Divide and Conquer

Vitor R. Greati*

December, 2015

1 Preliminaries

The divide-and-conquer approach usually falls on recursive solutions, and is based on three main steps:

Divide Divide the problem into smaller instances of the same problem.

Conquer Solve each minor instance using a straightfoward manner.

Combine Combine those solutions to finally solve the original problem.

Given the recursive aspect, **recurrences** are the most common ways to define functions for counting the total of operations performed by the solutions. They simply use smaller inputs to describe a function, which means that there is not a direct definition, but a description based on known outputs of that same function.

Three methods can be applied for solving recurrences:

- Substitution method
- Recursion-tree method
- Master method

Sometimes, details are ignored when formulating recurrences. For example, on Merge Sort algorithm's, there are floor and ceiling functions involved, but they turn out to disappear when theta notation is applied. Otherwise, there are some cases where this simplification can't occur.

*greati@ufrn.edu.br —— vitorgreati.me

2 Maximum-subarray problem

From Cormen: in a sequence of days, each day has a certain value, and a customer can buy and sell a thing each day according to that value. Which pair of days start, end would give the greatest profit at the end?

For this problem, we can have a brute-force solution, testing all pairs {begin, end}, giving $\Omega(n^2)$ of complexity.

Running away from this approach, a better way of thinking this problem is keeping in hands the change between days, not the value at a day. With that, the problem becomes finding the greatest sum of contiguous elements. In other words, it's the **maximum subarray problem**. Still, there's a $\emptyset(n^2)$ trivial solution, but a divide-and-conquer approach can produce a linear solution!