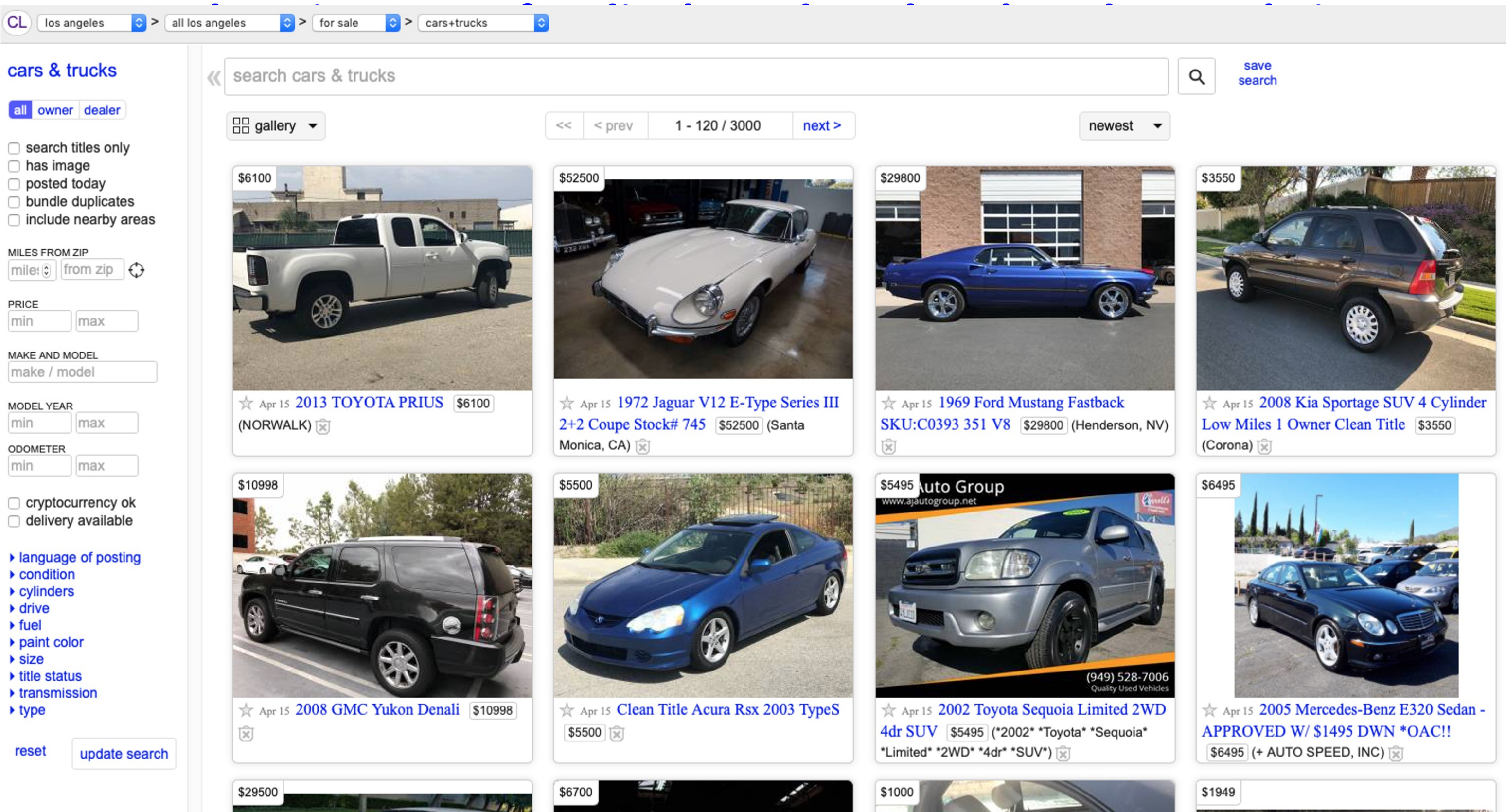
OUTLINE

- Web Advertising
  - Definition of Web-Advertising
  - Greedy algorithm
  - BALANCE algorithm
  - Generalized-BALANCE algorithm

