A glimpse of auction theory (continued) + distinct elements

Anna Karlin

Agenda

- Finish up glimpse of auction theory
- Distinct Elements

Auctions

- Companies like Google and Facebook make most of their money by selling ads.
- The ads are sold via auction.

Facebook Ads bidding... 🤔 Is this an auction?

Yes! That's the first thing you need to understand to master bidding management of Facebook Ads. **When you're creating a new campaign, you're joining a huge, worldwide auction.**

You'll be competing with hundreds of thousands of advertisers to buy what Facebook is selling: Real estate on the News Feed, Messenger, Audience Network, and mobile apps to display your ads to the users.

An auction is a ...

- Game
  - Players: advertisers
  - Strategy choices for each player: possible bids
  - Rules of the game – made up by Google/Facebook/whoever is running the auction

- What do we expect to happen? How do we analyze mathematically?

Special case: Sealed Bid single item auction

- Say I decide to run an auction to sell my laptop and I let you be the bidders.
- If I want to make as much money as possible – what should the rules of the auction be?

Some possibilities:

- **First price auction:** highest bidder wins; pays what they bid.
- **Second price auction:** highest bidder wins; pays second highest bid.
- **All pay auction:** highest bidder wins: all bidders pay what they bid.

Which of these will make me the most money?

Bidder	1	2	3	4
Bids	100	81	35	24

Payment	1st price	100	0	0	0
	2nd price	81	0	0	0
	All pay	100	81	35	24

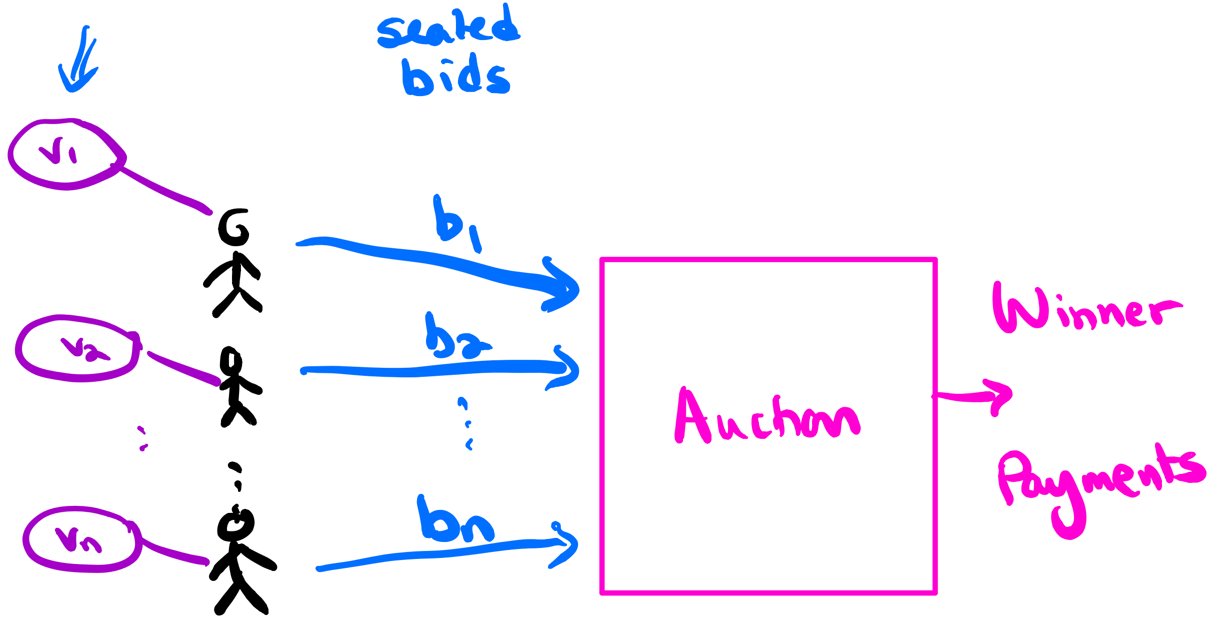
BIDDER MODEL

$v_i = 95$
indiff
$95 and
the item.

Each bidder has a value, say v_i for bidder i.

private information.

