# CS2440 Final Project - AdX

Alan Chen, Patrick Peng - Agent “banger”

# 1 Introduction

TAC AdX is a simulated ad exchange where users of various demographics are auctioned via second price auctions to companies that are attempting to generate profit and prove to be a reputable advertiser. To earn revenue, the companies are assigned campaigns that pay according to the amount of impressions received in a particular market segment. Our task was to develop an agent to play as one of 10 companies participating in the simulated ad exchange, hoping to generate the most profit over 10 days of advertisements.

# 2 Theoretical Analysis

During the development of theoretical analyses, we performed research online about the game to get an idea of what the key points in this game were. Some primary references are included in the references section.

The original TAC AdX game is too complex to be analytically solved in a short time frame. We make various simplifications to the game to construct a theoretical analysis of optimal strategies.

We first consider strategies for bidding on the incoming users, starting with the simplest model and progressively adding elements of the game until we feel that the game is sufficiently complex that it is not analytically tractable.

At the base level, for each user a second price auction is run. Of course, for second price auctions, bidding truthfully is the dominant strategy. If we consider $k$ repeated second price auctions, but still maintain that bids can only be submitted once (before all the auctions are ran), the game is clearly exactly the same as a one off auction, so bidding truthfully is still the dominant strategy.

## 2.1 Dominant Budgeting

Now, suppose we add in the element of *budgeting* to the repeated $k$ second price auction model. Formally, before the auctions are run, each bidder receives a budget $B_i$ as part of their type, meaning the bidder will only receive utility if their spending is below $B_i$. Any goods won by spending more than $B_i$ are valued at 0 i.e. the utility is negative of the amount paid for the good.

When the auctions are run, in addition to submitting a bid $\hat{b}_i$, each bidder $i$ also submits a budget $\hat{B}_i$ where the bidder can only spend up to $\hat{B}_i$. Once the bidder has spent at least $\hat{B}_i$ after winning some $k_i$ goods, they are removed from the auction. The auction continues to run on the remaining $k - k_i$ goods and bidders $[n] \setminus \{i\}$.

**Lemma 2.1.** *Submitting $\hat{B}_i = B_i$ is a dominant strategy up to $\hat{b}_i$.*

*Proof.* Consider alternative strategies. Notice that if the agent is not "in the money" ($\hat{b}_i$ is not high enough to win any goods), the submitted budget is irrelevant - the agent gets 0 utility regardless.

**Strategy 1:** submit $\hat{B}_i < B_i$. In this case, the bidder is unnecessarily limiting their utility. There are two cases with what happens.

1. Submitting $\hat{B}_i = B_i$ would not change the amount of items won. Then, the utility is the same.
2. Submitting $\hat{B}_i = B_i$ would cause $i$ to win more goods. Assuming each good brings them positive utility (namely if $\hat{b}_j \leq v_i$, where $j$ is the second highest bidder), which is always true

when $\rho$ linear, clearly being truthful is better. This assumption is usually applicable if the agent is playing rationally.

**Strategy 2:** submit $\hat{B}_i > B_i$. Notice, the bidder now unnecessarily exposes themselves to negative utility. There are two cases that can happen here.

1. Submitting $\hat{B}_i > B_i$ causes no change compared to submitting $\hat{B}_i = B_i$ in the number of goods that the agent wins. Then, the utility is the same.
2. However, if submitting $\hat{B}_i > B_i$ causes the agent to win $\Delta k_i$ extra goods compared to submitting $\hat{B}_i = B_i$, the agent has lowered their utility by $\Delta k_i \hat{b}_j$ unnecessarily, where $\hat{b}_j \geq 0$ is the bid of the second highest bidder.

So, it is always dominant up to $\hat{b}_i$ to submit $\hat{B}_i = B_i$. The $\hat{b}_i$ part is because there could be a case where you make an extra bid before you hit your budget which puts you over the budget, making the good have 0 utility. □

However, there does not exist dominant behavior for $\hat{b}_i$. Note that in some occasions, depending on the actions of the other bidders, bidding untruthfully may yield more utility. In other words, there is no dominant strategy as the best response depends on the actions of the other agents. Consider the following example.

