AdX Agent Design

Enrique Areyan Viqueira and Amy Greenwald

CS1951k Algorithmic Game Theory

April 14, 2020

AdX Game

- A market with multiple copies of heterogeneous goods (users' impressions)
- Your agent's job: procure **enough targeted** impressions at the **lowest** possible cost

Two Sources of uncertainty

- Impressions (Supply)
- Competition (Demand)

Variants of the AdX Game

- One-Day, One-Campaign
- Two-Days, Two-Campaigns

Modeling the game. Example: One-Day Game.

Game Elements:

- Let M be a set of **market segments**, and $\pi = \langle \pi_1, \pi_2, \ldots, \pi_{|M|} \rangle$ a probability mass function where π_m is the probability of drawing $m \in M$.
- A **user** belongs to a random market segment $m \in M$ (e.g., old, female) according to π.
- A **campaign** $C_j = \langle R_j, B_j, m_j \rangle$
  demands $R_j \sim r(\cdot)$ impressions,
  from a random market segment $m_j \in M$ such that $m_j \sim G(\cdot)$,
  and has budget $B_j \sim b(\cdot)$.
- An **agent** $j \in A$ is characterized by its campaign C_j.

Modeling the game. Example: One-Day Game (cont).

A **One-Day Game**, has N agents, each with a **single** campaign.

Let $\vec{x} = \langle x_1, x_2, \ldots, x_{|M|} \rangle$ be a bundle of impressions.
The **utility** u_j of agent j, as a function of bundle $\vec{x}$, is given by:

$$u_j(\vec{x}, C_j) = \rho(\mu(\vec{x}, C_j)) - p(\vec{x})$$

$p(\vec{x})$ is the total **cost** of bundle $\vec{x}$,

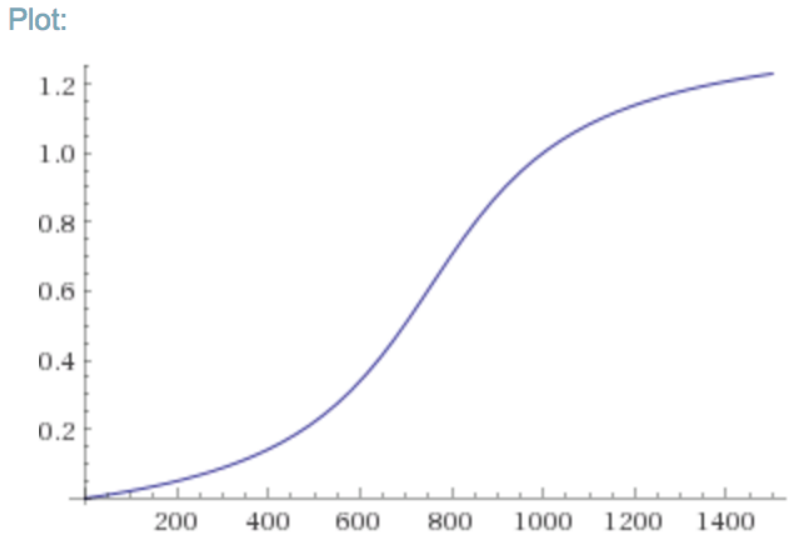
$\mu(\vec{x}, C_j) = \sum_{m \in M} x_m \mathbb{1}^j_m$ where $\mathbb{1}^j_m = 1$ if m matches m_j, 0 o/w.

$\rho(\cdot)$ is a **revenue function** mapping impressions to revenue.

Modeling the game. Example: One-Day Game (cont).

Revenue function

Figure 1: Example of revenue function for campaign with reach 1000.



Modeling the game. Example: One-Day Game (cont).

Game Dynamics

Stage 1: Agent j learns its own type C_j, but not others' types.

Stage 2: All agents compute and submit their bids.

Stage 3: The n users arrive in a random order

$$\langle \vec{m} \rangle = \left\langle m^1, m^2, \ldots, m^n \right\rangle, \text{ where } m^i \sim \pi$$

For each user i that arrives, a second-price auction is held.

The game ends and payoffs are realized.

Modeling the game. Example: One-Day Game (cont).

Strategies

Agent's j **strategy** $s_j(C_j)$ maps a campaign C_j to a tuple $\left\langle \vec{b}, \vec{l} \right\rangle$.

- $\vec{b} = \left\langle b_1, b_2, \cdots, b_{|M|} \right\rangle$ is a bid vector, where $b_m \in \mathbb{R}_+$ is the bid of agent j for market segment $m \in M$.

- $\vec{l} = \left\langle l_1, l_2, \cdots, l_{|M|} \right\rangle$ is a limit vector, where $l_m \in \mathbb{R}_+$ is the total spending limit of agent j in auctions matching market segment $m \in M$.

Notation

Denote by s_{-j} the strategies of agents other than j.

