CS 136 Assignment 6: Sponsored Search Auctions

David C. Parkes
School of Engineering and Applied Sciences, Harvard University
Team: Rangel Milushev and Tomislav Žabčić-Matić

Friday Sep 28, 2018

1 Introduction

For this assignment, we were asked to program bidding agents to participate in a generalized second-price (GSP) auction. The balanced bidding agent can be found in **angelslavbb.py**.

We also implemented our own version of a budget bidding agent to compete in a tournament against the rest of the class. It can be found in **angelslavbudget.py**.

2 Balanced Bidding Agent

2.1 Design

We designed our balanced bidding (BB) agent according to the problem specifications (to be found below):

Consider period t. Let b_{-i}^{t-1} denote the bids from the agents other than i in the period $t-1$. Suppose there are m positions. Balanced-bidding proceeds as follows:

- Given bids b_{-i}^{t-1}, agent i targets the position j^* that maximizes

$$\max_{j\in\{1,\dots,m\}}[pos_j \cdot (v_i - t_j)], \tag{1}$$

  where pos_j is the position effect (which you should estimate by using the number of clicks in the previous round) and t_j is the price the agent would pay for position j given bids b_{-i}^{t-1}. For example, in GSP auction, t_j equals the j-th highest bid in b_{-i}^{t-1}.

- (*Not expecting to win*) If price $t_{j^*} \geq v_i$ in this target position, then bid $b_i^t = v_i$ in period t.

- Otherwise:

  (a) (*Not going for the top*) If target position $j^* > 1$, then set bid b_i^t to satisfy

$$pos_{j^*}(v_i - t_{j^*}) = pos_{j^*-1}(v_i - b_i^t). \tag{2}$$

  (b) (*Going for the top*) If $j^* = 1$, then bid $b_i^t = v_i$.

We are making use of the pre-defined **slot_info** to help us with the calculation for expected utilities, taking the *min_bid* for position j of the previous round to serve as the price the agent i would pay for being in position j (t_j). The number of clicks for position j are the position effect for position j (or pos_j).

2.2 Experimental Analysis

In our analysis, we will be looking at the differences between agent populations in the GSP auction and how utilities and expenditures change, as we run the auction with different agents.

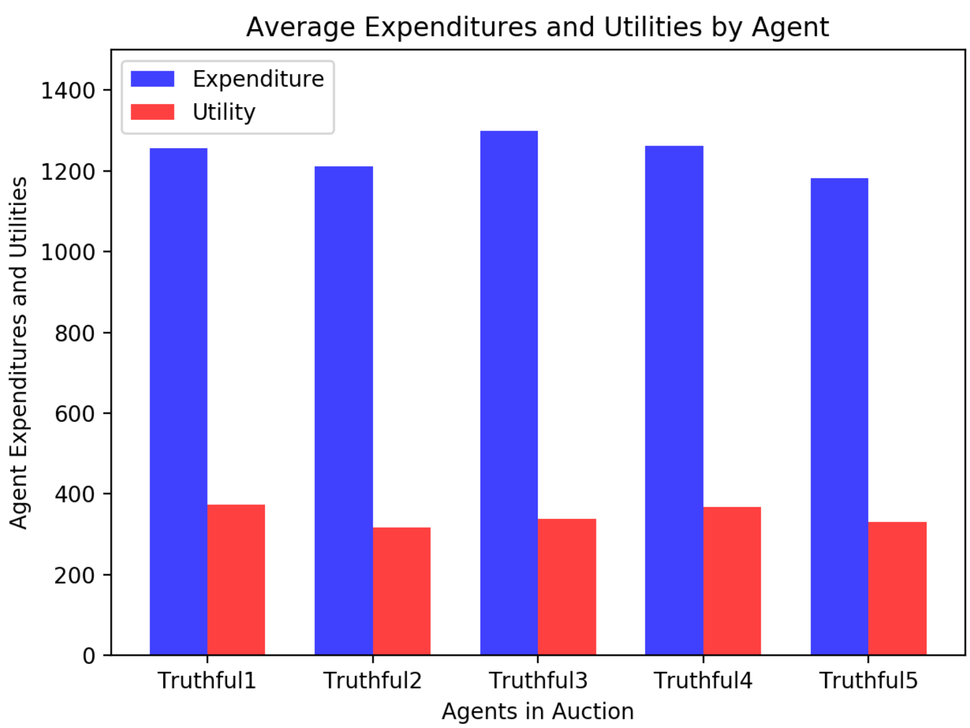
2.2.1 Population of Truthful Agents

When we run the auction on a population of Truthful Agents, we can see that their expenditures in the auction is much higher than their average utilities. We ran the auction with the following parameters:

```
python auction.py \
--num-rounds 48 \
--perms 1 --iters 200 \
--seed 2 \
--mech=GSP \
Truthful,5
```

