CS425: Algorithms for Web Scale Data

Lecture 7: Web Advertising

Most of the slides are from the Mining of Massive Datasets book. These slides have been modified for CS425. The original slides can be accessed at: www.mmds.org
Online Algorithms

- **Classic model of algorithms**
  - You get to see the entire input, then compute some function of it
  - In this context, “offline algorithm”

- **Online Algorithms**
  - You get to see the input one piece at a time, and need to make irrevocable decisions along the way
Online Bipartite Matching
Bipartite Graphs

- Bipartite graph:
  - Two sets of nodes: A and B
  - There are no edges between nodes that belong to the same set.
  - Edges are only between nodes in different sets.
Bipartite Matching

- Maximum Bipartite Matching: Choose a subset of edges $E_M$ such that:
  1. Each vertex is connected to at most one edge in $E_M$
  2. The size of $E_M$ is as large as possible

- Example: Matching projects to groups

$M = \{(1,a),(2,b),(3,d)\}$ is a matching
Cardinality of matching = $|M| = 3$
Bipartite Matching

- **Maximum Bipartite Matching:** Choose a subset of edges $E_M$ such that:
  1. Each vertex is connected to at most one edge in $E_M$
  2. The size of $E_M$ is as large as possible

- **Example:** Matching projects to groups

![Graph with edges connecting projects and groups. M = \{(1,c),(2,b),(3,d),(4,a)\} is a maximum matching. Cardinality of matching = |M| = 4]
Example: Bipartite Matching

$M = \{(1,c), (2,b), (3,d), (4,a)\}$ is a perfect matching

**Perfect matching** … all vertices of the graph are matched

**Maximum matching** … a matching that contains the largest possible number of matches
Problem: Find a maximum matching for a given bipartite graph

- A perfect one if it exists

There is a polynomial-time offline algorithm based on augmenting paths (Hopcroft & Karp 1973, see http://en.wikipedia.org/wiki/Hopcroft-Karp_algorithm)

But what if we do not know the entire graph upfront?
Online Bipartite Matching Problem

- Initially, we are given the set of projects
- The TA receives an email indicating the preferences of one group.
- The TA must decide at that point to either:
  assign a preferred project to this group, or
  not assign any projects to this group

- Objective is to maximize the number of preferred assignments

*Note: This is not how your projects were assigned 😊*
Greedy Online Bipartite Matching

- **Greedy algorithm**
  
  For each group $g$
  
  Let $P_g$ be the set of projects group $g$ prefers
  
  if there is a $p \in P_g$ that is not already assigned to another group
  
  assign project $p$ to group $g$
  
  else
  
  do not assign any project to $g$
Greedy Online Graph Matching: Example

\[(1, a)\]
\[(2, b)\]
\[(3, d)\]
For input $I$, suppose greedy produces matching $M_{\text{greedy}}$ while an optimal matching is $M_{\text{opt}}$

Competitive ratio =

$$\min_{\text{all possible inputs } I} \left( \frac{|M_{\text{greedy}}|}{|M_{\text{opt}}|} \right)$$

(what is greedy’s worst performance over all possible inputs $I$)
Analysis of the Greedy Algorithm

**Step 1:** Find a lower bound for the competitive ratio

**Definitions:**
- $M_o$: The optimal matching
- $M_g$: The greedy matching
- $L$: The set of vertices from A that are in $M_o$, but not in $M_g$
- $R$: The set of vertices from B that are connected to at least one vertex in $L$
Claim: All vertices in R must be in \( M_g \)

Proof:
- By contradiction, assume there is a vertex \( v \in R \) that is not in \( M_g \).
- There must be another vertex \( u \in L \) that is connected to \( v \).
- By definition \( u \) is not in \( M_g \) either.
- When the greedy algorithm processed edge \((u, v)\), both vertices \( u \) and \( v \) were available, but it matched none of them. This is a contradiction!

Fact: \( |M_o| \leq |M_g| + |L| \)

Adding the missing elements to \( M_g \) will make its size to be at least the size of the optimal matching.

Fact: \( |L| \leq |R| \)

Each vertex in \( L \) was matched to another vertex in \( M_o \)
Analysis of the Greedy Algorithm (cont’d)

- **Fact**: \(|R| \leq |M_g|\)
  
  *All vertices in \(R\) are in \(M_g\)*

- **Summary**:
  
  \(|M_o| \leq |M_g| + |L|\)
  
  \(|L| \leq |R|\)
  
  \(|R| \leq |M_g|\)

- **Combine**:
  
  \(|M_o| \leq |M_g| + |L|\)
  
  \(\leq |M_g| + |R|\)
  
  \(\leq 2 |M_g|\)

Lower-bound for competitive ratio:

\[
\frac{|M_g|}{|M_o|} \geq \frac{1}{2}
\]
We have shown that the competitive ratio is at least 1/2. However, can it be better than 1/2?