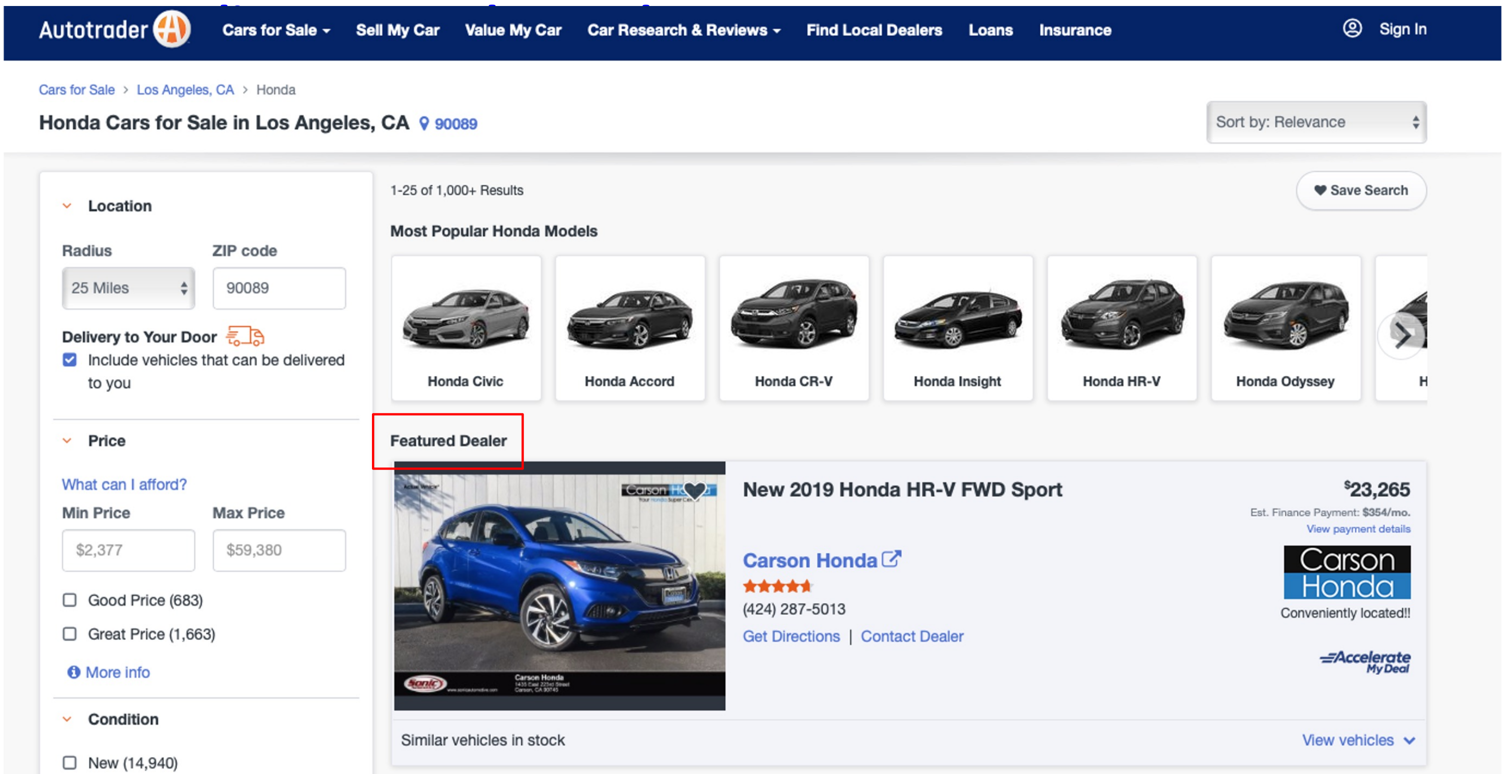
Types of Web Ads

- Advertisers post ads directly
  - Craig's List, auto trading sites, social networks



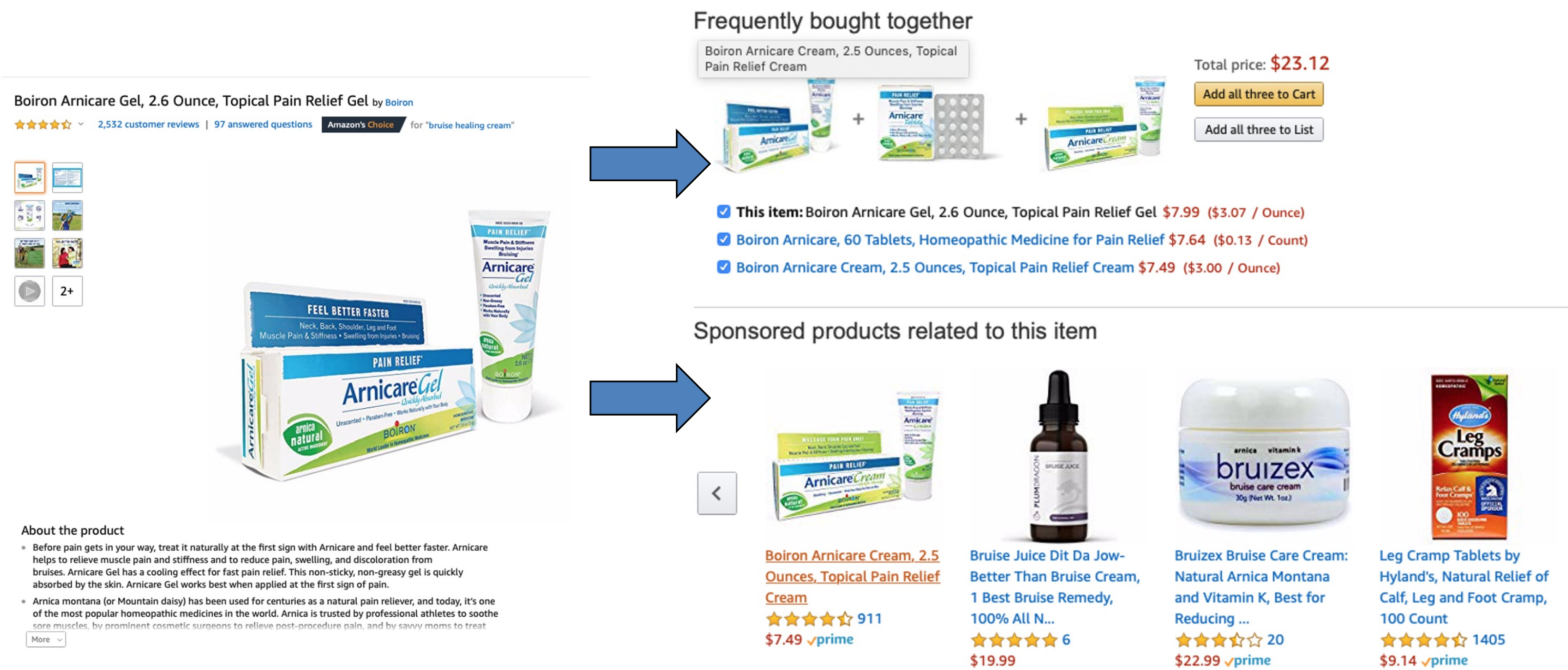
Types of Web Ads

- Advertisers pay for display ads to be placed on websites
  - Often has a fixed price per ***impression*** (one display of the ad with download of page by a user)