Here are some statistics:

- The mean of the average utilities was: \$345.584
- The mean of the average expenditure was: \$1253.20
- The average daily revenue was: \$6207.79
- The standard deviation for the daily revenue was: \$1457.17



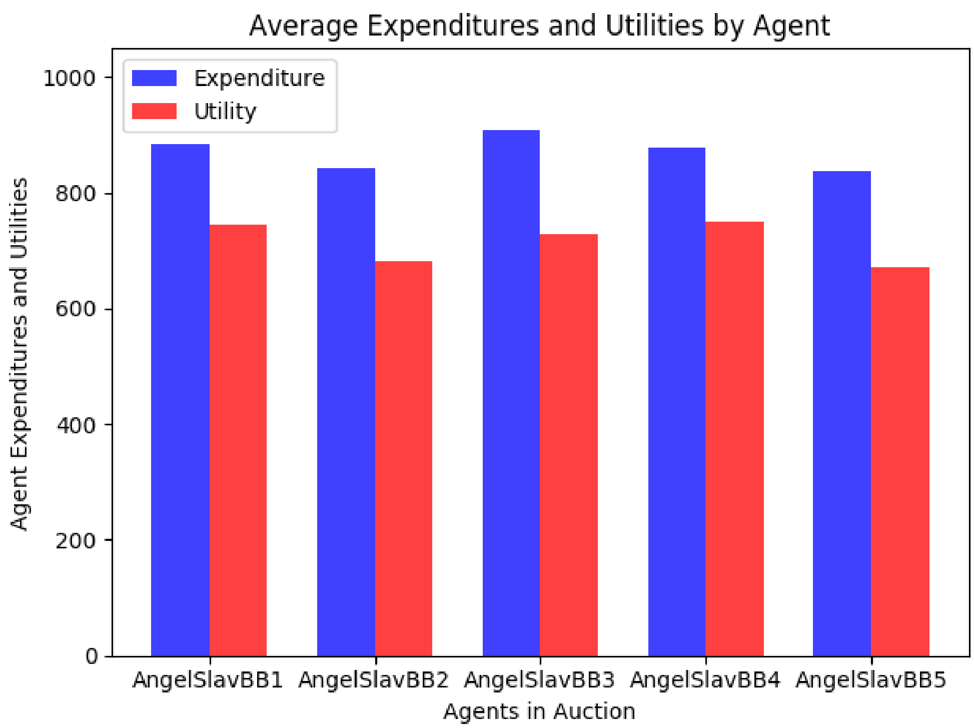
2.2.2 Population of Balanced Bidding Agents

When we run the auction on a population of BB Agents, we can see that the gap that we witnessed in the auction with truthful agents shrinks significantly. The expenditures in the auction is not much higher than their average utilities. We ran the auction with the following parameters:

```
python auction.py \
--num-rounds 48 \
--perms 1 --iters 200 \
--seed 2 \
--mech=GSP \
AngelSlavBB,5
```

Here are some statistics:

- The mean of the average utilities was: $714.90
- The mean of the average expenditure was: $870.726
- The average daily revenue was: $4353.64
- The standard deviation for the daily revenue was: $1373.14



