COS 445 - Strategy Design 4

Due online Monday, April 15th at 11:59 pm

Instructions:

- **You may not take late days on the Strategy Designs**. If it helps, think of the Strategy Designs as being due on Friday, except we have given everyone three free late days.
- You should aim to work in a team of two, but you are allowed to work alone or in a team of three. Your team should submit a *single* writeup, using the team feature on codePost. You should also submit a *single* code solution, using the team feature on TigerFile.
- Your goal in this assignment, and all strategy designs, is to **maximize your absolute payoff**. Your goal is *not* to outperform other submissions. Any justification you provide should explain why your strategy maximizes your absolute payoff (and only this kind of justification will contribute to your writeup score). Your code score is computed solely based on how you perform in comparison to "if your solution were replaced by a course staff submission" (that is, you are not being compared to a strategy that you play against, so it does you no good to harm the strategies you play against).[1]
- You are allowed to engage with other teams over Ed or in person (but this is neither encouraged nor discouraged). If this is part of your strategy, you should discuss what you did and why you did it in your writeup. You are allowed to coordinate with other teams, or trick other teams. You are *not* allowed to promise other teams favors (e.g. monetary rewards) or threaten punishment outside the scope of this assignment. For example, you are allowed to promise "if your code does X, our code will do Y." You are not allowed to promise "if your code does X, I will buy you a cookie." If this is part of your strategy, your justification should explain why it will help you *on this assignment*.
- Please reference the course collaboration policy here.
- Please reference the following document for further detail on how these assignments are evaluated: GradesForStrategy.pdf.
- This assignment is open-ended, **please ask questions on Ed to clarify expectations as needed**.

Reminder!

Please read the instructions at GradesForStrategy.pdf to better understand how the strategy design assignments are graded (which in turn should clarify how to answer the prompts).

[1] This claim is *slightly* inaccurate in order to save runtime while computing code scores. You are free to read the full details **here**, and to ask for clarification on Ed, but I'm comfortable advising that the best way to optimize your code score is to just optimize your own payoff and not to overthink subtleties in precisely how the code scores are computed.

Start Your Startup (35 points)

Your best friend (the same one from SD3) encountered wild success with your suggestions, and found their way onto the executive board for a new startup aimed at automating the process of gerrymandering congressional districts. As such, they now need to figure out how to advertise on search engines to reach the broadest possible audience. Your friend remembered that you were taking 445 (and also that you had a previous consulting gig in the industry), and asked for your help bidding in ad auctions.

In this challenge you'll play the following game: Each day, there will be a new keyword up for auction. On that day (and no earlier), your marketing research team will tell you how much you should value a click for this keyword. There are multiple ad slots for sale, auctioned off through a **Generalized Second Price (GSP) Auction**. Specifically:

Setup:

- Every team initially has a **budget of 500** USD.
- There are $T = 10000$ **rounds**.
- D is a distribution that takes values in $[1, 200]$. The CDF of D satisfies $F(x) = 1 - 1/x$ for all $x \in [1, 200]$. Note this means that a draw from D is equal to 200 with probability $1/200$.[2]

Each Round t:

1. v_{it} denotes the value of team i per round-t click. Each v_{it} is drawn independently from D.
2. Every team i learns v_{it} *only* at the start of round t. Team i *does not* learn v_{jt} for any $j \neq i$.
3. There are 10 ad positions up for auction.
   - Slot One has a click-through rate of 5%. If team i wins this slot, they add $v_{it} \cdot .05$ to their payoff *at the end of the entire game*.
   - Slot Two has a click-through rate of 3.5%. If team i wins this slot, they add $v_{it} \cdot .035$ to their payoff *at the end of the entire game*.
   - Slot Three has a click-through rate of 3%. If team i wins this slot, they add $v_{it} \cdot .03$ to their payoff *at the end of the entire game*.
   - Slots Four through Ten have identical click-through rates of 1.5%. If team i wins one of these slots, they add $v_{it} \cdot .015$ to their payoff *at the end of the entire game*.
4. The ad positions are auctioned via a **generalized second-price (GSP) auction** (the bullets below remind you of the definition of a generalized second-price auction):
   - Each team i submits a sealed bid $b_{it} \geq 0$.
   - The j^{th} highest bidder is awarded the j^{th} ad slot (ties are broken randomly).
   - The winner of the top slot pays the second-highest bid **per click**, the winner of the second-highest slot pays the third-highest bid **per click**, and in general the winner of the j^{th}-highest slot wins pays the $(j+1)^{st}$-highest bid **per click**. Teams which do not win a slot pay nothing.[3]

[2] This is called a "truncated equal revenue curve," similar to the equal revenue curve we saw in Lecture.

[3] For instance, if the bids were $\langle 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 \rangle$, then bidder 11 would win the top slot and pay $10 \cdot .05 = .5$. Bidder 10 would win the second slot and pay $9 \cdot .035 = .315$, etc.

- Any payments made by team i during the round-t auction are deducted *immediately* from team i's budget.

5. Team i may not submit a bid that could possibly cause them to overdraw their remaining budget. Specifically, if B_{it} denotes team i's remaining budget at the start of round t, any bid exceeding $B_{it} \cdot 20$ will be capped at $B_{it} \cdot 20$. Again, note that you do not receive payoff from clicks you win in round t until the end of the game, so your sum of payments across all rounds will not exceed your original budget of 500.

