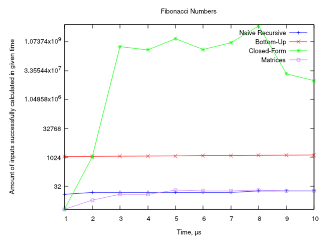
Homework 5

Rumen Mitov

March 16, 2024

Problem 5.1



For the results table please first build and run the program with **make all**, you will then find the table in the file *build/results.txt*.
The function *closed_form* will not always produce the same answers as the other functions because it uses floating point arithmetic, hence if the input variable, n, is large enough precision could be lost. Thus, an incorrect answer will be returned.

Problem 5.2

A brute force implementation of multiplication is just addition in repetition. Basically we are adding **b** to the result, **a** times. Since addition happens in $\Theta(n)$ and we are repeating this step **a** times, we can conclude that the overall time complexity of this operation will be $\Theta(n^2)$.

Algorithm 1 Multiplication(a, b)

---

if $a = 0$ or $b = 0$ **then return** 0
end if

if $a = 1$ **then return** b
end if

if $b = 1$ **then return** a
end if
return $Multiplication(\text{a}, (1 \text{<<b})) + Multiplication(\text{a}, (1 \text{<<b}))$

Since we are splitting b into 2 before the recursion step and addition and bit shifting both take $Theta(n)$ time, we estimate the time complexity to be:
$T(n) = 2 \cdot T(\frac{n}{2}) + \Theta(n)$

Using the recursion tree method, we know that our recursion tree will have a height of $\log n$, hence on each level of the tree we will have time complexity of $\Theta(n)$ due to the bit shifting and addition (after dropping constants), hence: $T(n) = \Theta(n \log n)$

Using the Master Theorem:
$a = 2$
$b = 2$
$\log_b(a) = 1 \implies n^{\log_b(a)} = n$
Since $n^{\log_b(a)} = f(n) = n$ we can conclude that: $T(n) = \Theta(n \log n)$