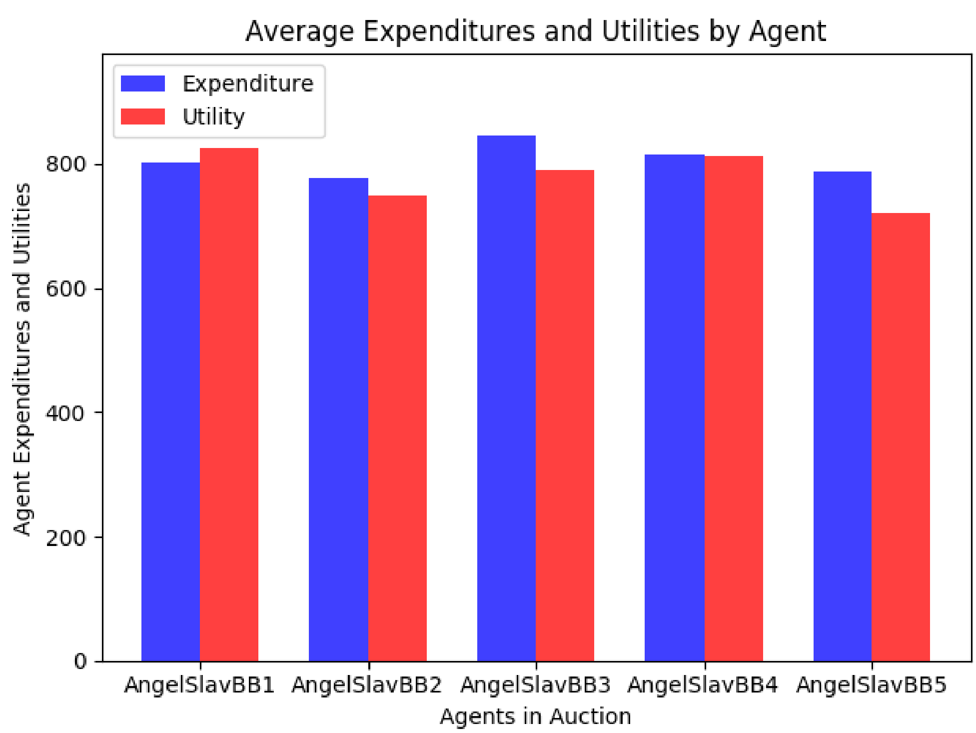
2.2.3 Population of Modified Balanced Bidding Agents

A slight modification of the balanced bidding strategy gave a surprising result, which we decided to include in our analysis. Instead of calculating the expected utility for the last slot with the minimum bid price, we calculated it with the reserve price. This gave a somewhat surprising increase in expected utility and decrease in expenditure of each agent in a population of modified agents. The modified version didn't violate any of the tests either, which we initially speculated. Please find the results below. We have commented out the code in case you're interested and wanted to run it. The parameters we used are the same as the ones from before:

```
python auction.py \
--num-rounds 48 \
--perms 1 --iters 200 \
--seed 2 \
--mech=GSP \
AngelSlavBB,5
```

Here are some statistics:

- The mean of the average utilities was: $779.58
- The mean of the average expenditure was: $805.87
- The average daily revenue was: $4029.39
- The standard deviation for the daily revenue was: $1237.73



2.2.4 Comparison of the above results

As we can see from the above charts, the average utility is much higher in a population of balanced bidding agents than in a population of truthful agents. In the unmodified version, the BB utility is still lower than the BB expenditure, yet the gap between expenditure and utility is much smaller compared to what we observe in a population of truthful agents. One possible explanation for why we see this improvement in both utility and expenditure for the balanced bidding agent is that the balanced bidding strategy is *envy-free* so it targets the slot which will maximize its own expected utility, and thus can outperform the utility of a truthful agent, which does not strategize in order to maximize utility. In terms of why we see the difference in expenditure, it is reasonable to believe that this follows from the maximization of expected utility, since utility is calculated as the agent's value minus its expenditure. Thus, it is reasonable to assume that in maximizing its own expected utility, the balanced bidding agent incurs less expenditure, which is part of why its utility is greater than that of the truthful agent.

