Web Economics Assignment Part A

Jun Wang and Emine Yilmaz
{j.wang, e.yilmaz}@cs.ucl.ac.uk

Department of Computer Science
University College London

This assignment is worth 5% of your overall mark
Submission deadline: 23:59 14/03/2016
Submission method: Submit your answer document online to Moodle
Format: Use LaTeX or MS Word and convert it into a single PDF
Filename: YourStudentID-YourSurname-YourFirstName-Assignment-PartA

The aim of this assignment is to help you understand some basic concepts and calculations in real-time bidding (RTB) based display advertising. Should you have any question about the assignment, please contact the TAs: Rishabh Mehrotra (r.mehrotra@cs.ucl.ac.uk), Bin Zou (bin.zou.14@ucl.ac.uk).

1 Real-time bidding (2%)

Real-time bidding (RTB) based display advertising uses the second price ad auction as mechanism and employs the cost-per-impression price scheme to charge advertisers.

1. For an advertiser i with true value r_i for each user click on her ad, what is her truthful bidding on a bid request with estimated click-through rate θ_i on her ad? If all the advertisers use the truth-telling bidding strategy, will they always bid the same price for each auction? Please explain the reasons.

2. Let us consider the case where the click-through rate is a constant value across all advertisers, and the bid request volume is large enough to exhaust the advertiser's budget within her campaign's lifetime, what is the right choice of changing her truth-telling bid price in order to maximise the ad click number? Higher, lower, or keep the bid price level. Please explain the reasons.

2 Sponsored Search Auction (3%)

In this assignment you will be working with a sponsored search auction and assigning bids to slots using the VCG mechanism. Your job is to implement the algorithm by editing the file *auction.py*.

Set Up
Download this Python project webEconAssignmentA.zip and import it appropriately to run in any of the Python IDEs. As a sanity check, select and run driver.py. The output should be few lines ending with:

	clicks	price	profit	bidder
slot	10.00	0.00	0.00	null
slot	8.00	0.00	0.00	null
slot	7.00	0.00	0.00	null
slot	5.00	0.00	0.00	null
sums	30.00	0.00	0.00	0.00

The 3 numeric columns above are the click-through rate, the price charged for that slot, and the profit expected for the bidder in this slot. The final column is the name of the bidder, and since you haven't done the work yet that column is all null right now. The last row is the expected number of clicks served, the revenue to the auctioneer, and the total profit to the bidders.

Getting Started with SSA
The file *driver.py* generates different examples of a Sponsored Search Auction. (Initially, most are commented out.) After constructing one of these objects, it will print out the slot table and the bid table, and then call **executeVCG** (which is what you will be writing).

Open the file *auction.py*. Inside the class, you will see these member fields: **query** (string) and **bids** (list of bids). The query variable says what phrase is being auctioned; you won't need it. The bids list holds a name and value (max price) for each player. The constructor for the auction object guarantees that bids are always arranged in decreasing order of value.

By perusing the *slot.py* file you will find that it retains a click-through rate (*clickThruRate*), the assigned bidder, the price the bidder offered, and the profit the bidder is expected to garner. You are going to fill in everything in this data structure except *clickThruRate*. The slots are always given in decreasing order of click-through rate.

Conducting the Auction
Your job is just to finish writing *executeVCG*. Allocate slots and prices according to the **Vickrey-Clarke-Groves** mechanism.

When you get your algorithm working, go back to driver and uncomment those lines in main so that you get more test cases. The first example has as many slots as bidders, but later ones may not be that way. Make sure your code can handle other cases as well. Peruse the results to see if you believe them. You might find some surprising effects by considering related auctions. For example, Rick's RollerRama altered something; what happened to their total profit? The profit of the auctioneer changed somewhere by doing something under their own control. What was it, and what effect did it have? Don't hand in the answers to these questions; just know the answers.

Finishing up
Copy the content of your *auction.py* file and paste it onto the document containing your solution to the first question on RTB above. Use Moodle to submit your solutions file which contains the answer to both questions above.