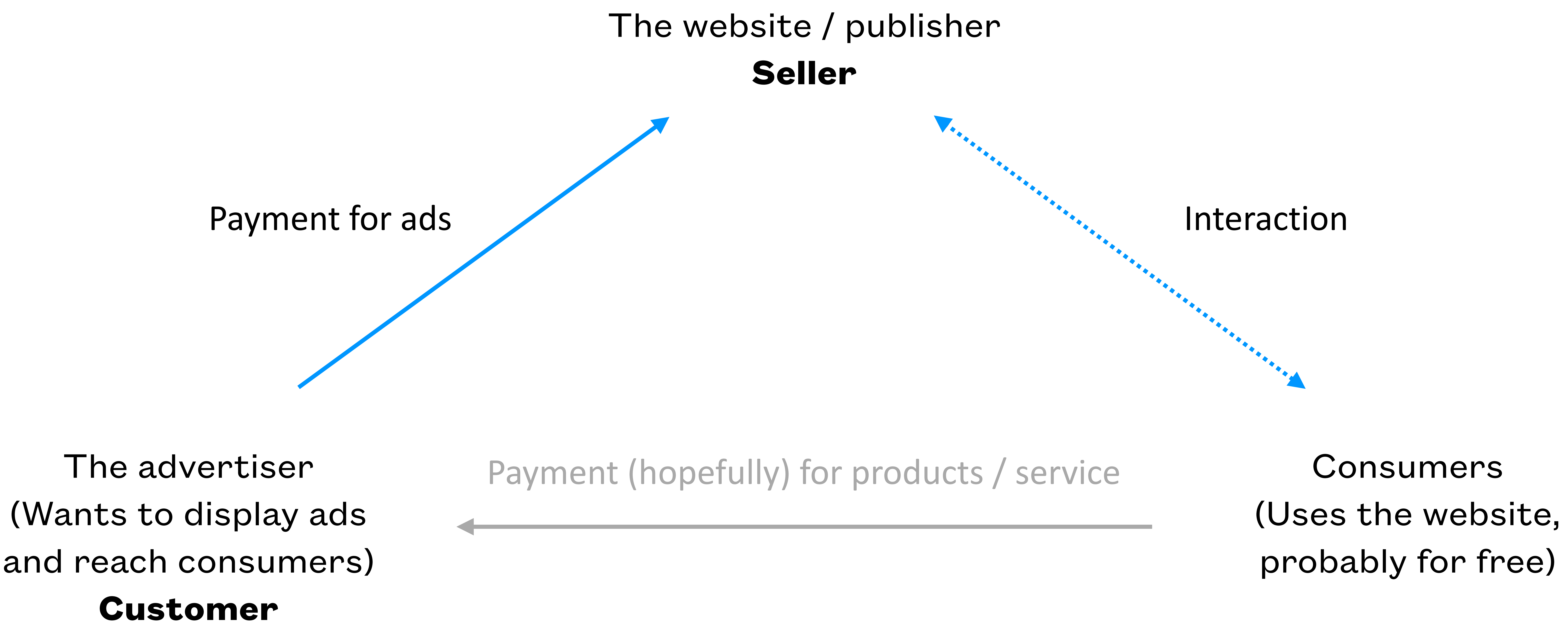
Advertising on the Web

Debapriyo Majumdar
Indian Statistical Institute
debapriyo@isical.ac.in

Online advertising

- Display ads
  - The primary method before search ads became popular
  - The host website charges *fees* for every 1000 “impressions” of ad
    - Called “CPM” (cost per mile — calculated as cost per 1000) rate
    - The advertiser pays the publisher when a user visits the website (and the ad is shown to the user)
    - Based on the model used in TV, magazine ads
  - Many types (how they are displayed)
    - Banner, Pop-up, Trick banner, Overlay, etc (we must have seen these all)
  - Untargeted, or demographically targeted
  - Low clickthrough rates
    - Low ROI for advertisers

The advertisement scenario



From cost-per-impression to cost-per-click

- Pay anyway, even if the user does not click
- Similar to TV (modernly streaming media), but in video viewers stay more glued to the screen

- Better option for the advertiser: pay only if the consumer clicked (showed initial interest) the ad (pay per click)
- The consumer's interest is the product
- (Pay only if you get the product)

For a history, read (recommended): https://www.launchpresso.com/what-happened-to-overture-com/