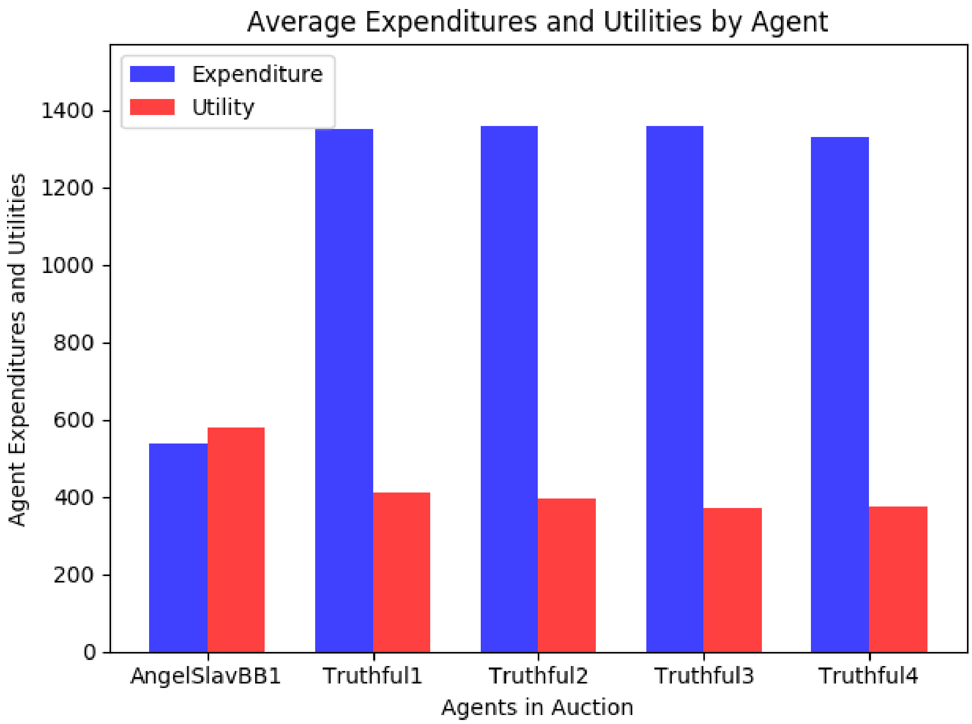
2.2.5 Balanced Bidding Agent in a Population of Truthful Agents

In this case, we run the auction in a population of truthful agents and one balanced bidding agent. As we can see on the graph below, the BB agent outperforms all the truthfuls when it comes to utility. The BB agent actually has a higher utility than expenditure which is another interesting observation. All the truthful agents have a much higher expenditure than the BB agent. We ran the auction with the following parameters:

```
python auction.py \
--num-rounds 48 \
--perms 10 \
--iters 200 \
--seed 2 \
--mech=GSP \
AngelSlavBB,1\
Truthful,4
```

Here are some statistics:

- The mean of the average daily utilities was: $425.76
- The average daily utility for AngelSlavBB was: $580.05
- The average daily utility for the Truthful agents was: $387.20
- The mean of the average daily expenditure was: $1185.75
- The average daily expenditure for AngelSlavBB was: $538.07
- The average daily expenditure for the Truthful agents was: $1347.25
- The average daily revenue was: $5928.76
- The standard deviation for the daily revenue was: $1310.31



2.2.6 Truthful Agent in a Population of Balanced Bidding Agents

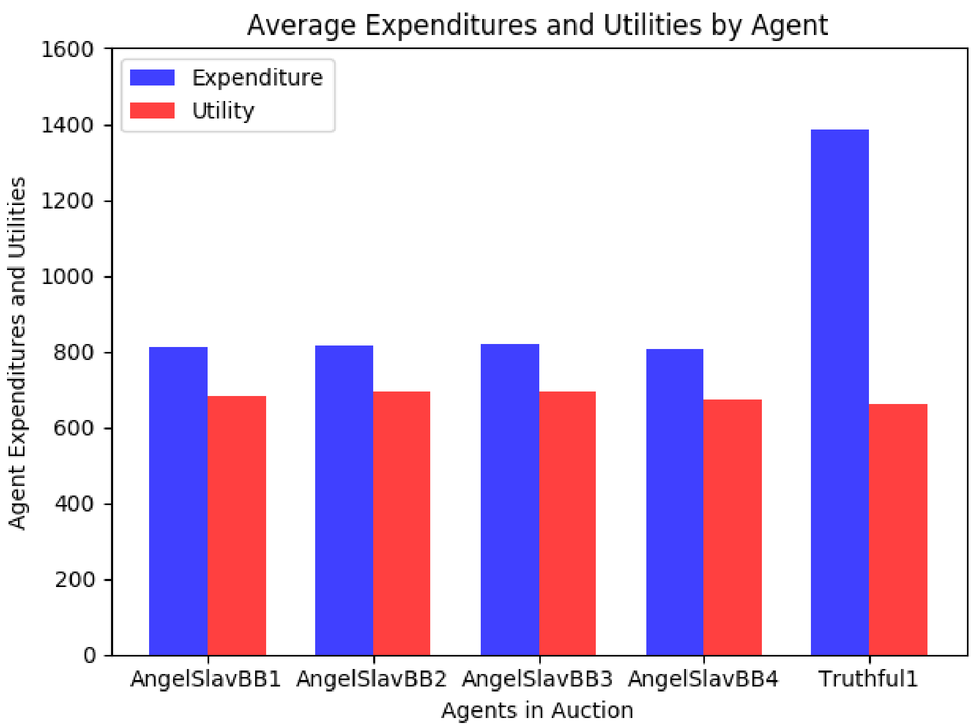
In this case, we run the auction in a population of BB agents and one truthful agent. An interesting observation here is that the BBs are forcing the truthful agent to spend more. Another interesting observation is that the utility for the Truthful agent does not seem to deviate too far off from the utility of the BB agents (refer to the statistics below). This time around, all the BB agents have a lower utility than their respective expenditure. Again, the truthful agent's expenditure is significantly higher than the expenditures of the BB agents. We ran the auction with the following parameters:

```
python auction.py \
--num-rounds 48 \
--perms 10 \
--iters 200 \
--seed 2 \
--mech=GSP \
AngelSlavBB,4\
Truthful,1
```

