

Objectives

- Review: Asymptotic running times
- Implementing Gale-Shapley algorithm
- Classes of running times

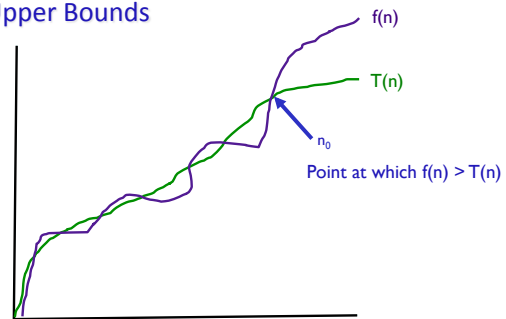
Review Asymptotic Bounds

- What does $O(f(n))$ mean?

Asymptotic Order of Growth: Upper Bounds

- $T(n)$ is the worst case running time of an algorithm
 - We say that $T(n)$ is $O(f(n))$ if there exist constants $c > 0$ and $n_0 \geq 0$ such that for all $n \geq n_0$, we have $T(n) \leq c \cdot f(n)$
- Annotations:
 - "order $f(n)$ " points to $f(n)$
 - "c cannot depend on n" points to c
 - "sufficiently large n" points to n_0
 - "T(n) is bounded above by a constant multiple of f(n)" points to $T(n) \leq c \cdot f(n)$
- T is **asymptotically upperbounded** by f

Asymptotic Order of Growth: Upper Bounds



Upper Bounds Example

- Find an upperbound for $T(n) = pn^2 + qn + r$
- p, q, r are positive constants

Idea: Let's inflate the terms in the equation so that all terms are n^2

Upper Bounds Example

- $T(n) = pn^2 + qn + r$
- p, q, r are positive constants
- For all $n \geq 1$,

$$\begin{aligned}
 T(n) &= pn^2 + qn + r \\
 &\leq pn^2 + qn^2 + rn^2 \\
 &= (p+q+r)n^2 \\
 &= cn^2
 \end{aligned}$$
- $T(n) \leq cn^2$, where $c = p+q+r$
- $T(n) = O(n^2)$
- Also correct to say that $T(n) = O(n^3)$

Notation

- $T(n) = O(f(n))$ is a **slight abuse of notation**
 - **Asymmetric:**
 - $f(n) = 5n^3; g(n) = 3n^2$
 - $f(n) = O(n^3) = g(n)$
 - But $f(n) \neq g(n)$.
 - **Better notation:** $T(n) \in O(f(n))$
- **Meaningless statement.** Any comparison-based sorting algorithm requires *at least* $O(n \log n)$ comparisons
 - Use Ω for lower bounds

Asymptotic Order of Growth: Lower Bounds

- Complementary to upper bound
- $T(n)$ is $\Omega(f(n))$ if there **exist constants** $\epsilon > 0$ and $n_0 \geq 0$ such that for all $n \geq n_0$, we have $T(n) \geq \epsilon \cdot f(n)$

ϵ cannot depend on n

sufficiently large n

$T(n)$ is bounded below by a constant multiple of $f(n)$

➔ T is **asymptotically lowerbounded** by f

Example: Lower Bound

- $T(n) = pn^2 + qn + r$
 - p, q, r are positive constants
- **Idea:** Deflate terms rather than inflate
- For all $n \geq 0$,

$$T(n) = pn^2 + qn + r \geq pn^2$$

$$\rightarrow T(n) \geq \epsilon n^2, \text{ where } \epsilon = p > 0$$

$$\rightarrow T(n) = \Omega(n^2)$$
- Also correct to say that $T(n) = \Omega(n)$

Tight bounds

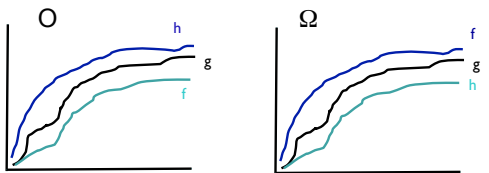
$T(n)$ is $\Theta(f(n))$ if $T(n)$ is both $O(f(n))$ and $\Omega(f(n))$