**Example**: Suppose we have three bidders in the repeated second price auction, and there are $k = 10$ identical goods that will be sequentially auctioned off. Bidder 1 (us) has a valuation of 1 and a budget of 10, bidder 2 has a valuation of 0.99 and a small budget of 1, and bidder 3 has a poor valuation of 0.51 and likewise a budget of 1. Assume the other two bid truthfully and consider our response to this scenario. In this case, a better response is to shave our valuation to 0.99, and claim as many goods as the randomness gets us, for an expected utility of around 3.96, whereas bidding truthfully would have a net utility of 0.08 as bidder 2 would always end up as the second price. This strategy only works because bidders are forced to drop out early by lack of budget.

However, if bidder 2 decides to underbid, say at 0.98, our best response is no longer to bid 0.99; instead we'd also want to bid 0.98.

Thus, it is clear that there is no dominant way to choose $\hat{b}_i$. We remedy this in our agent by attempting to use other factors to decide what $\hat{b}_i$ should be, such as quality score and additional assumptions.

So far, we have assumed that profit linearly scales with the number of goods won. In TAC AdX, that is not the case. We now analyze the optimal strategy for the TAC AdX profit scheme, which involves additional moving parts.

## 2.2 Optimal Target Reach Fraction

TAC AdX relaxes the assumption that the profit scales linearly and introduces the idea of a *reach*. TAC AdX scales profit according to a special effective reach function $\rho : [0, 1] \to \mathbb{R}$. Specifically, we consider when

$$\rho(\eta) = \frac{2}{a} \left( \arctan(a\eta - b) - \arctan(-b) \right)$$

for selected constants $a, b \in \mathbb{R}$. In the spec, these are set to be $a = 4.08577$ and $b = 3.08577$.

Consider a campaign with budget $B$ and reach $R$. Let $\eta$ be the fraction of $R$ that our agent wins/satisfies. Then, the profit is defined to be

$$P(\eta, B, R) = B\rho(\eta) - K, \tag{1}$$

where $K$ is the total cost spent to get $\eta R$ impressions and $\rho$ is the effective reach function. Clearly, $\eta = 1$ is not necessarily profit maximizing, because $\rho$ could grow slower than linear. In general, $P$ is not easy to maximize because $K$ is unknown. However, suppose we can estimate a fixed approximate *cost-per-impression* $k \in \mathbb{R}^+$. Then, the profit can be expressed as

$$P(\eta, B, R, k) = B\rho(\eta) - k\eta R. \tag{2}$$

Since $k$ is fixed, this is optimizable with respect to $\eta$. We want to solve

$$0 = \frac{\partial P}{\partial \eta} = B\frac{\partial \rho}{\partial \eta} - kR.$$

Letting $\rho$ be defined as in the specification,

$$kR = B\frac{2}{a}\frac{a}{1 + (a\eta - b)^2}.$$
$$\frac{2B}{kR} = 1 + (a\eta - b)^2.$$

Taking the appropriate square root, this can be solved to give an optimal target reach percentage of

$$\eta = \frac{b + \sqrt{\frac{2B}{kR} - 1}}{a}.$$

We need to be careful when implementing this $\eta$ in practice, since for some campaigns $\frac{2B}{kR} - 1$ may not be $\geq 0$. However, in those cases, it essentially means the campaign has extremely high estimated cost per impression and/or a very low $B/R$ ratio - our agent attempts to avoid these campaigns as we will see later on. So, in practice, it is sufficient to calculate the optimal $\eta$ as

$$\eta = \frac{b + \sqrt{\max\{0, \frac{2B}{kR} - 1\}}}{a} = \frac{3.08577 + \sqrt{\max\{0, \frac{2B}{kR} - 1\}}}{4.08577}.$$

## 2.3 Impression Valuation

Due to the fact that each agent can get contracts with budget equal to the reach, we assume 1 to be a 'true' valuation for each impression. Here, we ignore the additional contracts available for bidding because that makes this a lot more complex.

## 2.4 Campaign Auctions

For campaign auctions, we could not come up with any optimal strategy because there were too many factors at play. However, through some testing in labs and separately, we realized that quality score was crucial, so the completion difficulty of the contracts should be emphasized. Contracts with small reaches ($\delta = 0.3, 0.5$) were easier to complete and thus would help quality score, so we thought the optimal strategy would involve maintaining quality score via evaluating "good" contracts.