Payoffs:

- At the end of the game, your team receives payoff equal to your remaining budget, plus any value earned from clicks throughout the entire game. Specifically, if c_{it} denotes the click-through rate of the slot won by team i during round t ($c_{it} = 0$ if team i won no slot), and p_{it} denotes the payment made during round t, then team i's payoff at the end of the game is:

$$500 + \sum_{t=1}^{T} v_{it} \cdot c_{it} - p_{it}.$$

Your strategy should implement two functions. The first function will take as input a value v_{it} (your value per-click for that day) and return your bid for that day.

The second function will help you track data throughout the game. It will take as input:

- An array `bids`, where `bids`$[j-1]$ denotes the winning **bid** for slot j on day $t-1$.
- An integer `myBid`, which indicates the slot your strategy won (if it exists) on day $t-1$. If you did not win a slot on day $t-1$, you will receiv `myBid` $= -1$.
- If `myBid` ≥ 0, you will also be told $p_{i(t-1)}$, the price your team paid in round $t-1$.
- To be extra clear: the team that won slot j on day $t-1$ bid `bids`$[j-1]$ per click. You will not know the ID of the team that won each slot. If `myBid` ≥ 0, then your team won slot `myBid` $+ 1$, and paid p_{it}.
- You should use this function to help track the day t, and your remaining budget, and whatever information about previous payments you like.

Code it up according to the specifications below, and write a brief justification. You will implement the Bidder interface provided in `Bidder.java`, which requires the following methods, as documented in `Bidder.java`.

- `public double getBid(double dailyValue)` returns your bid for the current day and is called once per day before the auction.
- `public void addResults(List<Double> bids, int myBid, double myPayment)` lets you know if you won and how much the winners paid and is called once per day after the auction.

We've also provided a sample strategy, `Bidder_truthful.java`, and the usual construction of testing code: `Auctioneer.java` and a makefile to run it against varied strategies.

Your file must follow the naming convention `Bidder_netID1.java`, where `netID1` is the Net ID of the primary submitter. All advice from previous homeworks applies again to this homework, especially that you should consider the performance of your strategy against the strategies and distribution thereof which you expect your peers to play.

Your writeup should provide an overview of the main ideas in your code (remember that we also have your code — so you don't need to provide pseudocode or a step-by-step description of your algorithm), and justify why you think it will perform well, in addition to concrete answers to parts **a** and **b**.

Penalties may be given for code which does not compile, throws exceptions, or violates assertions. In particular, you will be considered to have submitted a bid of 0. Remember to test your code with settings besides the default settings of our testing environment. Extra credit may be awarded for reporting substantive bugs in our testing code.

Also submit a single PDF file, containing answers to the following three prompts. Recall that your grade for part c is the maximum of your grade on the writeup and your grade for your strategy's performance (any penalties are subtracted after taking the maximum).

Part a (10 points)

Prove that overbidding is a dominated strategy in the Generalized Second Price auction. That is, prove that any strategy that submits a bid of $b > v$ to the Generalized Second Price auction is dominated.

Note: For full credit, you should clearly state a precise outline (e.g. if bidding b is dominated, what strategy dominates it? Briefly explain why). But it is OK if you do not rigorously execute calculations.

Very briefly note why it might potentially make sense to overbid in the strategy design.[4]

Part b (10 points)

We saw in class that GSP is not truthful, so it may sometimes increase your payoff to submit a bid not equal to your value, even if there is only $T = 1$ round. We also saw in class that VCG is truthful, meaning that it cannot increase your payoff to submit a bid not equal to your value when there is only $T = 1$ round.

If this strategy design were otherwise identical, except the auction implemented during every round were VCG instead of GSP, and all bidders $j \neq i$ submit bids of $b_{jt} = v_{jt}$ every round, is it always a best response for bidder i to submit a bid of $b_{it} = v_{it}$ every round as well? Give a brief justification of why or why not.

Note: For full credit, you should clearly state a precise outline (e.g. if you are claiming it is a best response, briefly state why truth-telling in every round is a best response. If you are claiming it is not, what is one case where there is a better response, and what is that better response?). But it is OK if you do not rigorously execute calculations, freely assume that $T \gg n$, tie-break however you like instead of randomly, etc.

[4] To be clear, the course staff is suggesting neither that you overbid, nor that you don't overbid. Just that you think about it.

Part c (15 points)

Provide a brief justification for your strategy. Focus on convincing the grader that it is a good strategy, by explaining the main ideas and why you chose this strategy. You should aim to keep this under one page. This will not be strictly enforced, but the grader may choose not to read beyond one page. You should not think of this merely as a documentation explaining only *what* your code does. Instead, try to imagine that it's purpose is to convince your friend *why* they should adopt your strategy.

Note: For all Strategy Designs, it is a good idea to explicitly reason about how you believe your classmates will behave. For this Strategy Design in particular (because your utility is heavily dependent on what your classmates do), it may be a better-than-usual idea to explicitly reason about how you believe your classmates will behave. Note that you are allowed/encouraged to discuss your strategy publicly/privately with other teams (but you should still **not** share code).