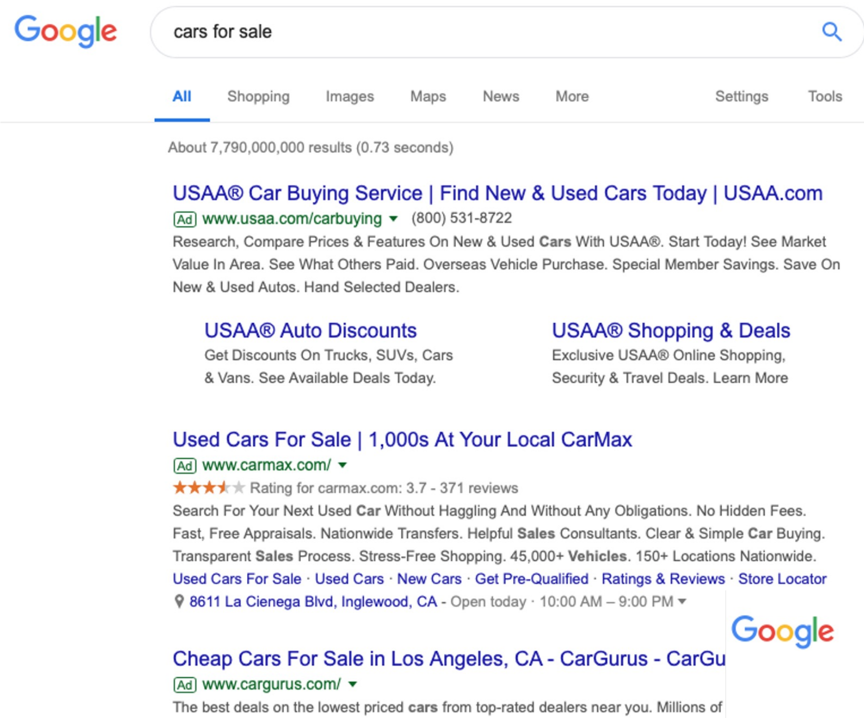


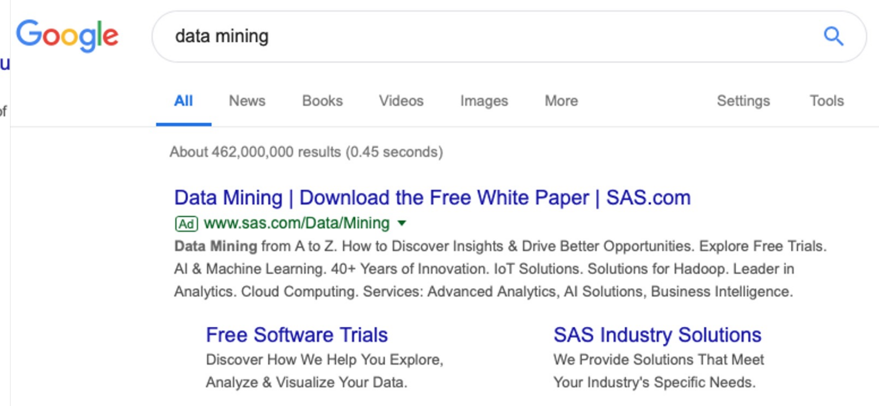
Types of Web Ads

- Online stores show ads
  - Amazon, Macy's, etc.
  - Selected by store to maximize probability customer will buy product
  - Collaborative Filtering



Types of Web Ads: Search Ads

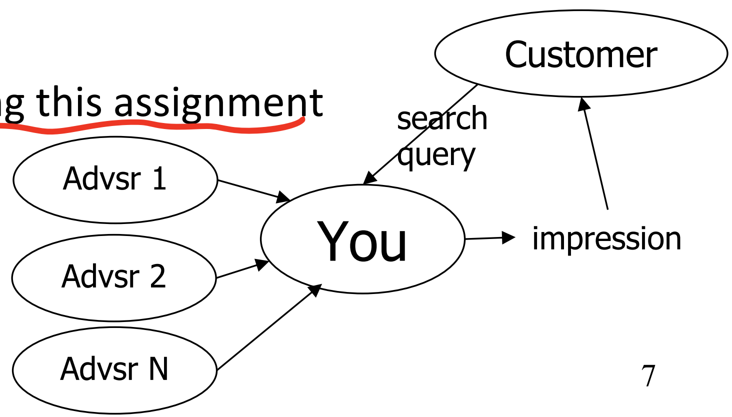




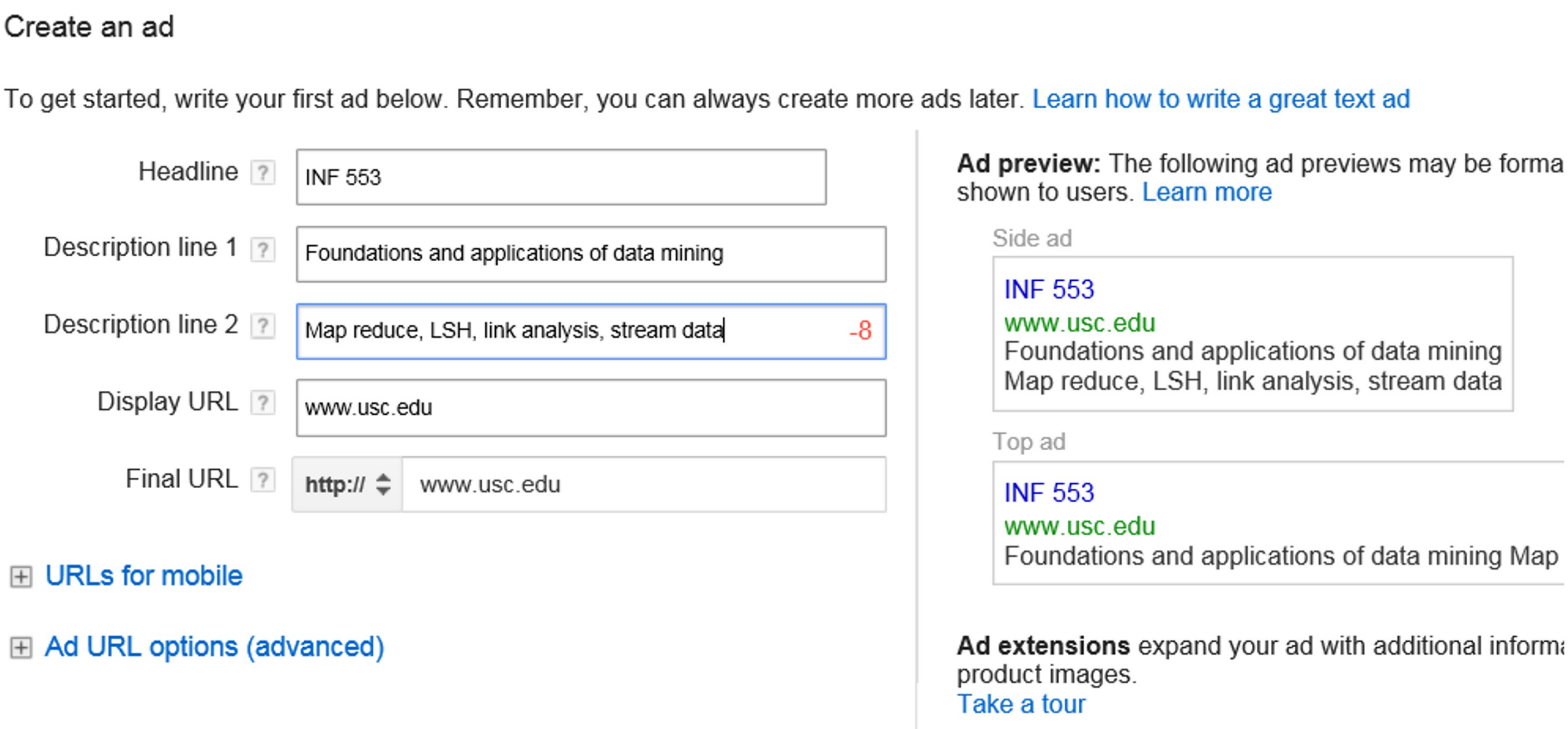
2. Search Ads Overview

- Most lucrative venue for online advertising: SEARCH
- Impression of an Ad
  - Ad is displayed
  - User clicked on the ad link to download the page
- Search engine charges advertisers for impression of their ads
- Adwords model (Google): matching search queries to advertisements
  - Require algorithms for optimizing this assignment
    - **Greedy** algorithms
    - **Online** algorithms

      can't predict



Google AdWords



Google AdWords

Select keywords

Your ad can show on Google when people search for the keywords you choose here. These keywords also determine which managed placements are good matches for your ads.

