Algorithm Design and Analysis





LECTURE 2 Analysis of Algorithms

- Stable matching problem
- Asymptotic growth

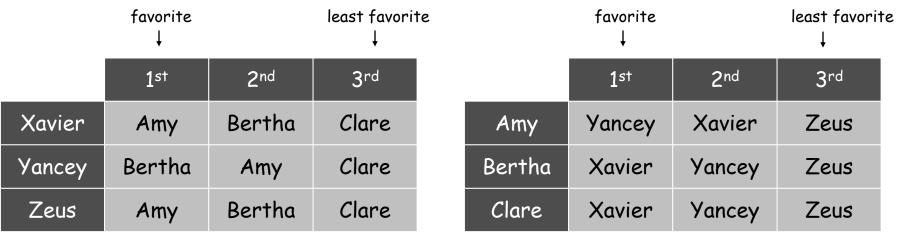
Sofya Raskhodnikova

Logistics

- Homework 1 will be posted tomorrow, due next Thursday
- Reading
 - KT Chapter 3
 - Reading Quizes on Canvas due Thursday & Sunday night
- Other stuff for you to do (if you just joined)
 - Background Quiz
 - Nameplate (from course page)
 - Sign up for Piazza for announcements

Stable Matching Problem

- Unstable pair: man *m* and woman *w* are unstable if
 - -m prefers w to his assigned match, and
 - -w prefers *m* to her assigned match
- Unstable pairs have an incentive to elope
- **Stable matching**: no unstable pairs.



Men's Preference Profile

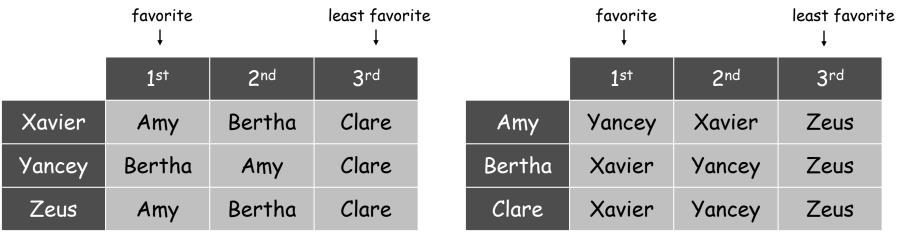
Women's Preference Profile

L2.3

8/24/2016

Stable Matching Problem

- Input: preference lists of *n* men and *n* women
- Goal: find a stable matching if one exists



Men's Preference Profile

Women's Preference Profile

8/24/2016

Review Questions

• In terms of *n*, what is the length of the input to the Stable Matching problem, i.e., the number of entries in the tables?

• How many bits do they take to store? (Answer: $2n^2$ list entries, or $2n^2\log n$ bits)

Review Questions

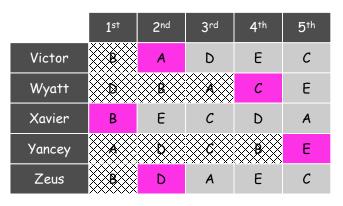
- **Brute force algorithm:** an algorithm that checks every possible solution.
- In terms of *n*, what is the running time of the brute force algorithm for checking whether a given matching is stable?
- In terms of *n*, what is the running time of the brute force algorithm for Stable Matching Problem? (Assume your algorithm goes over all possible perfect matchings.)

(Answer: $n! \times$ (time to check if a matching is stable) = $\Theta(n! n^2)$) ^{8/24/2016} S. Raskhodnikova; based on slides by E. Demaine, C. Leiserson, A. Smith, K. Wayne L2.6

Review question

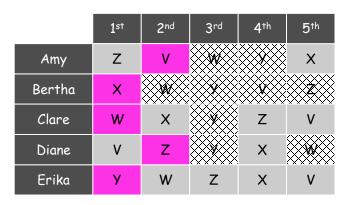
How many stable matchings are there for this instance?

- **A**. 0
- **B.** 1
- C. 2 or more.



