CS 341 — Lecture 5

Bartosz Antczak *Instructor: Bin Ma* *September 21, 2017*

5.1 More on Divide and Conquer

Example 5.1.1. *Closest Pair*

Given a set of n points on a plane, defined as $P_i = (x_i, y_i)$, return the shortest distance between any two points.

Algorithm I

The trivial algorithm: compare every pair's distance; runs in $O(n^2)$. When designing an algorithm, always start with the trivial one.

Algorithm II

Let's see if we can aim for $O(n \log n)$. We'll use divide and conquer!

```
Input P
Sort P into Px and Py

Input Px, pY
Split Px into Qx, Rx
Split Py into Qy, Ry (carefully)
Solve(Qx, Qy)
Solve(Rx, Ry)

Assume our given distance d is the minimum
Select points in Py within +-d of median line. Put in B
Solve(B)
```

This algorithm yeilds a runtime of $T(n) = 2 \cdot T(\frac{n}{2}) + O(n) \in O(n \log n)$.

Example 5.1.2. *Integer Multiplication*

Assume x, y are numbers that represent n bits. n is also very large. How would we multiply these two numbers together?

Algorithm I

Trivial multiplication. Takes $O(n^2)$ time.

Algorithm II

Let's try using a divide and conquer technique! Let $n = 2m$ (if n is odd, just add a trailing 0). We then split our number x represented as a bit array into two halves x_1 and x_2. We define

$$x = 2^m \cdot x_1 + x_2$$

and we apply the same procedure for y. We now have

$$\begin{aligned} x \cdot y &= (2^m \cdot x_1 + x_2)(2^m \cdot y_1 + y_2) \\ &= x_2 y_2 + 2^m (x_1 y_2 + x_2 y_1) + 2^{2m} x_1 y_1 \end{aligned}$$

Here we perform four simpler multiplication operations. By the Master Theorem, we have a run time of $T(n) = 4 \cdot T(\frac{n}{2}) + c \cdot n = O(n^2)$, which is no better than our trivial algorithm.

Algorithm II and a half: Karatsuba algorithm

Observe that

$$(x_1 + x_2) \cdot (y_1 + y_2) = x_1 y_1 + (x_1 y_2 + x_2 y_1) + x_2 y_2 \implies (x_1 y_2 + x_2 y_1) = (x_1 + x_2) \cdot (y_1 + y_2) - x_1 y_1 - x_2 y_2$$

This means that we can only perform three calculations rather than four, resulting in our algorithm having a recurrence relation of $T(n) = 3 \cdot T(\frac{n}{2}) + c \cdot n$, which results in a run time of $O(n^{\log_2 3}) = O(n^{1.59})$.