### 2.5 Optimal vs. Exploitative Play

In our agent, we experiment with various *exploitative* practices that are not necessarily rigorously dominant strategies. We use an example to motivate this decision.

**Example.** Consider a market segment where all participating bidders have one campaign where the budget is at most the reach i.e. $B_j \leq R_j$ for all $j \in [n]$. Then, the per-person valuation is $\leq 1$. Further, assume that bidder $i$'s campaign has a per-person valuation of exactly 1 i.e. $B_i = R_i$.

In this case, a perfectly valid but notably not game theory optimal strategy is to exploitatively *overbid*. By submitting a bid above 1 (their valuation), the bidder can guarantee first dibs on the market segment instead of leaving it up to chance among the likely multiple other bidders who submitted bids of 1. Since the auctions are second price, as long as there are no other overbidders, the agent will still break even or make money.

For an agent that prioritizes quality score, for example, this is a valid strategy that deviates from conventional play, as it exposes the agent to negative utility. However, the agent can play a more profitable strategy on the following days, using its strong quality score as leverage to win the campaigns it wants to win and dominate the game. Thus, depending on the goals of an agent, unconventional exploitative strategies may work better than "in the box" strategies.

In particular, an agent can play in an unexploitable manner by always bidding true valuations, never exposing itself to negative valuation. However, this will probably lead to a middle of the pack finish on the leaderboard. Employing exploitative strategies is absolutely essential to *winning* the competition.

## 3 Failed Ideas

Early on into the project, we struggled with figuring out a strategy without glaring issues. This section discusses some strategies that we at one point implemented but did not make it into our final agent (in their preliminary form) along with the insight we gained.

### 3.1 Tanking Quality Score

One strategy we investigated early on was if there was any exploits with the quality score computation. We hoped to develop a strategy where early on, the agent would develop a strong quality score, and then after the game went on for a few days, buy out all of the campaigns on all following days without a regard for whether or not they were feasible to complete so that the other agents would not have a way to make significant profit (other than the randomly assigned auction).

However, with $\alpha = 0.5$ as the moving average scaling parameter set by the game, the "historical" reputation of the agent was easily overridden by its performance on new days. Our agents atrocious performance in these experiments where we intentionally destroyed quality score to maximize profit motivated our first key principle described in subsection 4.1 that maintaining quality score should be first priority, even above profit.

### 3.2 Overbidding

As mentioned in the discussion of optimal vs. exploitative play, overbidding is a viable strategy targeted at exploiting the assumption that no one else will submit bids above 1 in order to establish

a strong quality score.

We implemented this strategy early on, varying how much to overbid and only overbidding for the early days, but the results were disappointing. We often ended up with $-1000$ or more average utility, likely because tuning the daily limit was difficult. Through this, we realized that for both performance and simplicity, overbidding should be used sparingly and only as a desperation measure to complete a campaign. See subsection 4.3 for how we utilize overbidding.

### 3.3 Proportional Bidding

A strategy that we attempted and worked well during the labs was *proportional bidding*, or bidding that was scaled based on the proportion of people in the underlying market segments.

For example, if a campaign's segment was `MALE_YOUNG`, we submitted different bids for the underlying segments `MALE_YOUNG_HIGH_INCOME` and `MALE_YOUNG_LOW_INCOME`. Because one segment (`MALE_YOUNG_LOW_INCOME`) is a much larger segment than the other (`MALE_YOUNG_HIGH_INCOME`), we bid lower on it since, it appears, that there is more freedom to undercut.

We decided to eliminate this strategy of bidding because it resulted in a lot of relatively arbitrary parameters in our code - we found it difficult to rigorously understand what our agent was bidding. Furthermore, we found empirically in both local runs and the live competition in class that a simpler undercutting strategy could achieve essentially the same results.

Instead, we bid using a much simpler strategy, defaulting the majority of the time to undercutting at a fixed percentage of the true per user valuation. See subsection 4.3.

## 4 Our Agent

### 4.1 Key Principles

Our agent operates on a few key principles.