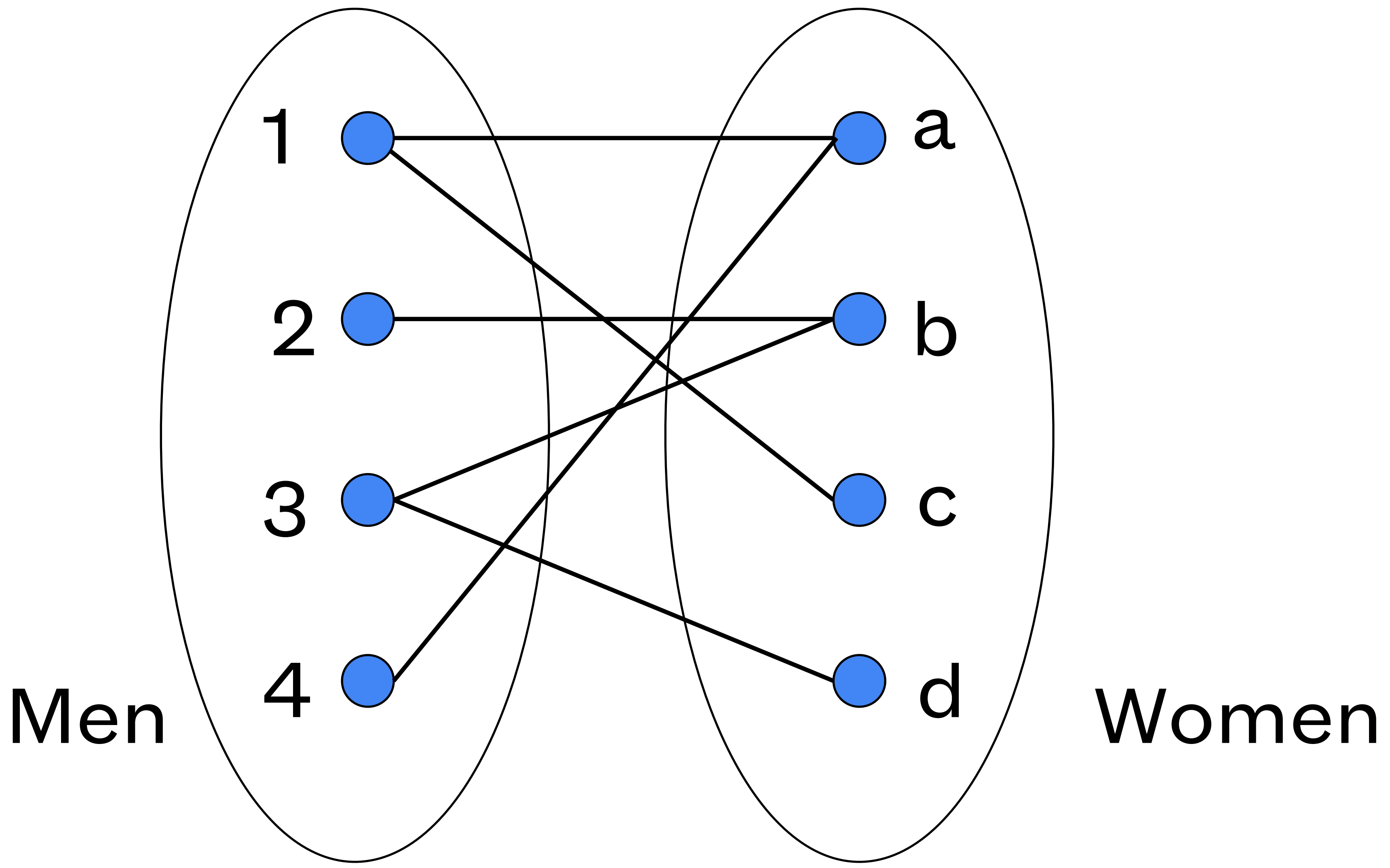
Search advertising

- Pay-per-click is seemingly good for the advertiser
- But guaranteed display of ads is no more profitable for the provider
  - What if some ads are almost never clicked on?
  - Website real estate space blocked, but no revenue
- **Search ads**: show ads based on what the user is interested in
  - Ads are relevant to the users $\implies$ much higher chance of clicking (ad revenue)
  - Good for advertisers too (their ads are targeted)
- The **Adwords** model
  - The advertisers bid on search queries: if my ad is shown when the user searched with keyword q, and the user clicks on my ad, I will pay $\$b$
  - For every search by some user (with query q), the search engine displays some of the ads that have a bid on the query

Online algorithms for search advertising

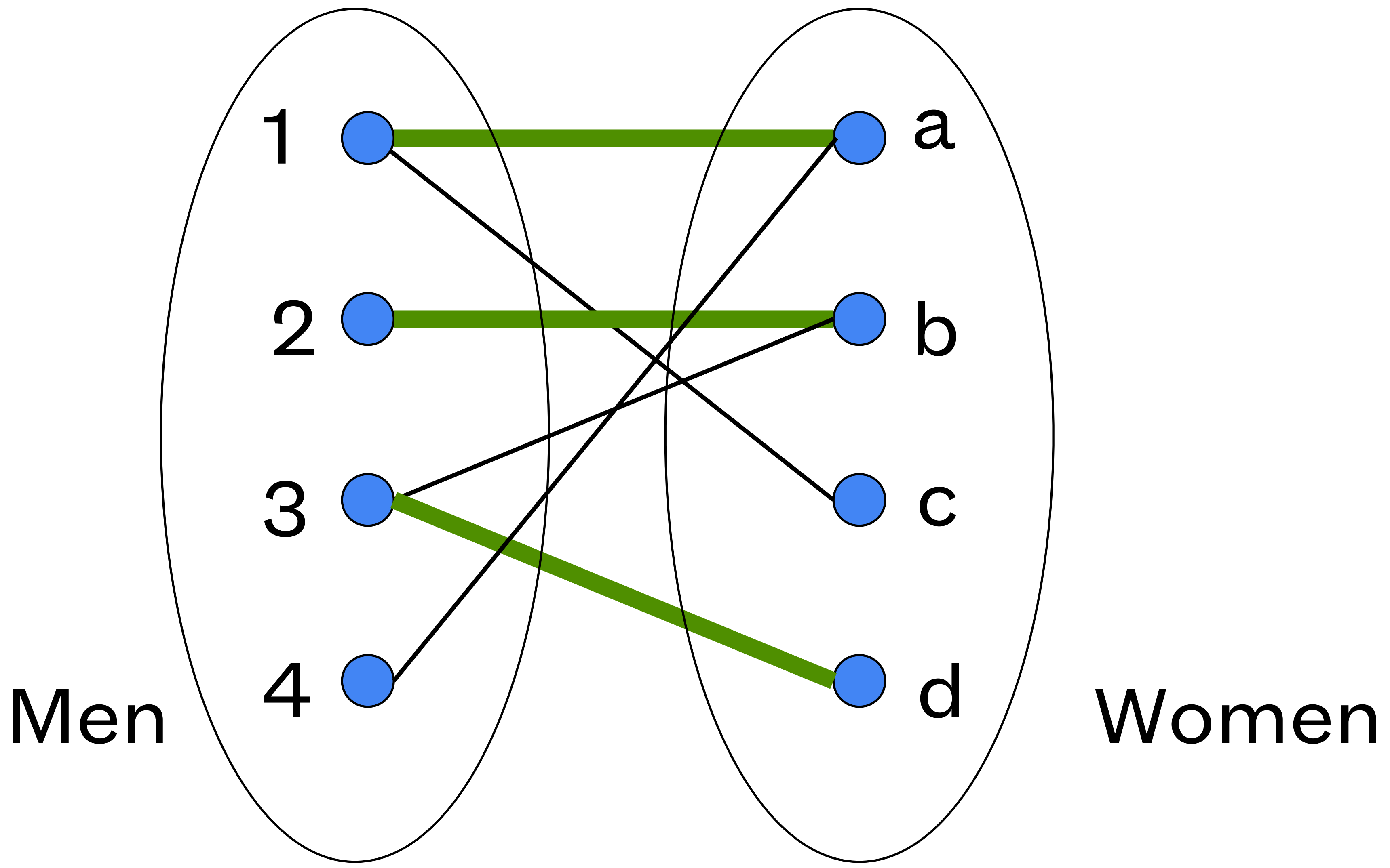
- Search advertising is a multi-billion-dollar industry
- Main technical problems:
  - Search engine: which ads to show for a search q?
  - Advertiser: which search terms should I bid on, and how much?
- Classic model of (offline) algorithms:
  - You get to see the entire input, then compute some function 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

