Lecture 5

Algorithms & Data Structures

Goldsmiths Computing

October 29, 2018

Outline

Introduction

Growth of functions

Recursion

Outline

Introduction

Growth of functions

Recursion

Lecture

- Data structures!
  - linked lists
  - stacks
  - queues
- How do operations scale?
- Does it matter how they are implemented?

Lab

- Be a data structure implementor
  1. implement linked lists
  2. implement methods on linked lists

VLE activities

Stacks and queues quiz

Statistics so far:

- 289 attempts: average mark 6.77
- 91 students: average mark 7.30
  - 11 under 4.00, 58 over 6.99, 18 at 10.00

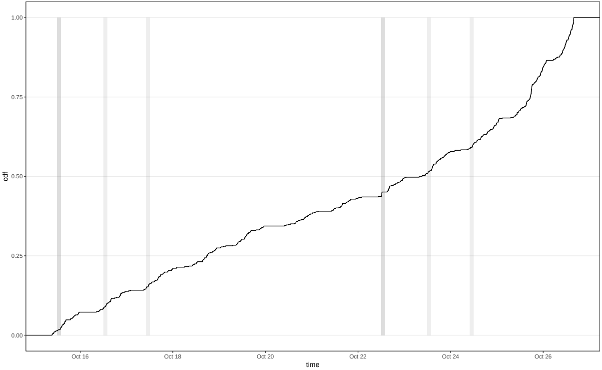
Quiz closes at 16:00 on Friday 2nd November

- no extensions
- grade is
  - 0 (for no attempt)
  - $30 + 70 \times (\text{score}/10)^2$

VLE activities (cont'd)

Big-O quiz

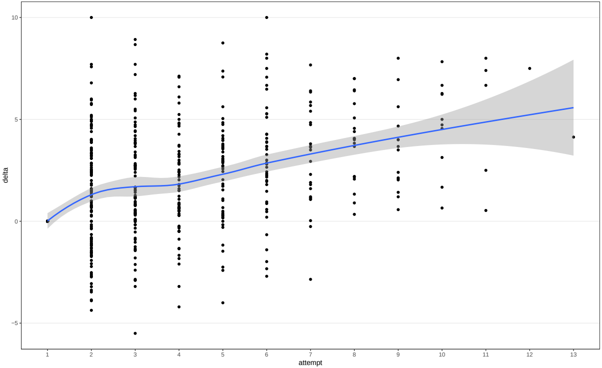
- 579 attempts: average mark 5.75
- 135 students: average mark 7.74
    - 11 under 4.00, 92 above 6.99, 36 at 10



VLE activities (cont'd)

Big-O quiz

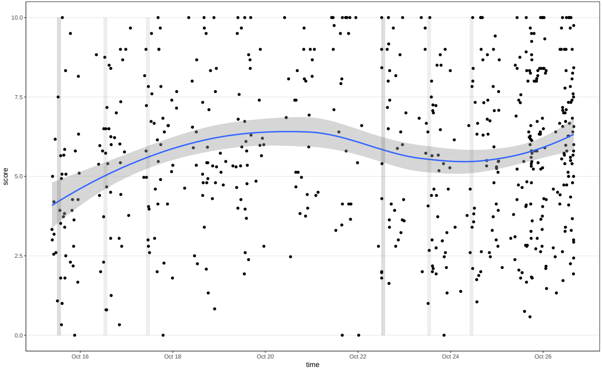
- 579 attempts: average mark 5.75
- 135 students: average mark 7.74
    - 11 under 4.00, 92 above 6.99, 36 at 10



VLE activities (cont'd)

