once on take enroll
→ email Vivek

Michael A Forbes
mforbes@illinois.edu

• ↦ 2019-08-27.1
2019-08-27.2 ← cs 473

cs 473 Algorithms: Lecture 1 (board) ~90

= Questions

motivation and goals

motivation: google is really useful — maps [[how A → B]]
— flights [[- cheapest A → B
- might not make it much flights]]
— search [[how ideas relate to each other
is there a right answer?]]

[[this lecture not sponsored by google
please consider bing ..]]

Q: how does google do it?

A: algorithms!

Q: can algorithms do everything?

A: no

fact (cs 374): exist computational problems that cannot be solved at all by computers
↳ "undecidable" [[but we care about solving now efficiently "
"

fact (cs 579): exist solvable

Q: which problems can be solved efficiently?

A: no idea [[see cs 579]]

this course: fundamental algorithmic paradigms for designing efficient algorithms [[just do it]]

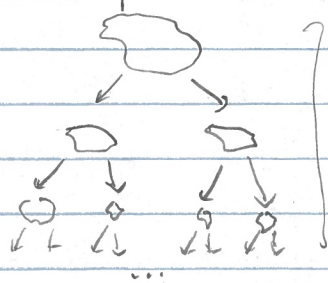
[[cs 374++]] - divide and conquer [[break into small problems recursively and recombine]]
- dynamic programming [[divide and conquer plus memoization]]
- randomization [[take a random walk around campus]]
- cuts and flows [[how quickly can information flow in a network
what are the bottlenecks]]
- linear programming [[optimizing function subject to constraints
[[not an algo]]
- intractability, and coping w/ it [[relaxing criteria for success]]

theme: road to efficient algo is winding, long, and filled w/ math

= questions

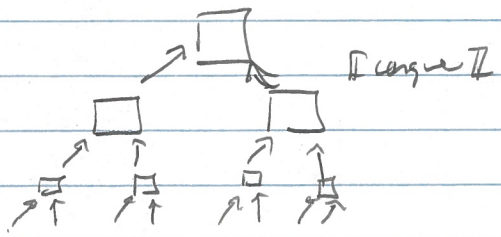
divide and conquer [[seen in cs 374]]

idea:



recursion tree

[[divide into two smaller problems]]



[[conquer]]

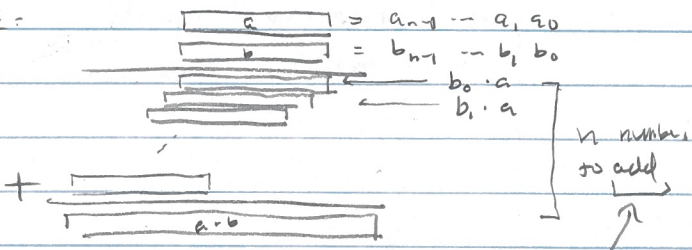
[[base of recursion]]

integer multiplication divide and conquer [[seen in cs 374]]

Michael A. Forbes
m.forbes@illinois.edu
2019-08-27.2 ← 2019-08-27.1
cs473 → 2019-08-27.3

recall: grade school multiplication of two n-bit numbers takes $O(n^2)$ time

ex: 111 × 101 = 1000 + 111 = 10001 | 1 [aka 7×5] [aka 35]

ex: 

[this is not divide and conquer]

$O(n)$ each

$\Rightarrow n \cdot O(n) = O(n^2)$

Q: can we do better?

lem: multiply two n-bit numbers in $O(n^2)$ time, using divide and conquer

pf: a, b n-bit $\quad a = a_1 \cdot 2^{n/2} + a_0 \quad b = b_1 2^{n/2} + b_0$ [need floor/ceil notation carefully do this in psets!]

$$a \cdot b = (a_1 \cdot 2^{n/2} + a_0)(b_1 \cdot 2^{n/2} + b_0) \quad \text{padding}$$
$$= a_1 b_1 \cdot 2^n + (a_0 b_1 + a_1 b_0) 2^{n/2} + a_0 b_0 \quad \text{conquer in 3 additions}$$

multiplication of $n/2$-bit #'s | divide

time complexity $T(n) \le 4 \cdot T(n/2) + O(n)$

$\le O(n^2)$ [rs4]

[do better]

lem [Karatsuba] $O(n^{\log_2 3}) = O(n^{1.584\ldots})$

pf: $a = a_1 \cdot 2^{n/2} + a_0 \quad b = b_1 \cdot 2^{n/2} + b_0$

$$a \cdot b = a_1 b_1 2^n + (a_0 b_1 + a_1 b_0) 2^{n/2} + a_0 b_0$$

key idea: $(a_1 - a_0)(b_1 - b_0) = a_1 b_1 + a_0 b_0 - (a_0 b_1 + a_1 b_0)$

$\Rightarrow$ recursively compute $a_1 b_1$, $a_0 b_0$, $(a_1 - a_0)\cdot(b_1 - b_0)$ $\Rightarrow$ get $a_0 b_1 + a_1 b_0$

($n/2$ bit)

$$T(n) \le 3 \cdot T(n/2) + O(n)$$
$$\le O(n^{\log_2 3})$$

rmk:
- $\Omega(n^2)$ conjectured necessary by Kolmogorov 1960
- Karatsuba 1960 disproved this! [can we do better?] [$k \to \infty$?]
- Toom 63 / Cook 66: split n-bit numbers into $k \ge 2$ parts $\Rightarrow$ multiplication in $n^{1+O(1/\lg k)}$ [for $k \le O(1)$]
- Gauss 1800's / Cooley-Tukey 1965 / Schonhage Strassen 1971: Fast Fourier Transform $\Leftrightarrow$ multiplication in $O(n \lg n \cdot \lg\lg n)$
- Furer 07: $O(n \lg n \cdot 2^{O(\lg^* n)})$ [iterated logarithm, very slowly growing]

Michael A. Forbes
MForbes@illinois.edu

2019-08-27.2 → 2019-08-27.3
2019-08-29.1 ← cs473

Harvey - van der Hoeven 2019: $O(n \lg n)$

Q: do better?

A: not believed likely

today = - logistics
- motivation [google]
- goals [fundamental algo paradigms]
- divide and conquer [Karatsuba matrix multiplication]