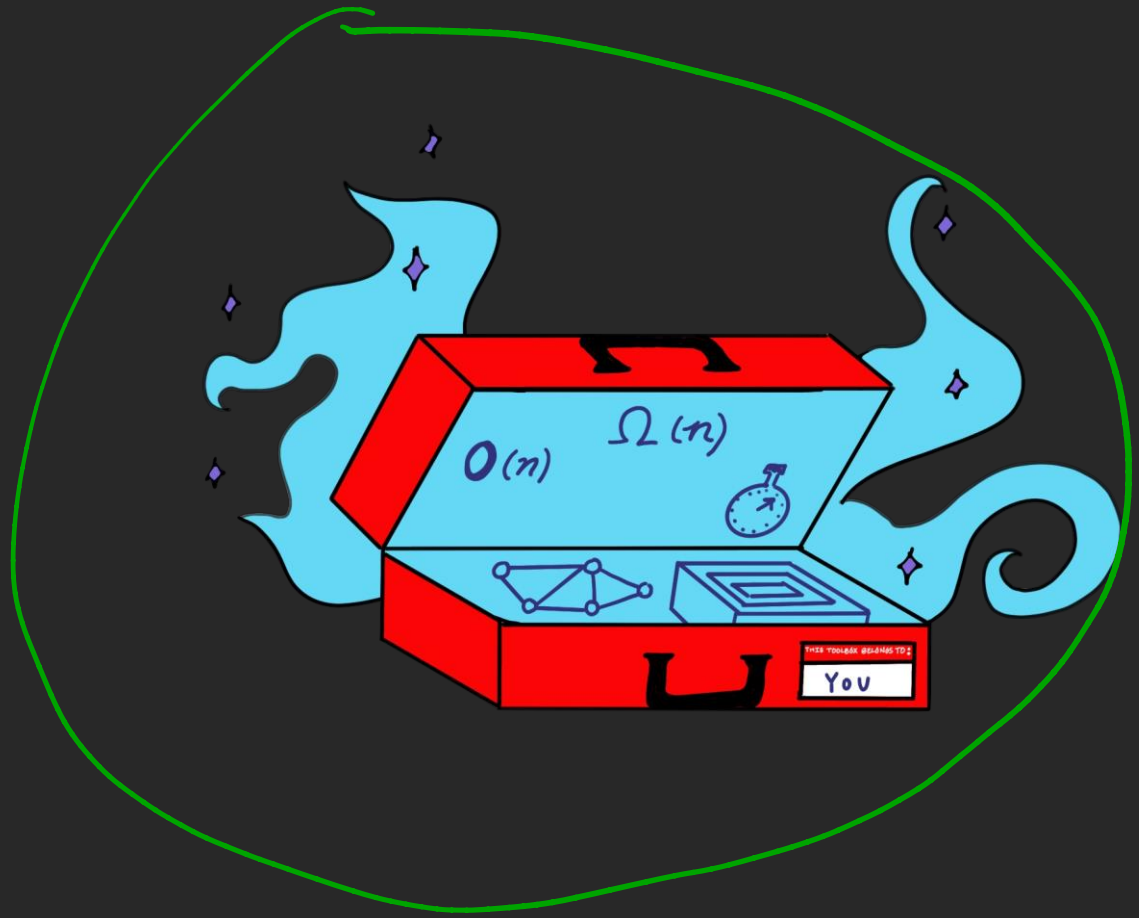
CSCI 3104
Lecture 2: Analyzing Algorithms