1. **Quality score is extremely important.** As aforementioned, we noticed through experiments and intuition early on that quality score was the most important descriptor of an agent - it should even be prioritized over profit, which is why this became part of our theorized optimal strategy and final agent. Recovering from low quality score is extremely difficult without suffering huge profit losses, whereas maintaining a strong quality score via sometimes tanking losses results in the agent having more freedom to do whatever it wants, dominating the rest of the agents with lower quality score.

   Low quality score also means that all sources of profit are essentially cut off: it becomes increasingly unlikely that the agent gets a free campaign and it is also extremely unlikely that the agent wins any contracts via auction.

   The only way back into the game with low quality score is by winning a campaign via auction by bidding a very low budget (since the effective bid is large due to low quality score) and then tanking a loss to complete the auction, potentially multiple times. With such a short game (10 days), it is nearly impossible to get back into and win the game if this is the case.

2. **Bidding on campaigns should depend on demand and profitability of the given campaigns.** Again, through tests, we noted that extra campaigns could be a significant detractor or contributor to profits; the budgets varied widely among the contracts the TA bots were

able to obtain. If we uphold the first principle and maintain a high quality score, we are also able to obtain contracts with larger budgets, so we focused on maximizing this in our agent.

## 4.2 Bidding on Campaigns

When evaluating how to bid on the available campaigns, it is important that we evaluate the profitability of each campaign. If the sector intersects with campaigns we already know to exist, it is less likely we will be able to fulfill the contract well; similarly, if the reach is too high it is also hard to fulfill it well. To maintain a high quality score, we assign more value to contracts which are easier to complete. So, we generate a private valuation for each campaign based on its sector, reach, and length.

There are two types of campaigns when viewed from the lens of our agent: known and unknown. In particular, known campaigns are campaigns where our agent has full information: its own auctions and the previously auctioned campaigns. On the other hand, unknown campaigns are campaigns where we lack information because of randomness.

For this section, we assume that all campaigns being auctioned are sold to some agent who aims to complete the campaign.

### 4.2.1 Known Campaigns

A campaign can be completely described by its market segment $S_i$, its $\delta_i$, and its length $\ell_i \in \{\ell_{\min}, \ell_{\min}+1, \ldots, \ell_{\max}\}$. With this knowledge, for any market segment, we can calculate a heuristic measure of how "competitive" that segment is based on a sum over known campaigns that contain this segment. Suppose that there are $m$ known campaigns that include this day $T$, each with parameters $(S_i, \delta_i, \ell_i)$. Then, the proportion of the segment that is covered by known auctions on the current day $T$ is defined to be

$$p_K(S, T) = \sum_{i=1}^{m} 1_{S \subseteq S_i} \frac{\delta_i |S|}{\ell_i |S_i|}. \tag{3}$$

### 4.2.2 Accounting for the Randomly Assigned Campaigns

There are also unknown auctions or auctions that will randomly appear and be randomly assigned as well. We can use expectations to approximate the contribution of the randomly assigned campaigns to the market segment of interest. In other words, we want to compute a probabilistic version of Equation 3, or

$$\mathbb{E}[\text{proportion of segment } S \text{ covered by campaigns at time } T]. \tag{4}$$

We can calculate this expectation by multiplying the average number of active campaigns owned by anyone with the average proportion of the current market segment covered by each campaign, noting that they are independent quantities. Let's begin with the simplest version of the expectation: assume that the campaign of interest is only 1 day long i.e. how difficult it is is determined purely by the contracts that will be running on the next day.

An important quantity at play is the quality score of the other agents, since it affects their probability of receiving a random contract on day $T$. Since it is impossible to know the quality scores of other agents, we specify a function $Q_{\text{avg}} : \mathbb{N}^+ \to [0, 1]$, where $Q_{\text{avg}}(T)$ represents our

guess at the approximate average clipped quality score on day $T$ (i.e. $\mathbb{E}^{\mathcal{A}\setminus i}[\min(Q,1)]$ taken over the other agents playing the game). In practice, we empirically determine values for this function by running ten equivalent bots that log their quality scores throughout the game, then averaging them. In a situation with about equal opponents, we estimate this to be a good upper bound for performance.