Here are some statistics:

- The mean of the average daily utilities was: \$681.06
- The average daily utility for AngelSlavBB agents was: \$685.79

- The lowest value for the average daily utility for the AngelSlavBB was: $674.71
- The average daily utility for the Truthful agent was: $662.15
- The mean of the average daily expenditure was: $927.24
- The average daily expenditure for AngelSlavBB agents was: $812.59
- The average daily expenditure for the Truthful agent was: $1385.80
- The average daily revenue was: $4636.18
- The standard deviation for the daily revenue was: $1185.86



2.2.7 Discussion of 2.2.5 and 2.2.6

As we observed in 2.2.4 and the preceding two sections, it would always be better to follow a balanced biding strategy. In our tests, the BB agent has always resulted in a higher utility and a lower expenditure than the Truthful agent. The truthful bidder always bids its true value whereas the BB agent targets the slots that would provide it with the highest expected utility. The truthful bidding strategy seems highly inefficient in the GSP, since the positive difference between the expenditure and utility is always significantly larger than that of the BB agent. Additionally, the BB agent does better than the Truthful agents across populations.

As we can see in 2.2.5, in a population of truthful bidders, the BB agent actually has a higher utility than expenditure. Since the GSP is not strategy proof, the BB does significantly better because it maximizes expected utility, whereas the truthful agent always bids their truthful value.

3 Auction Design and Reserve Prices

3.1 VCG Implementation

Please find the implementation of the VCG payment rule in the **vcg.py** file. We followed the suggestion in the problem specification and implemented the payment rule recursively.

3.2 GSP Reserve Price Experimental Analysis in BB Agent Population

In this section we answer the following questions:

1. What is the auctioneer's revenue under GSP with no reserve price when all the agents use the balanced-bidding strategy?
2. What happens as the reserve price increases?
3. What is the revenue-optimal reserve price?

Please find the answers in the following subsections below. An interesting finding was that as we hold the reserve price constant, increasing the number of agents results in diminishing marginal returns for the revenue.