Caleb Escobedo

Caleb.Escobedo@colorado.edu

Lecture Outline

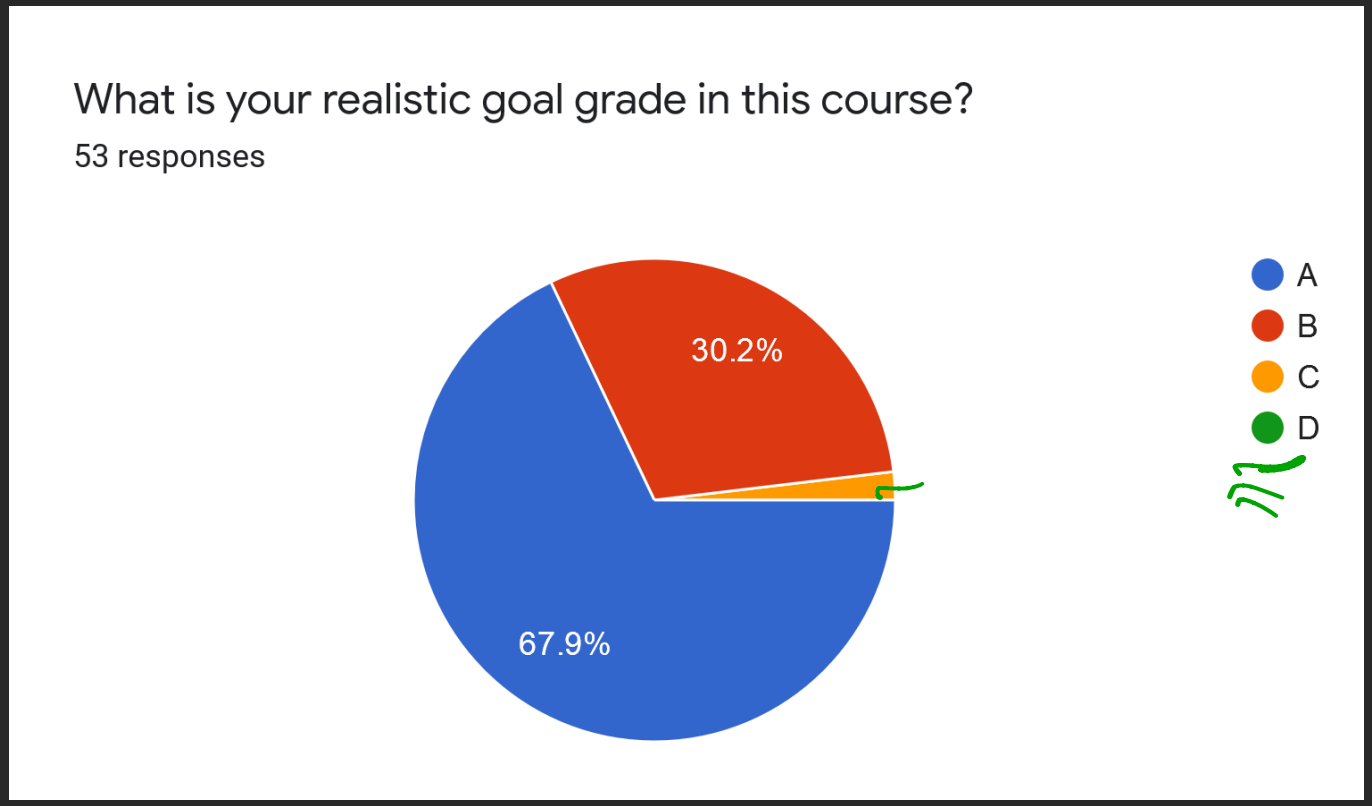
- Poll review
- Abstract model of computation
- Adversarial algorithm analysis
- Asymptotic analysis
  - Input to algorithms
  - Big-O
  - Big-Theta
  - Big-Omega
  - Small-o
  - Small-ω
- Useful functions: Example



Poll Review!

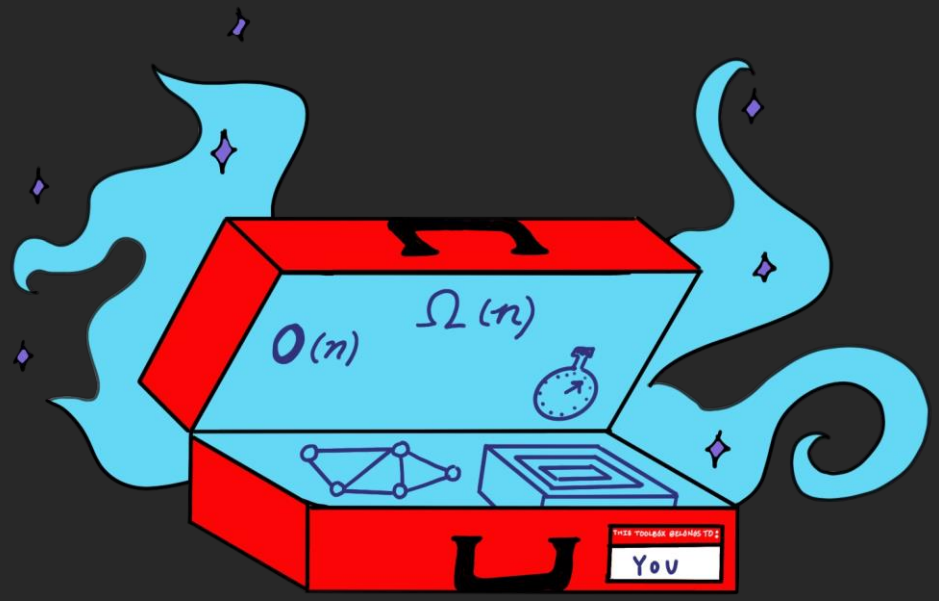
- Reading list – work in progress
- Gradescope: first.lastname@Colorado.edu
  - Entry code: M2V5Y3
- Practice problems in lecture
- Post slides before lecture
- Preferred languages
  - Python – easy to read and code
  - C++ – greater understanding of what the computer is doing

Poll Review!