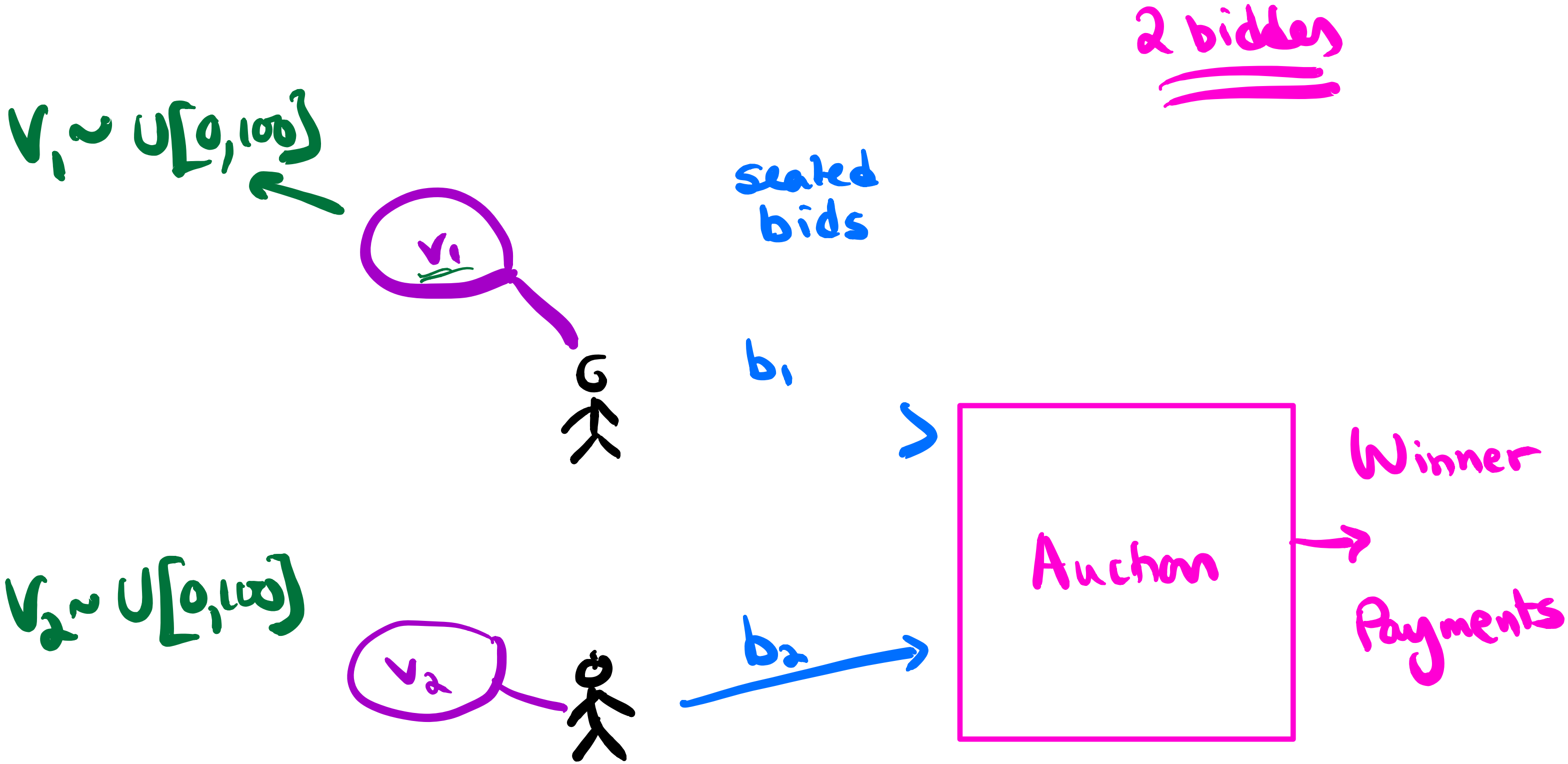
Bidder is trying to maximize their “utility” –
the value of the item they get – price they pay.



In a second price auction, it is always in your best interest to bid your value

$$b_i := v_i$$

2^{nd} price auction is <u>truthful</u>



	2nd price	1st price	All pay auction
v	bid $= v$	bid $= \frac{v}{2}$	bid $\frac{v^2}{200}$

equal bidding strategy

1st price auctn. If each bidder bids half their value, then they maximize E(utility) ⇐ value of other bidders

Bayes-Nash eq

best response in expectation.

$V_1 \sim U[0,100]$ $\quad$ $V_2 \sim U[0,100]$

$\Downarrow$ $\qquad$ $\Downarrow$

bid V_1 $\qquad$ bid. V_2

payment = $\min(V_1, V_2)$

Exp auctioneer revenue in 2nd price auction

a) $E(\max(V_1, V_2))$

b) $E\left(\frac{V_1+V_2}{2}\right)$

⇒ c) $\boxed{E(\min(V_1, V_2))}$

d) I don't know

(los 100)

2 bidd[illegible] U[0,10[illegible]

	2nd price	1st price	All pay auctn	
v	bid = v	bid = $\frac{v}{2}$	bid $\frac{v^2}{200}$	equil bidding strategy
exp auctioneer revenue	$E(\min(v_1,v_2))$	$E\left(\max\left(\frac{v_1}{2},\frac{v_2}{2}\right)\right)$	$E\left[\frac{v_1^2}{200}+\frac{v_2^2}{200}\right]$	