3.2.1 No Reserve Price

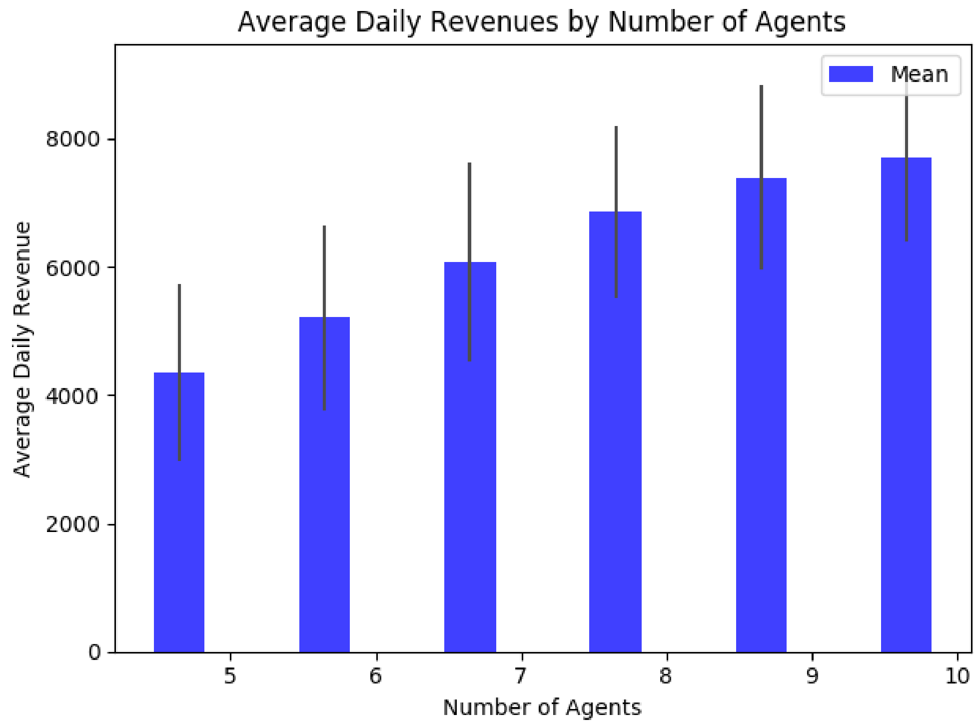
We are running the GSP auction in a population of BBAgents with the following parameters:

```
python auction.py \
--num-rounds 48 \
--perms 1 \
--reserve 0 \
--iters 200 \
--seed 2 \
--mech=GSP \
AngelSlavBB, i
```

where i varies from 5 to 10. Some statistics are:

- For 5 BB agents the average daily revenue is: $4353.64
- For 6 BB agents the average daily revenue is: $5205.35
- For 7 BB agents the average daily revenue is: $6071.97
- For 8 BB agents the average daily revenue is: $6864.33
- For 9 BB agents the average daily revenue is: $7389.07
- For 10 BB agents the average daily revenue is: $7702.66

The graph representation of our findings is:



3.2.2 Increasing Reserve Price

As the reserve price increases, the auctioneer's revenue gradually increases, up to a certain point, after which the revenue begins decreasing, eventually down to a revenue of 0. Thus, we conclude that revenue, as a function of reserve price, is approximately concave down, with some local fluctuation, and is non-monotonic.

3.2.3 Optimal Reserve Price

The optimal reserve price occurs at a reserve price of $90. The revenue to the auctioneer under this reserve prices is $5895.00 (in a population of 5 agents). We obtained this result by running an algorithm similar to binary search to obtain a range in which the reserve price is optimal, and then tested on each reserve price within that range (65 through 100).

3.3 VCG Reserve Price Experimental Analysis in Truthful Agent Population

In this section we compare the results with the above section and answer the following questions:

1. What is the auctioneer's revenue under VCG with no reserve price when all agents are truthful?

2. What happens as the reserve price increases?

3.3.1 No Reserve Price

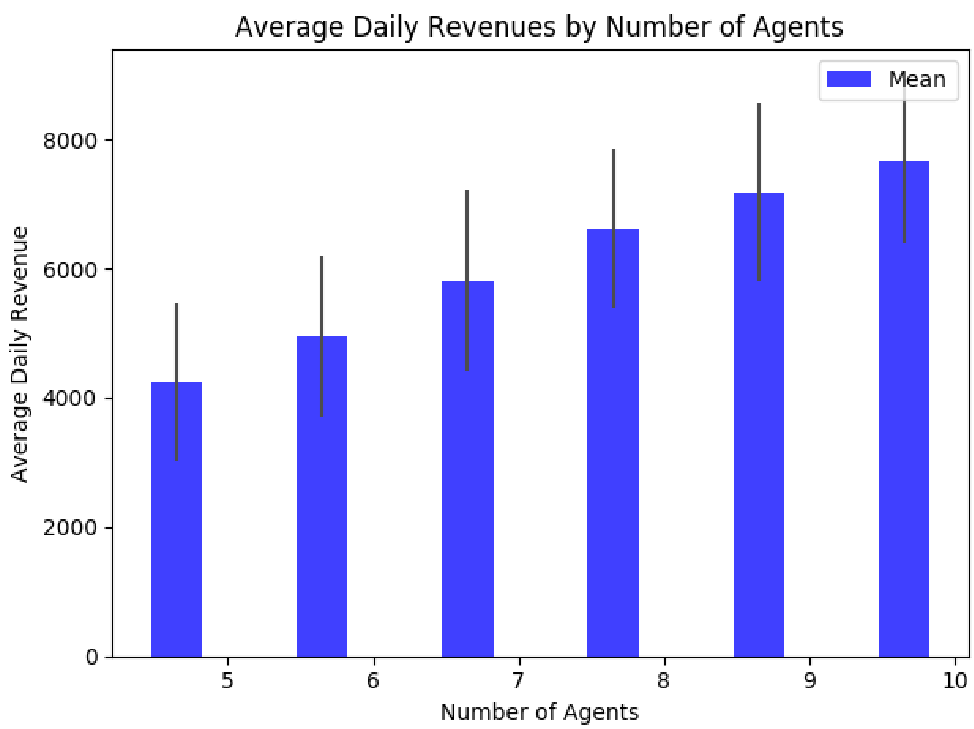
We are running the VCG auction in a population of Truthful agents with the following parameters:

```
python auction.py \
--num-rounds 48 \
--perms 1 \
--reserve 0 \
--iters 200 \
--seed 2 \
--mech=VCG \
Truthful, i
```