What is your realistic goal grade in this course?
53 responses
30.2%
67.9%
A
B
C
D

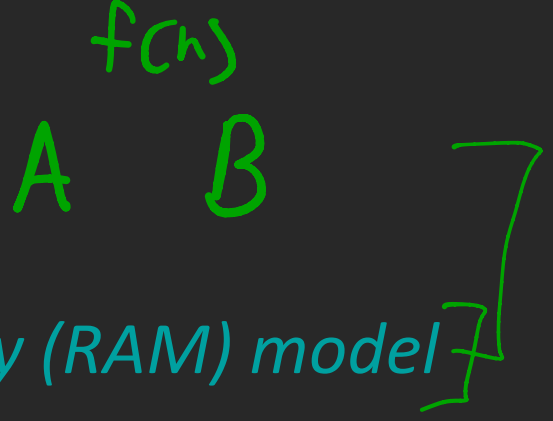
Lecture Outline

- ~~Poll review~~
- Abstract model of computation
- Adversarial algorithm analysis
- Asymptotic analysis
  - Input to algorithms
  - Big-O
  - Big-Theta
  - Big-Omega
  - Small-o
  - Small-ω
- Useful functions: Example



Abstract Model of Computation

- Machine independent
- *Random Access Memory (RAM) model*
  - Atomic operations
  - Higher-order operations
  - Atomic data types
  - Higher-order data types



Atomic Operations

O(1) +, −, =

if, set

Higher-Order Operations

for (i = 0; i <= n; i++) O(n)

f(n)