Tips

- Start with 10-20 keywords.
- Be specific: avoid one-word keywords. Choose phrases that customers would use to search for your products and services.
- By default, keywords are broad matched to searches to help you capture a wider range of relevant traffic. Use match types to control this.
- Learn more about choosing effective keywords.

Enter one keyword per line.

```
usc informatics
usc data mining
```

Category: University Of Southern California

« Add all from this category

« Add southern california university
« Add california southern university
« Add the university of southern california
« Add southern university of california
« Add universities in southern california
« Add university of southern california campus
« Add university southern california
« Add university in southern california
« Add where is university of southern california
« Add southern california universities
« Add where is the university of southern

3、

Matching Keywords with Searches

- Match types: exact, phrase, broad, negative

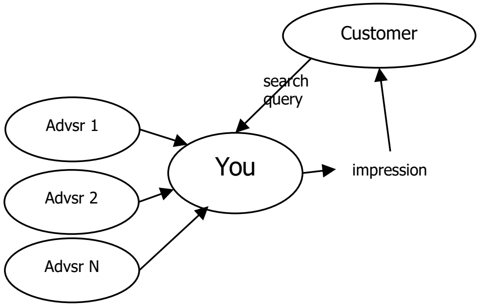
AdWords KeyWord Match Types

MATCH TYPE	SPECIAL SYMBOL	EXAMPLE KEYWORD	ADS MAY SHOW ON SEARCHES THAT	EXAMPLE SEARCHES
Broad Match	none	women's hats	includes misspellings, synonyms, related searches, and other relevant variations	*buy ladies hats*
Broad Match Modifier	+keyword	+women's +hats	contain the modified term (or closer variations, but not synonyms), in any order	*hats for women*
Phrase Match	"keyword"	"women's hats"	are a phrase, and close variations of that phrase	*buy women's hats*
Exact Match	[keyword]	[women's hats]	are an exact term and close variations of that exact term	*women's hats*

TryAdHawk.com

Source: Google

Online Algorithms



- **Classic model of algorithms**
  - Use the entire input to compute some result
    - "offline algorithm"
- **Online Algorithms**
  - You get to see the input one piece at a time, and need to make **irrevocable decisions along the way**
  - Make decisions without knowing the future
  - For search: only know past queries and current query; don't know what queries will come in later
  - Similar to handling data streams
- **An online algorithm cannot always do as well as an offline algorithm** → has query dataset and no new query

Example 8.1

- **Knowing the future could help**
- Manufacturer A of conventional furniture
  - bids 20 cents on both terms “sofa” and “chesterfield”
- Manufacturer B of antique furniture
  - bids 10 cents on search term “chesterfield”
- Both have monthly budget of $100
  - **B** can place its ad 1,000 times, **A** can place its ad 500 times
- Query “chesterfield” arrives
- Can only display one ad
- Might display A’s ad because A bid more, but…

Example 8.1

- **Knowing the future could help**
- Might display A's ad because A bid more
  - 20 cents vs 10 cents
- However, if there are many queries for "sofa" and few for "chesterfield," B will never spend its full budget
  - B only bids on "chesterfield"
- Sending "chesterfield" queries to B might increase the overall revenue for you
- **Without knowing the future, on-line algorithm may not perform as well as offline**

Offline Query-Ad Matching Problem

- Advertisers, each
  - Bids on keywords : "sofa": 10 cents/impression
  - Has a budget, e.g., $100/mon
- A set of queries in some month, say Sep 2015
  - e.g., 600 "chesterfield", 100 "sofa"
- Find assignments of queries to bids, such that
  - Total profit is maximized

Greedy Approach

- Consider two furniture manufacturers A and B
  - A: bids 20 cents on “chesterfield”; 10 cents on “sofa”
  - B: bids 10 cents on “chesterfield”
  - Both A and B have budget: $100/month

- Queries (expected): 600 “chesterfield”, 100 “sofa”
  - “chesterfield”: 500 to A => profit: $100 → budget of A used up
  - “chesterfield”: 100 to B => profit: $10

=> Total profit: $110

Optimal Solution (for offline)

- Consider two furniture manufacturers A and B
  - A: bids 20 cents on “chesterfield”; 10 cents on “sofa”
  - B: bids 10 cents on “chesterfield”
  - Both A and B have budget: $100/mon

- Queries (expected): 600 “chesterfield”, 100 “sofa”

- Optimal solution: assignment of queries to bids that generates the largest profit

- Queries (expected): 600 “chesterfield”, 100 “sofa”
  - “sofa”: 100 to A => profit: $10
  - “chesterfield”: 450 to A => profit: $90
  - “chesterfield”: 150 to B => profit: $15

  => Total profit: $115

Comparison

Bids	Chesterfield	Sofa	Budget
A	20 cents	10 cents	$100
B	10 cents		$100

Queries	Chesterfield (600)	Sofa (100)	Profit
A	500		$100
B	100		$10

Greedy, Total profit:
$110

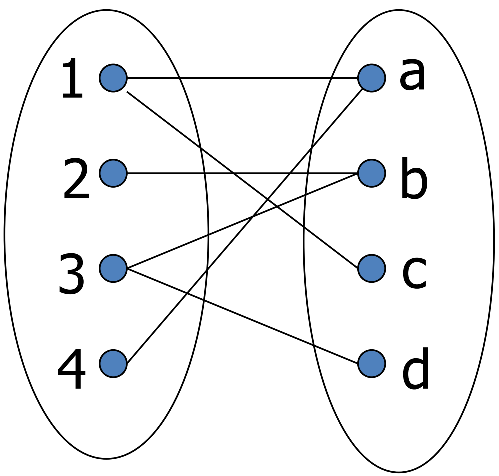
Queries	Chesterfield (600)	Sofa (100)	Profit
A	450	100	$100
B	150		$15

Non-Greedy (Optimal), Total profit: $115

Online Bipartite Matching

The Matching Problem

- Simplified version of the problem of matching ads to search queries
- Looking for "Maximal matching" in a bipartite graph
  - involves bipartite graphs with two sets of nodes
- All edges connect node on left set to node in right set

