

Online Matching

Anil Maheshwari

anil@scs.carleton.ca
School of Computer Science
Carleton University
Canada

Competitive Ratio

Competitive Ratio

Ratio of the value returned by an Online Algorithm in comparison to the (best) Offline algorithm.

What is the largest value of $c \leq 1$, such that

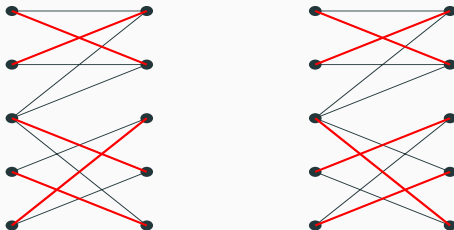
$$\frac{\text{Value (online algorithm)}}{\text{Value (off-line algorithm)}} \geq c$$

Bipartite Matching

Bipartite Matching

Let $G = (V = L \cup R, E)$ be a bipartite graph where the vertex set V consists of the sets L and R (referred to as 'left' and 'right' sets) and a set E of edges (v, w) where $v \in L$ and $w \in R$.

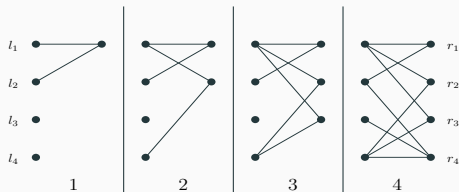
The set $M \subseteq E$ is a matching in G if no two edges in M share a vertex.



Online Matching Problem

Input: All the vertices in the set L are known in advance, but the vertices in R and the edges are presented over time. At each time instant $t \in \{1, 2, 3, \dots\}$, a new vertex $r_t \in R$ and all its incident edges arrive.

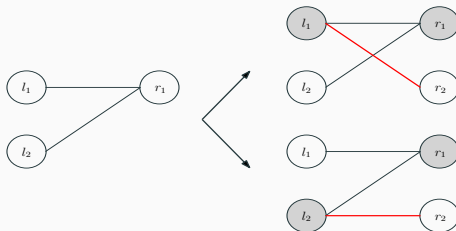
Task: The online matching algorithm needs to decide among all the currently unmatched neighbors of r_t in the set L to which vertex (if any) r_t should be matched. The vertex r_t remains matched to that vertex for the rest of the algorithm.



Output: Find a matching M of the largest possible size. Find M such that the ratio $\frac{|M|}{|M^*|}$ is as large as possible, where M^* is (offline) maximum matching in G .

Lower Bound on Deterministic Algorithms

Consider a bipartite graph on 4 vertices, where $L = \{l_1, l_2\}$ and $R = \{r_1, r_2\}$. At the first time step the algorithm is presented with the vertex r_1 and the two incident edges (r_1, l_1) and (r_1, l_2) .



Adversary chooses what to do in the next time stamp and therefore the Competitive Ratio = $\frac{1}{2}$

Greedy Bipartite Matching Algorithm

Greedy Online Matching Algorithm:

At time step t :

Match r_t to any of the unmatched neighbors in the set L

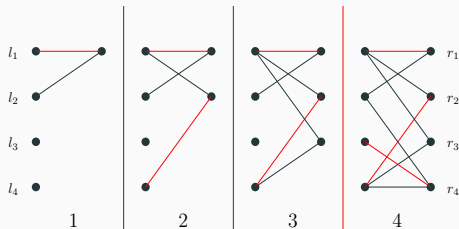


Figure 1: Competitive Ratio: $\frac{|M|}{|M^*|} = \frac{3}{4}$

We will show that the Greedy Online Matching Algorithm has a competitive ratio $\geq \frac{1}{2}$

Competitive Ratio of Greedy Algorithm

Competitive Ratio

Online greedy matching algorithm is $\frac{1}{2}$ -competitive.

Proof Sketch:

- Let M be the greedy matching and M^* be an optimal matching.
- Consider the following two disjoint subsets of vertices in L :
 - (a) Let $A \subseteq L$ be the vertices that are in M^* but not in M and
 - (b) $B \subseteq L$ that are both in M^* and M .
- We claim that $|A| \leq |N(A)| \leq |M|$ and $|B| \leq |M|$.
- Since $|M^*| = |A| + |B| \leq 2|M|$, or $|M| \geq |M^*|/2$. $\square$

Adwords

Setting:

1. A set of bids from advertisers for search queries
2. Budget for each advertiser
3. CTR for each advertiser
4. Limit on the number of displayed ads

Objectives:

1. Ensure that advertiser has bid for the query
2. If ad is clicked, advertiser has budget to pay
3. Maximize the total revenue = value of the bid $\times$ CTR for the ad.