Atomic Data Types

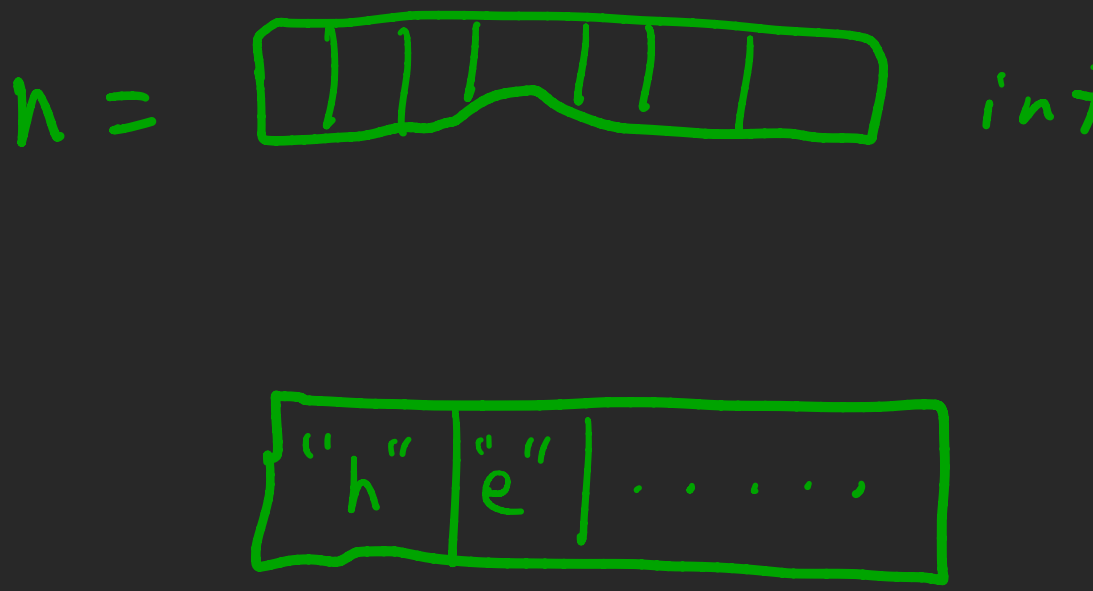
bool 1, 0

int 16

float 1.001

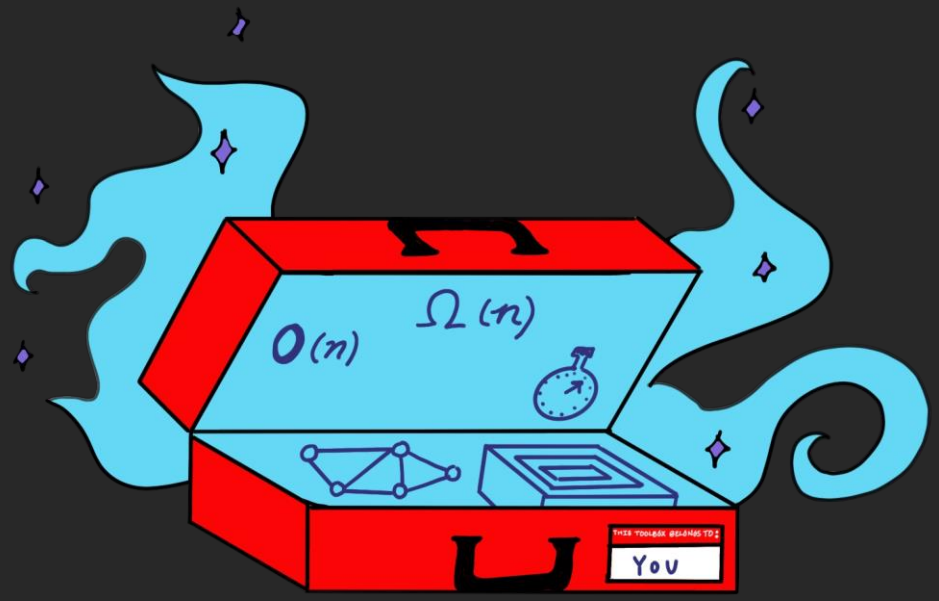
char "c"

Higher-Order Data Types



Lecture Outline

- ~~Poll review~~
- ~~Abstract model of computation~~
- Adversarial algorithm analysis
- Asymptotic analysis
  - Input to algorithms
  - Big-O
  - Big-Theta
  - Big-Omega
  - Small-o
  - Small-ω
- Useful functions: Example



Adversarial Algorithm Analysis

- Worst-case analysis
- Average case
- Best case

```
for (i=0; i<n; i++)
    if (i ≥ n[i])
        for (...)
```

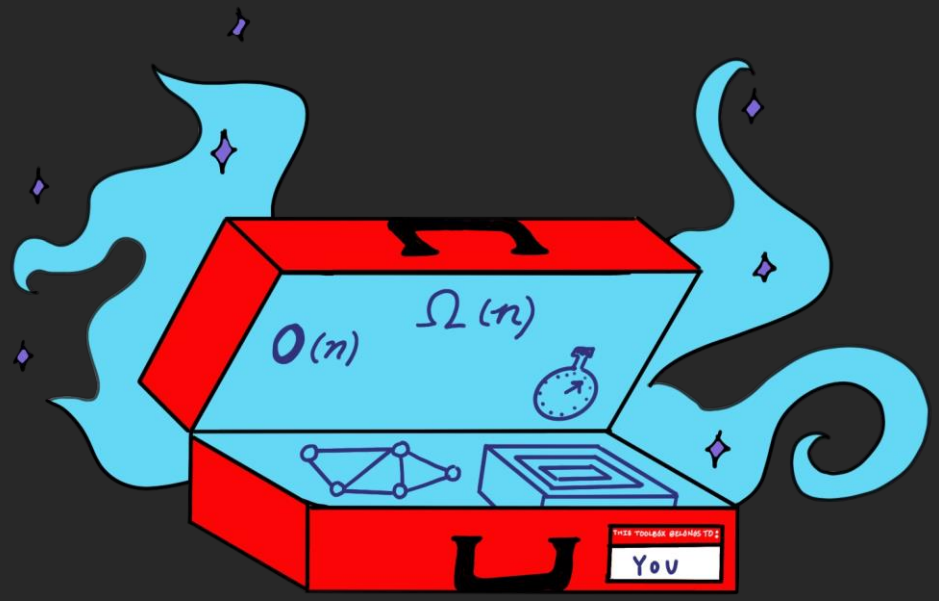
len

- n = 1, 2, 3, 4

  n = -1, -1, -1, -1

Lecture Outline

- ~~Poll review~~
- ~~Abstract model of computation~~
- ~~Adversarial algorithm analysis~~
- Asymptotic analysis
  - Input to algorithms
  - Big-O
  - Big-Theta
  - Big-Omega
  - Small-o
  - Small-ω
- Useful functions: Example