Men's preferences

Women's preferences



Review question

1)

Men's preferences

	1 st	2 nd	3 rd	4 th	5 th
Victor	В	А	D	E	С
Wyatt	D	В	А	С	E
Xavier	В	E	С	D	А
Yancey	А	D	С	В	E
Zeus	В	D	А	E	С

Women's preferences

	1 ^{s†}	2 nd	3 rd	4 th	5 th
Amy	Z	V	W	У	Х
Bertha	X	W	У	V	Z
Clare	W	х	У	Z	V
Diane	V	Z	У	х	W
Erika	У	W	Z	х	V

2)

Men's preferences

	1 ^{s†}	2 nd	3 rd	4 th	5 th
Victor	В	А	D	E	С
Wyatt	D	В	А	С	E
Xavier	В	E	С	D	A
Yancey	А	D	С	В	Е
Zeus	В	D	А	E	С

Women's preferences

	1 ^{s†}	2 nd	3 rd	4 th	5 th
Amy	Z	V	W	У	Х
Bertha	х	W	У	V	Z
Clare	W	х	У	Z	V
Diane	V	Z	У	х	W
Erika	У	W	Z	х	V

Brief Syllabus

- Reminders
 - Worst-case analysis
 - Asymptotic notation
 - Basic data structures
- Design Paradigms
 - Greedy algorithms, divide and conquer, dynamic programming, network flow, linear programming, randomization
- P, NP and NP-completeness

Useful Functions and Asymptotics

Permutations and combinations

• Factorial: "*n* factorial"

$$n! = n \cdot (n-1) \cdot \ldots \cdot 2 \cdot 1$$

= number of permutations of $\{1, ..., n\}$

• Combinations: "*n* choose *k*"

$$\binom{n}{k} = \frac{n \times (n-1) \times \dots \times (n-k+1)}{k \times (k-1) \times \dots \times 2 \times 1} = \frac{n!}{k!(n-k)!}$$

= number of ways of choosing an unordered subset of k items in $\{1, ..., n\}$ without repetition

8/24/2016

Review Question

• In how many ways can we select two disjoint subsets of {1, ..., n}, of size k and m, respectively?

• Answer:

$$\binom{n}{k}\binom{n-k}{m} = \binom{n}{m}\binom{n-m}{k} = \binom{n}{m+k}\binom{m+k}{k}$$

Asymptotic notation

O-notation (upper bounds): f(n) = O(g(n)) means there exist constants c > 0, $n_0 > 0$ such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0$. **EXAMPLE:** $2n^2 = O(n^3)$ $(c = 1, n_0 = 2)$

functions, not values

Asymptotic Notation

• **One-sided equality:** T(n) = O(f(n)).

-Not transitive:

- $f(n) = 5n^3$; $g(n) = 3n^2$
- $f(n) = O(n^3) = g(n)$
- but $f(n) \neq g(n)$.

-Alternative notation: $T(n) \in O(f(n))$.

Set Definition

 $O(g(n)) = \{ f(n) : \text{there exist constants} \\ c > 0, n_0 > 0 \text{ such} \\ \text{that } 0 \le f(n) \le cg(n) \\ \text{for all } n \ge n_0 \}$

EXAMPLE: $2n^2 \in O(n^3)$

(*Logicians:* $\lambda n.2n^2 \in O(\lambda n.n^3)$, but it's convenient to be sloppy, as long as we understand what's *really* going on.)

Examples

• $10^6 n^3 + 2n^2 - n + 10 = O(n^3)$

• $n^{1/2} + \log n = O(n^{1/2})$

• $n(\log n + \sqrt{n}) = O(n^{3/2})$

• $n = O(n^2)$

Ω -notation (lower bounds)

O-notation is an *upper-bound* notation. It makes no sense to say f(n) is at least $O(n^2)$.

 $\Omega(g(n)) = \{ f(n) : \text{there exist constants} \\ c > 0, n_0 > 0 \text{ such} \\ \text{that } 0 \le cg(n) \le f(n) \\ \text{for all } n \ge n_0 \}$

EXAMPLE: $\sqrt{n} = \Omega(\log n)$ (*c* = 1, *n*₀ = 16)

8/24/2016

Ω -notation (lower bounds)

• **Be careful:** "Any comparison-based sorting algorithm requires at least O(n log n) comparisons."

-Meaningless!

-Use Ω for lower bounds.