where i varies from 5 to 10. Some statistics are:

- For 5 Truthful agents the average daily revenue is: $4231.64
- For 6 Truthful agents the average daily revenue is: $4944.47
- For 7 Truthful agents the average daily revenue is: $5811.53
- For 8 Truthful agents the average daily revenue is: $6616.13
- For 9 Truthful agents the average daily revenue is: $7184.10
- For 10 Truthful agents the average daily revenue is: $7665.00

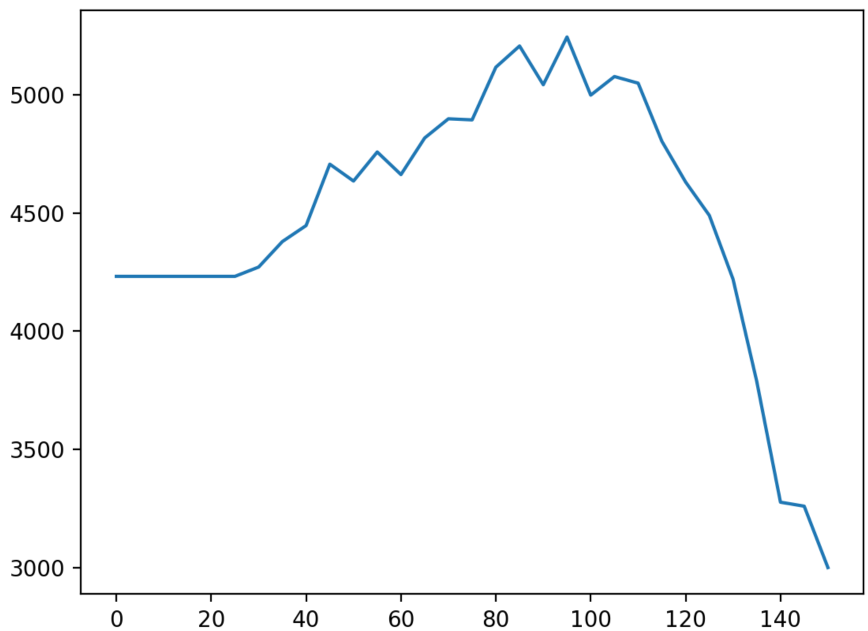
The graph representation of our findings is:



The revenue results are very similar to the revenue results that we got for the GSP auction with the BB agents, although there is some difference. The GSP seems to produce a little bit of a higher revenue with our current implementation of the BB agents.

3.3.2 Increasing the Reserve Price

As the reserve price in the VCG auction with a population of Truthful agents increases, the auctioneer's revenue gradually increases, up to a certain point, after which the revenue begins gradually decreasing, eventually down to a revenue of 0. Thus, we conclude that revenue, as a function of reserve price, is roughly concave down (though it does have some non-concavity around local optima which some searches for optimal reserve price may get stuck in), and is non-monotonic. See the graph below for a graphical representation. The y axis is the revenue and the x axis is the reserve price.