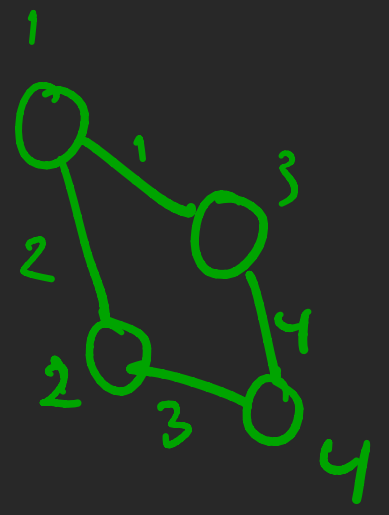


Asymptotic Analysis

$$\lim_{n\to\infty} T(n) = \Gamma(g(n))$$

$g(n)$		meaning
1	:	constant resource usage, independent of n
$\log n$	:	sublinear, specifically logarithmic resource usage
n^c for $c < 1$	:	sublinear resource usage
n	:	linear resource usage
$n \log n$	:	super-linear, but much less than quadratic
n^c for $c > 1$	:	polynomial resource usage, super-linear
c^n for $c > 1$	:	exponential resource usage
n^n	:	extremely fast-growing resource usage

Input to Algorithms



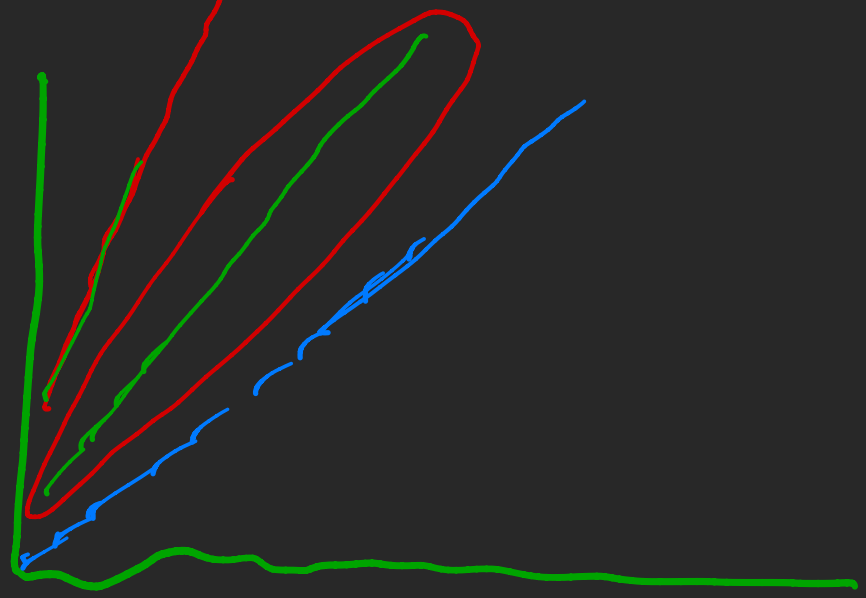
- F(n) = O(n)
- F(n, m) = O(nm), O(n+m)
- Not interested on constant multiplication or addition for runtimes!

$O(2n) = O(n)$

$O(n+2) = O(n)$

- What do we care about again?? Time and Space

Big-Θ (Theta) Tight Bound



$\Theta(g(n)) = \{f(n)$: there exist positive constants c_1, c_2, and n_0 such that $0 \le c_1 g(n) \le f(n) \le c_2 g(n)$ for all $n \ge n_0\}$.[1]

