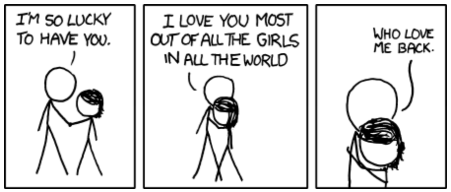
CSE 421: Algorithms

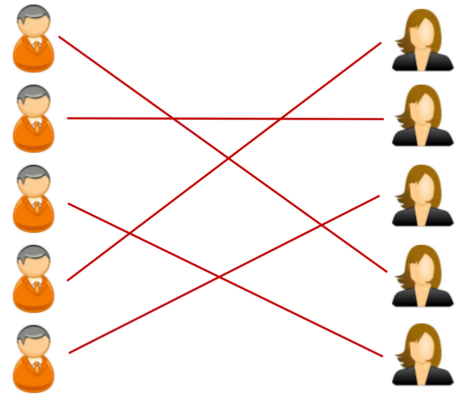
Winter 2014

Lecture 2: Stable matching

Reading: Chapter 2 of Kleinberg-Tardos



stable matching problem



propose-and-reject algorithm

Propose-and-reject algorithm. [Gale-Shapley 1962]

Intuitive method that is guaranteed to find a stable matching.

```
Initialize each person to be free
while (some man is free and hasn't proposed to every woman) {
    Choose such a man m
    w = 1st woman on m's list to whom m has not yet proposed
    if (w is free)
        assign m and w to be engaged
    else if (w prefers m to her fiancé m')
        assign m and w to be engaged, and m' to be free
    else
        w rejects m
}
```

http://mathsite.math.berkeley.edu/smp/smp.html
http://www.cs.columbia.edu/~evs/intro/stable/Stable.html
http://demonstrations.wolfram.com/StableMarriages/

proof of correctness: termination

- Observation 1. Men propose to women in decreasing order of preference.
- Observation 2. Once a woman is matched, she never becomes unmatched; she only "trades up."
- Claim. Algorithm terminates after at most n^2 iterations of while loop.
- Proof. Each time through the while loop a man proposes to a new woman. There are only n^2 possible proposals. ▪

	1st	2nd	3rd	4th	5th
Victor	A	B	C	D	E
Walter	B	C	D	A	E
Xavier	C	D	A	B	E
Yuri	D	A	B	C	E
Zoran	A	B	C	D	E

	1st	2nd	3rd	4th	5th
Amy	W	X	Y	Z	V
Brenda	X	Y	Z	V	W
Claire	Y	Z	V	W	X
Diane	Z	V	W	X	Y
Erika	V	W	X	Y	Z

$n(n-1) + 1$ proposals required

proof of correctness: perfection

- Claim: All men and women get matched.
- Proof:

proof of correctness: stability

Claim: No unstable pairs.
Proof: (by contradiction)
– Suppose A-Z is an unstable pair: each prefers each other to partner in Gale-Shapley matching S*.

summary

- Stable matching problem. Given n men and n women, and their preferences, find a stable matching if one exists.
- Gale-Shapley algorithm. Guarantees to find a stable matching for any problem instance.
- How to implement GS algorithm efficiently?
- If there are multiple stable matchings, which one does GS find?

implementation

- Problem size
  – $N = 2n^2$ words
    $2n$ people each with a preference list of length n
  – $2\ n^2 \log n$ bits
    specifying an ordering for each preference list: $n \log n$ bits
- Brute force algorithm
  – Try all $n!$ possible matchings
  – Do any of them work?
- Gale-Shapley Algorithm
  – n^2 iterations, each costing constant time as follows:

efficient implementation

Efficient implementation. We describe $O(n^2)$ time implementation.

Representing men and women.
- Assume men are named **1, ..., n**.
- Assume women are named **1', ..., n'**.

Engagements.
- Maintain a list of free men, e.g., in a queue.
- Maintain two arrays **wife[m]**, and **husband[w]**.
  - set entry to **0** if unmatched
  - if **m** matched to **w** then **wife[m]=w** and **husband[w]=m**

Men proposing.
- For each man, maintain a list of women, ordered by preference.
- Maintain an array **count[m]** that counts the number of proposals made by man **m**.

efficient implementation

Women rejecting/accepting.
- Does woman **w** prefer man **m** to man **m'**?

efficient implementation

Women rejecting/accepting.
- Does woman **w** prefer man **m** to man **m'**?
- For each woman, create inverse of preference list of men.
- Constant time access for each query after $O(n)$ preprocessing per woman. $O(n^2)$ total reprocessing cost.

Amy	1st	2nd	3rd	4th	5th	6th	7th	8th
Pref	8	3	7	1	4	5	6	2

Amy	1	2	3	4	5	6	7	8
Inverse	4th	8th	2nd	5th	6th	7th	3rd	1st

```
for i = 1 to n
   inverse[pref[i]] = i
```

Amy prefers man 3 to 6
since inverse[3]=2 < 7=inverse[6]

understanding the solution

For a given problem instance, there may be several stable matchings. Do all executions of Gale-Shapley yield the same stable matching? If so, which one?