**Step 2:** Find an upper bound for competitive ratio:

Typical approach: Find an example.

If there is at least one example that has competitive ratio of \( r \), it must mean that competitive ratio cannot be greater than \( r \).

Greedy matching: \((1, a), (2, b)\)

The optimal matching is: \((4, a), (3, b), (1, c), (2, d)\)

Competitive ratio = \( \frac{1}{2} \) for this example

So, competitive ratio ≤ \( \frac{1}{2} \)
Greedy Matching Algorithm

- We have shown that competitive ratio for the greedy algorithm is $1/2$.
  - We proved that both lower bound and upper bound is $1/2$

- **Conclusion**: The online greedy algorithm can result in a matching solution that has half the size of an optimal offline algorithm in the worst case.
Web Advertising
**Banner ads (1995-2001)**

- Initial form of web advertising
- Popular websites charged X$ for every 1,000 “impressions” of the ad
  - Called **“CPM”** rate (Cost per thousand impressions)
- Modeled similar to TV, magazine ads
- From **untargeted** to demographically targeted
- Low click-through rates
- Low ROI for advertisers
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
Ads vs. Search Results

**Geoico** Car Insurance. Get an auto insurance quote and save today...
Geoico auto insurance, online car insurance quote, motorcycle insurance quote, online insurance sales and service from a leading insurance company.
www.geoico.com/ - 21k - Sep 22, 2005 - Cached - Similar pages
  - Auto Insurance - Buy Auto Insurance
  - Contact Us - Make a Payment
More results from www.geoico.com >

Geoico, Google Settle Trademark Dispute
The case was resolved out of court, so advertisers are still left without legal guidance on use of trademarks within ads or as keywords.

Google and Geoico settle AdWords dispute | The Register
Google and car insurance firm Geoico have settled a trade mark dispute over ... Car insurance firm GEOICO sued both Google and Yahoo! subsidiary Overture in ...
www.theregister.co.uk/2005/09/09/google_geoico_settlement/ - 21k - Cached - Similar pages

Geoico v. Google
... involving a lawsuit filed by Government Employees Insurance Company (Geoico). Geoico has filed suit against two major Internet search engine operators, ...
www.consumeraffairs.com/news04/geoico_google.html - 19k - Cached - Similar pages
Performance-based advertising works!
  - Multi-billion-dollar industry

Interesting problem:
What ads to show for a given query?
  - (This lecture)

If I am an advertiser, which search terms should I bid on and how much should I bid?
  - (Not focus of this lecture)
Adwords Problem

- **Given:**
  1. A set of bids by advertisers for search queries
  2. A click-through rate for each advertiser-query pair
  3. A budget for each advertiser (say for 1 month)
  4. A limit on the number of ads to be displayed with each search query

- **Respond to each search query with a set of advertisers such that:**
  1. The size of the set is no larger than the limit on the number of ads per query
  2. Each advertiser has bid on the search query
  3. Each advertiser has enough budget left to pay for the ad if it is clicked upon
A stream of queries arrives at the search engine: $q_1, q_2, \ldots$

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

- **Simplification:** Instead of raw bids, use the “expected revenue per click” (i.e., Bid*CTR)

- **Clearly we need an online algorithm!**
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>

Click through rate | Expected revenue
## 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: Budget

- Two complications:
  - Budget
  - CTR of an ad is unknown

- Each advertiser has a limited budget
  - Search engine guarantees that the advertiser will not be charged more than their daily budget
Complications: CTR

- CTR: Each ad has a different likelihood of being clicked
  - Advertiser 1 bids $2, click probability = 0.1
  - Advertiser 2 bids $1, click probability = 0.5
  - Clickthrough rate (CTR) is measured historically
    - Very hard problem: Exploration vs. exploitation
      Exploit: Should we keep showing an ad for which we have good estimates of click-through rate
      or
      Explore: Shall we show a brand new ad to get a better sense of its click-through rate
Simplified Problem

- We will start with the following simple version of Adwords:
  - One ad shown for each query
  - All advertisers have the same budget $B$
  - All bids are $1$
  - All ads are equally likely to be clicked and $CTR = 1$

- We will generalize it later.
Greedy Algorithm

- Simple greedy algorithm:
  For the current query $q$, pick any advertiser who:
  1. has bid 1 on $q$
  2. has remaining budget

- What is the competitive ratio of this greedy algorithm?

- Can we model this problem as bipartite matching?
Bipartite Matching Model

**Online algorithm:**
For each new query $q$, assign a bid if available

Equivalent to the online greedy bipartite matching algorithm, which had competitive ratio $= 1/2$.

