Budgeting and Bidding in Ad Systems: Theory and Practice

Aranyak Mehta
Market Algorithms, Google Research, Mountain View, CA.
Outline

Topics:

1. **Budget Allocation:**
   - Algorithms based on Online Matching
   - Algorithms based on Reinforcement Learning

2. **Auto-Bidding:**
   - Algorithms
   - Equilibrium
Search Ads System Overview

1. Query on Google.com
2. Ads Inventory
3. Budget
4. Scoring
5. Auction
6. Return Ads

Advertiser response

Advertiser optimization

Reporting
Motivation: Demand constraints in Repeated Auctions

- Auction each arriving ad slot.
- Stateful because of budget constraints.
- Mismatched bidding components.

Targeting: “flowers”

Bids: $1 per click

Budget: $500 per day

Traffic = 1000 clicks!
Allocation on top of auction

- Can model it as a repeated online auction with demand constraint.
  - Impossibility results
  - Impractical

- Design: Allocation layer on top of online stateless auction:

  ![Diagram](image_url)

  - “Pure” Optimization
  - Mechanism design / Game theory
Two Methods

- Bid Lowering
  - “Your bid was too high.”

- Throttling
  - “Your targeting was too broad.”
Two Methods

- **Bid Lowering**
  - “Your bid was too high.”

- **Throttling**
  - “Your targeting was too broad.”

---

Targeting: “flowers”

Bids: $1 per click

Budget: $500 per day

Traffic = 1000 clicks!
Two Methods

- Bid Lowering
  - “Your bid was too high.”

- Throttling
  - “Your targeting was too broad.”

- Targeting: “flowers”
  - Bids: $1 per click
  - Budget: $500 per day
  - Traffic = 1000 clicks!
Two Methods

● **Bid Lowering**
  - “Your bid was too high.”
  - **Heuristic**: reduce bid by some multiplier.
  - **Theoretical abstraction**: How to incorporate the interaction across ads?

● **Throttling**
  - “Your targeting was too broad.”

![Diagram showing Targeting: “flowers”, Bids: $1 per click, Budget: $500 per day, Traffic = 1000 clicks!](image)
An abstraction: The “AdWords Problem”

Definition (M., Saberi, Vazirani, Vazirani, FOCS 2005, JACM 2007)

- $N$ advertisers, advertiser $a$ has budget $B(a)$
- $M$ search queries that arrive online, advertiser $a$ has bid $bid(a, q)$ for query $q$

Decision: Algorithm needs to allocate $q$ to one of the advertisers irrevocably (or discard). Allocated advertiser depletes budget by $bid(a, q)$

Goal: Maximize sum of values over all queries

Generalizes online bipartite matching [KVV’90]
The AdWords Problem

<table>
<thead>
<tr>
<th>Advertisers</th>
<th>Queries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Budgets = 100</td>
<td>100 copies each</td>
</tr>
</tbody>
</table>

- Advertisers
  - Budgets = 100

- Queries
  - 100 copies each

Graph:
- Edge: Budgets = 100 (0.99)
- Edge: 100 copies each (1.0)
- Edge: 100 copies each (1.0)
The AdWords Problem

Advertisers
Budgets = 100

Queries
100 copies each

“Greedy” solution would lead to ½ of the maximum potential.
The MSVV Algorithm

spent(a) = fraction of a’s budget already used up.

When query $q$ arrives, allocate it to an advertiser that maximizes

$$\text{bid}(a, q) \times \Psi(\text{spent}(a))$$

where

$$\Psi(x) \propto 1 - \exp(-(1 - x)).$$

**Theorem [MSVV05]**

Achieves optimal competitive ratio $1 - 1/e \sim 63\%$

**Note:** A worst-case guarantee, even if we do not have any estimates.
The AdWords Problem

Advertisers

Queries

Budgets = 100

100 copies each
What about stochastic input?

[Intuition:] [MSVV05] proof updates dual variables / bid multipliers as the sequence arrives (explicitly shown in [BJN07]). In iid or random order setting, you can sample and estimate duals.