Then, recalling that per specification the random campaigns are at most one day long (so that the only random campaigns that matter are the ones that come into existence on day $T$) and all random quantities are chosen independently,

$$\mathbb{E}[p(S,t)] = (|\mathcal{A}|-1)Q_{\text{avg}}(t)\mathbb{E}[\delta]\mathbb{E}^{S_i}[1_{S\subseteq S_i}], \tag{5}$$

where $\mathbb{E}^{S_i}$ means averaging over all potential choices of $S_i$, the market segment of an individual campaign.

For multiple-day-long campaigns, we additionally add in expectations for how many campaigns will be on auction that day, and how that will influence how many campaigns are active at a given time. For a given contract with market segment $S$ on a given day $j$ past the current day $t$, where we know all the campaigns that are available and their lengths, we use the formula:

$$p_U(S,t,j) = \mathbb{E}[p(S,t,j)] = 5\sum_{x=t}^{\min(j,t+2)} \min\left(1, \frac{10-x+1}{3}\right)\frac{(3-(t-x))}{3}\mathbb{E}[\delta]\mathbb{E}^{S_i}[1_{S\subseteq S_i}] \tag{6}$$

The first term scales if the day is the ninth or the tenth, representing less contracts available then; the next represents the chance a contract from a previous day $x$ actually makes it to day $t$ (the length is long enough), and the other terms come from the previous equation, with a multiple of 5 because there are 5 contracts up for auction per day (we assume all are won by some participating agent).

To combine both known and unknown and develop a complete score for a market segment $S$ over $[t, t+\ell-1]$, we simply average the sum of $p_U$ and $p_K$ as follows:

$$p(S,t,\ell) = \sum_{j=0}^{\ell-1}\frac{p_U(S,t,j)+p_K(S,t+j)}{\ell}. \tag{7}$$

#### 4.2.3 Evaluation

Given these metrics, we have a general idea of which sectors will be more popular than others, allowing us to develop a valuation for each campaign given the sectors it covers by calculating how much of that segment we expect to be in demand. With this valuation calculated, we then calculate the difficulty $c$ of our contract with market segment $S$, length $\ell$, and starting from day $t$ as

$$c(S) = \frac{\mathbb{E}[p(S,t,\ell)]}{\sqrt[3]{\ell}} \tag{8}$$

We add a fudging factor based on length of contract because we believe longer contracts are easier to complete, as we have more flexibility in when to acquire users.

#### 4.2.4 Final Decision on Campaigns

To make our final decision on what to bid on a campaign, we consider another factor: if there is a high demand for campaigns, we should gradually decrease our budget bids to look to win easy contracts, as our quality score should be enough to still net some profit on easy contracts. Here, we assume that with a high quality score we can still get a good budget, especially considering the formula for given budget in a second-price auction. On the other hand, if demand is low, we can win with higher budget bids, so we gradually increase the budget instead.

This makes our bid $b$ for contracts we believe to be easy enough as so:

$$b(S) = (\text{competitiveness factor})c(S), \tag{9}$$

where the competitiveness factor is a tuned parameter that is adjusted every time we win or lose a contract. The competitiveness factor is the inverse of the intuitive notion of "competitiveness" because the auction is second-*lowest* price. In general, if we find ourselves losing a lot of contracts, we make the competitiveness factor lower to be more aggressive, and if we are winning many contracts, we make the competitiveness factor higher to chill out and not overload our agent with too many contracts, as with every new contract we accept, we place ourselves in a position to lose quality score.

Another point must be mentioned here, which is the situation that occurs when only one bidder bids on a contract. In the class competition, we saw many agents with sky-high profits at the start (Figure 1). We were quite confused about this until we saw an edge case rule describing that when there is only one bidder on a contract, the budget for the contract is based on an average of the 3 lowest quality scores in the game.

We believed that these agents must be getting extremely high budgets on campaigns that nobody else bids on, since our initial strategy was to not bid on campaigns we deemed particularly unprofitable. To counter this, even if the contract is difficult to complete, we bid the maximum on every auction so that nobody can exploit this rule to achieve an equilibrium of sorts. On the off chance we win, we win with a large budget given our quality score, and we are usually able to maintain our quality score while making some profit. This diverges from our preliminary strategy because we bid on contracts even if they're hard. However, we justify the decision by claiming we are *forced* to bid on them to prevent other agents from achieving easy profits with extremely high-budgeted contracts.

