Recommended book: * Introduction to Algorithms [CRLS] Additional: Algorithm Design [KT]

Assessment

20% coursework in week five or six.

80% summer exam

Introduction

First, consider the inputs that your algorithm will receive and then what the output should look like.

Multiplication

School algorithm

This algorithm takes each individual digit in the second number and multiplies it with the each digit of the first number.

This means that, for two numbers of length 'n', the number of multplications you must do at least n x n multplications. As a result, the algorithm running time scales quadratically. See L1 notes for Konrad's explanation.

See Kolmogorov's conjectured that there is no algorithm shorter than the n x n running time of the school algorithm. In other words, that it was asymptotically optimal.

A week later, Karatsuba found his eponymous algorithm which is also referred to as *divide-and-conquer*.

Al-Khwarizmi is the of the Persian mathematician who derived the arabic numeral system. This became algorisme to describe the system in French and eventually came to describe an executable series of instructions.

Divide and conquer

Take the n digit numbers and split them into two n/2 digit numbers.

This is a recursive algorithm as it calls itself during its running.

For two numbers of length n (assuming n is a power of two), T(n) = 4 x T(n/2) + k x n for some k $\in \ltimes$

So T(n) = 4 T(n/2) + kn = 4^2 T(n/2^2) + 3kn ... = 4^c T(n/2^c) + (2^c - 1)kn = n^2 (T(1) + k) - kn

Big-O notation

See lecture notes for the correct algebra for assertaining the Big-O time of an algoritm.

Sorting algorithms

Merge sort

Conceptually, a merge sort works as follows:

- Divide the unsorted list into n sublists, each containing one element (a list of one element is considered sorted).
- Repeatedly merge sublists to produce new sorted sublists until there is only one sublist remaining. This will be the sorted list.

Insertion sort

nsertion sort iterates, consuming one input element each repetition, and growing a sorted output list. At each iteration, insertion sort removes one element from the input data, finds the location it belongs within the sorted list, and inserts it there. It repeats until no input elements remain.

Insertion sort is fast for small lsits because the algorithm is simple.

Loop invariants

The algorithmic analog of proof by induction.

for example, with insertion sort: Induction hypothesis: At the start of the i-th iteration, A[0:i-1] is sorted.

Induction step: i implies i + 1

See notes/slides for induction technique.

There are three key steps: * Initalisation: it is true prior to the first iteration of the first loop. * Maintenance * Termination

Measuring running time

Slide in lec 2

Worst case analysis