Big-O quiz

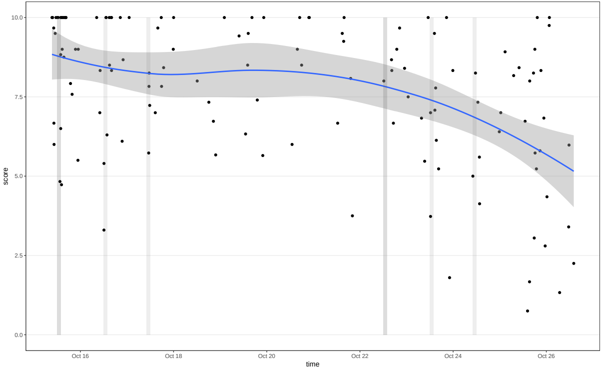
- 579 attempts: average mark 5.75
- 135 students: average mark 7.74
  - 11 under 4.00, 92 above 6.99, 36 at 10



VLE activities (cont'd)

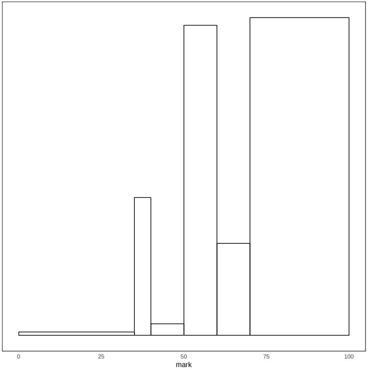
Big-O quiz

- 579 attempts: average mark 5.75
- 135 students: average mark 7.74
    - 11 under 4.00, 92 above 6.99, 36 at 10



VLE activities (cont'd)

- 126 final uploads: average mark 83.26



Outline

Motivation

Turning empirical measurements into scaling hypotheses, or *vice versa*

Common functional classes

1. power-law
2. logarithmic and linear-logarithmic
3. exponential

Power-law

$$f(n) \propto n^k$$

$$f(n) = An^k$$

Power-law

$$f(n) \propto n^k$$

$$f(n) = An^k$$

Given $f(n_1)$ and $f(n_2)$, estimate k (and A):

$$\frac{f(n_1)}{f(n_2)} = \left(\frac{n_1}{n_2}\right)^k$$

Power-law

$$f(n) \propto n^k$$
$$f(n) = An^k$$

Given $f(n_1)$ and $f(n_2)$, estimate k (and A):

$$\frac{f(n_1)}{f(n_2)} = \left(\frac{n_1}{n_2}\right)^k$$

$$k = \frac{\log\left(\frac{f(n_1)}{f(n_2)}\right)}{\log\left(\frac{n_1}{n_2}\right)}$$

Logarithmic

$$f(n) \propto \log(Bn)$$

$$f(n) = A \log(Bn)$$

Logarithmic

$$f(n) \propto \log(Bn)$$

$$f(n) = A\log(Bn)$$

Given $f(n_1)$ and $f(n_2)$, estimate A (and B):

$$f(n_1) - f(n_2) = A(\log(n_1) - \log(n_2))$$

Logarithmic

$$f(n) \propto \log(Bn)$$

$$f(n) = A \log(Bn)$$

Given $f(n_1)$ and $f(n_2)$, estimate A (and B):

$$f(n_1) - f(n_2) = A(\log(n_1) - \log(n_2))$$

$$A = \frac{f(n_1) - f(n_2)}{\log(n_1) - \log(n_2)}$$

Exponential

$$f(n) \propto 2^{cn}$$

$$f(n) = A2^{cn}$$

Exponential

$$f(n) \propto 2^{cn}$$

$$f(n) = A2^{cn}$$

Given $f(n_1)$ and $f(n_2)$, estimate c (and A):

$$\log(f(n_1)) - \log(f(n_2)) = c(n_1 - n_2)$$

Exponential

$$f(n) \propto 2^{cn}$$

$$f(n) = A2^{cn}$$

Given $f(n_1)$ and $f(n_2)$, estimate c (and A):

$$\log(f(n_1)) - \log(f(n_2)) = c(n_1 - n_2)$$

$$c = \frac{\log(f(n_1)) - \log(f(n_2))}{n_1 - n_2}$$

Work

1. Do growth of functions quiz
    - open until 16:00 9th November 2018
    - no extensions

Outline

Motivation

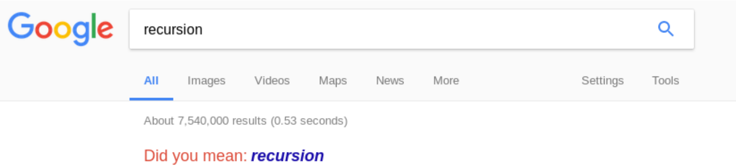
A way to describe solutions of problems that makes them