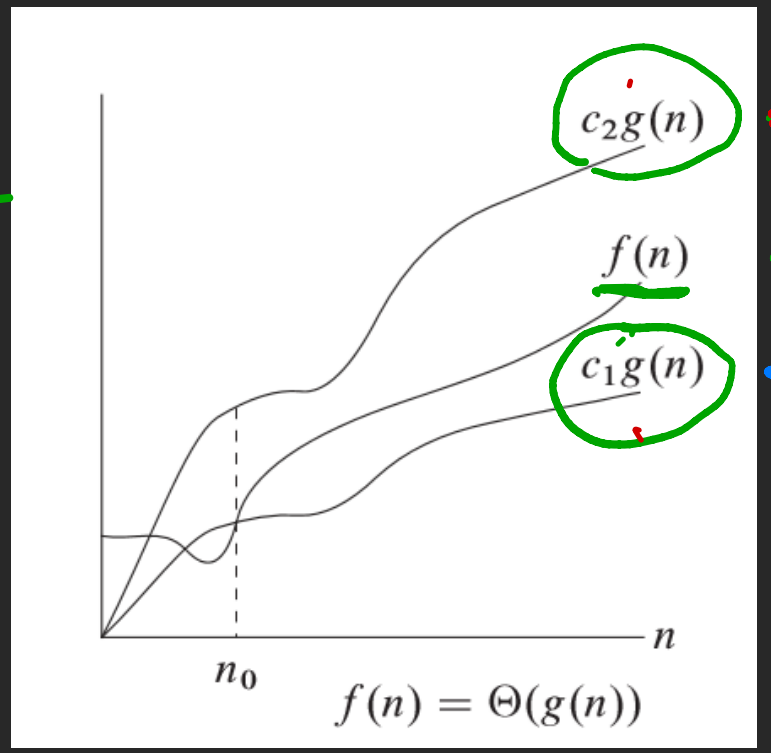
$f(n) = 3n + 1 \quad g(n) = n$

$f(n) = \Theta(g(n))$

$0 \le c_1 n \le 3n + 1 \le c_2 n$

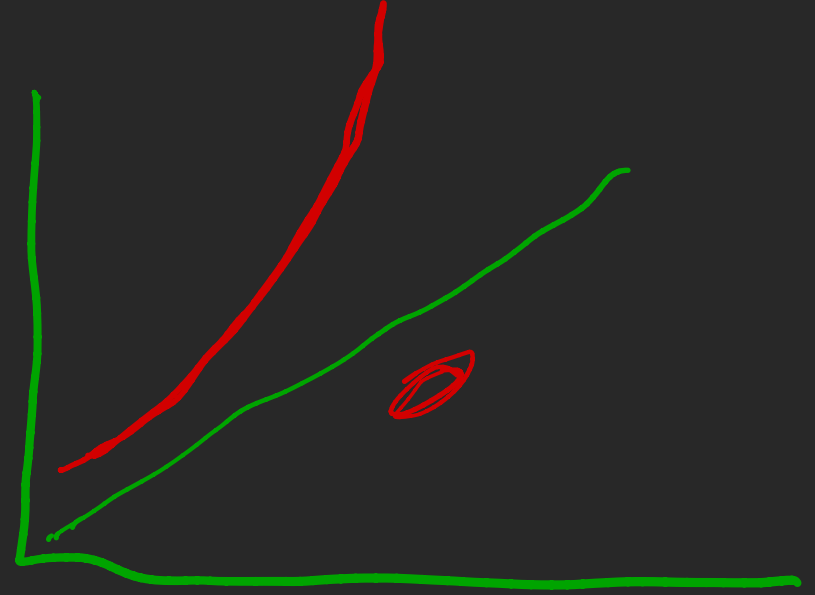
$0 \le c_1 \le 3 + \frac{1}{n} \le c_2, \quad n_0 = 1$

$0 \le c_1 \le 4 \le c_2, \quad c_1 = 1, \quad c_2 = 5$



Big-O Upper Bound

$O(g(n)) = \{f(n)$: there exist positive constants c and n_0 such that $0 \leq f(n) \leq cg(n)$ for all $n \geq n_0\}$.



$f(n) = \Theta(g(n)) \Rightarrow f(n) = O(g(n))$

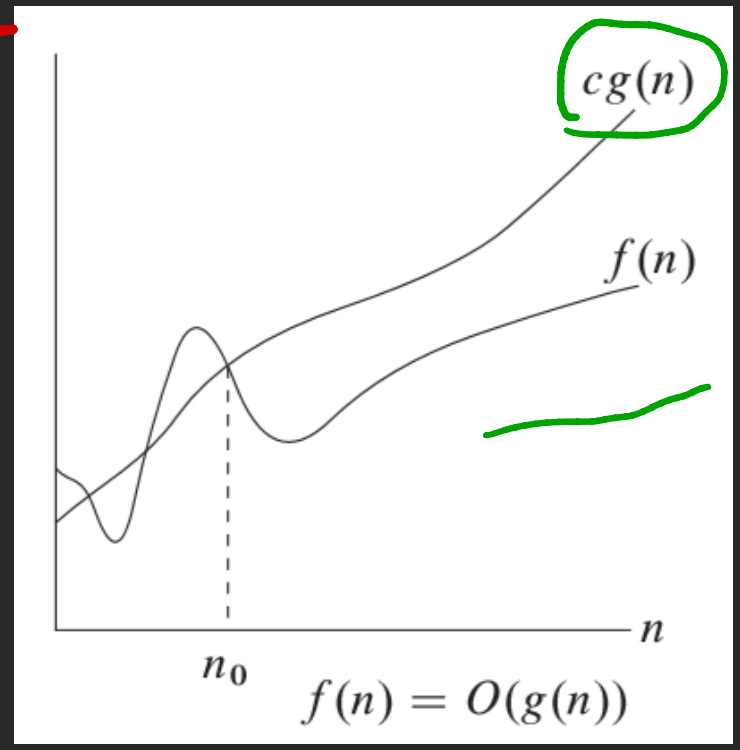
$f(n) = n$

$g(n) = n^3$

$0 \leq n \leq cn^2 \Rightarrow$

$0 \leq 1 \leq cn, \; n = 1$

$0 \leq 1 \leq c$



Small-o Not Asymptotically Tight Upper

$o(g(n)) = \{f(n) :$ for any positive constant $c > 0$, there exists a constant $n_0 > 0$ such that $0 \leq f(n) < cg(n)$ for all $n \geq n_0\}$.