$\frac{1}{2}\underline{E(\max(v_1,v_2))}$

$\frac{2}{3}\cdot 100$

$= \frac{100}{3}$ $\quad$ $= \frac{100}{3}$ $\quad$ $= \frac{100}{3}$

$E[\min(v_1,v_2)] + E[\max(v_1,v_2)] = E(v_1+v_2)$

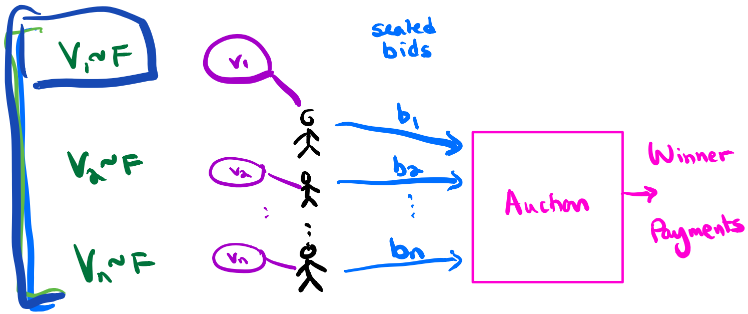
$\frac{2}{3}100$

$= E(v_1) + E(v_2)$

$\frac{100}{2} + \frac{100}{2}$

100

$$E[\min(v_1, v_2)] = 100 - \frac{2}{3}100 = \frac{100}{3}$$



Revenue equivalence Thm

In equilibrium
E(auctioneer rev) same in all 3 auctions.

$\Rightarrow$ E[payment of each bidder same in all 3 auctions]

2nd price auction with reserve price of r

highest bidder wins if $b \geq r$ pay max(2nd highest bid, r)

100 81 75

r = 90

Distinct elements

Anna Karlin
With many slides by Luxi Wang, Shreya Jayaraman,
Alex Tsun and Jeff Ullman

Data mining

- In many data mining situations, the data is not known ahead of time.
- Examples:
    - Google queries
    - Twitter or Facebook status updates
    - Youtube video views
- In some ways, best to think of the data as an infinite stream that is non-stationary (distribution changes over time)

Stream model

- Input elements (e.g. Google queries) enter/arrive one at a time.
- We cannot possibly store the stream.

Question: How do we make critical calculations about the data stream using a limited amount of memory?

Sources of this kind of data

- Sensor data
    - E.g. millions of temperature sensors deployed in the ocean
- Image data from satellites or surveillance camers
    - E.g. London
- Internet and web traffic
    - E.g. millions of streams of IP packets
- Web data
    - E.g. Search queries on Google, clicks on Bing, etc.

Example Applications

- Mining query streams
  - Google wants to know which queries are more frequent today than yesterday.
- Mining click streams
  - Facebook wants to know which of its ads are getting an unusual number of hits in the last hour.
- Mining social network news feeds
  - E.g., looking for trending topics on Twitter and Facebook, trending videos on TikTok

More Applications

- Sensor networks
  - Many sensors feeding into a central controller.
- IP packets
  - Gather congestion information for optimal routing
  - Detect denial-of-service attacks

Problem

user ids.

- Input: sequence of N elements $x_1, x_2, \ldots, x_N$ from a known universe U (e.g., 8-byte integers).
- Goal: perform a computation on the input, in a single left to right pass where
  - Elements processed in real time
  - Can't store the full data. => use minimal amount of storage while maintaining working "summary"

What can we compute?

32, 12, 14, 32, 7, 12, 32, 7, 32, 12, 4,

32 12 12 12 7 7 7 7 . . . 4

- Some functions are easy:
  - Min
  - Max
  - Sum
  - Average

Counting distinct elements

x_1 x_2 x_3 x_4

32, 12, 14, 32, 7, 12, 32, 7, 32, 12, 4,

32, 12, 14, 7, 4

Applications:

- IP packet streams: How many distinct IP addresses or IP flows (source+destination IP, port, protocol)
    - Anomaly detection, traffic monitoring
- Search: How many distinct search queries on Google on a certain topic yesterday
- Web services: how many distinct users (cookies) searched/browsed a certain term/item
    - Advertising, marketing trends, etc.

Another application

You are the content manager at YouTube, and you are trying to figure out the **distinct** view count for a video. How do we do that?

Note: A person can view their favorite videos several times, but they only count as 1 **distinct** view!

Counting distinct elements

32, 12, 14, 32, 7, 12, 32, 7, 32, 12, 4,

- Want to compute number of **distinct** keys in the stream.
- *How to do this without storing all the elements?*
- *Yet another super cool application of probability (and hashing)*

A naive solution, counting!

Store the n **distinct** user IDs in a hash table.

Space requirement: O(n)

#distinct elts in stream.

x_i

$h(x_i)$

insert

Count

Considering the number of users of YouTube, and the number of videos on YouTube, this is not feasible.

Consider a hash function $h : \mathcal{U} \rightarrow [0, 1]$
For distinct values in $\mathcal{U}$, the function maps to iid (independent and identically distributed) Unif(0,1) random numbers.

Note that, if you were to feed in two equivalent elements, the function returns the **same** number.

h(32) = 0.43
h(12) = 0.19
h(14) = 0.85
h(7) = 0.61

32, 12, 14, 32, 7, 12, 32, 7, 32, 12, 4,

0.43 0.19 0.85 0.43 0.61 0.19 ...

hash vals

track min hash value seen so far

n distinct elts in stream

min hash value $= \min(U_1, U_2, \ldots, U_n)$

Min of IID Uniforms

If $Y_1, \ldots, Y_m$ are iid $Unif(0,1)$, where do we "expect" the points to end up?

$m = 1$

$m = 2$

$m = 4$