[Algorithm:]
○ Sample initial segment
○ Solve the LP for the sample
○ Use those duals for the rest of the sequence.

[Theorem:] 1-epsilon in random order model
Display ads

[FKMMP WINE 2009]

- Original solution: LP / max flow on estimated graph.
- Algorithm 1
  \[ w' = w - \text{penalty}(\text{usage}, \text{capacity}) \]
- Algorithm 2: Learning duals a la DH09

Targeting: “NYTimes front page”

Bids: $1 per imp

Capacity: 5M imps
Two Methods

- **Bid Lowering**
  - “Your bid was too high.”

- **Throttling**
  - “Your targeting was too broad.”

Targeting: “flowers”

Bids: $1 per click

Budget: $500 per day

Traffic = 1000 clicks!
Throttling

- Extreme of bid lowering
  - bid multiplier either 0 or 1.

- "Vanilla" Throttling:

  Probability of participation in each auction = Budget / Max-Spend-estimate
Throttling

- **Optimized Throttling** [Karande, Mehta, Srikant WSDM 2013]
  - Provide an optimized set of options for the advertiser, rather than random.

- Knapsack formulation

\[
\max_S \sum_{i \in S} ctr_i \\
\text{s.t. : } \sum_{i \in S} spend_i \leq B
\]

Greedy heuristic: Participate in auctions with best \( \frac{ctr}{spend} = 1/cpc \)
Optimized Throttling

Expected spend

Budget

Threshold

Metric (e.g., 1/cpc)

Estimate offline, implement online
Optimized Throttling

% Change in BC Clicks per Dollar

- OT-CTR
- OT-Clicks
- BidScaling
- LP-Clicks

head
all
A lot more work in this direction.

Budget Allocation | Reinforcement Learning
CAN

DEEP REINFORCEMENT LEARNING

DESIGN

WORST CASE

ONLINE OPTIMIZATION ALGORITHMS?

Part of a broader theme
[A New Dog learns Old Tricks, Kong, Liaw, M., Sivakumar, ICLR 2019.]
"AdWords MDP"

State at time $t$

$\text{spend}(1)$, $\text{spend}(2)$, ..., $\text{spend}(N)$

$\text{bid}(1, t)$, $\text{bid}(2, t)$, ..., $\text{bid}(N, t)$

Next State

Action: which ad to allocate to

Reward

Ad 1

$\text{spend}(1) + \text{bid}(1, t)$, ..., $\text{bid}(1, t+1)$, ...

Ad 2

Ad N
Learning an Agent

**Goal:** Learn agent’s policy function that maps state to action.

**Network:** Standard 5-layer 500-neuron-per-layer network with ReLU non-linearity.

**Training:** Standard REINFORCE policy-gradient learning with learning rate $1e^{-4}$, batch size 10.

Takes few hours typically on single-threaded standard Linux desktop.

**Punch line:** It works!
Training Set: Universal Distribution

Two expanded versions of the Z-graph
How does the network solve it?

Did it “Find the MSVV Algorithm”? How to evaluate?

Probing the network as a black box.

Warm-up: 0/1 bids

Pretend we’re in the middle of execution for an instance. We’re at an item arrival.

All advertisers have bid=1
All except advertiser i have spend=0.5.

x-axis: spend
y-axis: Probability that advertiser i wins the item
How does the network solve it?

Did it “Find the MSVV Algorithm”? How to evaluate?

1. **Probing the network as a black box.**

**General Case:**

All advertisers except advertiser 0 have bid=1, spend=0.5.

- **x-axis:** spend(0)
- **y-axis:** Minimum bid to win the item.

**Blue:** Learned Agent

**Green:** OPT (MSVV)
Table 3: This table compares the performance of the learned algorithm compared the BALANCE in the discretized state space. Here, the agent is trained on the adversarial graph with the ad slots arriving in a permuted order. The agent was only trained on the input instance with 20 advertisers and a common budget of 20 but tested on instances with up to $10^6$ ad slots.

