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 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

1. M = {(1,a),(2,b),(3,d)} is a matching
2. Cardinality of matching = |M| = 3

Matching Algorithm

- 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 upfront?
Online problem

- Initially, we are given the set Boys
- In each round, one girl’s choices are revealed
- At that time, we have to decide to either:
  - Pair the girl with a boy
  - Don’t pair the girl with any boy
- Example of application: assigning tasks to servers

Greedy algorithm

- Pair the new girl with any eligible boy
  - If there is none, don’t pair girl
- How good is the algorithm?

Competitive Ratio

- For input I, suppose greedy produces matching \( M_{\text{greedy}} \) while an optimal matching is \( M_{\text{opt}} \)
- Competitive ratio = 
  \[
  \min_{\text{all possible inputs } I} \left( \frac{|M_{\text{greedy}}|}{|M_{\text{opt}}|} \right)
  \]

Analyzing the greedy algorithm

- Consider the set \( G \) of girls matched in \( M_{\text{opt}} \) but not in \( M_{\text{greedy}} \)
- Then it must be the case that every boy adjacent to girls in \( G \) is already matched in \( M_{\text{greedy}} \)
- There must be at least \( |G| \) such boys
  - Otherwise the optimal algorithm could not have matched all the G girls
- Therefore
  \[
  |M_{\text{greedy}}| \geq |G| = |M_{\text{opt}} - M_{\text{greedy}}| \\
  |M_{\text{greedy}}|/|M_{\text{opt}}| \geq 1/2
  \]
History of web advertising

- Banner ads (1995-2001)
  - Initial form of web advertising
  - Popular websites charged X$ for every 1000 "impressions" of ad
    - Called "CPM" rate
  - Modeled similar to TV, magazine ads
  - Untargeted to demographically tagged
  - 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”

Ads vs. search results

- 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 to bid?

Adwords problem

- A stream of queries arrives at the search engine
  - q1, q2,...
- Several advertisers bid on each query
- When query q, 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!

Web 2.0

Greedy algorithm

- Simplest algorithm is greedy
- It’s easy to see that the greedy algorithm is actually optimal!
Complications (1)
- 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 measured historically
- Simple solution
  - Instead of raw bids, use the "expected revenue per click"

Complications (2)
- Each advertiser has a limited budget
  - Search engine guarantees that the advertiser will not be charged more than their daily budget

Simplified model (for now)
- Assume all bids are 0 or 1
- Each advertiser has the same budget B
- One advertiser 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: xxxxyyyy
  - Worst case greedy choice: BBBB____
  - Optimal: AAAABBBB
  - Competitive ratio = ½
- Simple analysis shows this is the worst case

BALANCE algorithm [MSVV]
- [Mehta, Saberi, Vazirani, and Vazirani]
- For each query, pick the advertiser with the largest unspent budget
  - 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: xxxxyyyy
  - BALANCE choice: ABABBB____
  - Optimal: AAAABBBB
  - Competitive ratio = ¾
Analyzing BALANCE

- Consider simple case: two advertisers, A1 and A2, each with budget B (assume B > 1)
- Assume optimal solution exhausts both advertisers’ budgets
- BALANCE must exhaust at least one advertiser’s budget
  - If not, we can allocate more queries
  - Assume BALANCE exhausts A2’s budget

**General Result**

- In the general case, worst competitive ratio of BALANCE is 1 – 1/e = 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

**Worst case for BALANCE**

- N advertisers, each with budget B ≫ N ≫ 1
- NB queries appear in N rounds of B queries each
- Round 1 queries: bidders A1, A2, …, AN
- Round 2 queries: bidders A2, A3, …, AN
- Round i queries: bidders Ai, …, AN
- Optimum allocation: allocate round i queries to Ai
  - Optimum revenue NB

**BALANCE allocation**

After k rounds, sum of allocations to each of bins A1, …, AN is
\[ S_k = S_{k+1} = \ldots = S_N = \frac{B}{N^k} \]

If we find the smallest k such that \( S_k \geq B \), then after k rounds we cannot allocate any queries to any advertiser

**BALANCE analysis**

\[
\begin{align*}
B/1 & \quad B/2 & \quad B/3 & \quad \ldots & \quad B/(N-k+1) & \quad \ldots & \quad B/(N-1) & \quad B/N \\
1/1 & \quad 1/2 & \quad 1/3 & \quad \ldots & \quad 1/(N-k+1) & \quad \ldots & \quad 1/(N-1) & \quad 1/N \\
\end{align*}
\]

\[
\begin{align*}
S_1 & = B \\
S_i & = 1 \\
\end{align*}
\]
BALANCE analysis

- Fact: $H_n = \sum_{i=1}^{n} \frac{1}{i} \approx \log(n)$ for large $n$
  - Result due to Euler

$\frac{1}{1} \quad \frac{1}{2} \quad \frac{1}{3} \quad \ldots \quad \frac{1}{(N-k+1)} \quad \ldots \quad \frac{1}{N}$

$log(N)$

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

$S_k = 1$ implies $H_{N-k} = \log(N) - 1 = \log(N/e)$

$N-k = N/e$

$k = N(1-1/e)$

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
- Consider query $q$, advertiser $i$
  - Bid = $x_i$
  - Budget = $b_i$
- BALANCE can be terrible
  - Consider two advertisers $A_1$ and $A_2$
    - $A_1$: $x_1 = 1$, $b_1 = 110$
    - $A_2$: $x_2 = 10$, $b_2 = 100$

Generalized BALANCE

- 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 $(1-1/e)$