➤ The "right" bound

Property: Transitivity

- If $f = O(g)$ and $g = O(h)$ then $f = O(h)$
- If $f = \Omega(g)$ and $g = \Omega(h)$ then $f = \Omega(h)$
- If $f = \Theta(g)$ and $g = \Theta(h)$ then $f = \Theta(h)$

Proofs in book



Property: Additivity

- If $f = O(h)$ and $g = O(h)$ then $f + g = O(h)$
- If $f = \Omega(h)$ and $g = \Omega(h)$ then $f + g = \Omega(h)$
- If $f = \Theta(h)$ and $g = \Theta(h)$ then $f + g = \Theta(h)$

Proofs in book

Sketch proof for O:
 By defn, $f \leq c \cdot h$
 By defn, $g \leq d \cdot h$
 $f + g \leq c \cdot h + d \cdot h = (c + d) \cdot h = c' \cdot h$
 $\rightarrow f + g$ is $O(h)$

Practice:
Asymptotic Order of Growth

What are the upper bounds, lower bounds, and tight bound on $T(n)$?

- $T(n) = 32n^2 + 17n + 32$

Practice:
Asymptotic Order of Growth

- $T(n) = 32n^2 + 17n + 32$
 - $T(n)$
 - $O(n^2), O(n^3)$
 - $\Omega(n^2), \Omega(n)$
 - $\Theta(n^2)$
 - $T(n)$ is **not** $O(n), \Omega(n^3), \Theta(n),$ or $\Theta(n^3)$

ASYMPTOTIC BOUNDS FOR CLASSES OF ALGORITHMS

Asymptotic Bounds for Polynomials

- $a_0 + a_1n + \dots + a_dn^d \in \Theta(n^d)$ if $a_d > 0$
 - ➔ Runtime determined by highest-order term
- **Polynomial time.** Running time is $O(n^d)$ for some constant d that is independent of the input size n
- Other examples of polynomial times:
 - $O(n^{1/2})$
 - $O(n^{1.58})$
 - $O(n \log n) \leq O(n^2)$

Asymptotic Bounds for Logarithms

- **Logarithms.** $\log_b n = x$, where $b^x = n$
 - Approximate: To represent n in base- b , need $x+1$ digits

N	b	x
100	10	
1000	10	
100	2	
1000	2	

Asymptotic Bounds for Logarithms

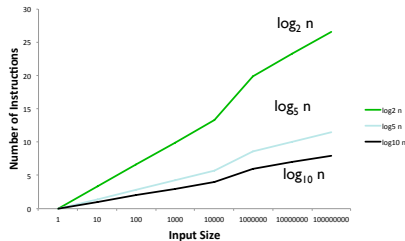
- **Logarithms.** $\log_b n = x$, where $b^x = n$
 - Approximate: To represent n in base- b , need $x+1$ digits

N	b	x
100	10	2
1000	10	3
100	2	6.64
1000	2	9.92

Describe the running time of an $O(\log n)$ algorithm as the input size grows. Compare with polynomials.

Asymptotic Bounds for Logarithms

- **Logarithms.** $\log_b n = x$, where $b^x = n$
 - x is number of digits to represent n in base- b representation



Jan 15

19

Asymptotic Bounds for Logarithms

- **Logarithms.** $\log_b n = x$, where $b^x = n$

➔ Slowly growing functions

- **Identity:** $\log_a n = \log_b n / \log_b a$

➢ Means that

$$\log_a n = 1 / \log_b a * \log_b n$$

Constant!

