Computational Advertising

Greedy Algorithms
Competitive Algorithms
Picking the Best Ad
The Balance Algorithm
Online Algorithms

- Classic model of *(offline)* algorithms:
  - You get to see the entire input, then compute some function of it.
- *Online algorithm*:
  - You get to see the input one piece at a time, and need to make irrevocable decisions along the way.
  - Similar to data stream models.
Example: Bipartite Matching

- Two sets of nodes.
- Some edges between them.
- Maximize the number of nodes paired 1-1 by edges.
Bipartite Matching – (2)

\[ M = \{(1,a),(2,b),(3,d)\} \text{ is a } \text{matching} \text{ of cardinality } |M| = 3. \]
Bipartite Matching – (3)

$M = \{(1,c),(2,b),(3,d),(4,a)\}$ is a **perfect matching** (all nodes matched).
Problem: Find a maximum-cardinality matching for a given bipartite graph.

- A perfect one if it exists.

There is a polynomial-time offline algorithm (Hopcroft and Karp 1973).

But what if we don’t have the entire graph initially?
Initially, we are given the set of men. In each round, one woman’s set of choices is revealed. At that time, we have to decide either to:

- Pair the woman with a man.
- Don’t pair the woman with any man.

Example applications: assigning tasks to servers or Web requests to threads.
Online Matching – (2)

(1,a)
(2,b)
(3,d)
Pair the new woman with any eligible man.
  - If there is none, don’t pair the woman.
- How good is the algorithm?
Competitive Ratio

- For input I, suppose greedy produces matching $M_{\text{greedy}}$ while an optimal matching is $M_{\text{opt}}$.

$$\text{Competitive ratio} = \min_{\text{all possible inputs I}} \left( \frac{|M_{\text{greedy}}|}{|M_{\text{opt}}|} \right).$$
Let O be the optimal matching, and G the matches produced by a run of the greedy algorithm.

Consider the sets of women:

- A: Matched in G, not in O.
- B: Matched in both.
- C: Matched in O, not in G.
During the greedy matching, every woman in C found her match in the optimal solution taken by another woman.

Thus, $|A| + |B| \geq |C|$.  

Surely, $|A| + |B| \geq |B|$.  

Thus, $|G| = |A| + |B| \geq (|B| + |C|)/2 = |O|/2$.  

If you’re greater than each of two things, you are greater than their average.
Worst-Case Scenario

|Greedy| = 2;
|Opt| = 4.
History of Web Advertising

  - Initial form of web advertising.
  - Popular websites charged X$ for every 1000 “impressions” of ad.
    - Called “CPM” rate.
    - Modeled on TV, magazine ads.
  - Untargeted to demographically targeted.
  - Low clickthrough rates.
    - low ROI for advertisers.
Performance-Based Advertising

- Introduced by Overture around 2000.
  - Advertisers “bid” on search keywords.
  - When someone searches for that keyword, the highest bidder’s ad is shown.
  - Advertiser is charged only if the ad is clicked on.
- Similar model adopted by Google with some changes around 2002.
  - Called “Adwords.”
Performance-based advertising works!

- Multi-billion-dollar industry.

Interesting problems:

- What ads to show for a search?
- If I’m an advertiser, which search terms should I bid on and how much should I bid?
Adwords Problem

- A stream of queries arrives at the search engine
  - q1, q2, ...
- Several advertisers bid on each query.
- When query q_i arrives, search engine must pick a subset of advertisers whose ads are shown.
- **Goal**: maximize search engine’s revenues.
- Clearly we need an online algorithm!
- Simplest online algorithm is Greedy.
Each ad has a different likelihood of being clicked.

**Example:**
- Advertiser 1 bids $2, click probability = 0.1.
- Advertiser 2 bids $1, click probability = 0.5.
  - Click-through rate measured by historical performance.

**Simple solution:**
- Instead of raw bids, use the “expected revenue per click.”
The Adwords Innovation

<table>
<thead>
<tr>
<th>Advertiser</th>
<th>Bid</th>
<th>CTR</th>
<th>Bid * CTR</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>$1.00</td>
<td>1%</td>
<td>1 cent</td>
</tr>
<tr>
<td>B</td>
<td>$0.75</td>
<td>2%</td>
<td>1.5 cents</td>
</tr>
<tr>
<td>C</td>
<td>$0.50</td>
<td>2.5%</td>
<td>1.125 cents</td>
</tr>
</tbody>
</table>
## The Adwords Innovation

<table>
<thead>
<tr>
<th>Advertiser</th>
<th>Bid</th>
<th>CTR</th>
<th>Bid * CTR</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>$0.75</td>
<td>2%</td>
<td>1.5 cents</td>
</tr>
<tr>
<td>C</td>
<td>$0.50</td>
<td>2.5%</td>
<td>1.125 cents</td>
</tr>
<tr>
<td>A</td>
<td>$1.00</td>
<td>1%</td>
<td>1 cent</td>
</tr>
</tbody>
</table>
Complications – (2)

- Each advertiser has a limited budget
  - Search engine guarantees that the advertiser will not be charged more than their daily budget.
Assume all bids are 0 or 1.
Each advertiser has the same budget B.
One advertiser is chosen per query.
Let’s try the greedy algorithm:
- Arbitrarily pick an eligible advertiser for each keyword.
Bad Scenario For Greedy

- Two advertisers A and B.
- A bids on query x, B bids on x and y.
- Both have budgets of $4.
- Query stream: x x x x y y y y.
- Possible greedy choice: B B B B _ _ _ _.
- Optimal: A A A A B B B B.
- Competitive ratio = 1/2.
  - This is actually the worst case.
[Mehta, Saberi, Vazirani, and Vazirani].

