O-notation

CS 146 - Spring 2017

Review question

Write a recursive function to clean a hotel

but

at each step, clean the middle room (half way between lo and hi)

Today

- How to measure the efficiency of an algorithm?
- O-notation the definition
- Comparing asymptotic growth of functions
- Simplifying expressions
- Points to be careful about

How to measure the efficiency of an algorithm?

 Method 1: run it on a computer, time it.

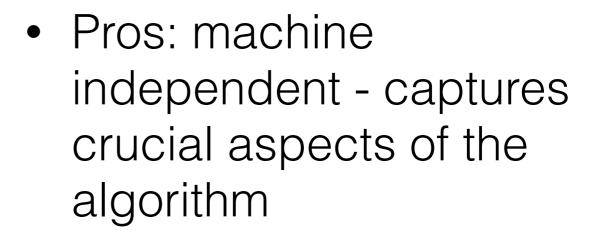


- Cons: machine dependent - duration will depend on the specs of the computer
- Captures more than algorithmic efficiency



How to measure the efficiency of an algorithm?

- Method 2: analyze the algorithm
- ie count number of steps





Cons: counting is hard!

How to cope with messy counting

- n^3 operations is about the same as n^3 1
- n^3 operations is way smaller than e^n

Ignore details

- focus on drastic differences -> asymptotic growth
- machine independent -> ignore constant factors

What is O-notation?

O-notation: discussion

- Definition
- We use it to compare functions
- We use it to simplify complicated expressions

How many meows?

```
void int talk(int n) {
   for (int i = 0; i < n; i++)
       for (int j = 0; j < i; j++)
            meow();
```

Use it in a sentence

- $f(n) \in O(g(n))$ very formal
- |f(n) is O(g(n)) we will use this
- f(n) = O(g(n)) rather informal, but common

Useful math for O-notation

- log rules
- exponent formulas
- Gauss sum formula
- geometric sum formulas

Homework stuff

 What day of the week should be the homework due date?

- Homework will be posted on the course website
- http://www.jennylam.cc/courses/146-s17/
- Homework will be submitted electronically on Canvas

extra slides

How many steps?

```
/*
* Clean rooms numbered lo (inclusive), up to hi (inclusive)
*/
public static void cleanHotel(int lo, int hi) {
    for (int i = lo; i <= hi; i++)</pre>
        System.out.printf("cleaning room %d\n", i);
}
public static void recursiveCleanHotel(int lo, int hi) {
    // base case: when there's one room left
    if (lo == hi) {
        System.out.printf("cleaning room %d\n", lo);
        return;
    // do a little bit of work: clean 1 room
    System.out.printf("cleaning room %d\n", lo);
    // let the recursion do the rest
    recursiveCleanHotel(lo+1, hi);
}
```

Analyzing by counting steps - the fine print

- when deciding what is a step, we are assuming a model of computation
- a model of computation is a simplification of reality
- the model may not be perfect, but we can gain insight from it.

True or false?

$$n^3 + 1 is O(2^n)$$

n is
$$O(n/10 + 1)$$

n is O(n/logn)

O-notation is not a simplification operation!

- true that: often used to simplify functions (like rounding)
- by definition: used to compare functions (think <=)

Fun with O, Theta, Omega

- f(n) is Theta(g(n)) means
 - f(n) is O(g(n)) and g(n) is O(f(n))
- f(n) is Omega(g(n)) means
 - g(n) is O(f(n))