Uncertainty: Online Search queries

Simplifications:

1. Only display one advertisement per search query
2. Each advertiser has the same starting budget
3. All CTR's are the same
4. All bids are binary (either 0 or 1)

Greedy Ad Algorithm

On the arrival of a search query q :

Among the advertisers who have some remaining budget, pick any of them who has bid 1 for q

Advertiser	Search Query	Initial Budget
A_1	x	\$4
A_2	x, y	\$4

GA for Ads

Greedy algorithm is $\frac{1}{2}$ -competitive

Proof: Homework.

Idea: Look at the proof of the greedy online bipartite matching algorithm.

Balance Algorithm

Balance Algorithm

Balance Algorithm

Assign the query to the advertiser who bids for the query and has the highest remaining budget.

Break the ties deterministically.

Advertiser	Search Query	Initial Budget
A_1	x	\$4
A_2	x, y	\$4

2-Advertisers

The balance algorithm is $\frac{3}{4}$ -competitive for 2-advertisers.

Proof Sketch: Consider the scenario outlined on the previous page, and assume that offline revenue is $2B$, where B is the starting budget of the advertisers.

- Assume that we have in total B queries of type x and B queries of type y .
- Note that budget of A_2 is fully consumed by Balance.
- If at least $1/2$ of x 's are assigned to A_1 , then the online revenue will be $\geq B + 1/2B = 3/2B$.
- Show that the number of queries of type x that are assigned to A_1 is at least $B/2$.
- Use the properties of Balance, and argue if either of A_1 or A_2 gets the last query of type x , the consumed budget of $A_1 \geq B/2$. $\square$

Many Advertisers $\rightarrow$ b -Matching

A bipartite graph $G = (L \cup R, E)$

Vertices in R come in an online manner along with the edges incident to them.

Parameter $b > 0$ is a fixed positive integer.

When a vertex $w \in R$ is revealed to the algorithm, possibly match it one of its neighbors $v \in L$ provided that the number of vertices matched to v so far by the algorithm is $< b$

Whatever decision that we make for w cannot be altered on the arrival of future vertices of R .

BALANCE Algorithm

For each vertex $w \in R$ in order of its appearance:

Among all the neighbors of w in L that have been matched $< b$ times,
match w to that neighbor (if any) that is matched to the fewest.

An alternate view of the problem:

1. Vertices in $L = \{1, 2, \dots, N\}$ are advertisers, where each of them have a daily budget of \$1
2. Each advertiser bids a small amount $\epsilon > 0$ for a set of keywords of their liking.
3. The set R comprises of keyword queries that arrive in an online manner.
4. Each query keyword needs to be assigned to an advertiser (if any) who has bid for that keyword and has some remaining budget $\geq \epsilon$.
5. If the query is assigned to an advertiser, its budget is decreased by ϵ and we generate a revenue of ϵ .

BALANCE algorithm assigns the query to the advertiser who has

1. Bid for that keyword
2. Has remaining budget $\geq \epsilon$
3. Among all those advertisers has the largest remaining budget.

Problem: Maximize the revenue generated by the algorithm, i.e., maximize the sum total of the budget spent by the advertisers.

An Example

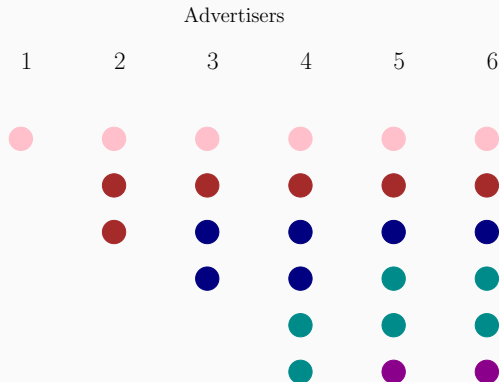


Figure 2: BALANCE with 6 advertisers numbered 1 to 6. Each has a budget of \$1 and can pay for 6 queries. Advertiser i bids for keywords $\{K_1, \dots, K_i\}$. Thirty-six online queries arrive: first 6 for K_1 (pink dots), followed by next 6 for K_2 (dark red),... BALANCE handles 26 queries whereas optimal can handle all 36 queries.

