CS112 Lab 05, Feb 18, 22 2010

http://cs-people.bu.edu/deht/cs112_spring11/lab05/

Diane H. Theriault

deht@cs.bu.edu

http://cs-people.bu.edu/deht/

Computational Complexity and Run-time Analysis

- **Which algorithm will be faster?**

- How many steps does your program take to finish, given an input of a particular size?

Computational Complexity and Run-time Analysis

- “Size” of an input depends on context
  - Number of Wallies after N steps
  - Find the smallest number in a list of N items
  - Number of items to sort
  - Length of two strings to compare

Computational Complexity and Run-time Analysis

- Run-time is expressed as a function of the input size (however that is measured)
- These functions are called by their usual math names

1	Constant
log N	Logarithmic
N	Linear
N^2, N^3, N^k	Quadratic, Cubic, Polynomial
2^N	Exponential

Example Linear Algorithm

```
int function(int N)
{
    int numSteps = 0;
    for(int i=0; i<N; i++)
    {
      numSteps ++;
    }
    return numSteps;
}
```

Example Quadratic Algorithm

```
int function(int N)
{
    int numSteps = 0;
    for(int i=0; i<N; i++)
    {
            for(int j=0; j<N; j++)
            {
              numSteps ++;
            }
    }
    return numSteps;
}
```

Example Constant Algorithm

```
int function(int N)
{
    return 1;
}
```

Example Logarithmic Algorithm

```
int function(int N)
{
    int numSteps = 0;
    for(int i=N; i>=1; i/=2)
    {
      numSteps ++;
    }
    return numSteps;
}
```

Example Exponential Algorithm

```
int fibonacci(int N)
{
    if(N == 0 || N == 1) return N;
    return fibonacci(N-1) + fibonacci(N-2);
}
```

“Big O” notation

- Computer science way of writing down the relationships between functions.
- What is the term that “dominates” the function?
- Don’t count
  - “constant factors”
  - “lower order terms”

“Big O” notation

- Run-time of some algorithm: $5\ N^2 + 3N - 5$

- We only care about the term with the largest rate of growth.
- Constant factors are discarded.

- $5\ \mathbf{N^2} + 3N - 5 = O(\mathbf{N^2})$

“Big O” notation

- f(n) = O(g(n)) if (and only if)
- Limit f(n) / g(n) converges
- f(n) and g(n) never differ by more than a constant factor.

Comparing functions

- What does it mean for the limit of a function $h(x)$ to converge?
- As x gets very large, $h(x)$ asymptotically approaches some value
  - $H(x) = 1/x$ converges to 0
  - $h(x) = (4x + 5) / 2x$ converges to 2
  - $h(x) = x$ does not converge!

Comparing functions

- When algebraic manipulation fails, we can use l’Hôpital’s rule to help see the relationship between functions

- limit(f(n) / g(n)) = limit(f’(n) / g’(n))

- Keep applying this until your numerator or denominator is a constant

Example

- $F(x) = 5x^2 + 2x + 3$
- $G(x) = x^2$

Example

- $F(x) = 5x^2 + 2x + 3$
- $G(x) = x^2$
- lim $(5x^2 + 2x + 3)$ / x^2 =
  lim (10x + 2) / 2x =
  lim 10 / 2

Example

- $F(x) = 5x^2 + 2x + 3$
- $G(x) = x^2$
- lim $(5x^2 + 2x + 3)$ / x^2 =
  lim (10x + 2) / 2x =
  lim 10 / 2
- Converges!
- $F(x) = O(G(x))$

Example

- $F(x) = 5x^2 + 2x + 3$
- $G(x) = 20x$

Example

- $F(x) = 5x^2 + 2x + 3$
- $G(x) = 20x$
- $\lim (5x^2 + 2x + 3) / 20x =$
  $\lim (10x + 2) / 20 =$
  $\lim .5x + .1$

Example

- $F(x) = 5x^2 + 2x + 3$
- $G(x) = 20x$
- $\lim (5x^2 + 2x + 3) / 20x =$
  $\lim (10x + 2) / 20 =$
  $\lim .5x + .1$
- Does not converge! $F(x) != O(G(x))$

Example

- $F(x) = 5x^2 + 2x + 3$
- $G(x) = 20x$
- $\lim (5x^2 + 2x + 3) / 20x =$
  $\lim (10x + 2) / 20 =$
  $\lim .5x + .1$
- Does not converge! $F(x) \mathrel{!}= O(G(x))$
- BUT, $\lim 1 / (.5x+.1)$ does converge
  $G(x) = O(F(x))$

Theta (Θ) notation

- Expresses "equality" between functions

- If $f(x) = O(g(x))$
  and $g(x) = O(f(x))$
- Then, $f(x) = \Theta(g(x))$

- $10x^2 = \Theta(5x^2 + 2x + 3) = \Theta(100x^2 + 10x)$

Logarithms and Exponentials

- Logarithmic and exponential functions are very important in computer science (usually can assume base 2)
- Some simple rules are useful for manipulating them:

$2^a * 2^b = 2^{a+b}$ $2^a / 2^b = 2^{a-b}$	$(2^a)^b = 2^{ab}$
$\log(a*b) = \log(a) + \log(b)$ $\log(a/b) = \log(a) - \log(b)$	$\log(a^b) = b \log(a)$
$\log_a a^n = n$ $\log 2^n = n$	$a^{\log_a n} = n$ $2^{\log n} = n$

A fun party trick

- How high can you count on your fingers?
- It’s not ten.

Worked examples

- 1 vs 1037
- $n^3 + 2n^2 + 1000$ vs n^5
- log n vs log log n
- 2^{2n} vs 2^n

On your homework:

- Use the tools we talked about today
- Put each “O” on a separate line
- Put all the Θ’s on the same line
- You don’t need to submit proofs