	1st	2nd	3rd
Xavier	A	B	C
Yuri	B	A	C
Zoran	A	B	C

	1st	2nd	3rd
Amy	Y	X	Z
Brenda	X	Y	Z
Claire	X	Y	Z

An instance with two stable matchings.
- A-X, B-Y, C-Z.
- A-Y, B-X, C-Z.

understanding the solution

- For a given problem instance, there may be several stable matchings. Do all executions of Gale-Shapley yield the same stable matching? If so, which one?
- Def. Man **m** is a valid partner of woman **w** if there exists some stable matching in which they are matched.
- Man-optimal assignment. Each man receives best valid partner (according to his preferences).

understanding the solution

- For a given problem instance, there may be several stable matchings. Do all executions of Gale-Shapley yield the same stable matching? If so, which one?
- Def. Man **m** is a valid partner of woman **w** if there exists some stable matching in which they are matched.
- Man-optimal assignment. Each man receives best valid partner (according to his preferences).
- Claim. All executions of GS yield a man-optimal assignment, which is a stable matching!
  - No reason a priori to believe that man-optimal assignment is perfect, let alone stable.
  - Simultaneously best for each and every man.

man optimality

Claim. GS matching S^* is man-optimal.

- Proof. (by contradiction)
  - Suppose some man is paired with someone other than his best partner. Men propose in decreasing order of preference $\Rightarrow$ some man is rejected by a valid partner.
  - Let **Y** be the man who is the first such rejection, and let **A** be the women who is first valid partner that rejects him.
  - Let **S** be a stable matching where **A** and **Y** are matched.

stable matching summary

Stable matching problem. Given preference profiles of n men and n women, find a stable matching.

Gale-Shapley algorithm. Finds a stable matching in $O(n^2)$ time.

Man-optimality. In version of GS where men propose, each man receives best valid partner.

Does man-optimality come at the expense of the women?

measuring efficiency: the RAM model

- RAM = Random Access Machine
- Time ≈ # of instructions executed in an ideal assembly language
  - each simple operation (+,*,-,=, if, call) takes one time step
  - each memory access takes one time step

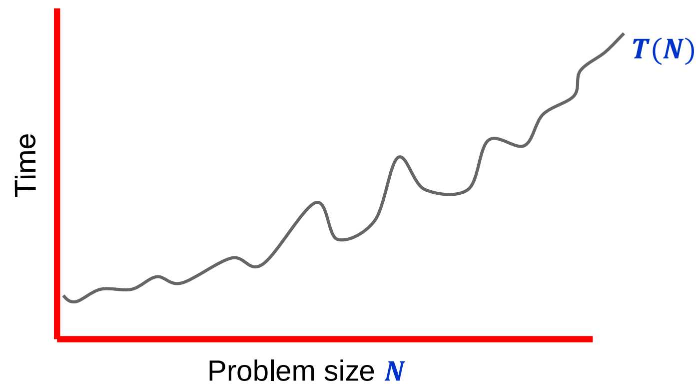
complexity analysis

- Problem size N
  - **Worst-case complexity**: max # steps algorithm takes on any input of size N
  - **Average-case complexity**: average # steps algorithm takes on inputs of size N

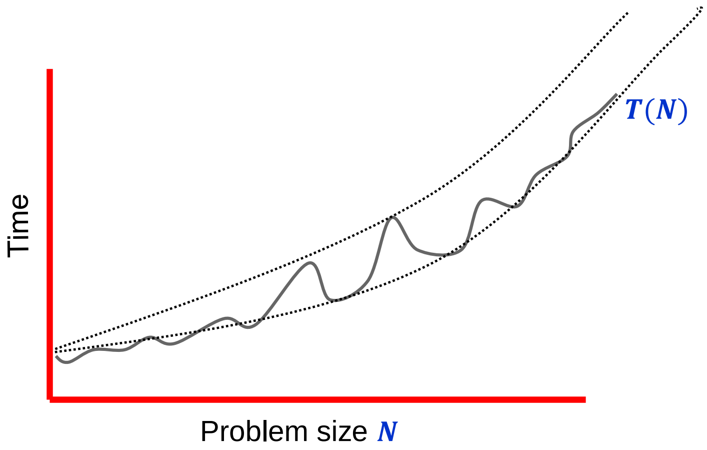
complexity

- The complexity of an algorithm associates a number T(N), the worst/average-case/best time the algorithm takes, with each problem size N.
- Mathematically,
  - $T: \mathbb{N} \rightarrow \mathbb{R}_{\geq 0}$ is a function that maps positive integers giving problem size to positive real numbers giving number of steps

complexity



complexity



complexity

complexity

complexity