Background: bipartile matching



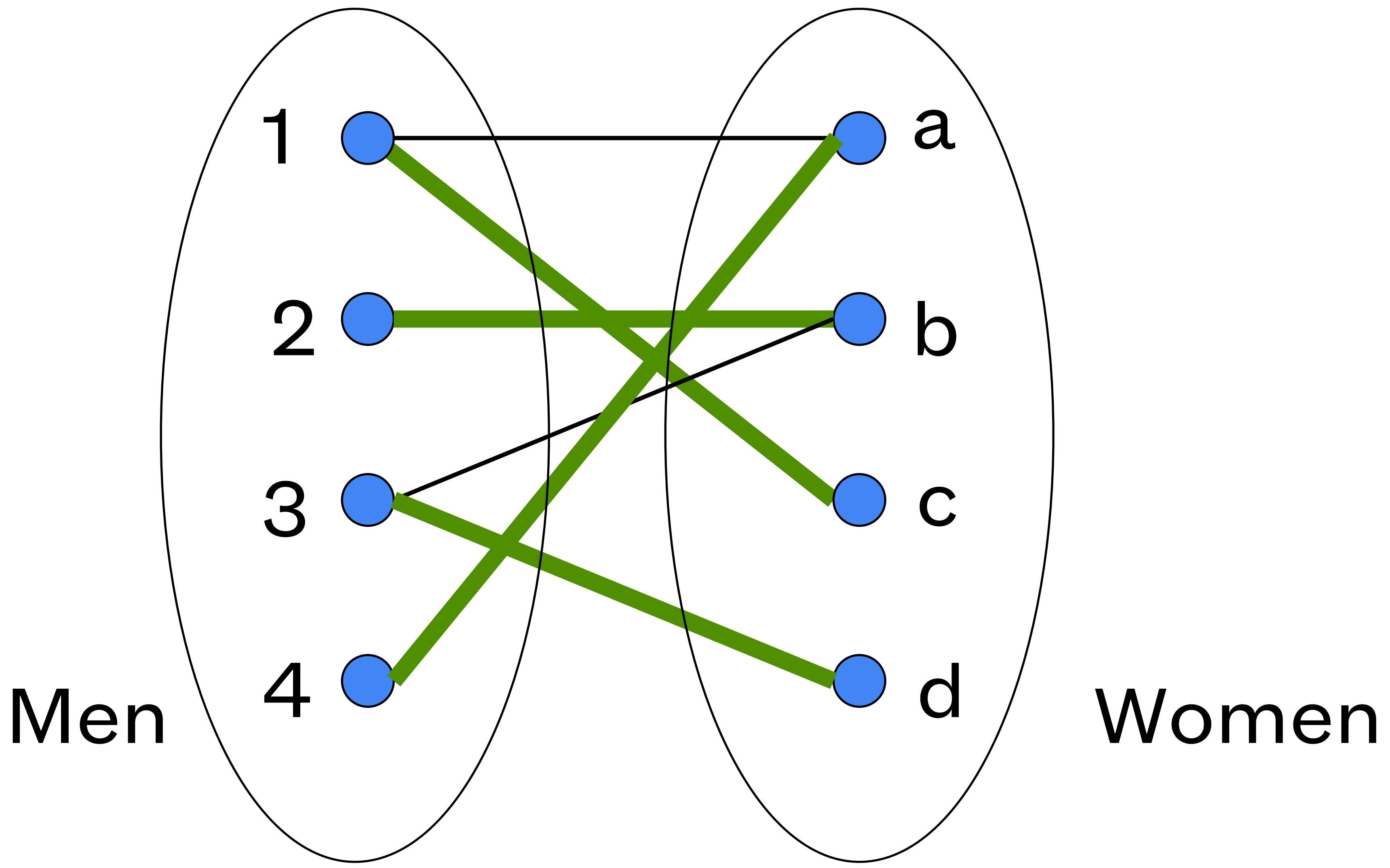
- A bipartile graph with two sets of nodes V_1 and V_2 with edges only across the sets
- A matching is a subset M of the edges such that no two edges in M have a common node
- In other words, a 1-1 pairing of men and women in the picture
- **Goal**: maximize the number of nodes paired this way

Maximal matching



- Example: $M = \{(1,a), (2,b), (3,d)\}$ is a matching
- Observe: no more edges can be added to M
  - So, M is a *maximal* matching
- Cardinality of M is 3
- Can there be another matching with a higher cardinality (in other words, is M a *maximum cardinality* matching?)

Maximum cardinality matching

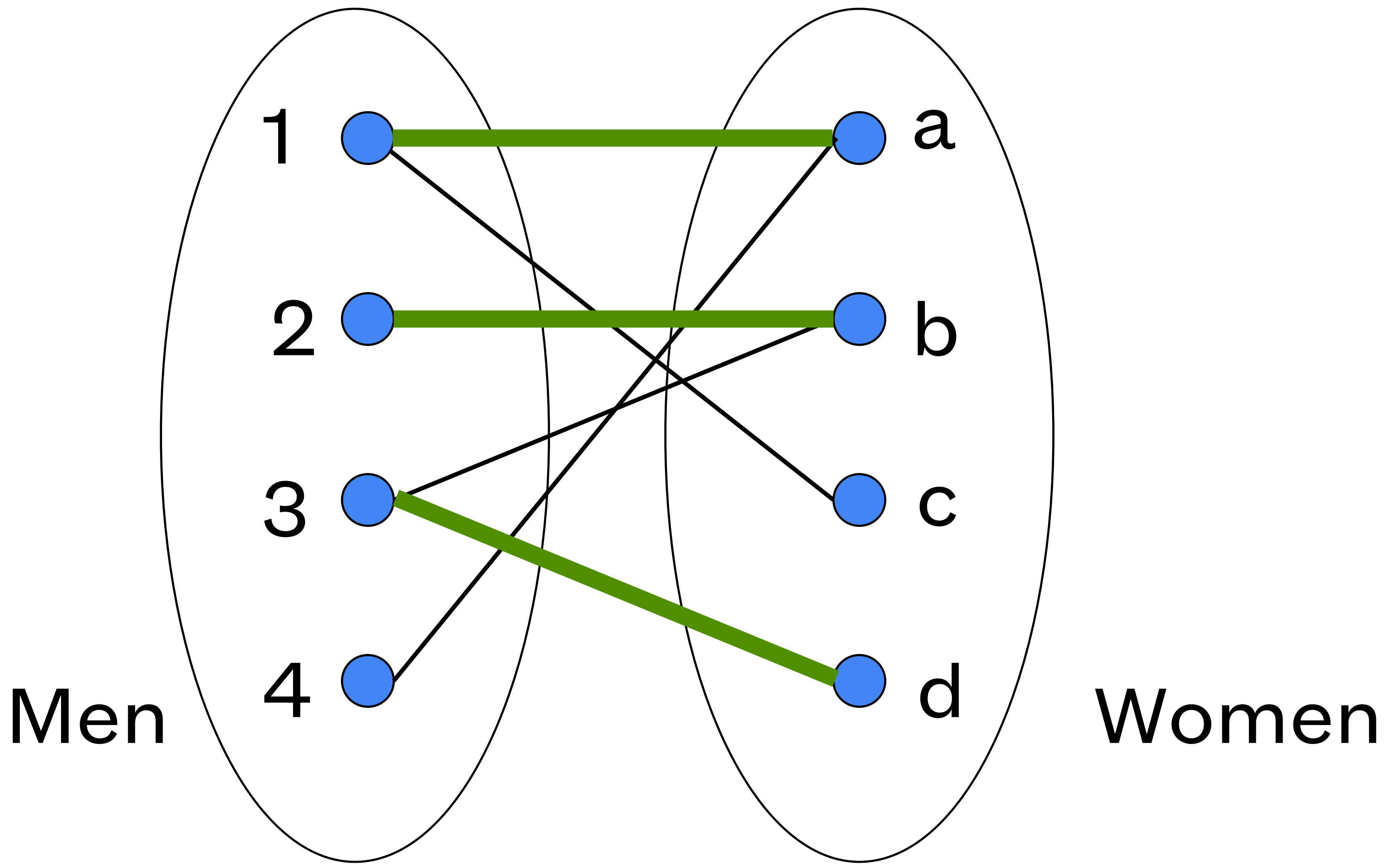


- Example 2: $M_2 = \{(1,c), (2,b), (3,d), (4,a)\}$ is a matching of cardinality 4
- M_2 is maximal as well as of maximal cardinality
- In fact M_2 is a perfect matching (all nodes are part of some edge of the matching)
- The problem we want to consider: given a bipartile graph, find a maximal cardinality matching (a perfect one if it exists)
- Offline: polynomial time algorithm exists
- Online: we don't have the entire graph initially