### 4.3 Bidding on Users

Bidding on users is a key portion of our agent, since we must attempt to remediate the lack of a dominant strategy using ideas developed in our optimal theoretical strategy.

First, we define the *clipped* optimal target reach fraction as

$$\eta^* = \min\{\eta_{\text{high}}, \max\{\eta_{\text{low}}, \eta\}\} \tag{10}$$

where $\eta$ is calculated according to subsection 2.2. Intuitively, we need to clip $\eta$ to a reasonable range that would preserve quality score. We set $\eta_{\text{low}} = 0.9$ and $\eta_{\text{high}} = 1.3$, following [1]. These values ensure that we complete the campaign to a degree that results in a strong quality score.

The rough ideas of our bidding algorithm are outlined here:

1. First, set $\hat{B}$ to a shaded and fixed percentage of $B - K$, where $K$ is the cost so far.

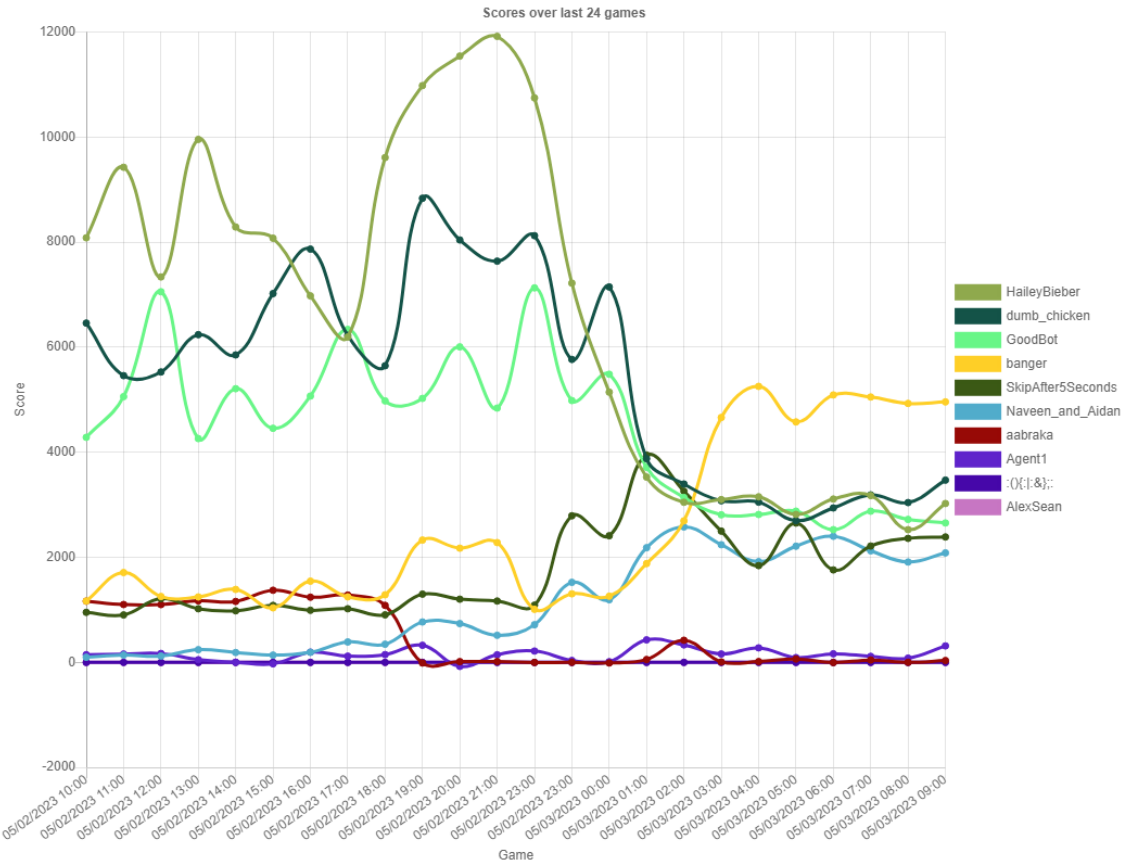


Figure 1: Results with abnormally high utility (our agent is "banger," in yellow). We suspected it was because there were campaigns won via an auction where only one bidder submitted a budget. Between 2 and 3 AM on 5/3, we submitted a new agent with many revisions, including our new strategy for bidding on auctions; after that no agent was able to derive such high profits.