O-notation (tight bounds)

$$\Theta(g(n)) = \Theta(g(n)) \cap \Omega(g(n))$$

EXAMPLE:
$$\frac{1}{2}n^2 - 2n = \Theta(n^2)$$

Polynomials are simple: $a_d n^d + a_{d-1} n^{d-1} + \dots + a_1 n + a_0 = \Theta(n^d)$

8/24/2016

o-notation and w-notation

O-notation and Ω -notation are like \leq and \geq . *o*-notation and ω -notation are like < and >.

 $o(g(n)) = \{ f(n) : \text{ for every constant } c > 0, \\ \text{ there is a constant } n_0 > 0 \\ \text{ such that } 0 \le f(n) < cg(n) \\ \text{ for all } n \ge n_0 \}$

EXAMPLE: $2n^2 = o(n^3)$ $(n_0 = 2/c)$

8/24/2016

Overview of Asymptotic Notation

Notation	means	Think	E.g.	$\operatorname{Lim} f(n)/g(n)$
f(n)=O(n)	$ \begin{array}{l} \exists \ c > 0, n_0 > 0 \\ \forall \ n > n_0: \\ 0 \leq f(n) < cg(n) \end{array} $	Upper bound	$100n^2$ = O(n^3)	If it exists, it is < ∞
$f(n)=\Omega(g(n))$	$ \exists c > 0, n_0 > 0, \forall n > n_0 : \\ 0 \le cg(n) < f(n) $	Lower bound	$2^n = \Omega(n^{100})$	If it exists, it is > 0
$f(n) = \Theta(g(n))$	both of the above: $f=\Omega(g)$ and $f = O(g)$	Tight bound	$log(n!) = \Theta(n \log n)$	If it exists, it is > 0 and $<\infty$
f(n)=o(g(n))	$ \begin{aligned} \forall c > 0, \ \exists n_0 > 0, \ \forall n > n_0 : \\ 0 \leq f(n) < cg(n) \end{aligned} $	Strict upper bound	$n^2 = o(2^n)$	Limit exists, =0
$f(n)=\omega(g(n))$	$ \begin{aligned} \forall c > 0, \ \exists n_0 > 0, \ \forall n > n_0 : \\ 0 \leq cg(n) < f(n) \end{aligned} $	Strict lower bound	$n^2 = \omega(\log n)$	Limit exists, =∞

Common Functions: Asymptotic Bounds

- **Polynomials.** $a_0 + a_1 n + \dots + a_d n^d$ is $\Theta(n^d)$ if $a_d > 0$.
- Polynomial time. Running time is $O(n^d)$ for some constant d independent of the input size n.
- Logarithms. $\log_a n = \Theta(\log_b n)$ for all constants a, b > 0.

can avoid specifying the base

log grows slower than every polynomial

For every x > 0, $\log n \stackrel{\star}{=} o(n^x)$.

Every polynomial grows slower than every exponential

- **Exponentials.** For all r > 1 and all d > 0, $n^d = o(r^n)$.
- Factorial. By Sterling's formula,

$$n! = \left(\sqrt{2\pi n}\right) \left(\frac{n}{e}\right)^n \left(1 + o(1)\right) = 2^{\Theta(n\log n)}$$

grows faster than every exponential

8/24/2016

Exercise: Show that $log(n!) = \Theta(n log n)$

• Upper bound: $\log(n!) = \sum \log(i)$ i=1 $< n \log(n)$ • Lower bound: $\log(n!) = \sum \log(i)$ i = 1 $\geq \sum \log(i)$ i=1 $\geq \frac{n}{2}\log(\frac{n}{2}) = \frac{n}{2}\log(n) - \frac{n}{2}$

8/24/2016

Exercise: Show that $log(n!) = \Theta(n log n)$

• Stirling's formula:

$$n! = \left(\sqrt{2\pi n}\right) \left(\frac{n}{e}\right)^n \left(1 + o(1)\right)$$

$$\log(n!) = \log(\sqrt{2\pi n}) + n\log(n) - n\log(e) + \underbrace{\log(1+o(1))}_{-\infty}$$

 $=\log(1)+o(1)$ since log is continuous

$$= n \left(\log(n) - \log(e) + \frac{\log(2\pi n)}{n} \right) + o(1)$$

= $n (\log(n) - O(1)) + o(1)$
= $n \log n (1 - O(\frac{1}{\log n})) + o(1)$
= $n \log n (1 \pm o(1)) = \Theta(n \log n)$