Online Matching

- Initially, we know 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 among her choices
  - Don't pair the woman with any man
- Example applications:
  - Assigning tasks to servers
  - Web requests to threads
  - Assigning ads to search queries

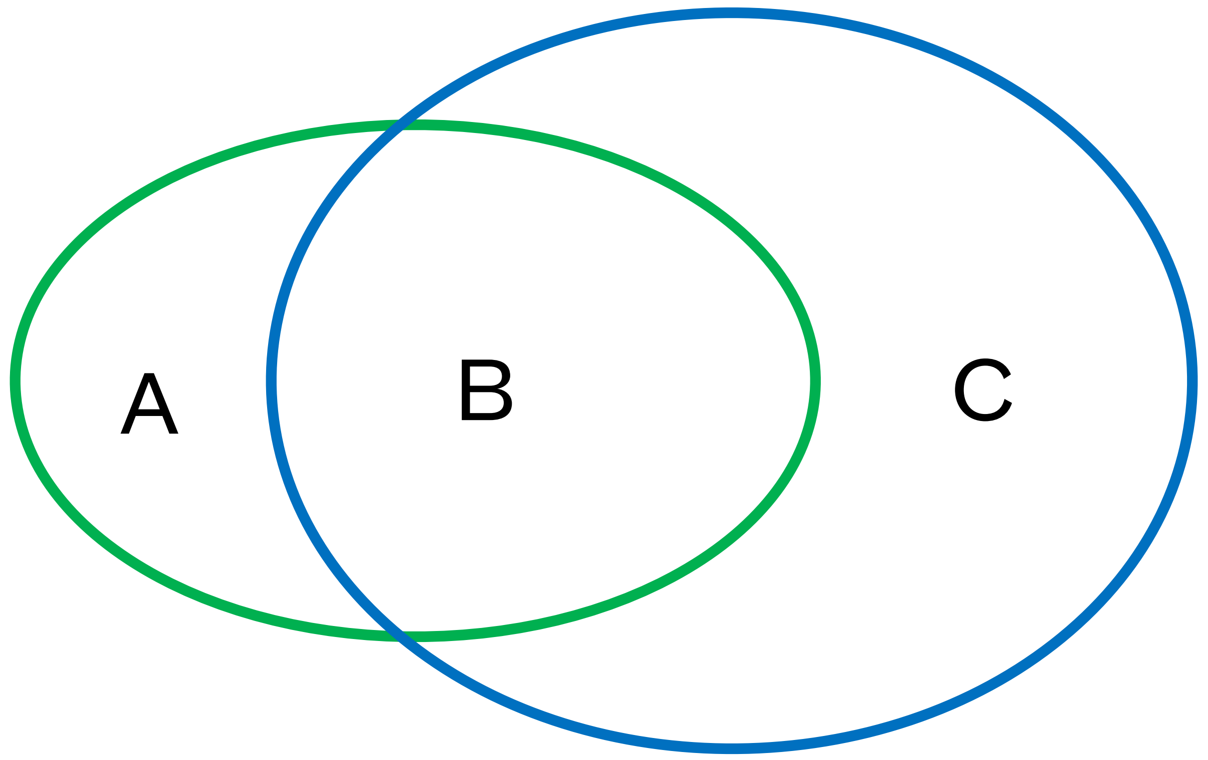
The greedy algorithm



- The set of men are known
- As a new woman arrives with her choices, pair the woman with some man of her choice
  - If any such man is still available
- Simple to execute
- But how good (or bad) is this algorithm?

- Measured by the the competitive ratio of a matching algorithm
- If M_{greedy} is a matching by the greedy algorithm and M_{opt} is an optimal matching, then competitive ratio = $\min_G \{ |M_{\text{greedy}}| / |M_{\text{opt}}| \}$ taken over all possible inputs G

The greedy algorithm has competitive ratio 1/2



- Let O be the optimal matching, and G the matches produced by a run of the greedy algorithm
- We need to show $2|G| \geq |O|$
- Consider the sets of women:

  A: Matched in G, not in O

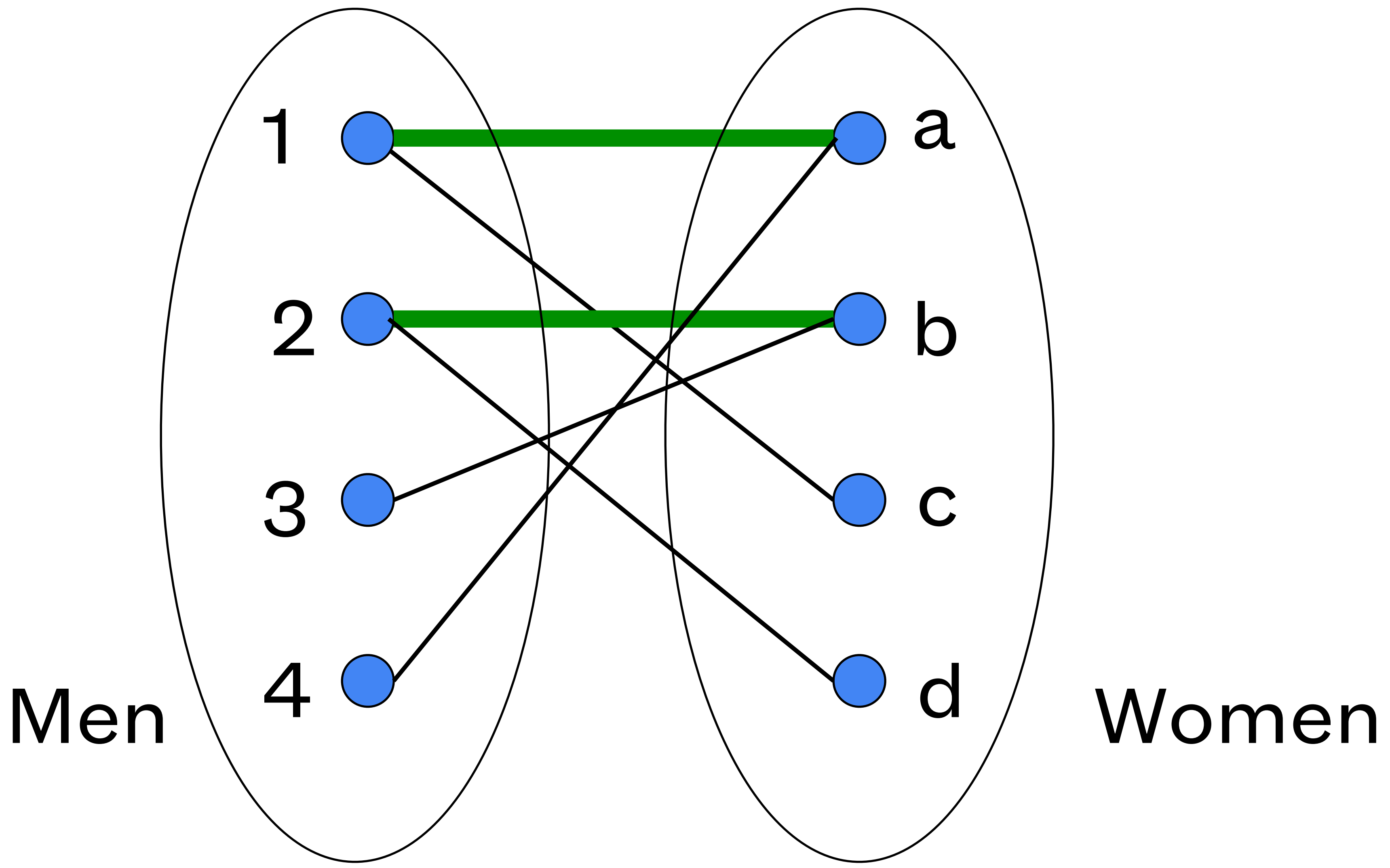
  B: Matched in both

  C: Matched in O, not in G

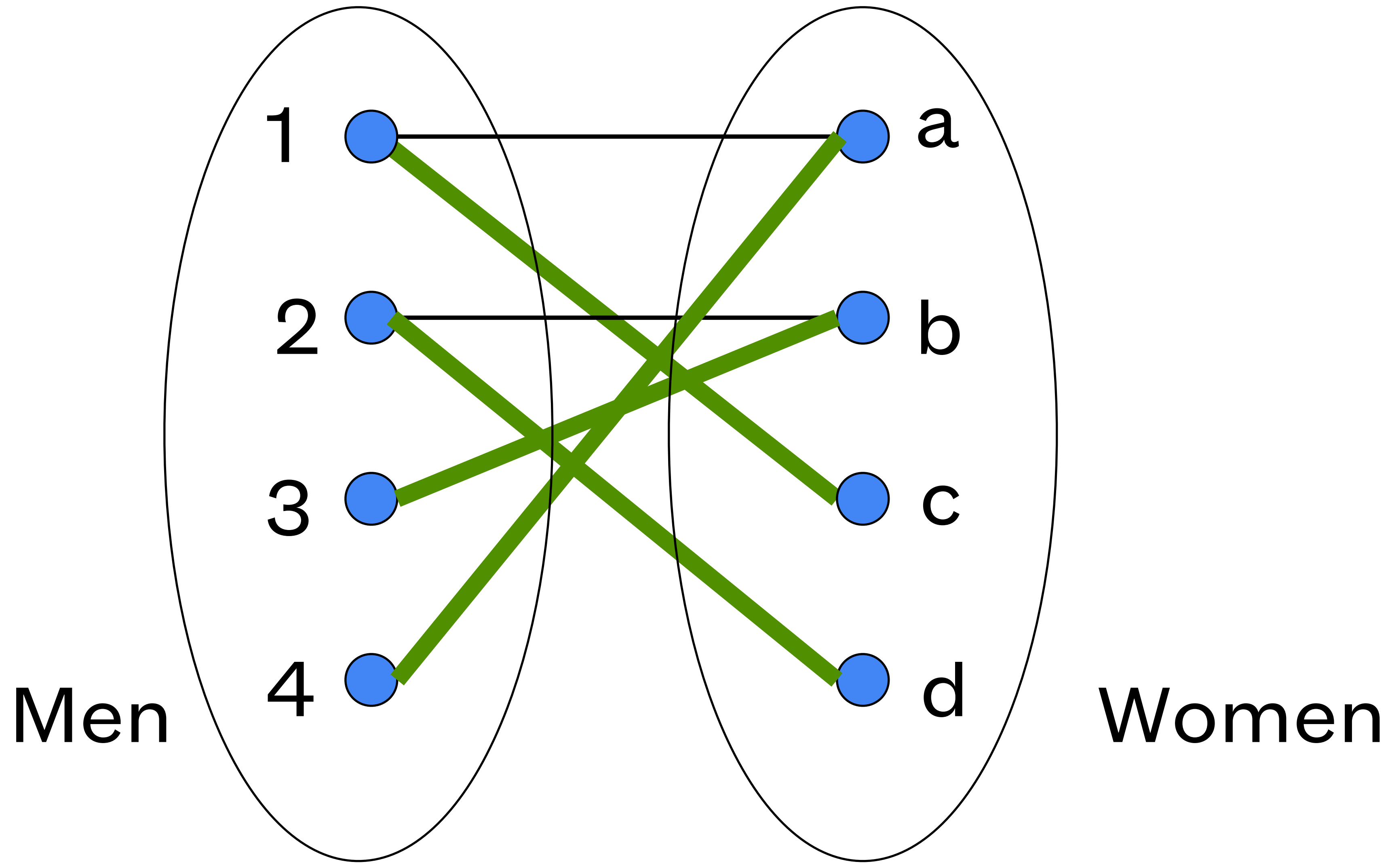
- During the greedy matching, no woman in C could be matched (but they could be matched in O)
- In greedy every $w \in C$ her optimal match taken by another woman already
  - Those matches are taken by women in A and B
- So, $|A| + |B| \geq |C|$
- Then we have, $2|G| = 2|A| + 2|B| = |A| + |B| + |A| + |B|$

  $\geq |B| + |C| = |O|$

The the worst case scenario is indeed 1/2



Greedy matching
with cardinality 2



Optimal matching
with cardinality 4

The **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 (online)
- Goal: maximize the revenue of the search engine

The **Adwords** problem

- Goal: maximize the revenue of the search engine
- **Further complication 1**: 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 (bid × click-through rate)
  - However, for simplicity, we will use “bid” instead of bid × ctr in our algorithms
- **Further complication 2:** each advertiser has limited budget
- The search engine cannot charge the advertiser more than the set budget

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

↓

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

Adwords: simplified model

- Assume all bids are 0 or 1
- Each advertiser has the same budget
- Only one advertiser is chosen per query

Greedy algorithm on the simplified adwords problem

- Let's try the greedy algorithm:
  - Arbitrarily pick an eligible advertiser for each keyword
- 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.