2. If this is the last day of the auction and the *clipped* optimal target reach fraction $\eta^*$ has not yet been reached, aggressively up scale $\hat{B}$.

3. If $\eta^*$ has been reached, set $\hat{B}$ to something low, in case we get lucky and can make some easy profits.

4. Report $\hat{B}/R_{\text{rem}}$ as the bid, where $R_{\text{rem}}$ is the remaining reach in the campaign and $\hat{B}$ as the daily limit.

The fixed percentage we used was 0.51, with the aggressive scaling then being $\hat{B} = \hat{B} \times 2$ (overbidding, based on our theorized value of 1). This strategy can intuitively be thought of "undercutting until panic."

Note that with this rule campaigns of length 1 will always be aggressively bid on. This is because campaigns of length 1 have minimal room for error. In order to guarantee that our quality score remains consistent, we must aggressively bid on them. In a sense, we prioritize completion rather than the alternative of undercutting and gambling for additional profit when it comes to the short term campaigns.

## 4.4 Computational Efficiency

In this section, we discuss the time complexity of the various heuristics and algorithms we calculate in a game. We skip talking about initialization because this takes constant time, apart from the initialization of a set of the three-characteristic market segments, which takes time linear to the number of these (8 in this version of the game.) We will write the number of these market segments as $m$ and the number of campaigns up for auction each day as $n$ for simplicity. We also do not include the number of days each contract can run for in the calculations, as it will usually just result in a constant multiple of $n$ in any calculation.

### 4.4.1 Campaign Bids Function

As getting campaign bids is the first action requested of the agent each day, we perform some setup work here.

1. First, we clear the campaigns in our known campaigns that have already expired, i.e. their end date is before the current day. These campaigns no longer have influence on our calculations, so we discard them. To make this more efficient, we use a priority queue for storing known campaigns, sorted on their end date. The only campaigns in this list are the one free campaign we get every day and the campaigns we see up for auction in previous days; we assume they are all in the list because we bid on all of them, meaning they must have a winner. This makes the time it takes to remove the expired campaigns $O(n)$, as the removal time for each element is constant but we expect a constant fraction of the campaigns from the previous days to have expired by the current day.

2. Next, we add our free campaign for the day to the known campaigns; adding to a priority queue like this which contains some multiple of $n$ campaigns in expectation takes $O(\log(n))$ time.

3. Third, we update our heuristics for the competitiveness and pricing of the market. To update our competitiveness factor, we check our active campaigns to see if we've won any from the

last day. This operation takes $O(n)$ time. Then, we iterate over each of the three-characteristic market segments to estimate the pricing for each based on active campaigns. For each segment, we first check if we've cached the result for the current day in our map (impossible, as we reset the cache each day.) This takes $O(\log(n))$ time. Then, we iterate over each campaign in the known campaign list and add the expected reach if the segment is a subset of the campaign's target segment. We then add a constant representing the expected amount of competition from other agents. These calculations are outlined in subsubsection 4.2.1. These calculations are essentially multiplying some constants together, so over all the market segments, the time complexity is $O(m)$. So, the update of our pricing takes $O(n \log(n) + nm)$ time.

4. Finally, we form our bids for the day. For each campaign available on a given day, we calculate its difficulty based on the market segments it covers. The difficulty is computed based on the average pricing of the segments it covers over the days the campaign covers. We cache the results here for each segment on each day, so we can expect this to take $O(n \log(n) + m)$ time: $n$ contracts, with each one's difficulty calculation taking $\log(n)$ time, plus a possible need to calculate and cache expected difficulties for each segment a few days in advance ($O(m)$.) The computations after determining the difficulty of a contract are constant. So, the bid calculations take $O(n \log(n) + m)$ overall.