<table>
<thead>
<tr>
<th>No. of advertisers</th>
<th>Budgets (common)</th>
<th>No. of ad slots</th>
<th>Approx. of BALANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>10</td>
<td>100</td>
<td>0.9</td>
</tr>
<tr>
<td>20</td>
<td>20</td>
<td>400</td>
<td>0.92</td>
</tr>
<tr>
<td>30</td>
<td>30</td>
<td>900</td>
<td>0.88</td>
</tr>
<tr>
<td>10</td>
<td>2000</td>
<td>20000</td>
<td>0.85</td>
</tr>
<tr>
<td>10</td>
<td>4000</td>
<td>40000</td>
<td>0.85</td>
</tr>
<tr>
<td>25</td>
<td>4000</td>
<td>100000</td>
<td>0.84</td>
</tr>
<tr>
<td>50</td>
<td>400</td>
<td>20000</td>
<td>0.84</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>10000</td>
<td>0.85</td>
</tr>
<tr>
<td>100</td>
<td>1000</td>
<td>100000</td>
<td>0.85</td>
</tr>
<tr>
<td>200</td>
<td>100</td>
<td>20000</td>
<td>0.85</td>
</tr>
<tr>
<td>500</td>
<td>50</td>
<td>25000</td>
<td>0.85</td>
</tr>
<tr>
<td>1000</td>
<td>100</td>
<td>10000</td>
<td>0.84</td>
</tr>
<tr>
<td>25</td>
<td>40000</td>
<td>1000000</td>
<td>0.84</td>
</tr>
</tbody>
</table>
What does this mean for practice?

- RL can potentially find worst case algorithms.
- We know RL can adapt to real distributions / data well.
- Opens up potential to merge ML and Algorithms to work more in tandem.
Auto-Bidding: Algorithms and Equilibrium

[Aggarwal, Badanidiyuru, M., 2019]
Performance Auto-Bidding products

Fine Grained bidding:
- Keywords: Bids
- Budget
Performance Auto-Bidding products

High level expressivity:
- Goals
- Constraints

Advertiser ➔ Autobidder ➔ Auctions

Auction: bid
## Performance Auto-Bidding products

<table>
<thead>
<tr>
<th></th>
<th>Goal</th>
<th>Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Budget Optimizer</td>
<td>Clicks</td>
<td>Budget</td>
</tr>
<tr>
<td>Target CPA</td>
<td>Conversions</td>
<td>Avg cost-per-conversion</td>
</tr>
<tr>
<td>Other potential</td>
<td>Post-install-events</td>
<td>Avg cost-per-install</td>
</tr>
<tr>
<td>examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
A General Framework

\[
\max \sum_i x_i \text{ctr}_i v_i
\]

subject to

\[
\sum_i x_i \text{ctr}_i \text{cpc}_i \leq B_c + \sum_i x_i \text{ctr}_i w_{ic} \quad \forall c
\]

\[
x_i \in \{0, 1\}
\]

Should you buy the i-th click?

The value for the i-th click

Constraint specific constants

Expected Spend
A General Framework

- **Budget Optimizer:**
  - $v_i = 1$, $B = \text{budget}$, $w_i = 0$

- **Target CPA:**
  - $v_i = \text{pCVR}$, $B = 0$
  - $\frac{\sum_i x_i \text{ctr}_i \text{cpc}_i}{\sum_i x_i \text{ctr}_i \text{cvr}_i} \leq T \Rightarrow w_i = T \times \text{cvr}_i$

- **Target CPC constraint:**
  - $\frac{\sum_i x_i \text{ctr}_i \text{cpc}_i}{\sum_i x_i \text{ctr}_i} \leq M \Rightarrow w_i = M$

\[
\begin{align*}
\text{Max} & \quad \sum_i x_i \text{ctr}_i v_i \\
\text{s.t.} & \quad \sum_i x_i \text{ctr}_i \text{cpc}_i \leq B_c + \sum_i x_i \text{ctr}_i w_{ic} \quad \forall c \\
& \quad x_i \in \{0, 1\}
\end{align*}
\]
Optimal Bidding Algorithm

● Given the LP and all the data, including CPCs, we can solve to say which items you want to pick.