Setup:

- L has N vertices (advertisers) $1, \dots, N$, each with a budget of \$1
- N keywords $K_1, \dots, K_N$
- Advertiser i bids only for the keywords $\{K_1, \dots, K_i\}$
- Set $\epsilon = \frac{1}{N}$
- Each advertiser can pay for at most N queries

Query Sequence:

- Total of N^2 queries
- First N queries are for the keyword K_1
- Next N queries are for the keyword K_2
- ...
- ...
- Last N queries are for the keyword K_N

Offline Revenue = N

- First N queries corresponding to the keyword K_1 are distributed evenly among all the advertisers
- Next N queries corresponding to the keyword K_2 are distributed among the advertisers $2, \dots, N$
- In general, N queries for the keyword K_i are distributed evenly among advertisers $i, \dots, n$ provided that they have sufficient remaining budget

Which queries the advertiser N receives?

- at least one query of type K_1
- at least one query of type K_2
- ...
- at least $\lfloor \frac{N}{N-i} \rfloor$ queries of type K_i

When does N -th advertiser runs out of budget ?

$$\begin{aligned} N &\leq \left\lfloor \frac{N}{N} \right\rfloor + \left\lfloor \frac{N}{N-1} \right\rfloor + \cdots + \left\lfloor \frac{N}{N-i} \right\rfloor \\ &\leq N \left(\frac{1}{N} + \frac{1}{N-1} + \cdots + \frac{1}{N-i} \right) \end{aligned}$$

Condition on i

$$i \approx N(1 - \frac{1}{e})$$

Recall that n -th Harmonic Number $H_n = \sum_{i=1}^n \frac{1}{i} \approx \ln n$.

Express $\frac{1}{N} + \frac{1}{N-1} + \cdots + \frac{1}{N-i}$ as the difference of two Harmonic numbers.

Competitive Ratio

BALANCE algorithm's competitive ratio is at most $1 - \frac{1}{e}$

Proof: The above example illustrates that the revenue of Balance is $N(1 - \frac{1}{e})$.

Offline, we will assign first N queries to Advertiser 1, next N queries to Advertiser 2, and so on.

Total offline revenue = N .

Thus, Competitive Ratio = $\frac{N(1 - \frac{1}{e})}{N} = 1 - \frac{1}{e}$.

□

Non-uniform bids

Suppose 2-advertisers bid for the same set of queries but they have different bid amounts:

Advertiser	Bid Amount	Budget
A_1	\$1	\$110
A_2	\$10	\$100

Consider a set of 10 queries:

Revenue(offline) = 100

Revenue(Balance Algorithm) = 10

Non-uniform bids (contd.)

New Idea: For query q :

Suppose advertiser A_i bids amount x_i for q :

Let f_i be the **fraction** of the unspent budget of A_i

Define $\Psi_i = x_i(1 - e^{-f_i})$

Assign q to the advertiser that has the largest value of Ψ_i

Non-uniform bids - Example

Advertiser	Bid Amount	Budget
A_1	$x_1 = \$1$	\$110
A_2	$x_2 = \$10$	\$100

$$\Psi_i = x_i(1 - e^{-f_i})$$

Advertiser	1st Query	2nd Query	3rd Query
Ψ_{A_1}	$1(1 - e^{-1}) = 0.63$	$1(1 - e^{-1}) = 0.63$	$1(1 - e^{-1}) = 0.63$
Ψ_{A_2}	$10(1 - e^{-1}) = 6.3$	$10(1 - e^{-.9}) = 5.9$	$10(1 - e^{-.8}) = 5.5$

Conclusions

1. The competitive ratio of Balance is $1 - \frac{1}{e}$
2. Non-uniform bid algorithm has a competitive ratio of $1 - \frac{1}{e}$
3. No online algorithm can have a competitive ratio greater than $1 - \frac{1}{e}$
4. Main Reference: Aranyak Mehta, Amin Saberi, Umesh V. Vazirani, Vijay V. Vazirani: AdWords and generalized online matching. Journal of ACM 54(5): 22 (2007)
5. Bala Kalyanasundaram, Kirk Pruhs: An optimal deterministic algorithm for online b-matching. Theoretical Computer Science 233(1-2): 319-325 (2000)