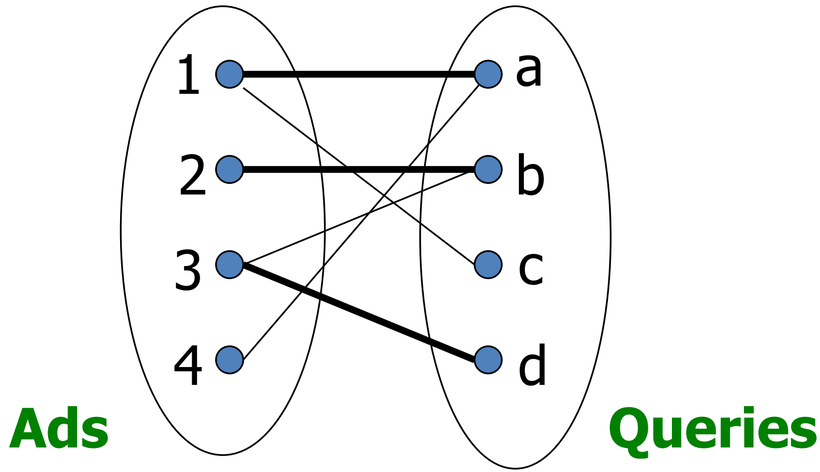


Ads **Queries**

Nodes: Queries and Ads

Goal: Match queries to ads so that maximum number of matchings are made

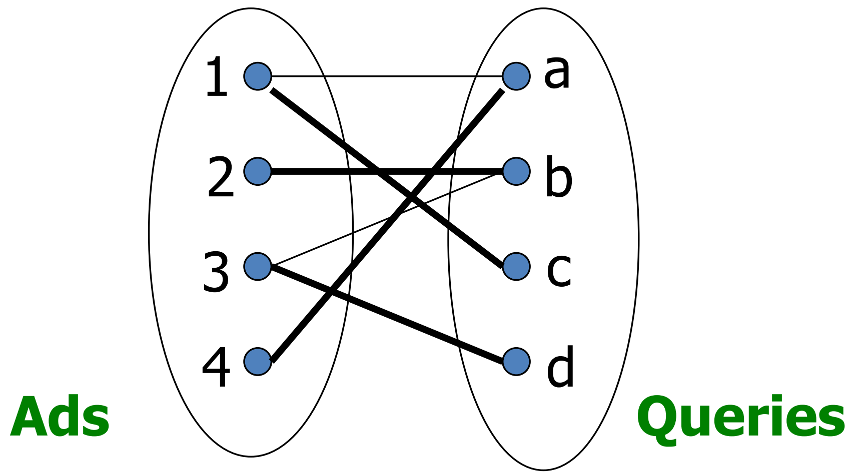
Example: Bipartite Matching



M = {(1,a),(2,b),(3,d)} is a **matching**

Cardinality **of matching = |M| = 3**

Example: Bipartite Matching



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

Maximal matching: a matching that contains the largest possible number of matches

Perfect matching: all vertices of the graph are matched

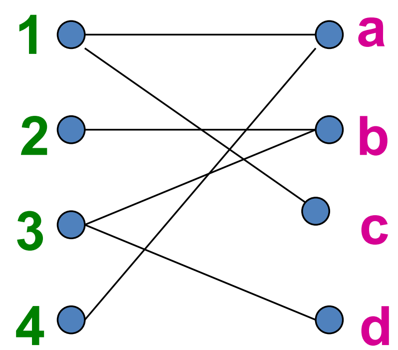
2、

Matching Algorithm

- **Problem: Find a maximal 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 Graph Matching Problem

- Initially, we are given the set **ads**
- In each **round**, **one set of query terms is added**
  - Relevant **edges** are revealed
  - Indicate which advertisers have bid on those query terms
- **At that time, we have to decide to either:**
  - Pair the **query** with an **ad**
  - Do not pair the **query** with any **ad**



(1,a)
(2,b)
(3,d)

3.

Greedy Algorithm

- **Greedy algorithm for the online graph matching problem:** pick the one pays more.
  - Pair the new query with **any** eligible ad
    - If there is none, do not pair query

- **How good is the algorithm?**

4, evaluation

Competitive Ratio

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

Competitive ratio =

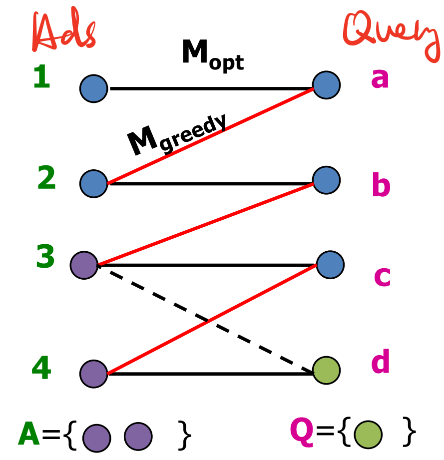
$$\min_{\text{all possible inputs } I} (|M_{greedy}| / |M_{opt}|)$$

greedy's worst performance over all possible inputs *I*

Analyzing the Greedy Algorithm

- Consider a case: $M_{greedy} \neq M_{opt}$
- Consider the set Q of queries ("wasted") matched in M_{opt} but not in M_{greedy}
- **A is the set of ads** that are linked ⇒ adjacent to a non-matched query in Q, and A ("blocking") already matched in M_{greedy}
  - If there exists such a non-matched (by M_{greedy}) ad linked to a non-matched query, then greedy would have matched them
- Since ads A are already matched in M_{greedy} then

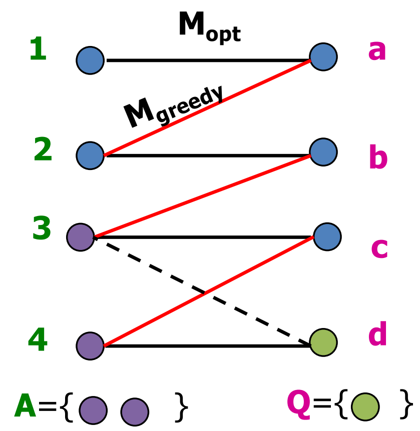
(1) $|M_{greedy}| \geq |A|$



ads that are adjacent to a query in Q but are already matched to another query

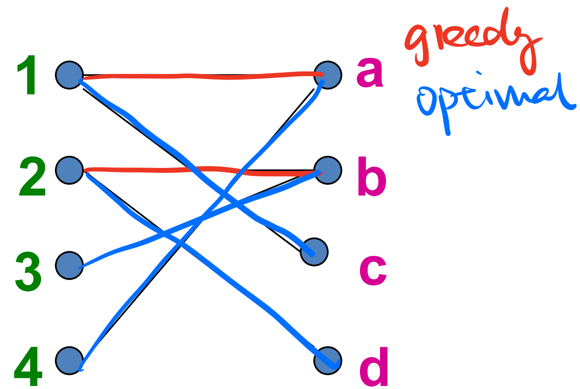
Analyzing the Greedy Algorithm

- **Summary so far:**
  - Queries Q matched in M_{opt} but not in M_{greedy}
  - (1) $|M_{greedy}| \geq |A|$
- There are at least $|Q|$ such ads in A ($|Q| \leq |A|$) otherwise the optimal algorithm couldn't have matched all queries in Q
  - **So:** $|Q| \leq |A| \leq |M_{greedy}|$
- Q': matched in M_{opt} and also in M_{greedy}
  - $|M_{opt}| = |Q| + |Q'|$ and $|Q'| \leq |M_{greedy}|$
  - $|M_{opt}| \leq |Q| + |M_{greedy}|$
  - Worst case is when $|Q|$ is maximum, $\Rightarrow |Q| = |A| = |M_{greedy}|$
- $|M_{opt}| \leq 2|M_{greedy}|$ **then** $|M_{greedy}|/|M_{opt}| \geq ½$
- **Competitive Ratio = ½**
- **This is Greedy's worst performance over all possible inputs *I***