- easy to prove correct
- easy to compute how they scale

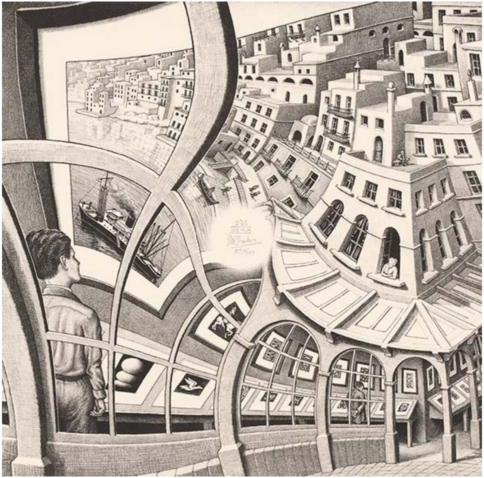
Definition

The definition of a problem or solution in terms of (variant forms of) itself

Illustration



Illustration



– M.C. Escher, *Print Gallery* (1956)

Ingredients

base case non-recursive condition (possibly more than one)

recursive steps rules reducing the problem towards the base case

Examples

factorial $n! = n \times (n-1)!$ and $0! = 1$

fibonacci numbers $F(n) = F(n-1) + F(n-2)$ and $F(0) = 0$, $F(1) = 1$

Examples

factorial $n! = n \times (n-1)!$ and $0! = 1$

fibonacci numbers $F(n) = F(n-1) + F(n-2)$ and $F(0) = 0$, $F(1) = 1$

Tower of Hanoi audience participation!!

Examples: list algorithms

Search

Is the object o present in the list l?

base case is the object o present in the list NIL?

Examples: list algorithms

Search

Is the object o present in the list l?

base case is the object o present in the list NIL?

recursive step is the object o equal to the first element of the list? If not, is it in the rest of the list?

Examples: list algorithms

Selection

Return the maximum of the objects in the list l

base case what is the maximum element of the empty list?

Examples: list algorithms

Selection

Return the maximum of the objects in the list l

base case what is the maximum element of the empty list?

alternative base case what is the maximum element of a list with one element?

Examples: list algorithms

Selection

Return the maximum of the objects in the list l

base case
: what is the maximum element of the empty list?

alternative base case
: what is the maximum element of a list with one element?

recursive step
: how does the first element compare with the maximum of the rest of the list?

Examples: list algorithms

Selection

Return the k^{th} biggest of the objects in the list l

base case what is the k^{th} biggest element of a list with k elements?

Examples: list algorithms

Selection

Return the k^{th} biggest of the objects in the list l

base case	what is the k^{th} biggest element of a list with k elements?
recursive step	how does the first element compare with the k^{th} biggest element of the rest of the list?

Examples: list algorithms

Selection

Return the k^{th} biggest of the objects in the list l

base case what is the k^{th} biggest element of a list with k elements?

recursive step how does the first element compare with the k^{th} biggest element of the rest of the list?

base case, second try what are the k^{th} biggest elements of a list with k elements?

Examples: list algorithms

Selection

Return the k^{th} biggest of the objects in the list l

base case what is the k^{th} biggest element of a list with k elements?

recursive step how does the first element compare with the k^{th} biggest element of the rest of the list?

base case, second try what are the k^{th} biggest elements of a list with k elements?

recursive step, second try how does the first element compare with the k^{th} biggest elements of the rest of the list?

Work

1. Reading
    - CLRS, section 2.3

Onward

1. This week:
    - stacks and queues quiz deadline
2. Next week:
    - growth of functions deadline
    - linked lists lab submission
    - reading
    - practice