- Assume all bids are 0 or 1
- Each advertiser has the same budget
- Only one advertiser is chosen per query

Balance Algorithm

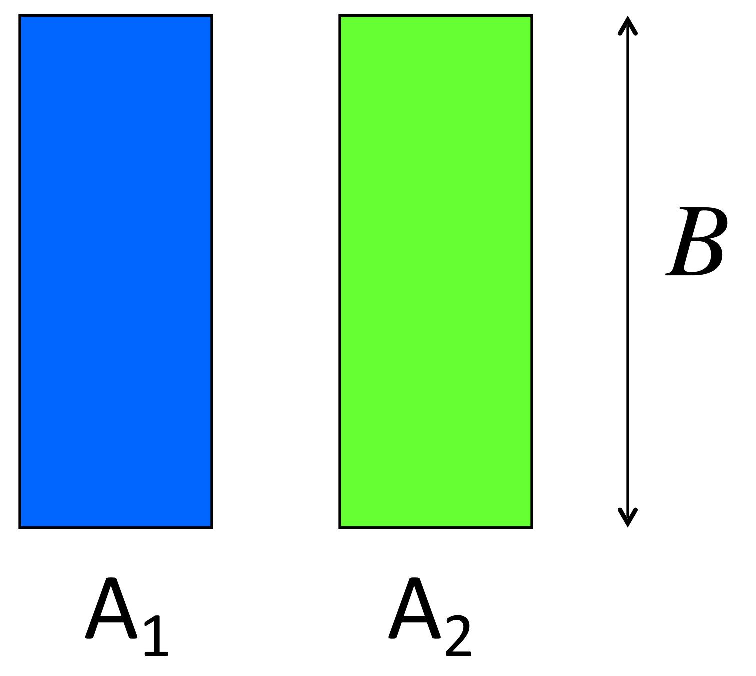
- Algorithm: for each query, pick the advertiser with the largest unspent budget who bid on this query
  - Break ties arbitrarily

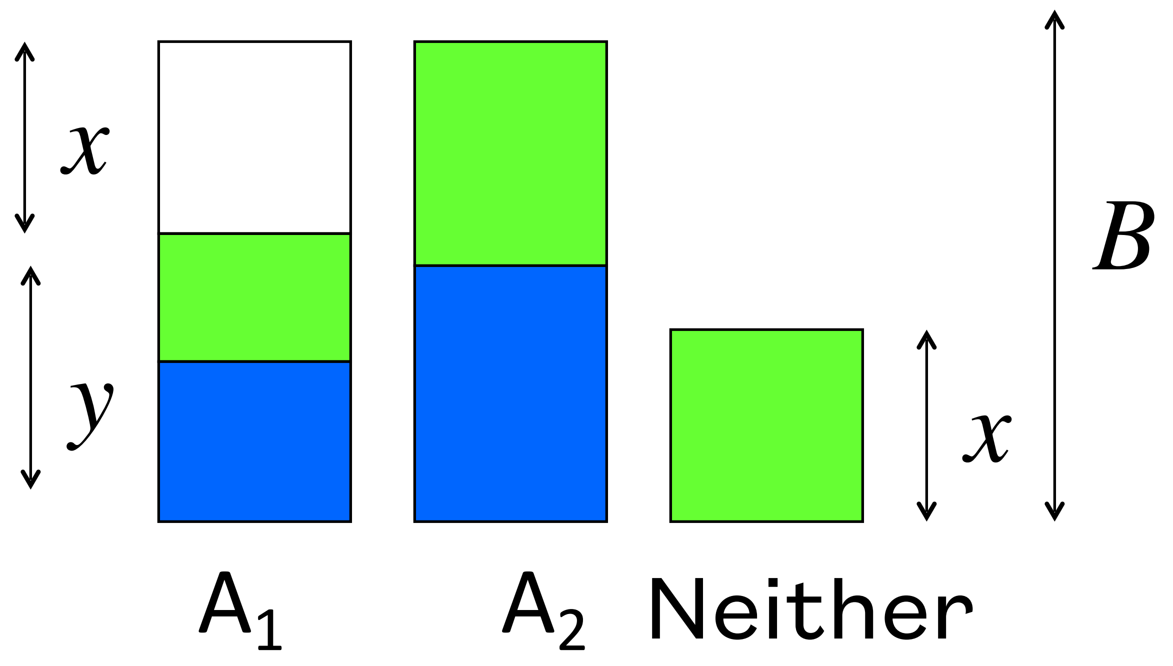
- Two advertisers A and B
- A bids on query x, B bids on x and y
- Both have budgets of $4
- Query stream: x x x x y y y y
- Balance choice: B A B A B B _ _
- Optimal: A A A A B B B B
- Competitive ratio = 3/4

Analyzing Balance

- Consider simple case: two advertisers A_1 and A_2
- Each with budget $B > 1$ (B is an even number)
- 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





Balance allocation

■ Queries allocated to A_1 in optimal solution

■ Queries allocated to A_2 in optimal solution

Optimal 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: $y \geq x$

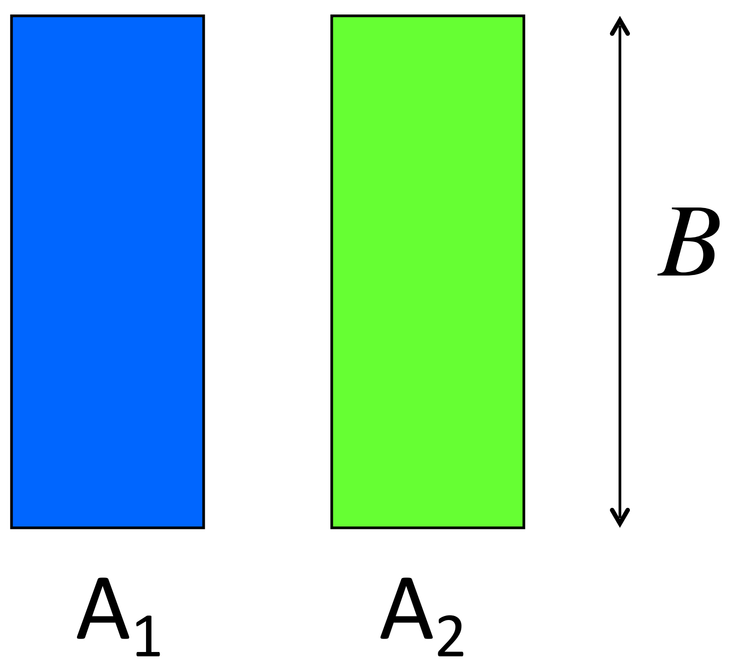
If we can prove that, then:

