50001 - Algorithm Analysis and Design - Lecture 1

Oliver Killane

12/11/21

Lecture Recording

Lecture recording is available here

Introduction

An algorithm is a method of computing a result for a given problem, at its core in a systematic/-mathematical means.

This course extensively uses haskell instead of pesudocode to express problems, though its lessons still apply to other languages.

Fundamentals

Insertion Problem

Given an integer x and a sorted list ys, produce a list containing $x : ys$ that is ordered.

Note that this can be solved by simply using $sort(x : ys)$ however this is considered wasteful as it does not exploit the fact that ys is already sorted.

An example algorithm would be to traverse ys until we find a suitable place for x

```haskell
insert :: Int -> [Int] -> [Int]
insert x [] = [x]
insert x yss@(y:ys)
    | x <= y    = x:yss
    | otherwise = y : insert x ys
```

Call Steps

In order to determine the complexity of the function, we use a **cost model** and determine what steps must be taken.

For example for $insert\ 4\ [1, 3, 6, 7]$

$insert\ 4\ [1, 3, 6, 7]$
$\rightsquigarrow$ { definition of $insert$ }
$1 : insert\ 4\ [3, 6, 7]$
$\rightsquigarrow$ { definition of $insert$ }
$1 : 3 : insert\ 4\ [6, 7]$
$\rightsquigarrow$ { definition of $insert$ }
$1 : 3 :: 4 : [6, 7]$

Hence this requires 3 call steps.

We can use recurrence relations to get a generalised formula for the worst case (maximum number of calls):

$$\begin{aligned} T_{insert} 0 &= 1 \\ T_{insert} 1 &= 1 + T_{insert}(n-1) \end{aligned}$$

We can solve the recurrence:

$$\begin{aligned}
T_{insertn} &= 1 + T_{insert}(n-1) \\
T_{insertn} &= 1 + 1 + T_{insert}(n-2) \\
T_{insertn} &= 1 + 1 + \cdots + 1 + T_{insert}(n-n) \\
T_{insertn} &= n + T_{insert}(0) \\
T_{insertn} &= n + 1
\end{aligned}$$

More complex algorithms

```
isort :: [Int] -> [Int]
isort []      = []
isort (x:xs) = insert x (isort xs)
```

$$\begin{aligned}
T_{isort}0 &= 1 \\
T_{isort}n &= 1 + T_{insert}(n-1) + T_{sort}(n-1)
\end{aligned}$$

Hence by using our previous formula for *insert*

$$T_{isort}n \quad = 1 + n + T_{sort}(n-1)$$

And by recurrence:

$$\begin{aligned}
T_{isort}n &= 1 + n + (1 + n - 1) + T_{isort}(n-2) \\
T_{isort}n &= 1 + n + (1 + n - 1) + (1 + n - 2) + \cdots + T_{isort}(n-n) \\
T_{isort}n &= 1 + n + (1 + n - 1) + (1 + n - 2) + \cdots + T_{isort}(0) \\
T_{isort}n &= n + n + (n-1) + (n-2) + \cdots + (n-n) + 1 \\
T_{isort}n &= 1 + n + \textstyle\sum_{i=0}^{n} i \\
T_{isort}n &= \textstyle\sum_{i=0}^{n+1} i \\
T_{isort}n &= \frac{(n+1) \times (n+1)}{2}
\end{aligned}$$