For each query, pick the advertiser with the largest unspent budget who bid on this query.

- Break ties arbitrarily.
Example: Balance

- Two advertisers A and B.
- A bids on query x, B bids on x and y.
- Both have budgets of $4.
- Query stream: x x x x y y y y.
- Balance choice: B A B A B B _ _.
- Optimal: A A A A B B B B.
- Competitive ratio = 3/4.
Consider simple case: two advertisers, \( A_1 \) and \( A_2 \), each with budget \( B > 1 \), an even number.

We’ll consider the case where the optimal solution exhausts both advertisers’ budgets.

- I.e., optimal revenue to search engine = 2B.
- Balance must exhaust at least one advertiser’s budget.
  - If not, we can allocate more queries.
  - Assume Balance exhausts \( A_2 \)’s budget.
Analyzing Balance

Opt revenue = 2B
Balance revenue = 2B-x = B+y

Note: only green queries can be assigned to neither. A blue query could have been assigned to $A_1$.

We claim $y \geq x$ (next slide).
Balance revenue is minimum for $x=y=B/2$.
Minimum Balance revenue = $3B/2$.
Competitive Ratio = $3/4$. 
Case 1: At least half the blue queries are assigned to $A_1$ by Balance.

- Then $y \geq B/2$, since the blues alone are $\geq B/2$.

Case 2: Fewer than half the blue queries are assigned to $A_1$ by Balance.

- Let $q$ be the last blue query assigned by Balance to $A_2$. 

Balance allocation
Since $A_1$ obviously bid on $q$, at that time, the budget of $A_2$ must have been at least as great as that of $A_1$.

Since more than half the blue queries are assigned to $A_2$, at the time of $q$, $A_2$’s remaining budget was at most $B/2$.

Therefore so was $A_1$’s, which implies $x \leq B/2$, and therefore $y \geq B/2$ and $y \geq x$.

Thus Balance assigns $\geq 3B/2$. 

Balance allocation
In the general case, competitive ratio of Balance is $1 - 1/e = \text{approx. } 0.63$.

Interestingly, no online algorithm has a better competitive ratio.

Won’t go through the details here, but let’s see the worst case that gives this ratio.
N advertisers, each with budget $B >> N >> 1$.

$N \times B$ queries appear in $N$ rounds.

Each round consists of a single query repeated $B$ times.

- **Round 1 queries**: bidders $A_1, A_2, ..., A_N$.
- **Round 2 queries**: bidders $A_2, A_3, ..., A_N, ...$
- **Round i queries**: bidders $A_i, ..., A_N, ...$
- **Round N queries**: only $A_N$ bids.

**Optimum allocation**: round i queries to $A_i$.

- Optimum revenue $N \times B$. 
After $i$ rounds, the first $i$ advertisers have dropped out of the bidding.

- Why? All subsequent queries are ones they do not bid on.
- Thus, they never get any more queries, even though they have budget left.
Balance Allocation

After $k$ rounds, sum of allocations to each of $A_k, \ldots, A_N$ is
$$S_k = S_{k+1} = \ldots = S_N = \sum_{1 \leq i \leq k} \frac{B}{(N-i+1)}.$$

If we find the smallest $k$ such that $S_k \geq B$, then after $k$ rounds we cannot allocate any queries to any advertiser.
Each width represents the amount of budget spent by $A_k$ after $k$ rounds.

Or in terms of fractions (dividing by $B$):

$$\frac{1}{1} \quad \frac{1}{2} \quad \frac{1}{3} \quad \ldots \quad \frac{1}{N-k+1} \quad \ldots \quad \frac{1}{N-1} \quad \frac{1}{N}$$
Fact: $H_n = \sum_{1 \leq i \leq n} \frac{1}{i} \sim \log_e(n)$ for large $n$.  
- Result due to Euler.

$1/1 \quad 1/2 \quad 1/3 \ldots \quad 1/(N-k+1) \quad \ldots \quad 1/(N-1) \quad 1/N$

$log(N)$

$log(N) - 1$ $S_k = 1$

$S_k = 1$ implies $H_{N-k} = \log(N) - 1 = \log(N/e)$.  
$N-k = N/e$ [Why? $\log(N-k) = H_{N-k} = \log(N/e)$].  
k = $N(1-1/e) \sim 0.63N$.  

Euler  Line above
Balance Analysis

- So after the first $N(1-1/e)$ rounds, we cannot allocate a query to any advertiser.
- Revenue = $BN(1-1/e)$.
- Competitive ratio = $1-1/e$. 
**General Version of Problem**

- Arbitrary bids, budgets.
- Balance can be terrible.
- **Example**: Consider two advertisers $A_1$ and $A_2$, each bidding on query $q$.
  - $A_1$: $x_1 = 1$, $b_1 = 110$.
  - $A_2$: $x_2 = 10$, $b_2 = 100$.
- First 10 occurrences of $q$ all go to $A_1$, and $A_1$ then gets 10 $q$’s for every one that $A_2$ gets.
  - What if there are only 10 occurrences of $q$?
    - Opt yields $100$; Balance yields $10$. 
Generalized Balance

- Arbitrary bids; consider query q, bidder i.
  - Bid = x_i.
  - Budget = b_i.
  - Amount spent so far = m_i.
  - Fraction of budget remaining f_i = 1-m_i/b_i.
- Define \( \psi_i(q) = x_i(1-e^{-f_i}) \).
- Allocate query q to bidder i with largest value of \( \psi_i(q) \).
- Same competitive ratio (1-1/e).