Design and Analysis of Algorithms

G. Philip & Samir Dutta

Contents

1 Lecture 1

Read the moodle page of this course for list of TAs, book recommendations and other stuffs.

1.1 What are algorithms

A series of steps which takes inputs from some well defined sources and gives an output which has some relation with the input

1.2 Properties of Algorithm

1. ***Input:*** Zero or more objects from a well-defined set.
2. ***Output:*** One or more objects from a well-defined set must have a specifies relation to inputs.
3. ***Finiteness:*** Runs in a finite amount of time.
4. ***Definiteness:*** Each step is well-defined (no ambiguity).
5. ***Effectiveness:*** Each steps is suffieciently well-defined that it in principle can be executed in pen and paper.

Example which has (4) but not (5): If Reimann Hypothesis is true return yes else no

1.3 Euclidean Division

Input: Non-negative integer x, positive integer y - Both given as binary number

Output: The quotient q and remainder r when x is divided by y

$x = (q \times y) + r$

```
Divide(x,y):
    if x = 0:
        return (0,0)
    z = x // 2
    (q,r) = Divide(z,y)
    q = 2 * q
    r = 2 * r
    if x is odd:
        r = r + 1
    if r >= y:
        r = r - y
        q = q + 1
    return (q,r)
```

Base Case: $x = 0$

Inductuve Assumption: `Diviede(x,y)` returns the correct value of q & r when $x \leq k$ for some $k \geq 0$

Inductuve Stp Assuming the above hypothesis prove that `Divide(x,y)` returns correct values for $x = k+1$ [Exercise]

1.4 Exercises

1. Proof the above algorithm with induction on the number of bits of x
2. Write pseudocode for a recursive function OddOrEven(x) which takes a positive integer x as argument and returns whether x is even or odd.
3. Consider the following set of operations to compute a number x:
    1. Initialization $x = 1$
    2. Do either one of the follwong steps, a finite number of times.
        2.1. $x = x \times 2$

2.2. $x = x + 3$

These steps can be interleaved. You can choose (2.1) in one steo, (2.2) in the next step Write the pseudo code for a recursive function `IsValid(y)` that takes a positive integer y as argument and returns

- True if y can be by some finite sequence of these steps, starting from $x = 1$
- False otherwise

Prove using induction that your function is correct.