- $O(\log_a n) = O(\log_b n)$ for any constants $a, b > 0$

Jan 19, 2016

Sprengle - CSCI211

20

Asymptotic Bounds for Logarithms

- **Logarithms.** $\log_b n = x$, where $b^x = n$
 - ➔ Slowly growing functions
- $O(\log_a n) = O(\log_b n)$ for any constants $a, b > 0$
 - ➔ Don't need to specify the base
- For every $x > 0$, $\log n = O(n^x)$
 - ➔ Log grows slower than every polynomial

Jan 19, 2016

Sprengle - CSCI211

21

Asymptotic Bounds for Exponentials

- **Exponentials:** functions of the form $f(n) = r^n$ for constant base r
 - Faster growth rates as n increases
- For every $r > 1$ and every $d > 0$, $n^d = O(r^n)$
 - ➔ Every exponential grows faster than every polynomial

Jan 19, 2016

Sprengle - CSCI211

22

Summary of Asymptotic Bounds

- In terms of growth rates
 - Logarithms < Polynomials < Exponentials**
- Practice comparing functions on next problem set
 - See Chapter 2 solved exercise

Jan 19, 2016

Sprengle - CSCI211

23

Review: Our Process

1. Understand/identify problem
 - Simplify as appropriate
2. Design a solution
3. Analyze
 - Correctness, efficiency
 - May need to go back to step 2 and try again
4. Implement
 - Within bounds shown in analysis

Jan 19, 2016

Sprengle - CSCI211

24

IMPLEMENTING GALE-SHAPLEY ALGORITHM

Jan 19, 2016 Sprenkle - CSCI211 25

Review: Gale-Shapley Stable Matching Algorithm

```

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
    
```

Jan 19, 2016 Sprenkle - CSCI211 26

How Can We Implement The Algorithm Efficiently?

- What is our goal for the implementation's runtime?
- What do we need to model?
- How should we represent them?

Jan 19, 2016 Sprenkle - CSCI211 27

How Can We Implement The Algorithm Efficiently?

- What is our goal for the implementation's runtime?
 - $O(N^2)$
- What do we need to model?
- How should we represent them?

Jan 19, 2016 Sprenkle - CSCI211 28

Stable Matching Implementation


- What do we need to represent?
- How should we represent them?

Data	How represented
Men, Women	
Preference lists	
Unmatched men	
Who men proposed to	
Engagements	

What's the difference between an array and a list?

Jan 19, 2016 Sprenkle - CSCI211 29

Arrays



- *Fixed* number of elements
- What is the runtime of
 - Determining the value of the i^{th} item in the array?
 - Determining if a value e is in the array?
 - Determining if a value e is in the array if the array is sorted?

Jan 19, 2016 Sprenkle - CSCI211 30

Array Operations' Running Times

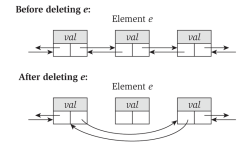
Operation	Running Time
Value of i^{th} item	$O(1)$ → direct access
If e is in the array	$O(n)$ → look through all the elements
If e is in the array if sorted	$O(\log n)$ → binary search

Limitation of arrays?

Fixed size, so difficult to add/delete elements

Lists

- Dynamic set of elements
 - Linked list
 - Doubly linked list
- What is the running time to
 - Add an element to the list?
 - Delete an element from the list?
 - Find an element e in the list?
 - Find the i^{th} element in the list?



List Operations' Running Time

Operation	Running Time
Add element	$O(1)$
Delete element	$O(1)$
Find element	$O(n)$
Find i^{th} element	$O(i)$

Disadvantage of list instead of array?

Finding i^{th} element is slower

Converting between Lists and Arrays (and Vice Versa)

- What is the running time of converting a list to an array?
- An array to a list?

$O(n)$

Stable Matching Implementation

- What do we need to represent? How should we represent them?

Data	How represented
Men, Women	
Preference lists	
Unmatched men	
Who men proposed to	
Engagements	

Stable Matching Implementation

- What do we need to represent? How should we represent them?

Data	How represented
Men, Women	Integers (like ids)
Preference lists	Array of arrays (2d array)
Unmatched men	List
Who men proposed to	Integer for each man → Array of integers
Engagements	2 Arrays

Looking Ahead

- Review Chapter 2
 - [Finishing up on Thursday](#)
- Return surveys – Thursday class
- Problem Set 1 due by 5 p.m. Friday