CSOR4231

Analysis of Algorithms

Algorithms are Everywhere

Examples

- Maps
- Fedex
- Biology
- Physics
- Computer Operating Systems
- Self-Driving Cars
- Determining if you should get a job/loan/school admission
- Regulating your heart
- Space Shuttle
- Machine Learning
- . . .

See, for example:

- `https://personal.denison.edu/~havill/algorithmics/algs.html`
- `https://en.wikipedia.org/wiki/List_of_algorithms`

Why is this the right time to study algorithms?

- Mathematical understanding
- fast computers
- ability to get algorithm implementations to users
- good interfaces

What do we study in this class

- Given a problem, we find the right algorithm

- We use math

- We prove that our work is right

- We keep an eye on practice/implementation, but our goal is to solve the clean well-defined problem.

What are the skills most people need

• Given a new problem, how do we design an algorithm

• Knowing what is efficient and what is not, to help you

   – model problems

   – use existing algorithms

   – decide which algorithms to extend

   – realize when a problem is too hard to solve quickly

First problem to consider: Multiplication

How do we multiply 2 n bit numbers?

```
          3 9 4
      x   5 1 7
      ─────────
        2 7 5 8
        3 9 4 0
    1 9 7 0 0 0
    ─────────────
    2 0 3 6 9 8
```

This algorithm uses roughly:

- n^2 multiplications

- n additions

Can one use fewer than about n^2 basic operations?

Multiplication

Can one use fewer than about n^2 basic operations for multiplication?

- Kolmogorov conjectured, in 1960, that you couldn't use fewer.

- In a seminar, a 23 year old student, Karatsuba, showed that you could use divide-and-conquer to multiply more efficiently.

- We will show his algorithm, and show how knowing how to analyze algorithms leads to (non-obvious) improvements

Karatsuba's algorithm

Observation 1:

- **An** n **digit number** x **can be split into** 2 $n/2$ **bit numbers,** x_1, x_0 , $x = 10^{n/2}x_1 + x_0$

- **e.g.** $12345678 = (1234)(10000) + 5678$

Karatsuba's algorithm

Observation 1:

- **An** n **digit number** x **can be split into** 2 $n/2$ **bit numbers,** x_1, x_0 **,**
  $x = 10^{n/2}x_1 + x_0$

- **e.g.** $12345678 = (1234)(10000) + 5678$

Let's multiply

$$x = 10^{n/2}x_1 + x_0$$

$$y = 10^{n/2}y_1 + y_0$$

$$
\begin{aligned}
xy &= (10^{n/2}x_1 + x_0)(10^{n/2}y_1 + y_0) \\
   &= 10^n x_1 y_1 + 10^{n/2}(x_1 y_0 + x_0 y_1) + x_0 y_0
\end{aligned}
$$

Karatsuba's algorithm

$$x = 10^{n/2}x_1 + x_0$$

$$y = 10^{n/2}y_1 + y_0$$

$$
\begin{aligned}
xy &= (10^{n/2}x_1 + x_0)(10^{n/2}y_1 + y_0) \\
&= 10^n x_1 y_1 + 10^{n/2}(x_1 y_0 + x_0 y_1) + x_0 y_0
\end{aligned}
$$

Rewrite

- $H = x_1 y_1$
- $M = x_1 y_0 + x_0 y_1$
- $L = x_0 y_0$
- $xy = 10^n H + 10^{n/2}M + L$

Analysis

- 4 multiplications
- 2 shifts (multiply by power of 10)
- 3 additions

Use Recursion

$$xy = (10^{n/2}x_1 + x_0)(10^{n/2}y_1 + y_0)$$
$$= 10^n x_1 y_1 + 10^{n/2}(x_1 y_0 + x_0 y_1) + x_0 y_0$$

- 4 multiplications
- 2 shifts (multiply by power of 10)
- 3 additions

Do the multiplications recursively

- Let $T(n)$ be the time to multiply 2 n bit numbers
- Shifts and addtions take linear time

$$T(n) = 4T(n/2) + 2n + 3n$$

We will see that this recurrence solves to $\Theta(n^2)$

Recurrences

$$T(n) = 4T(n/2) + 2n + 3n = 4T(n/2) + 5n$$

- We will soon be able to, in our heads, solve such recurrences.

- We will also realize that decreasing the 4 but increasing the 5 will decrease the solution.

- Can we use fewer than 4 $n/2$ by $n/2$ multiplications?

Karatsuba's Algorithm

We need to compute:

- $H = x_1 y_1$

- $L = x_0 y_0$

- $M = x_1 y_0 + x_0 y_1$

Suppose that after computing H and L, we compute, using 1 multiplication and 2 additions:

$$
\begin{aligned}
Z &= (x_0 + x_1)(y_0 + y_1) \\
&= x_0 y_0 + x_0 y_1 + x_1 y_0 + x_1 y_1
\end{aligned}
$$

We now observe that $M = Z - H - L$

$$
\begin{aligned}
Z - H - L &= (x_0 y_0 + x_0 y_1 + x_1 y_0 + x_1 y_1) - x_1 y_1 - x_0 y_0 \\
&= x_1 y_0 + x_0 y_1 \\
&= M
\end{aligned}
$$

Karatsuba's Algorithm

To summarize, the algorithm to multiply $x = 10^{n/2}x_1 + x_0$ by $y = 10^{n/2}y_1 + y_0$ is :

- $H = x_1 y_1$

- $L = x_0 y_0$

- $Z = (x_0 + x_1)(y_0 + y_1)$

- $M = Z - H - L$

- $xy = 10^n H + 10^{n/2} M + L$

Analysis

- 3 multiplications of $n/2$ bit numbers

- 6 additions/subtractions

- 2 shifts

Recurrence:

$$T(n) = 3T(n/2) + 8n$$

Solving the Recurrence

Recurrence:

$$T(n) = 3T(n/2) + 8n$$

Solution:

$$T(n) = O(n^{\log_2 3}) = O(n^{1.58})$$

- Even faster algorithms are possible.

- FFT can do $O(n \log n)$ time

Course Logistics

Another Problem

Investing for someone who knows the Future: You are given the prices of a stock for each of the next n days. You can buy once and sell once and you want to maximize your profit.

Example

Day	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Price	70	90	40	27	69	80	13	50	35	75	51	53	56	10	15	41

Questions:

- How long does the naive algorithm take?
- Can we improve this with divide and conquer?