(exam)

Worst-case Scenario



(1,a)
(2,b)

- **Worst case** is when $|\boldsymbol{Q}| = |\mathbf{A}| = |\boldsymbol{M}_{greedy}|$
- Q = {c,d} – queries with no matching ad
- A = {1,2} – ads that are adjacent to a query in Q but are already matched to another query
- $|\boldsymbol{M}_{greedy}| = 2$, $|\boldsymbol{Q}| = 2$, $|\mathbf{A}| = 2$
- **Optimal matching: (1,c), (2,d), (3,b), (4,a)**
- $|\boldsymbol{M}_{opt}| = 4$
- $\boldsymbol{|M_{greedy}|/|M_{opt}| = ½}$ **(competitive ratio)**

“Performance-Based” Web Advertising

“Showing is not enough, must be clicked”

History of 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 Return on Investment (ROI) for advertisers

CPM…cost per *mille*

Mille…thousand in Latin

"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 first**
  - Advertiser is charged only if the ad is *clicked* on

- Similar model adopted by Google with some changes around 2002
  - Called **Adwords**

Web 2.0

- **Performance-based advertising works!**
  - Multi-billion-dollar industry
- **Interesting problem:**
  What ads to show for a given query?
  - (Today's lecture)
- **If I am an advertiser, which search terms should I bid on and how much should I bid?**
  - (Not focus of today's lecture)

2. Adwords Problem

- **Given:**
  - **1.** A set of bids by advertisers for search queries
  - **2.** A click-through rate (CTR) 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

Adwords Problem

- 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**
  - **Simple solution:** Instead of raw bids, use the "**expected revenue per click**" (i.e., **Bid*CTR**)
- **Clearly we need an online algorithm!**

The Adwords Innovation

Advertiser	Bid	CTR	Bid * CTR
A	$1.00	1%	1 cent
B	$0.75	2%	1.5 cents
C	$0.50	2.5%	1.125 cents

Click through rate

Expected revenue

(2). # Complications: Budget

- Two complications:
  - Budget
  - Click-through rate (CTR) of an ad is unknown

- Each advertiser has a ① limited budget
  - Search engine guarantees that the advertiser will not be charged more than their daily or monthly budget

Complications: CTR

2

- 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
  - **Click-through 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

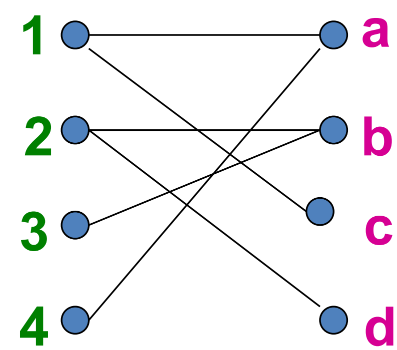
3、

Greedy Algorithm

(1)

- **Our setting: Simplified environment**
  - There is **1** ad shown for each query
  - All advertisers have the same budget ***B***
  - All ads are equally likely to be clicked
  - Value of each ad is the same (=**1**)

- **Simplest algorithm is greedy:**
  - For a query pick any advertiser who has bid **1** for that query
  - **Competitive ratio of greedy is 1/2**



(2)

Bad Scenario for Greedy

- **Two advertisers A and B**
  - ***A*** bids on query ***x***, and ***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 B B B B**
  - **Competitive ratio = ½**
- **This is the worst case!**
  - **Note:** Greedy algorithm is deterministic – it always resolves draws in the same way

Greedy algorithm with non-equal bids

- Greedy algorithm would **assign the query to the highest bidder who still has budget left**

Greedy Example: Two advertisers bid on a query q

- **Bidder A_1:** **bid $x_1 = 20$** **budget $b_1 = 40$**
- **Bidder A_2:** **bid $x_2 = 10$** **budget $b_2 = 50$**
- Assume ties are broken in favor of A_1 → if tie, pick A_1

Query q	Assigned to Bidder (A_1, A_2 or No Ad)	Remaining Budget for A_1	Remaining Budget for A_2
At start	----	40	50
1st query q	A1	20	50
2nd query q	A1	0	50
3rd query q	A2	0	40
4th query q	A2	0	30
5th query q	A2	0	20
6th query q	A2	0	10
7th query q	A2	0	0
8th query q	No ad	0	0

BALANCE Algorithm [MSVV]

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

BALANCE: Prefer the bidder who has the deepest pocket, not who pays the most ☺

Example: BALANCE

- **Two advertisers A and B**
  - **A** bids on query ***x***, and **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 B _ _**
  - Optimal: **A A A A B B B B**

- **In general:** For **BALANCE** on **2** advertisers
  Competitive ratio = ¾

BALANCE Example: Two advertisers bid on a query q

- **Bidder A_1:** **bid $x_1 = 20$** **budget $b_1 = 40$**
- **Bidder A_2:** **bid $x_2 = 10$** **budget $b_2 = 50$**
- Assume ties are broken in favor of A_1

Query q	Assigned to Bidder (A_1, A_2 or No Ad)	Remaining Budget for A_1	Remaining Budget for A_2
At start	----	40	50
1st query q	A2	40	40
2nd query q tie	A1	20	40
3rd query q	A2	20	30
4th query q	A2	20	20
5th query q tie	A1	0	20
6th query q	A2	0	10
7th query q	A2	0	0
8th query q	No Ad	0	0

(2)

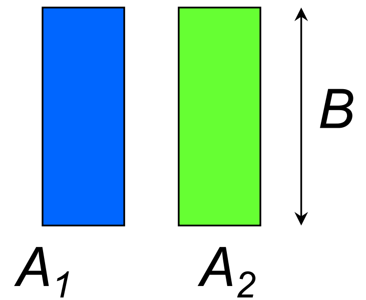
Analyzing BALANCE

- **Consider simple case (w.l.o.g.):**
  - **2** advertisers, $\mathbf{A_1}$ and $\mathbf{A_2}$, each with budget **B** (≥1)
  - Optimal solution exhausts both advertisers' budgets

- **BALANCE must exhaust at least one advertiser's budget:**
  - **Because optimal exhausts both**
  - **If not, we can allocate more queries**
    - Whenever both advertisers bid on the query, chosen advertiser's unspent budget only decreases