#### 4.4.2 Impression Bidding

When calculating our optimal reach and bid pricing, nearly all of these operations happen in constant time. The only complexity is when we use our campaign difficulty to estimate the optimal reach to target; this takes $\log(n)$ time as we will have the data for each segment already cached at this point.

#### 4.4.3 Overall Complexity

Overall, summing these complexities together and reducing, we get that the time complexity for our agent's runtime each day is $O(n \log(n) + nm)$, which is more than fast enough.

### 4.5 Changes During Live Competition

As some teams did not develop their bots until later, we were able to observe the effects of a single team joining the competition with a competitive agent when their previous agent had 0 utility. Essentially, this simulated a new agent joining the competition. We observed that in the multiple cases where this happened, everyone's utilities dropped, which makes sense: there was more competition and prices rose. To account for this, we decided to change the fixed percentage in subsection 4.3 to 0.75, a higher percentage, as we believed that more competition meant we would have to try harder for impressions. We then lowered the multiplier on the last day to 1.36, which would bring our bid just above 1 on the last day, as we had previously done. We immediately saw a rebound in results; this whole experience is shown in Figure 2. This decision also somewhat followed our theorized optimal strategy–in order to maintain quality score and therefore competitiveness, we had to bid higher.

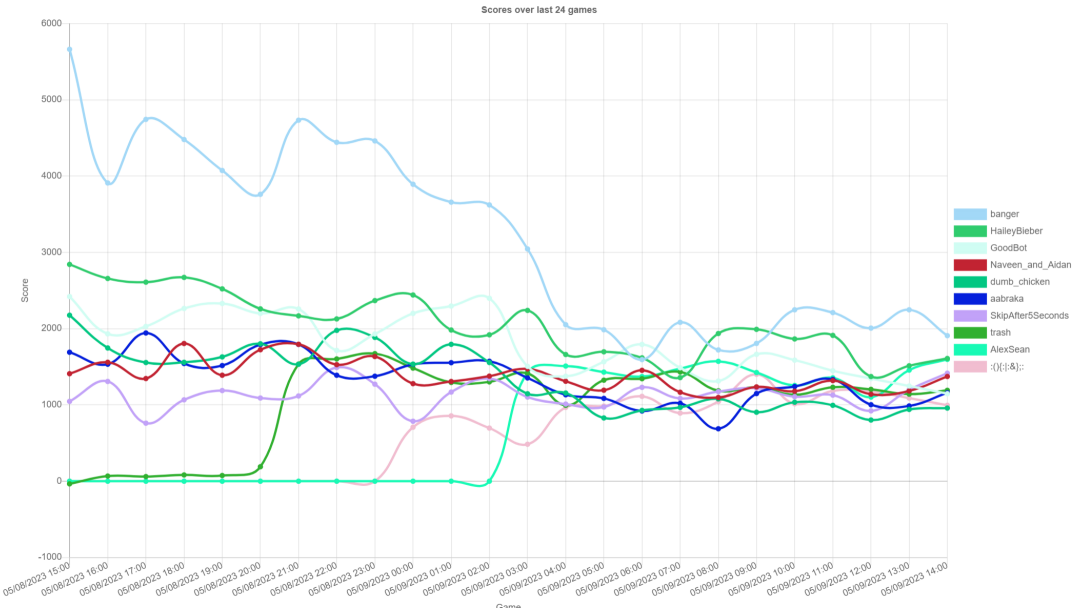


Figure 2: With the addition of multiple new agents (trash, AlexSean, :(){:—:&};:), we see our agent banger's performance trending downwards. We implemented the change in bidding percentage before the 12:00 update, and immediately our results separated from the rest of the teams.

# 5 Acknowledgements

We would like to thank Akash for his mentoring help and Prof. Amy for a great semester.

# References

[1] Tao, Bingyang, Fan Wu and Guihai Chen. “TAC AdX’14: Autonomous Agents for Realtime Ad Exchange.” Adaptive Agents and Multi-Agent Systems (2015).

[2] Greenwald, Tomer. Thesis. The Adx Game. Thesis, Prof. Yishay Mansour, 2018. https://www.tau.ac.il/ mansour/students/Tomer-Greenwald-MSc-thesis.pdf.