CS 136 Assignment 6.1
Sponsored Search Auctions (Programming)

David C. Parkes
School of Engineering and Applied Sciences, Harvard University
Out: Thursday October 21, 2021
Due: **5pm** sharp: **Friday, October 29, 2021**
Submissions to `Gradescope`

1 The Balanced Bidding Agent

1. [**2 Points**] What is your team name?

   We are team Dakz.

2. [**20 Points**] **Experimental Analysis**

   To answer the following questions, run the simulation with 5 agents. By default the budget is $5000, which is not binding. Leave it this way!

   (a) [**10 Points**] What is the average utility of a population of truthful agents? What is the average utility of a population of balanced bidding agents? *Compare the two cases and explain your findings.*

   Make use of the `--perms`, `--seed`, and `--iters` commands, e.g. `--perms 1 --seed 2 --iters 200` would be a good starting point.

   i. **All Truthful:** First, we run the auction on a set of 5 Truthful agents as follows:
   `python3 auction.py --perms 1 --seed 2 --iters 200 Truthful,5`

   Below are some statistics from this run:
   - **Mean of Average Utilities:** 337.41
   - **Mean of Average Amount Spent:** 1127.84
   - **Average Daily Revenue:** 6139.20
   - **Stdev Daily Revenue:** 1540.01

   We see the individual values for the agents here as well:

Bidder	Average Utility	Average Spent
Truthful1	344.98	1323.18
Truthful2	355.38	1223.25
Truthful3	328.17	1141.17
Truthful4	329.51	1302.07
Truthful5	329.01	1149.53

Overall, we see that there is a large gap between the amount a bidder spends and their average utility.

ii. **All Balanced:** Next, we run the auction on a set of 5 balanced bidding agents as follows:

```
python3 auction.py --perms 1 --seed 2 --iters 200 Dakzbb,5
```

Below are some statistics from this run:

- **Mean of Average Utilities:** 675.30
- **Mean of Average Amount Spent:** 888.92
- **Average Daily Revenue:** 4441.47
- **Stdev Daily Revenue:** 1435.40

We see the individual values for the agents here as well:

User	Average Utility	Average Spent
Dakzbb1	714.06	952.23
Dakzbb2	704.59	874.27
Dakzbb3	645.96	821.55
Dakzbb4	674.48	955.21
Dakzbb5	637.43	838.20

As shown in this table, the difference between average utility and average amount spent is much lower in the auction with all truthful agents. The average amount spent is also lower than in the auction with all truthful agents.

iii. **Analysis:** We see higher utilities and lower amounts spent in the auction with all balanced bidders. According to the balanced bidding strategy, this makes sense, as a balanced bidding strategy should produce an envy-free outcome in the GSP auction. Thus, balanced bidding attempts to optimize for utility, while truthful bidding does not. The lower amount spent is likely also caused by the fact that balanced bidding optimizes for utility.

(b) **[10 Points]** In addition, what is the average utility of one balanced-bidding agent against 4 truthful agents, and one truthful agent against 4 balanced-bidding agents? For the new experiment, make use of the `--seed`, and `--iters` commands, but you will now want to run multiple permutations. Note that you can add multiple agents types using:

```
> python auction.py  --perms 10 --iters 200 Truthful,4 abxybb,1
```

i. **4 Truthful 1 BB:** First, we run the auction on a set of 4 Truthful agents and 1 balanced bidding agent as follows:

```
python3 auction.py --perms 10 --seed 2 --iters 200 Truthful,4 Dakzbb,1
```

Below are some statistics from this run:

- **Mean of Average Utilities:** 423.97
- **Mean of Average Amount Spent:** 1143.79
- **Average Daily Revenue:** 5690.08
- **Stdev Daily Revenue:** 1241.36
- **Mean of Average Utilities for Truthful:** 390.22
- **Mean of Average Amount Spent for Truthful:** 1290.0

We see the individual values for the agents here as well:

Bidder	Average Utility	Average Spent
Truthful1	386.50	1283.45
Truthful2	385.69	1291.15
Truthful3	383.40	1283.88
Truthful4	395.57	1301.53
Dakzbb1	558.98	530.08

Overall, we see that the balanced bidder performs much better than the truthful bidders in terms of both utility and amount spent.

ii. **4 BB 1 Truthful:** Next, we run the auction on a set of 5 balanced bidding agents as follows:

```
python3 auction.py --perms 10 --seed 2 --iters 200 Dakzbb,4 Truthful,1
```

Below are some statistics from this run:

- **Mean of Average Utilities:** 667.27
- **Mean of Average Amount Spent:** 887.62
- **Average Daily Revenue:** 4438.10
- **Stdev Daily Revenue:** 1126.44
- **Mean of Average Utilities for BB:** 666.34
- **Mean of Average Amount Spent for BB:** 769.43

We see the individual values for the agents here as well:

User	Average Utility	Average Spent
Dakzbb1	660.20	765.64
Dakzbb2	671.05	774.64
Dakzbb3	669.14	781.30
Dakzbb4	664.97	756.14
Truthful1	670.98	1360.68

As shown above, all the agents have about the same utility, but the average amount spent for the truthful agent is significantly higher than the other balanced bid agents.

iii. **Analysis:** When a balanced bidder is in an auction with all truthful bidders, the balanced bidder has a higher average utility and lower average amount spent than all of the truthful bidders – in fact, in the **4 Truthful 1 BB** example, the balanced bidder actually gains a utility that is higher than the amount that it spends.

When a truthful bidder is in an auction with all balanced bidders, all of the bidders now have similar utilities (and utilities that are much higher than those in the **4 Truthful 1 BB** auction). However, the one truthful bidder still has a much higher amount spent than the balanced bidders, and now, none of the balanced bidders have greater utility than amount spent as well.

What does this suggest about the incentives to follow the truthful vs. the balanced bidding strategy?

The results of these two auctions, especially those of the **4 Truthful 1 BB** auction, suggest that there is an incentive to deviate from truthful bidding to the balanced bidding strategy, as the balanced bidder will outperform all other bidders in a group of truthful bidders in both utility and amount spent.

2 Experiments with Revenue: GSP vs VCG auctions

In this section we compare the revenue properties of the GSP and VCG auctions with different reserve prices. A reserve price, $r > 0$, sets a minimum price for any position. Used carefully, reserve prices can increase revenue. The reserve price in the GSP works as follows: only bids (weakly) above r can be allocated. The agent in the lowest allocated position pays the maximum of the reserve price and the maximum bid of unallocated bidders. The VCG auction works in a similar way and is explained below.[1]

4. **[55 Points] Auction Design and Reserve Prices**

   Run all simulations with 5 agents. Leave the budget to its default of $5000.

   (a) **[20 Points]** Complete the code that runs the VCG auction in `vcg.py`. The allocation rule is already implemented. You need to implement the payment rule. Because of the reserve price, it is most convenient to use the recursive form of the VCG payment rule (see the proof of Theorem 10.23 in the reading).

   In particular, suppose there are 3 bidders with bids weakly greater than the reserve price, and $b_1 \geq b_2 \geq b_3$, and say that b_4 is the fourth highest bid. Bidders 1–3 are allocated the top 3 positions.

[1] One way to think about it is that the reserve is made operational by including a "dummy bidder" whose bid is the amount of the reserve, and ignoring all bids with value less than r.

Let $t_{\text{vcg},i}(b)$ denote the expected payment by bidder i in a single auction given bid profile b. For bidder 3, this is $t_{\text{vcg},3}(b) = pos_3 \max(r, b_4)$. For bidders 1 and 2, in positions 1 and 2 respectively, this is $t_{\text{vcg},i}(b) = (pos_i - pos_{i+1})b_{i+1} + t_{\text{vcg},i+1}(b)$.

(b) **[10 Points]** What is the auctioneer's revenue under GSP with no reserve price when all the agents use the balanced-bidding strategy? What happens as the reserve price increases? What is the revenue-optimal reserve price?

You can set the reserve price in the simulation with the command line argument `--reserve INT` (where INT is the reserve price in cents). Also use `--perms 1` and `--iters 200`.

Solution

We will run the GSP auction on a set of 5 balanced-bidding agents, while altering the reserve (in cents) to get the average daily revenue, or the auctioneer's revenue as follows:

```
python auction.py --loglevel=info --mech=gsp --reserve=0 --perms=1
--iters=200 Dakzbb,5
```

In the following table, we see the reserve prices and the corresponding average daily revenue for each price.

Reserve Price	Auctioneer's Revenue
\$0.00	**\$4572.45**
\$0.50	\$4835.64
\$0.75	\$4991.21
\$0.80	\$5005.90
\$0.84	**\$5180.71**
\$0.85	\$5078.79
\$1.00	\$4942.73
\$1.50	\$2889.52
\$1.75	\$203.00
\$2.00	\$0.00

As we see in the table, as the reserve price increases beginning at \$0.00, we see the revenue increase until it reaches the revenue-optimal reserve price around \$0.75. After this, the revenue falls and as it increases past around \$2.00 the revenue remains at \$0.00 as the reserve price continues to increase. Thus we see that the revenue-optimal reserve price is around \$0.75.