3.4 Switching from GSP to VCG

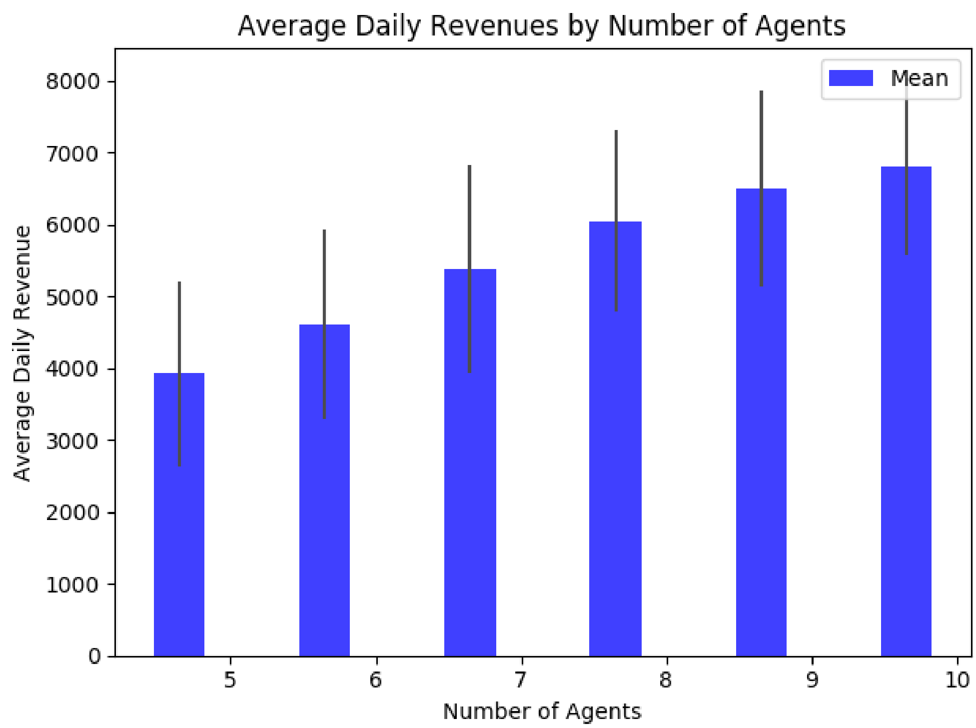
We are fixing the reserve price to zero. Let's explore what might happen if a search engine switched over from the GSP to VCG design. For this, we are running the BB agents in GSP, and at period 24, we switch to VCG, by using the `--mech=switch` parameter. We ran this with the following parameters:

```
python auction.py \
--num-rounds 48 \
--perms 1 \
--iters 200 \
--seed 2 \
--mech=switch \
AngelSlavBB,i
```

where i varies from 5 to 10. Some statistics are:

- For 5 BB agents the average daily revenue is: \$3924.32
- For 6 BB agents the average daily revenue is: \$4612.18
- For 7 BB agents the average daily revenue is: \$5377.12
- For 8 BB agents the average daily revenue is: \$6050.89
- For 9 BB agents the average daily revenue is: \$6508.88
- For 10 BB agents the average daily revenue is: \$6811.85

See the graph below for a visual representation. One thing that is to be observed is an average decrease of daily revenue of about \$300 to \$400. Although we do not have a plot to show the following phenomenon occuring, since getting that data was overly complicated, we were able to log revenue values by timestep (intra-iteration), and we noticed that when running the switch mechanism, at the moment when the system switched from GSP to VCG, there was a sharp drop in the per-timestep revenue, and then a slow increase back to a value that was almost that of the GSP system by the end of the 48-period iteration.



4 Our Budget Agent: AngelSlavBudget

In our budget agent implementation we modified the initial bid price to be 0. This worked better than any other initial bid price, and we theorize that this is the case because it prevents us from bidding anything before gauging what the other agents are doing. We also calculate our expected utility for a certain slot by making use of both the min and max bid prices. The update is as follows:

```
slot_payment = min_bid + (max_bid - min_bid)*alpha
```

where alpha is distributed uniformly at random between 0.5 and 0.99, and minbid and maxbid are the minimum and maximum bids for that slot in the previous round. We then calculate our utility for a slot using this information. This results in a more accurate prediction and a better placed bid when we target a slot (in a population of BB agents we are more likely to get it). Additionally, this results in us placing higher bids and increases the price for our competitors in the following rounds when we bid. We assume most of our peers would be using some variation of the balanced bidding strategy in the tournament, so we decided that this implementation is going to do rather well. You can see on the plot below how it did against our standard implementation of the balanced bidder. Our utility is always higher than our expenditure.