$f(n) = n$

$g(n) = n$ NO

$g(n) = n^2$ yes

$0 \leq n < Cn$

$0 \leq 1 < C$

$0 \leq n < Cn^2$

$0 \leq 1 < Cn$

Big-Ω (Omega) Lower Bound

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

$f(n) = \log_{10}(n^n)$

$g(n) = n$

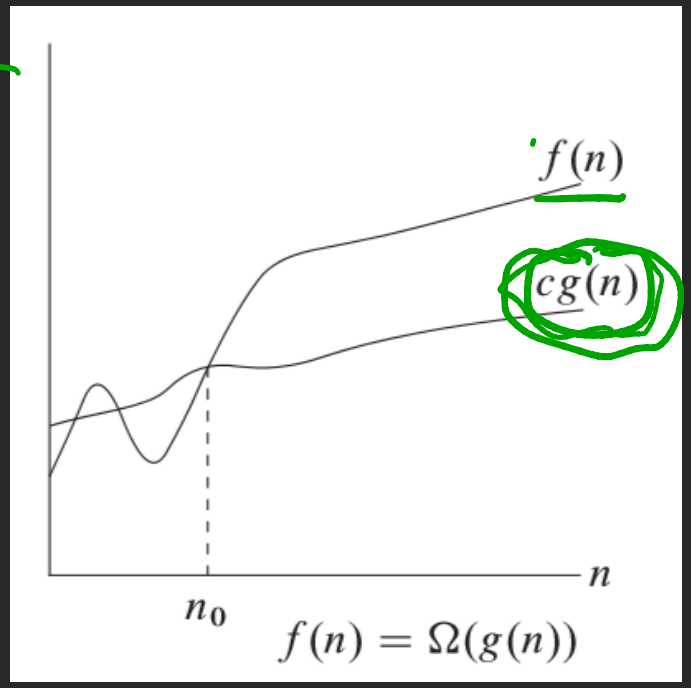
$f(n) = \Omega(g(n))$

$0 \le cn \le \log_{10}(n^n)$

$0 \le cn \le n \log_{10}(n)$

$0 \le c \le \log_{10}(n)$, $n_0 = 10$

$0 \le c \le 1$



Small-ω (Omega) Not Asymptotically Tight Lower

$\omega(g(n)) = \{f(n) :$ for any positive constant $c > 0$, there exists a constant $n_0 > 0$ such that $0 \leq cg(n) < f(n)$ for all $n \geq n_0\}$.