Modeling the game. Example: One-Day Game (cont).

The bundle $\vec{x} = \vec{x}(s_j, s_{-j}, \vec{m})$ procured by agent j depends on:

- its strategy s_j,
- other agents' strategies s_{-j},
- and the realization of the users (types and ordering).

We can now state our goal!

Find s_j^* that maximizes j's interim expected utility:

$$s_j^* \in \arg\max_{s_j} \left\{ \mathop{\mathbb{E}}_{\substack{C_{-j} \\ \vec{m} \sim \pi^n}} [u_j(\vec{x}(s_j(C_j), s_{-j}(C_{-j}), \vec{m}), C_j)] \right\}$$

Building an interim expected utility maximizing agent

Are we done? (Hint: nope, not even close...)

- Can we evaluate $s_{-j}(C_{-j})$?
- Only if know all other agent's strategies.
- We don't! So we proceed by making assumptions...
- Make sure you clearly state any assumptions you make!
- You should also try to justify your assumptions, even when their only justification is computational tractability.

Assumption!

We will be making assumptions about the behavior of other agents in the game (as well as other assumptions)!

Supply assumption

Supply Assumption!

Assume that, for each market segment $m \in M$, there are exactly as many users as expected according to distribution π, i.e., the number of users belonging to market segment m is $n\pi_m$.

We reduce the number of random events to think about by assuming a fixed, deterministic supply.

Demand assumption

Demand Assumption!

Assume (for the moment), we know other agents' campaigns C_{-j}.

In reality we know only the distribution of other agents' campaigns. We work with the demand assumption in what follows and later discuss ways to lift this assumption.

Game of complete information

Supply assumption + Demand assumption =

Game of complete information

Equilibrium Approach

High-Level Idea

- Compute an equilibrium and use it as your agent's prediction of the outcome of the game.
- Program your agent to play its part.
- Works well when an equilibrium is unique!

Equilibrium Assumption!

An equilibrium is a good prediction of the outcome of the game. (Justification: Repeat play would lead to an equilibrium!)

Two Equilibrium Approaches

Market Equilibrium Approach

- Compute a competitive (or Walrasian) equilibrium of the market induced by the game.
- Market equilibria do not always exist. May need to make simplifying assumptions (e.g., linear Fisher markets).

Game-Theoretic Approach

- Predict a game-theoretic equilibrium, such as Bayes-Nash or EPNE (if it exists)
- It is well-known that computing Nash equilibria is computationally complex.
- Iterative approach (may not converge): predict the behavior of the other agents in the game, and then compute your agent's move as a best-response to this prediction. Repeat.

Market Equilibrium Approach

High-Level Idea

Compute a competitive (or **Walrasian**) equilibrium of the market induced by the game, and then program your agent to **bid its part** of this equilibrium.

Definition: Walrasian equilibrium

An allocation and a pricing such that:

- All agents are utility-maximizing
- The market clears, a.k.a., supply meets demand.

Market Equilibrium Approach (cont.)

In combinatorial markets, we can compute a Walrasian Equilibrium with the following ILP. Recall WE maximizes welfare!

$$\text{maximize} \sum_j \sum_{S \subseteq U} v_j(S) x_{jS}$$

$$\text{subject to} \sum_{S \subseteq U} x_{jS} \leq 1, \quad \forall j$$

$$\sum_j \sum_{S \subseteq U : i \in S} x_{jS} \leq 1, \quad \forall i \in U$$

$$x_{jS} \in \{0, 1\}, \quad \forall j, \forall S \subseteq U$$

Market Equilibrium Approach (cont.)

Prices are in the dual ILP!!

$$\text{minimize} \sum_j u_j + \sum_i p_i$$

$$\text{subject to} \quad u_j + \sum_{i \in S} p_i \leq 1, \quad \forall j, \forall S \subseteq U$$

$$u_j, p_i \geq 0, \quad \forall i, \forall j$$

Market Equilibrium Approach (cont.)

Great! But can we actually solve the ILP in a reasonable amount of time? Note:

- The program requires a variable for *every* campaign and *every* bundle.
- ILP quickly becomes intractable, we have thousands of users.
- Is the ILP in P? in NP?

WE computation for combinatorial markets

Answer to the last question: it depends on how we model campaigns' valuations. (Let's say in P for now, ask me for details if you want to know more).

Market Equilibrium Approach (cont.)

Question

How would you relax (i.e., what assumptions would you make) to be able to solve for a WE in a reasonable amount of time?

Some remarks

- Note that WE computation is *independent* of agents' strategies!
- Most useful in a setting where other-agent strategies are difficult to predict.
- But what if we have reasonable predictions of other agents strategies?

Game-Theoretic Approach

High-Level Idea

Compute a game-theoretic equilibrium, and then program your agent to **bid its part** of this equilibrium.