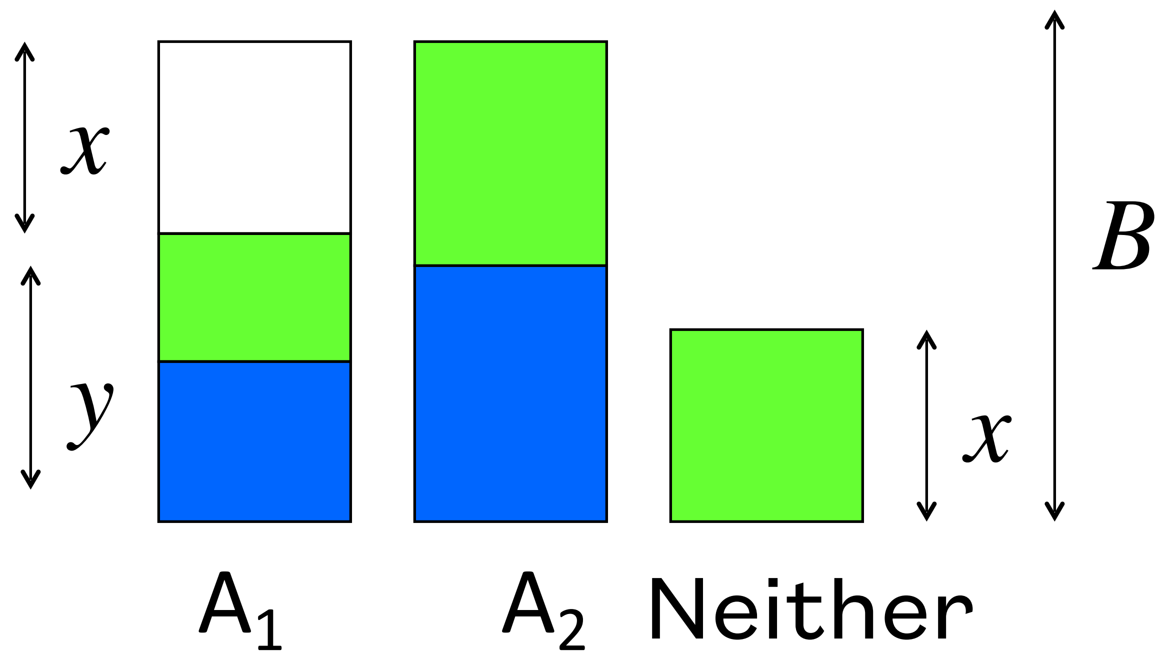
Balance revenue is minimum for $x = y = B/2$

Minimum Balance revenue = $3B/2$

Competitive Ratio = $3/4$

Analyzing Balance



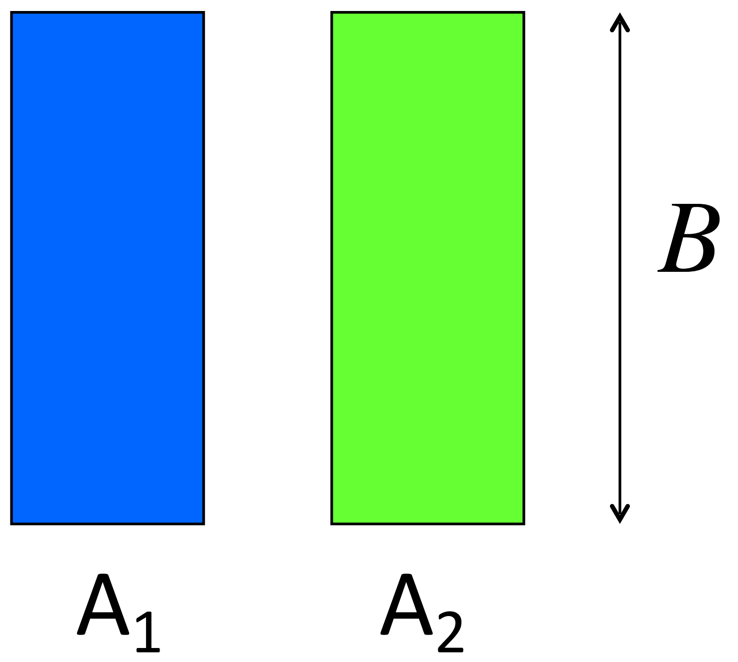


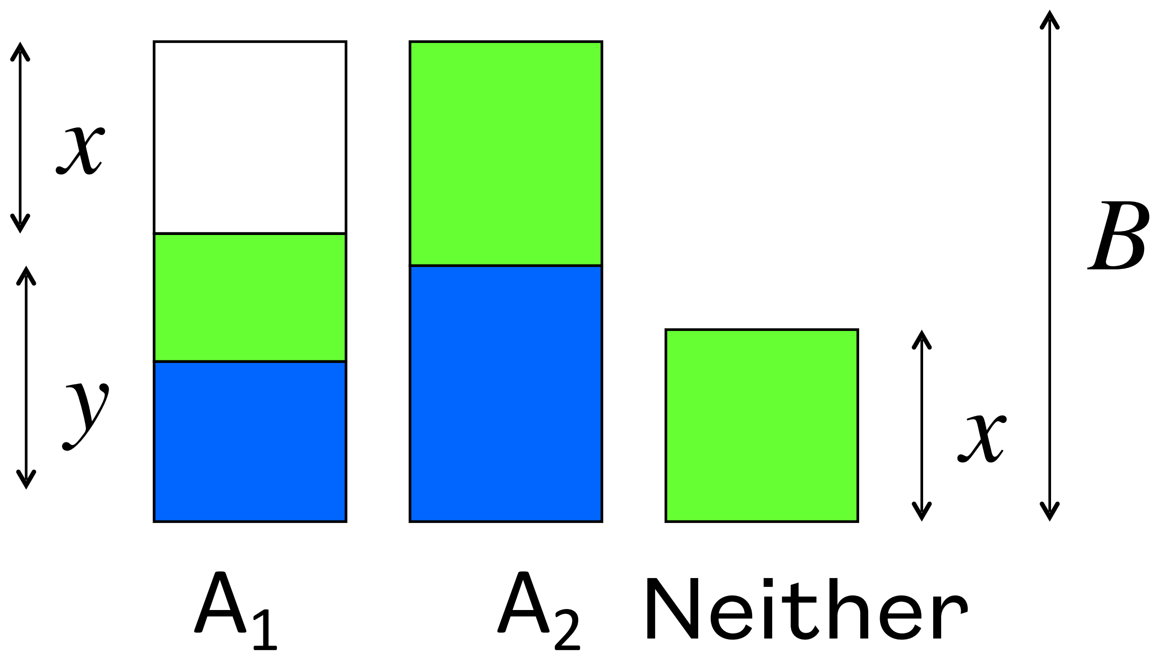
Balance allocation

- Queries allocated to A_1 in optimal solution
- Queries allocated to A_2 in optimal solution

