1v1: [u1925912, true_value]: 20 out of 20 wins, 100%
- exact_value_2 1v1: [u1925912, exact_value_2]: 20 out of 20 wins, 100%
- exact_value_3 1v2: [u1925912, exact_value_3, exact_value_3]: 20 out of 20 wins, 100%

5 rooms with 5 up to 9 bots were created. To make the testing more difficult (exact_value_3 and exact_value_4 were placed in every room, and the other bots were chosen at random from the list of bots provided above,

- 5-bot-room: [u1925912, exact_value_3, exact_value_4, exact_value_4, flat_bot_20]: 20 out of 20 wins, 100%
- 6-bot-room: [u1925912, exact_value_3, exact_value_4, exact_value_4, true_value_late, true_value_late]: 20 out of 20 wins, 100%
- 7-bot-room: [u1925912, exact_value_3, exact_value_4, exact_value_2, exact_value_2, true_value_late, true_value]: 2 out 20 wins, 10%
- 8-bot-room: [u1925912, exact_value_3, exact_value_4, exact_value_4, random_choice_bot, flat_bot_100, flat_bot_100, true_value_late]: 8 out of 20 wins, 40%
- 9-bot-room: [u1925912, exact_value_3, exact_value_4, exact_value_2, exact_value_4 flat_bot_50, flat_bot_50, flat_bot_50, true_value_late]: 10 out of 20 wins, 50%

A. When the bot still struggles

My bot seems to struggle whenever there is too much-remaining budget at the end of the auction and at the same time, there are bots in the auction that exploit that and bet consistently at the paintings from the start, regardless of how much budget is remaining between the other bots. Particularly exact_value_3 (with fewer bots in the room) and exact_value_4 (with more bots in the room) seem to exploit the weaker bots better.

This issue becomes hard to solve, especially when there are many other bots in the room. It becomes increasingly difficult to distinguish between the bots that start bidding really late (like true_value_late) and the bots that are never going to win any round of the auction because the bids are too low (like exact_value_2). While measures were taken to prevent this from happening so often and detect the weaker bots early, a certain amount of caution still needs to be taken when identifying the weaker bots so as not to misclassify the late-bidding bots. For rooms with more rational bidders that have the incentive to spend all their money during the auction, this problem would not occur.