- Assume BALANCE exhausts A_2's budget, but allocates x queries fewer than the optimal
- **Revenue:** $BAL = 2B - x$

Analyzing Balance



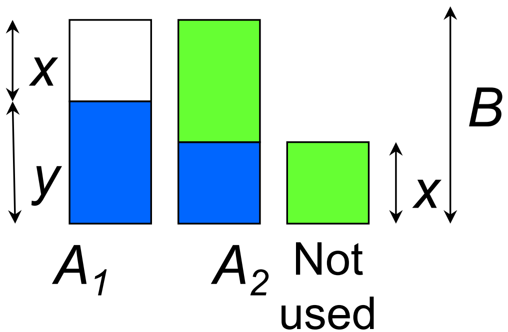
Queries allocated to A_1 in the optimal solution

Queries allocated to A_2 in the optimal solution

Optimal revenue = **2B**

Balance Algorithm:

Assume Balance gives revenue **= 2B-x or B+y**



Unassigned queries can only be assigned to A_2

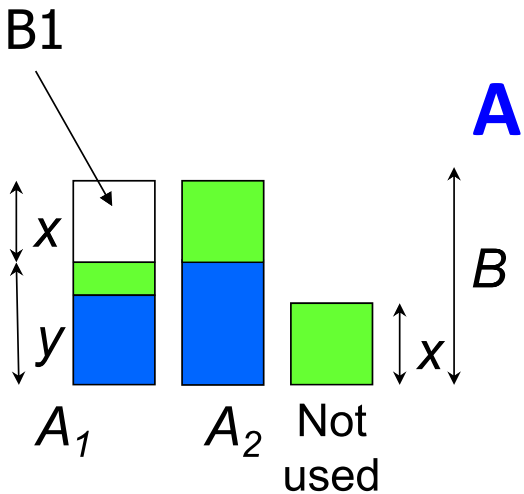
(if we could assign to A_1 we would, since A1 still has budget)

Goal: Show we have **$y \geq x$**

Case 1) ≤ ½ of A_1's queries got assigned to A_2

then y >= B/2, so y >= x (because y+x = B)

BALANCE exhausts A_2's budget



Analyzing Balance

Queries allocated to A_1 in the optimal solution

Queries allocated to A_2 in the optimal solution

Optimal revenue = **2B**

Balance Algorithm:

Assume Balance gives revenue **= 2B-x or B+y**

Unassigned queries can only be assigned to A_2

Goal: Show we have $\mathbf{y \geq x}$

Case 2) > ½ of $\mathbf{A_1}$'s queries got assigned to $\mathbf{A_2}$, consider the last of $\mathbf{A_1}$'s queries assigned to $\mathbf{A_2}$:

1) B2 >= B1 since Balance chose $\mathbf{A_2}$
2) B2 <= B/2 (since at least ½ of $\mathbf{A_1}$'s queries got assigned to $\mathbf{A_2}$)
3) Thus, B1<=B2 <= B/2, so x (or B1) < B/2, and x + y = B -> y >= x

Balance revenue is minimum for x=y=B/2 (i.e., Max x = B/2)

Minimum Balance revenue = 3B/2

$x \leq \frac{1}{2}B \Rightarrow$ worst case : $x = \frac{1}{2}B$

Competitive Ratio = 3/4 // [3B/2] / 2B = 3/4

(3).

BALANCE: General Result

- **For Balance algorithm with many bidders**
- **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!

- **Let's see the worst case example that gives this ratio**

lose opportunity to charge people with less budget

Kalyanasundaram, B., & Pruhs, K. R. (2000). An optimal deterministic algorithm for online b-matching. *Theoretical Computer Science*, *233*(1-2), 319-325.

(4).

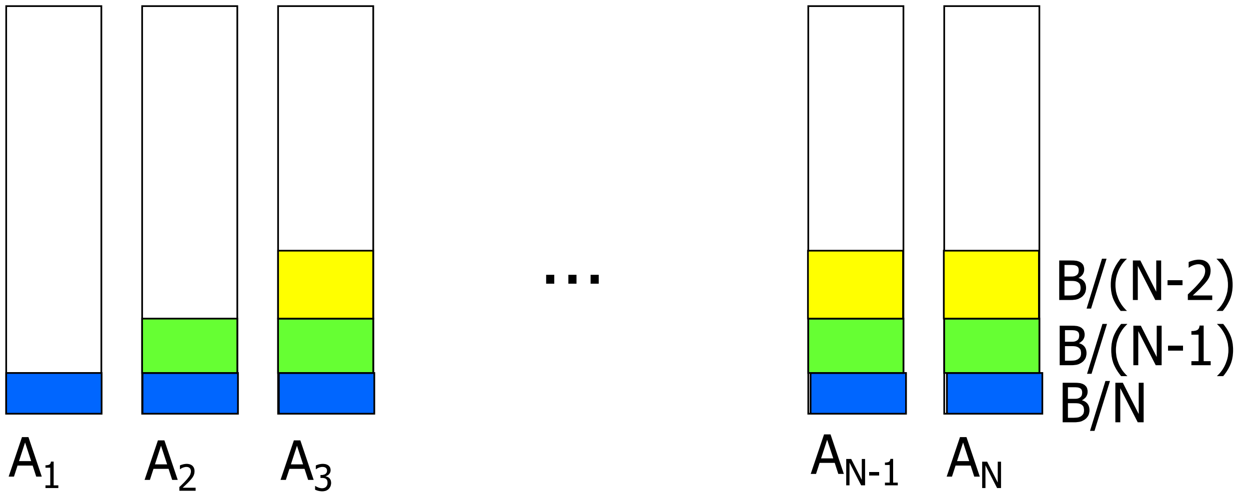
Worst case for BALANCE

- ***N* advertisers: $A_1, A_2, \ldots A_i, \ldots A_N$**
  - Each has a budget ***B = i*** // budgets are 1,2,3,..., N
- **Queries:**
  - ***N·B*** queries appear in ***N*** rounds of ***B*** queries each
- **Bidding (requirement or qualification):**
  - Round **1** queries can be bid by **A_1, A_2, ..., ..., A_N**
  - Round **2** queries can be bid by **A_2, A_3, ..., ..., A_N** → opportunity
  - Round ***i*** queries can be bid by **A_i, ..., A_N**
- **Optimum allocation:** // from front to back

  Allocate round ***i*** queries to ***A_i*** // even though there are other bidders
  - Optimum revenue ***N·B*** = 1+2+3+...+N
- **BALANCE:** // from back to front
  - Assigns query in round 1 to N advertisers equally, since all can bid on q1
    - But prefer/select the bidder with the largest remaining budget, e.g., A_N
  - For q2, only $A_2, A_3, \ldots, A_N$ can bid, so still prefer the larger bidders;
  - For each query qi, only $A_i, \ldots, A_N$ still prefer the back bidders