- 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

Analyzing Balance





Balance allocation

Queries allocated to A_1 in optimal solution

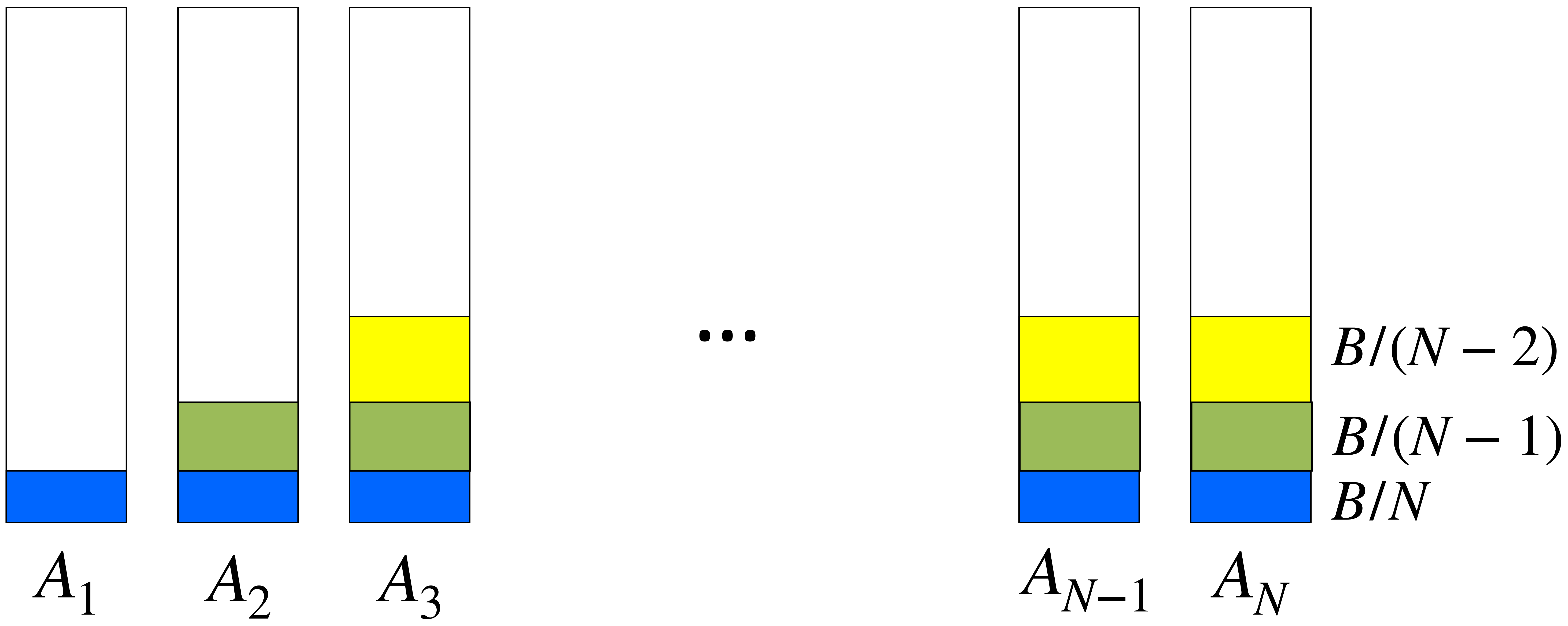
Queries allocated to A_2 in optimal solution

- 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 uses $\geq 3B/2$

Many bidders: an example

- N advertisers, each with budget B
- N distinct queries $q_1, q_2, \ldots, q_N$
- The i-th round: queries $q_i, q_i, \ldots, q_i$ (the same query q_i repeated B times)
- Total NB queries appear in N rounds
- Advertiser A_i bids on$q_1, q_2, \ldots, q_i$ only
- Bidders for round 1: all advertisers $A_1, A_2, \ldots, A_N$
- Bidders for round 2: all advertisers except A_1, i.e., $A_2, A_3, \ldots, A_N$
- Bidders for round i: advertisers $A_i, A_{i+1}, \ldots, A_N$
- Bidders for round N: only A_N
- Optimal allocation: all the queries in round i goes to A_i
  - Optimum revenue $= NB$ (all budget exhausted)

Balance algorithm on this example



After k rounds, sum of allocations to each of $A_k, \ldots, A_N$ is $S_k = S_{k+1} = \ldots = S_N$

$$= \sum_{i=1}^{k} B/(N-i+1) = B \sum_{i=1}^{k} 1/(N-i-1)$$

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

- The algorithm cannot allocate any queries after k rounds if $B \sum_{i=1}^{k} 1/(N-i+1) \geq B$,
- In other words, $\sum_{i=1}^{k} 1/(N-i+1) = \frac{1}{N} + \frac{1}{N-1} + \ldots + \frac{1}{N-k+1} \geq 1$
- When does this happen?

Balance algorithm: competitive ratio

- Fact (Euler): $H_n = 1 + \frac{1}{2} + \ldots + \frac{1}{n} \to \log_e n$ for large n
- We want to find k so that $\frac{1}{N} + \frac{1}{N-1} + \ldots + \frac{1}{N-k+1} \geq 1$
- The condition is same as $\left(\frac{1}{N} + \frac{1}{N-1} + \ldots + \frac{1}{2} + 1\right) - \left(\frac{1}{N-k} + \frac{1}{N-k-1} + \ldots + \frac{1}{2} + 1\right) \geq 1$
- For large n, the L.H.S. $\approx \log_e(N) - \log_e(N-k) = \log_e\left(\frac{N}{N-k}\right)$, and we need that to be ≥ 1
- This happens when $(N-k)/N \leq 1/e$, or $k \geq N(1 - 1/e)$
- Hence, no balance cannot make any further allocation after $N(1 - 1/e)$ rounds
- Revenue (balance) $= BN(1 - 1/e)$
- The competitive ratio of balance is at best $(1 - 1/e) \approx 0.63$
- Results (Kalyanasundaram and Pruhs, 2000) :
  - The competitive ratio of balance (where all advertisers have same budget) is indeed $1 - 1/e$
  - No deterministic online algorithm can have a better competitive ratio

General case: when advertisers have different budgets

- Different bidders have different budgets
- The algorithm would allocate queries to the advertiser with largest unspent budget
- Suppose both A_1 and A_2 bid on query q
- Query stream: query q comes 10 times
- Balance will allocate all 10 queries to A_1
- Revenue: 10
- Optimal: allocate all 10 queries to A_2
- Revenue: 100
- Competitive ratio = 1/10
- In fact, we can make this ratio as small as we want by constructing curated examples

Bidder	Bid	Budget
A_1	1	110
A_2	10	100

Bidder	Bid	Budget
A_1	1	210
A_2	20	200

Generalized Balance

- Todo 1: Be biased in favor of higher bids
- Todo 2: Consider fraction of remaining budget, not absolute remaining budget
- Generalized balance
  - Suppose advertiser A_i has bid x_i (can be zero) for query q
  - Fraction of A_i's budget currently remaining is f_i
  - Then define: $\Psi_i = x_i(1 - e^{-f_i})$
  - Allocate query q to the advertiser for whom Ψ_i is maximum (break ties arbitrarily)
- Result: This algorithm achieves competitive ratio = $(1 - 1/e) \approx 0.63$

Sources and Acknowledgements

1. J. Leskovec, A. Rajaraman and J. Ullman. "*Mining of massive datasets", Chapter 8.* Cambridge University Press, 2011. http://www.mmds.org
   - Some of the slides are used from the original authors, with minor modification
2. Tim. "What Happened To Overture.com?", May 2019: https://www.launchpresso.com/what-happened-to-overture-com/
3. B. Kalyanasundaram and K.R. Pruhs, "An optimal deterministic algorithm for b-matching," Theoretical Computer Science 233:1–2, pp. 319–325, 2000.
4. A Mehta, A. Saberi, U. Vazirani, and V. Vazirani, "Adwords and generalized on-line matching," IEEE Symp. on Foundations of Computer Science, pp. 264–273, 2005.