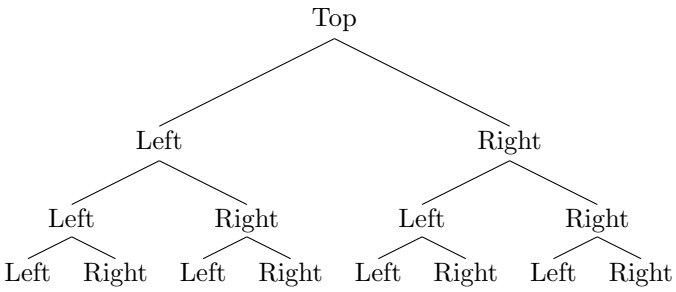

Top
Left
Right
Left
Right
Left
Right
Left
Right
Left
Right
Left
Right
Left
Right

TDT4120
Fall 2018

Henry S. Sjoen

September 8, 2018

Contents

1 Algorithm Design

1.1 Divide and Conquer

Divide and Conquer is an Algorithm design paradigm based on multi-branch recursion. A divide and conquer algorithm works by recursively breaking down a problem into two or more sub-problems of the same or related type, until these become simple enough to be solved directly. The solutions to the sub-problems are then combined to give a solution to the original problem.

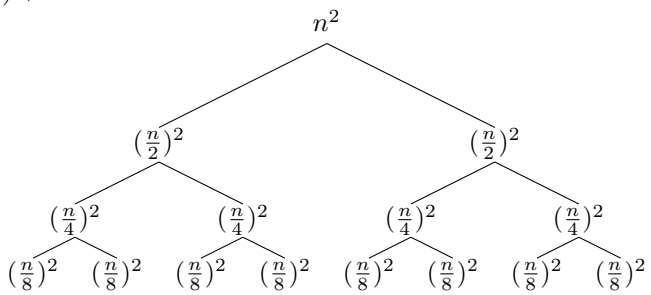
2 Loose Recurrences

2.1 The Master-Theorem

Generic form $T(n) = aT(\frac{n}{b}) + f(n)$

2.2 Recursion Trees

$T(n) = 4T(\frac{n}{2}) + n^2$



This is supposed to be a tree...

n^2
−(n/2)^2
−−(n/4)^2
−−(n/4)^2
−−(n/4)^2
−−(n/4)^2
−(n/2)^2
−−(n/4)^2
−−(n/4)^2
−−(n/4)^2
−−(n/4)^2
−(n/2)^2
−−(n/4)^2
−−(n/4)^2
−−(n/4)^2
−−(n/4)^2
−(n/2)^2
−−(n/4)^2
−−(n/4)^2
−−(n/4)^2
−−(n/4)^2

2.3 Variable-switching

$T(n) = 2T(\sqrt{n}) + \log n$

3 Sorting Algorithms

3.1 Mergesort

$\theta(n \log n)$

3.2 Quicksort

Howto:

Listing 1: Julia example

```
function traverse_recursive_max(node, start_value)
    highest_value = start_value
    if (node.value > highest_value)
        highest_value = node.value
    end
    if node.next == nothing
        return highest_value
    end
    traverse_recursive_max(node.next,highest_value)
end

traversemax(node) = traverse_recursive_max(node,node.value)
```

4 Code example overview

Listings