Round 1 has 1 queries
Round 2 has 2 queries
Round 3 has 3 queries
...
Round *i* has *i* queries
...
Round N has N queries

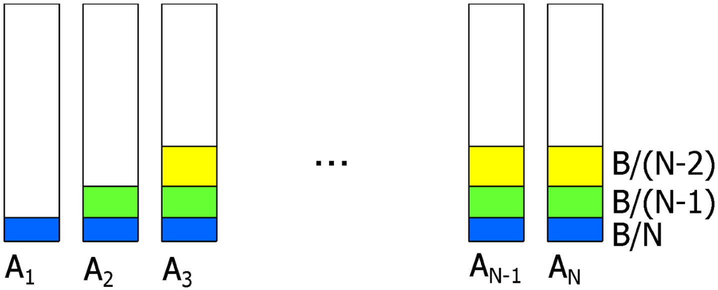
BALANCE Allocation



- **Eventually, budgets of higher-numbered advertisers exhausted**
- j is approximate value where all advertisers are out of budget or cannot bid on the remaining queries

Each round has revenue B so the approx. total revenue is B x j = BN(1-1/e)
1+1/2+…+1/n
1+1/2+…+1/(n-j)
…
1/(n-j+1)+…+1/n

BALANCE Allocation



- **Eventually, budgets of higher-numbered advertisers exhausted**
- j is approximate value where all advertisers are out of budget or do not bid on remaining queries

Last round, the last bidder used up all its budget, B:

$$B\left(\frac{1}{N}+\frac{1}{N-1}+\cdots+\frac{1}{N-j+1}\right)\geq B$$

Euler showed that as k gets large, $\sum_{i=1}^{k} 1/i$ approaches $\log_e k$.

$$\log_e N - \log_e(N-j) = 1$$

(1+1/2+…+1/n) - (1+1/2+…+1/(n-j)) = 1/(n-j+1)+…+1/n

- **So we want j such that ln(N) - ln(N-j) = 1 (approximately)**
- j = N(1-1/e) -> Each round has revenue B so the approx. total revenue is B x j
- **Approximate revenue of Balance Algorithm is BN(1-1/e)**
- **Competitive ratio is 1-1/e**

Kalyanasundaram, B., & Pruhs, K. R. (2000). An optimal deterministic algorithm for online b-matching. *Theoretical Computer Science*, *233*(1-2), 319-325.

5.

General Version of the Problem

- **Balance works well when bids are 1,0**
- **In practice, bids and budgets can be arbitrary**
- **In a general setting, BALANCE can perform poorly**
- **Example 8.9:** Consider two advertisers A_1 and A_2
  - A_1: bid_1 = **1**, $budget_1$ = **110**
  - A_2: bid_2 = **10**, $budget_2$ = **100**
  - Consider: we see **10** instances of **q**
  - BALANCE always selects A_1 because it has largest remaining budget
  - Earns total revenue = **10**
  - **Favors advertiser with larger remaining budget**
  - Optimal earns **100**

Modifications Needed to BALANCE Algorithm

- **Bias choice of ad in favor of higher bids**
- **Consider the fraction of budget remaining,** so we **bias toward using some of each advertiser's budget**
- More "risk averse": don't leave too much of any advertiser's budget unused

b-

Generalized BALANCE Algorithm

- **Arbitrary bids:** consider query q, bidder i
  - Bid = x_i
  - Budget = b_i
  - Amount spent so far = m_i
  - **Fraction of budget left over $f_i = 1 - (m_i/b_i)$**

$m_i\uparrow, f_i\downarrow, e^{-f_i}\uparrow, \psi(q)\downarrow$

- Define $\psi_i(q) = x_i * (1-e^{-f_i})$

ψ (psi)

  - ***bid * (1-e^{-(fraction of budget left)})***

- Allocate query q to bidder i with largest value of $\psi_i(q)$
- **Same competitive ratio (1-1/e)**

Example 8.10

- **Bidder A_1: $x_1 = 1$, $b_1 = 110$**
- **Bidder A_2: $x_2 = 10$, $b_2 = 100$**
- First occurrence of query q: **fraction 1 of budgets b_1 and b_2 remain**
- $\psi_1(q) = x_1(1-e^{-f1}) = 1(1-e^{-1}) = 1 – 1/e = 0.63$
- $\psi_2(q) = x_2(1-e^{-f2}) = 10(1-e^{-1}) = 6.3$ √
- **So first q is awarded to A_2**
- **$\psi_2(q)$ decreases, but for the next 9 instances of q: $\psi_2(q) > \psi_1(q)$ and queries are awarded to A_2**
- For 10th instance of q, **remaining fraction of budget b_2 is 1/10**
- $\psi_2(q) = x_2(1-e^{-f2}) = 10(1-e^{-1/10}) = 0.95$, *which is > 0.63*
- **After 10 queries q, have spent all of A_2's budget, and additional queries q will be awarded to A_1**
- **Total revenue for 10 queries q = 100**
- **Generalized Balance Algorithm: Successfully biased toward higher bids, took into account fraction of budget remaining**

7

Additional Observations

- Algorithm as described **does not account for possibility that click-through rate differs for different ads**
- **Multiply bid by CTR when computing ψ**

- Also can **consider historical frequency of queries**
  - **Use historical frequency to predict future frequency**

8.

Adwords Aspects Not in Our Model

Matching bids and search queries:

1. **In our simplified model, advertisers bid on sets of words**
2. An advertiser's bid is eligible to be shown for search queries with **exactly the same set of words as advertiser's bid**

- In reality, **Google, Yahoo, Microsoft all offer advertisers "broad matching": inexact matches of the bid keywords**
- **Examples: subsets, supersets, words with very similar meanings**

- Charge advertisers based on **complicated formulas that take into account how closely related the search query is to the advertiser's bids**
- **Proprietary algorithms**

Adwords Aspects Not in Our Model

Charging Advertisers for Clicks

- **In our simplified model, when a user clicks on an ad, the advertiser is charged the amount they bid**
- Known as a **first-price auction**
- **In reality, search engines use a more complicated system known as a second-price auction** increase willingness to bid high
- **Each advertiser pays approximately the bid of the advertiser who placed immediately behind them in the auction**
  - Example: **First-place advertiser would pay the bid of the second-place advertiser plus one cent**
- **Less susceptible to being gamed by advertisers than first-price auctions**
- **Lead to higher revenues for search engines**
  - https://blogs.cornell.edu/info2040/2012/10/27/google-adwords-auction-a-second-price-sealed-bid-auction/