Best-reply dynamics

- Predict the behavior of the other agents in the game
- Compute your agent's best-response to this prediction
- Repeat

Game-Theoretic Approach

Deterministic bid assumption!

Suppose agent j, for every segment matching its campaign:

- bids $\rho_j \left(\frac{B_j}{R_j}\right)$, where $\rho_j \in [0, 1]$ is a bid shading parameter
- has a spending limit equal to its campaign's budget B_j

Compute best response

- Now we **have** $s_{-j}(C_{-j})$!
- In particular, for a fixed C_{-j} we can simulate the game
- Can we solve for the optimal bid given C_{-j}?

Game-Theoretic Approach (cont.)

Let's analyze a **single market segment** m with initial supply N_m. Suppose there are $|A|$ other agents.

Order bids: $\rho_1 \left(\frac{B_1}{R_1}\right) \geq \rho_2 \left(\frac{B_2}{R_2}\right) \geq \cdots \geq \rho_a \left(\frac{B_{|A|}}{R_{|A|}}\right)$

The kth bidder gets x_k impressions and pays p_k.

$$x_k = \underbrace{\min\left(\underbrace{B_k\left(\frac{1}{\rho_{k+1}}\right)\left(\frac{R_{k+1}}{B_{k+1}}\right)}_{\text{Affordable \# Impr.}}, \underbrace{N_m - \sum_{t=1}^{k-1} x_t}_{\text{remaining supply}}\right)}_{\substack{k\text{th agent either spends its entire budget} \\ \text{or takes all the remaining supply}}}; p_k = x_k \rho_{k+1}\left(\frac{B_{k+1}}{R_{k+1}}\right)$$

Game-Theoretic Approach (cont.)

Algorithm 1 Simulate Auctions

procedure GETALLOCATIONANDPAYMENT(k)

  $x_i \leftarrow 0, p_i \leftarrow 0$ for all i; $curSupply \leftarrow N_m$

  Insert bid in kth position, $b \in \left(\rho_k \frac{B_k}{R_k}, \rho_{k-1} \frac{B_{k-1}}{R_{k-1}}\right)$

  for $i = 1; i \leq k; i++$ **do**

    $x_i \leftarrow \min\left(B_i \frac{1}{\rho_{i+1}} \frac{R_{i+1}}{B_{i+1}}, curSupply\right)$

    $p_i \leftarrow x_i \rho_{i+1} \frac{B_{i+1}}{R_{i+1}}$

    $curSupply \leftarrow curSupply - x_i$

  return (x_k, p_k)

Game-Theoretic Approach (cont.)

Choose optimal position k^*

$$k^* \in \underset{k \in \{1,2,\ldots,|A|\}}{\arg\max} \{u(x_k, p_k)\} \quad ,$$

where $u(x_k, p_k)$ is your utility for x_k impressions at price p_k.

Finally, choose your bid: $b \in \left(\rho_{k^*}\left(\frac{B_{k^*}}{R_{k^*}}\right), \rho_{k^*-1}\left(\frac{B_{k^*-1}}{R_{k^*-1}}\right)\right)$

Define $\frac{B_0}{R_0} = \infty$.

Game-Theoretic Approach (cont.)

Some remarks

- How might the other agents best-respond to our agent's choice of k^*, and its corresponding bids?
- Is this process guaranteed to converge? Probably not, so what should we do if/when it does not?

Some questions

- What is the computational complexity of finding a best response as described, under the deterministic bid assumption?
- How might we generalize this approach to handle multiple market segments? (It seems making an assumption about other agents' budgets, as well as bids, would help.)
- What about the many possible different orderings of the users? How can we manage this uncertainty?

Some General Remarks

Multiple days

The discussion so far has been for the one-day game.
Can you think of ways to generalize for multiple days?

Incomplete information

Going back to our assumptions, can we lift some of them?

Dealing with uncertainty

Given an approach for a complete-information setting (e.g., market or game-theoretic equilibria), how would you used it in the case of incomplete information?

Simple approach

- Sample the uncertainty of the game n times.
- For each sample, compute a bid vector and limits.
- Aggregate (e.g., average) these bid vectors and limits.
- Report bids and limits based on the aggregate.

But maybe aggregating across samples is not a great idea?

If in 50% of the samples, the best move is to go left, and in the other 50%, the best move is to go right, should a robot walk straight ahead? What if just ahead lies a tree?

Dealing with uncertainty (cont.)

Given an approach for a complete-information setting (e.g., market or game-theoretic equilibria), do scenario analysis.

Scenario evaluation

- Generate a set of candidate strategies S.
- Use the strategy $s \in S$ that maximizes some estimate of profit, averaged over a set of n samples (i.e., scenarios).

But how to generate candidate policies?

- Sample the uncertainty of the game m times.
- For each sample σ, compute a bid vector and limits.
- Add the strategy s_σ induced by sample σ to the set S.