Lecture 14: Asymptotic Growth

CS150: Computer Science
University of Virginia
Computer Science

David Evans
http://www.cs.virginia.edu/evans

Proof Techniques

- Theorem:
  There exists a polygon with four sides.
- Proof:

It is a polygon.
It has four sides.
QED.

What kind of proof is this?

Proof by Construction

- We can prove a "there exists an X with property Y" theorem, but showing an X that has property Y
- $O(f)$ means "**there are** positive constants c and n_0 such that $g(n) \le cf(n)$ **for all** $n \ge n_0$
- So, to prove g is in $O(f)$ we need to find c and n_0 and show that $g(n) \le cf(n)$ **for all** $n \ge n_0$

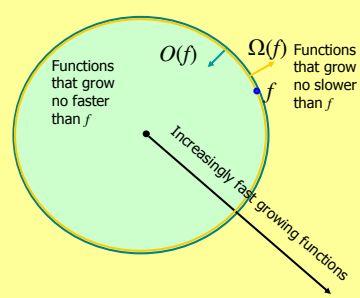
Dis-Proof by Construction

- To prove g is **not** in $O(f)$:
- $O(f)$ means: there are positive constants c and n_0 such that $g(n) \le cf(n)$ for all $n \ge n_0$
- So, to prove g is **not** in $O(f)$ we need to find a way given **any** c and n_0, to find an $n \ge n_0$ such that $g(n) > cf(n)$.

Growth Notations

- $g \in O(f)$ ("Big-Oh")
  g grows **no faster than** f (upper bound)
- $g \in \Theta(f)$ ("Theta")
  g grows **as fast as** f (tight bound)
- $g \in \Omega(f)$ ("Omega")
  g grows **no slower than** f (lower bound)

The Sets $O(f)$ and $\Omega(f)$



O and Ω Examples

g is in $O(f)$ iff there are positive constants c and n_0 such that $g(n) \le cf(n)$ for all $n \ge n_0$.

g is in $\Omega(f)$ iff there are positive constants c and n_0 such that $g(n) \ge cf(n)$ for all $n \ge n_0$.

- n is in $\Omega(n)$
  - Yes, pick $c = 1$
- $10n$ is in $\Omega(n)$
  - Yes, pick $c = 1$
- Is n^2 in $\Omega(n)$?
  - Yes! (pick $c = 1$)

- n is in $O(n)$
  - Yes, pick $c = 1$
- $10n$ is in $O(n)$
  - Yes, pick $c = 10$
- n^2 is **not** in $O(n)$
  - Pick $n > c$

Example

- Is n in $\Omega(n^2)$?

$n \ge cn^2$ for all $n \ge n_0$
$1 \ge cn$ for all $n \ge n_0$

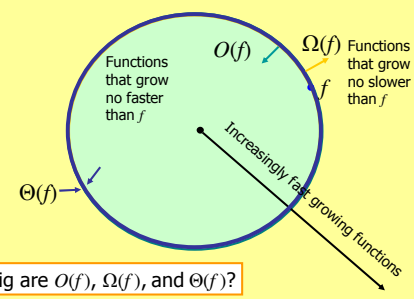
No matter what c is, I can make this false by using $n = (1/c + 1)$

g is in $\Omega(f)$ iff there are positive constants c and n_0 such that $g(n) \ge cf(n)$ for all $n \ge n_0$.

Θ ("Theta"): Tight Bound

g is $\Theta(f)$ iff
g is in $O(f)$
and g is in $\Omega(f)$

The Sets $O(f)$, $\Omega(f)$, and $\Theta(f)$



How big are $O(f)$, $\Omega(f)$, and $\Theta(f)$?

Θ Examples

- Is $10n$ in $\Theta(n)$?
  - Yes, since $10n$ is $\Omega(n)$ and $10n$ is in $O(n)$
    - Doesn't matter that you choose different c values for each part; they are independent
- Is n^2 in $\Theta(n)$?
  - No, since n^2 is not in $O(n)$
- Is n in $\Theta(n^2)$?
  - No, since n^2 is not in $\Omega(n)$

Sorting Cost

- What grows?
  - n = the number of elements in lst
- How much work are the pieces?
  - find-best: work scales as n (increases by one)
  - delete: work scales as n (increases by one)
- How many times does sort evaluate find-best and delete? n
- Total cost: scales as n^2

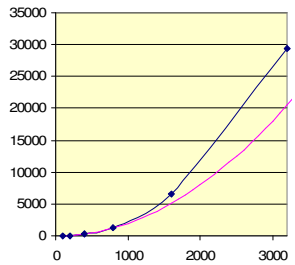
Sorting Cost

```
(define (sort lst cf)
  (if (null? lst) lst
      (let ((best (find-best lst cf)))
        (cons best (sort (delete lst best) cf)))))
(define (find-best lst cf)
  (if (= 1 (length lst)) (car lst)
      (pick-better cf (car lst) (find-best (cdr lst) cf))))
```

If we double the length of the list, the amount of work *approximately* quadruples: there are twice as many applications of find-best, and each one takes twice as long

The running time of this sort procedure is in $\Theta(n^2)$

Timing Sort



measured times

$= n^2/500$

$\Theta(n^2)$

Is our sort good enough?

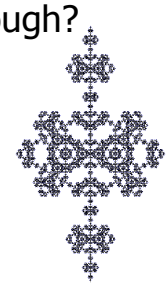
Takes over 1 second to sort 1000-length list. How long would it take to sort 1 million items?

1s = time to sort 1000
4s ~ time to sort 2000

1M is 1000 * 1000

Sorting time is n^2
so, sorting 1000 times as many items will take
1000^2 times as long = 1 million seconds ~ 11 days

Note: there are 800 Million VISA cards in circulation.
It would take 20,000 years to process a VISA transaction at this rate.



Eyes
by John Devor
and Eric Montgomery

Which of these is true?

- ~~Our sort procedure is too slow for VISA because its running time is in $O(n^2)$~~
- Our sort procedure is too slow for VISA because its running time is in $\Omega(n^2)$
- Our sort procedure is too slow for VISA because its running time is in $\Theta(n^2)$

Knowing a running time is in $O(f)$ tells you the running time is not *worse* than f. This can *only* be good news. It doesn't tell you anything about how bad it is. (Lots of people and books get this wrong.)

Charge

- Read Chapter 6 and 7 of the course book
- PS4 is due Monday