So, the competitive ratio of this algorithm is also $1/2$. 
Example: 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:** \( x x x x y y y y \)
  - Worst case greedy choice: \( B B B B \ _ _ _ _ \)
  - Optimal: \( A A A A A B B B B \)
  - Competitive ratio = \( \frac{1}{2} \)
- **This is the worst case!**
  - **Note:** Greedy algorithm is deterministic – it always resolves draws in the same way
**BALANCE Algorithm [MSVV]**

- **BALANCE** Algorithm by Mehta, Saberi, Vazirani, and Vazirani
  - For each query, pick the advertiser with the largest unspent budget
    - Break ties arbitrarily *(but in a deterministic way)*
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: A B A B B _ _
  - Optimal: A A A A B B B B

- Competitive ratio $\leq \frac{3}{4}$
Analyzing BALANCE: Simple Case

- Try to prove a lower bound for the competitive ratio
  - i.e. Consider the worst-case behavior of BALANCE algorithm

- Start with the simple case:
  - 2 advertisers $A_1$ and $A_2$ with equal budgets $B$
  - Optimal solution exhausts both budgets
  - All queries assigned to at least one advertiser in the optimal solution
    - Remove the queries that are not assigned by the optimal algorithm
    - This only makes things worse for BALANCE

![Diagram]

- Queries allocated to $A_1$ in the optimal solution
- Queries allocated to $A_2$ in the optimal solution
Analysis of BALANCE: Simple Case

- **Claim**: BALANCE must exhaust the budget of at least one advertiser
  - **Proof by contradiction**: Assume both advertisers have left over budgets
    - Consider query q that is assigned in the optimal solution, but not in BALANCE.
    - Contradiction: q should have been assigned to at least the same advertiser because both advertisers have available budget.

Goal: Find a lower bound for: \[ \frac{|S_{balance}|}{|S_{optimal}|} \]
Without loss of generality, assume the whole budget of $A_2$ is exhausted.

Claim: All blue queries (the ones assigned to $A_1$ in the optimal solution) must be assigned to $A_1$ and/or $A_2$ in the BALANCE solution.

Proof by contradiction: Assume a blue query $q$ not assigned to either $A_1$ or $A_2$. Since budget of $A_1$ is not exhausted, it should have been assigned to $A_1$. 
Analysis of BALANCE: Simple Case

- Some of the green queries (the ones assigned to \( A_2 \) in the optimal solution) are not assigned to either \( A_1 \) or \( A_2 \). Let \( x \) be the # of such queries.
- Prove an upper bound for \( x \)
  - Worst case for the BALANCE algorithm.
Consider two cases for $z$:

- **Case 1**: $z \geq B/2$

  
  - $\text{size (A}_1) = y + z \geq B/2$
  
  
  - $\text{size (A}_1 + A_2) = B + y + z \geq 3B/2$
Case 2: z < B/2
Consider the time when last blue query was assigned to A₂:

A₂ has remaining budget of ≤ B/2
For A₂ to be chosen, A₁ must also have remaining budget of ≤ B/2
Analysis of BALANCE: Simple Case

- **Case 2**: $z < B/2$

  - $\text{size } (A_1) \geq B/2$
  - $\text{size } (A_1 + A_2) = B + \text{size}(A_1) \geq 3B/2$
Analysis of BALANCE: Simple Case

- **Conclusion:**
  
  \[
  \frac{|S_{balance}|}{|S_{optimal}|} \geq \frac{3B}{2B} = \frac{3}{4}
  \]

  *Assumption: Both advertisers have the same budget \( B \)*

- Can we generalize this result to any 2-advertiser problem?
  - The textbook claims we can.
  - **Exercise:** Find a counter-example to disprove textbook’s claim.
    - **Hint:** Consider two advertisers with budgets \( B \) and \( B/2 \).
For multiple advertisers, worst competitive ratio of BALANCE is $1 - 1/e = \text{approx. } 0.63$

- Interestingly, no online algorithm has a better competitive ratio!

- See textbook for the worst-case analysis.
Arbitrary bids and arbitrary budgets!

In a general setting BALANCE can be terrible

- Consider two advertisers $A_1$ and $A_2$
  - $A_1$: $x_1 = 1$, $b_1 = 110$
  - $A_2$: $x_2 = 10$, $b_2 = 100$
- Assume we see 10 instances of $q$
- BALANCE always selects $A_1$ and earns 10
- Optimal earns 100
*Arbitrary bids:* consider query $q$, bidder $i$

- Bid = $x_i$
- Budget = $b_i$
- Amount spent so far = $m_i$
- Fraction of budget left over $f_i = 1 - m_i/b_i$
- 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* $(1 - 1/e)$
Conclusions

- **Web Advertising**: Try to maximize ad revenue from a stream of queries
- **Online algorithms**: Make decisions without seeing the whole input set
- **Approximation algorithms**: Theoretically prove upper and lower bounds w.r.t. the optimal solutions.