8/24/2016

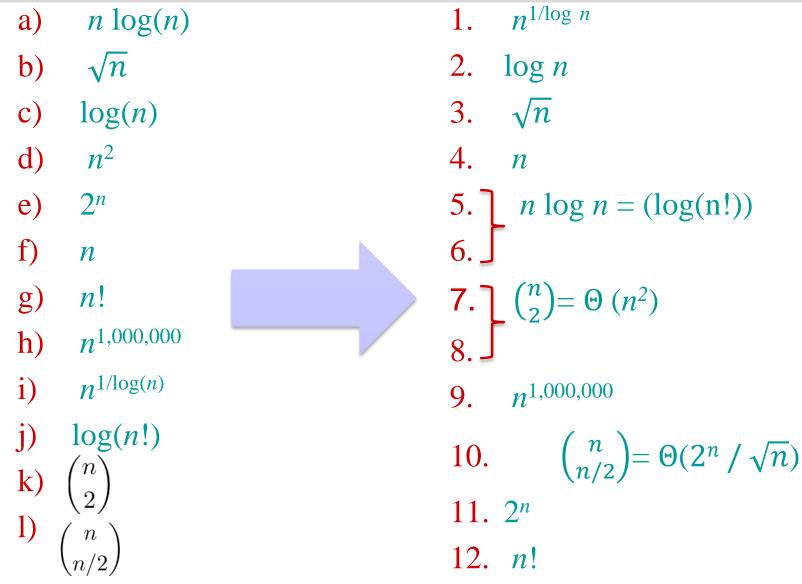
Sort by asymptotic order of growth

- a) $n \log(n)$
- b) \sqrt{n}
- c) $\log(n)$
- d) n^2
- e) 2^{*n*}
- f) *n*
- g) *n*!
- h) $n^{1,000,000}$
- i) $n^{1/\log(n)}$

j)
$$\log(n!)$$

k) $\binom{n}{2}$
l) $\binom{n}{n/2}$

Sort by asymptotic order of growth



8/24/2016

S. Raskhodnikova; based on slides by E. Demaine, C. Leiserson, A. Smith, K. Wayne

L2.26

Review question

• True or false?

1.
$$n^2 = O\left(\frac{n^2}{2}\right)$$

2. $n^2 = \omega\left(\frac{n^2}{2}\right)$
3. $n^2 = \Omega\left(\frac{n^2}{2}\right)$
4. $n^2 = O\left(2^{3\log_2 n}\right)$

Properties

- •Transitivity.
 - -If f = O(g) and g = O(h) then f = O(h).
 - -If $f = \Omega(g)$ and $g = \Omega(h)$ then $f = \Omega(h)$.
 - –Similarly, for Θ -, o- and ω -notation.
- •Additivity.
 - -If f = O(h) and g = O(h) then f + g = O(h).
 - -If $f = \Omega(h)$ and $g = \Omega(h)$ then $f + g = \Omega(h)$.
 - –Similarly, for Θ -, o- and ω -notation.

Question

- Let *f*, *g* be nonnegative functions.
- Consider the statement:

"either f(n) = O(g(n)) or g(n) = O(f(n))

(or both)"

Is this statement:

- 1. True for all functions *f* and *g*?
- 2. True for some, but not all, functions f and g?
- **3**. False for all functions *f* and *g*?

Conventions for formulas

Convention: A set in a formula represents an anonymous function in the set.

Example: $f(n) = n^3 + O(n^2)$ (right-hand side) means $f(n) = n^3 + h(n)$ for some $h(n) \in O(n^2)$.

Convention for formulas

Convention: A set in a formula represents an anonymous function in the set.

EXAMPLE:

(left-hand side)

 $n^{2} + O(n) = O(n^{2})$ means for any $f(n) \in O(n)$: $n^{2} + f(n) = h(n)$ for some $h(n) \in O(n^{2})$.

Review question

• True or false?

1.
$$2\binom{n}{2} = n^2(1 + o(1))$$

2. $\log_2(100 n^2) = \log_2(n) + O(1)$
3. $n^3 + O(n) = \Omega(n^2)$