$f(n) = n^3$

$g(n) = n$ yes

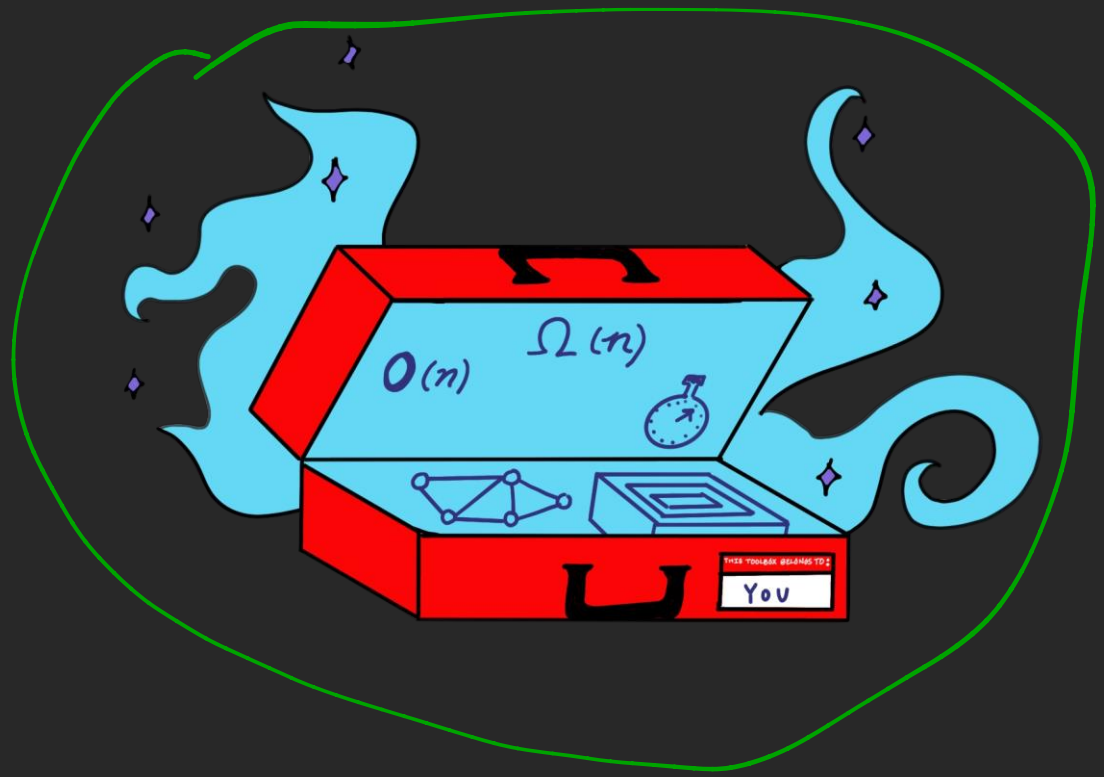
$g(n) = n^3$ NO

$0 \leq cn^3 < n^3$

$0 \leq c < 1$

Lecture Outline

- ~~Poll review~~
- ~~Abstract model of computation~~
- ~~Adversarial algorithm analysis~~
- ~~Asymptotic analysis~~
  - ~~Input to algorithms~~
  - ~~Big O~~
  - ~~Big Theta~~
  - ~~Big Omega~~
  - ~~Small o~~
  - ~~Small ω~~
- Useful functions: Example



Useful functions: Example question!

$f(n) = 4\log_{10}(n^n)$

$g(n) = 2n\log_2(n)$

$f(n) = \Theta(g(n))$

$0 \le C_1 2n\log_2(n) \le 4\log_{10}(n^n) \le C_2 2n\log_2(n)$

$\le 4n\log_{10}(n) \le$

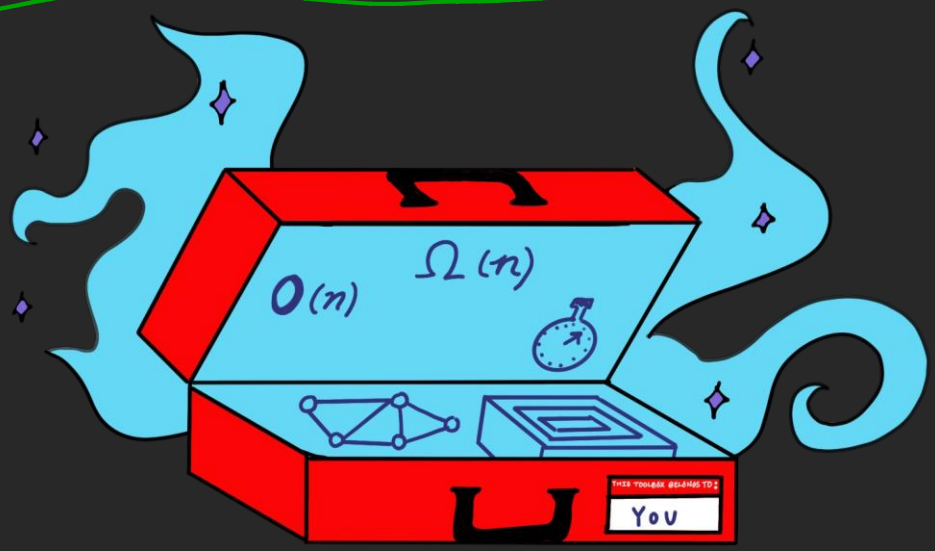
$0 \le C_1\log_2(n) \le 2\log_{10}(n) \le C_2\log_2(n)$

$0 \le C_1 \frac{\log(n)}{\log(2)} \le 2\frac{\log(n)}{\log(10)} \le C_2\frac{\log(n)}{\log(2)}$

1. $(x^y)^z = x^{yz}$
2. $x^y x^z = x^{y+z}$
3. $\log_x y = z \implies x^z = y$
4. $x^{\log_x y} = y$ by definition
5. $\log(xy) = \log x + \log y$
6. $\log(x^c) = c\log x$
7. $\log_c(x) = \log x / \log c$

Closing Thoughts

- Start HW1A early!
- For job interviews, do the extra credit
- Fill out poll!



1. $(x^y)^z = x^{yz}$
2. $x^y x^z = x^{y+z}$
3. $\log_x y = z \implies x^z = y$
4. $x^{\log_x y} = y$ by definition
5. $\log(xy) = \log x + \log y$
6. $\log(x^c) = c \log x$
7. $\log_c(x) = \log x / \log c$