● Can a simple bidding formula lead to the same outcomes?

● Does the answer depend on the underlying auction properties?
Bidding Algorithm

- Complementary slackness conditions say that you want to take all the items with

\[ c_{pc} \leq \frac{v_i + \sum_c \alpha_c w_{ic}}{\sum_c \alpha_c} \]

- Can implement it by setting bid:

\[ b(i) := \frac{v_i + \sum_c \alpha_c w_{ic}}{\sum_c \alpha_c} \]

Not entirely new, studied in various forms earlier, e.g., [Agrawal-Devanur'15]

---

Primal Linear Program

\[
\begin{align*}
\max \sum_i x_{is} c_{tris} v_i \\
\forall c, \sum_{i,s} x_{is} c_{tris} p_{c_{is}} & \leq B_c + \sum_{i,s} x_{is} c_{tris} w_{ic} \\
\forall i, \sum_s x_{is} & \leq 1 \\
\forall i,s,x_{is} & \geq 0
\end{align*}
\]

Dual Linear Program

\[
\begin{align*}
\min \sum_i \delta_i + \sum_c \alpha_c B_c \\
\forall i,s, \delta_i + \sum_c \alpha_c c_{tris} (c_{pc_{is}} - w_{ic}) & \geq c_{tris} v_i \\
\forall i, \delta_i & \geq 0 \\
\forall c, \alpha_c & \geq 0
\end{align*}
\]
Bidding Algorithm

**Theorem:** With the correct setting of the parameters \( \alpha_c \) the bidding formula is optimal iff the auction is truthful.

**Note:** The parameters can be learned from past data and updated online.
**Intuition**

Target CPA + Budget

\[
b(i) = \frac{\sum_c \alpha_c w_{ic}}{\sum_c \alpha_c} \cdot \frac{v_i + \alpha \cdot T \cdot v_i}{\alpha} = \gamma v_i
\]

Target CPA + Target CPC + Budget

\[
b(i) = \frac{\sum_c \alpha_c w_{ic}}{\sum_c \alpha_c} \cdot \frac{v_i + \alpha_1 \cdot T \cdot v_i + \alpha_2 \cdot M}{\alpha_1 + \alpha_2} = \gamma v_i + \eta M
\]
Bidding equilibrium

- What happens when everyone adopts autobidding?
  - Is there an equilibrium?
  - Do we get good overall value in equilibrium, or can it result in bad dynamics leading to low value and revenue?
Does there exist an Equilibrium?

Not Obvious due to interactions.

**Theorem:** An approx equilibrium exists s.t. each bidder bids almost optimally, given what other bidders are bidding.

**Proof:** Using Brouwer’s fixed point theorem.

\[
\phi(\kappa_a) = \kappa_a \cdot (1 + \epsilon)^{(\text{Target CPA} - \text{Realized CPA})}
\]
Performance in equilibrium: Price of Anarchy

**Efficiency == Weighted sum of advertiser goals**

E.g., for tCPA: \[ \text{Efficiency} = \sum_a \text{tCPA}(a) \times \text{Conversions}(a) \]

(total value of conversions)

GLOBAL OPT: Give q to ad with highest tCPA * pcvr
(and charge first price / for free).
Price of Anarchy

How much value do we lose by allowing one agent per bidder?

For the general autobidding problem, $POA = 2$.

You do not lose more than 50% value in the worst case, and there are instances in which you could lose 50%.

Due to multiple constraints (e.g., budgets), we use the "Liquid POA" definition.
Proof Idea

A := Queries s.t.
   Equilibrium-Ad = OPT-Ad

Equilibrium >= OPT(A)

B := Queries s.t.
   Equilibrium-Ad /= OPT-Ad

Use:
- Second-price auction
- Bid >= tCPA

Equilibrium >= OPT(B)

2 * Equilibrium >= OPT
Questions?