(c) **[10 Points]** What is the auctioneer's revenue under VCG with no reserve price when all agents are truthful? What happens as the reserve price increases? Explain your findings and compare with the results of part (b).

Again use the `--perms`, `--seed`, and `--iters` commands, e.g. `--perms 1 --seed 2 --iters 200`.

We will now run the VCG auction on a set of 5 truthful agents, while altering the reserve (in cents) to get the average daily revenue, or the auctioneer's revenue as follows:

```
python auction.py --loglevel=info --mech=vcg --reserve=0 --perms=1 --seed=2
--iters=200 Truthful,5
```

In the following table, we see the reserve prices and the corresponding average daily revenue for each price.

Reserve Price	Auctioneer's Revenue
$0.00	**$4345.24**
$0.50	$4465.68
$0.75	$4849.22
$0.80	$5044.00
$0.87	**$5170.12**
$0.90	$5109.71
$1.00	$4857.88
$1.50	$2808.90
$1.75	$210.00
$2.00	$0.00

Similar to the GSP auction we see that as the reserve price increases up to a point, the auctioneer's revenue continues to increase. Once we reach the revenue-optimal reserve at $0.87, the revenue falls until at around $2.00 the revenue stays fixed at $0.00 as reserve continues to increase. We see that the GSP auction with the balanced-bidding agent has a slightly higher revenue than the VCG auction with truthful agents, and both have an optimal reserve price between 800 and 900 cents.

(d) **[10 Points]** Fix the reserve price to zero. Explore what might happen if a search engine switched over from the GSP to VCG design. For this, run the balanced-bidding agents in GSP, and at period 24, switch to VCG, by using the `--mech=switch` parameter. What happens to the revenue?

Again use the `--perms`, `--seed`, and `--iters` commands, e.g. `--perms 1 --seed 2 --iters 200`.

When using the switch parameter on a population of 5 balanced agents, we get a revenue of 4041.25, which is lower than what we get when we just run the GSP auction on the same population (4441.05). Thus, we can conclude that the switch decreases total revenue.

(e) **[5 Points]** In one paragraph, state what you learned from these exercises about agent design, auction design, and revenue? (There is no specific right answer).

We've seen empirically through the exercises many of the things that we understand to be true theoretically: for instance, we saw that balanced bidding improved utility and revenue in GSP, as balanced bidding can lead to an envy-free outcome. Additionally, we evidence that the revenue in GSP with balanced bidders is higher than that of VCG in the truthful outcome, which is what we expect, and we also showed that switching from GSP to VCG is also not beneficial. Thus, from a revenue standpoint, it may be best for an auctioneer to use GSP over VCG.

3 The Competition

5. **[30 Points] Budget constraints**

   The balanced-bidding agent does not consider the budget constraint when deciding how to bid. In the final part of the assignment, your task is to design a *budget-aware agent.*

   This agent will compete in the simulated GSP auction against the agents submitted by other groups. You might like to test your design against a variety of other strategies.

   For example, you could write an agent that measures competition, or tries to drive up the payments of other agents so that they pay more and exhaust their budgets! Example ideas include:

   - Trying not to bid too much when prices are high.
   - Try to bid when other agents are not bidding very much and the price is lower.

   For the purpose of the competition, the daily budget constraint will be set to $600 (use the `--budget` flag). We will run a GSP auction with a small reserve price. Auctions will contain around 5 agents.

   The competition will be structured as a tournament, with agents placed into groups of 5 or 6 and the top few agents making it into a semi-final and final. The precise structure of the tournament will depend on the number of agents submitted.

   (a) **[25 Points]** In `TEAMNAMEbudget.py`, write your competition agent.
   Describe in a few sentences how it works, why it is designed this way, and how you expect it to perform in the class competition.

   In order to create our budget agent, we made a small modification to our balanced bidding agent by attempting to follow a strategy where the agent bids just above reserve price for periods close to 24, where the number of clicks is lower in order to not waste as much spending power on periods when clicks are low (around 24), and increases the bid to the truthful value again in later rounds (> 36) when there should be higher amounts of clicks. Otherwise, we bid according to the balanced bidding strategy. Thus, we try to take advantage of our knowledge of how the number of clicks works. In a small test against balanced bidding agents, our budget agent seems to gain more utility.

   You are not expected to spend many hours on writing an optimal agent, unless you want to. Consider a few possible strategies, try them out, pick the best one.

   (b) **[5 Points]** Win the competition (!) Likely parameters for the competition are
   `python auction.py --num-rounds 48 --mech=gsp --reserve=10 --iters 200`
   (followed by the list of submitted agents)