finished 15 min early
clock in-room wrong

Michael A. Forbes
miforbes@illinois.edu
2024-01-15.1

cs473 Algorithms: Lecture 1 (2024-01-16)

logistics:
- pset 0 out Friday 3pm
- signup
  - piazza
  - gradescope

today:
- introduction [logistics] [motivation and goals]
- divide and conquer; integer multiplication

[move to front]
[laptop policy]

lecture: TR 14-15:15 Siebel 1404

staff: instructor: Prof. Michael A. Forbes (miforbes)
- TA: Christian Howard (howard28)
- TA: Shubhang Kulkarni (smkulka2)

	[after lecture]
T 15:30	TBA
W	TBA
R	TBA

[full details] website

resources: webpage: courses.engr.illinois.edu/cs473/sp2024
- materials

calendar: - lectures - readings
- psets

forum (piazza):
- course announcements [!]
- peer discussion [public, subject to collab policy]
- contacting staff [private] [no email]

submissions (gradescope):
- coursework submission/return/regrades
- gradebook

course materials: - lectures:
- lecture materials
  - boardwork [this sheet]
  - recordings [lectures are recorded]
  - suggested reading for each lecture
- textbooks
  - ↳ 90% from Kleinberg-Tardos

grades: - psets (25%)
- 12 psets, 3 problems each
- out/due F17
- $\boxed{\text{no}}$ late psets, $\boxed{\text{but}}$ lowest psetscores dropped [webpage]
- pset groups: - submit psets in groups ≤3 [encouraged!]
  - $\boxed{\text{except}}$ pset0, done $\boxed{\text{individually}}$
  - [webpage]
  - integrity

- exams (45%)
  - 2 × 22.5%
  - non cumulative
  - dates - 2024-02-26 19-21:30
    - 2024-04-08 19-21:30

- final (30%) cumulative

prereq: formal: - cs173 (discrete math)
- cs225 (data structures)
- cs374 (algo, models of computation)

Michael A. Forbes
mforbes@illinois.edu
2024-01-15.2

wrt web

informal - formal proofs [induction, ...]
- basic algo [recursion, loops, ...]
- data structures [arrays, lists, ...]
- graph algo [DFS, BFS, Dijkstra, ...]
- probability [rand var, expectation, variance, ...]
- models of comp [Turing machines, ...]

[first thm]

thm: reading the course webpage makes you a better student

[not a real proof technique]

pf: by authority

cor: ________ " ________ often ________ " ________ even ________ " ________ [and pizza]

= [Q]

[why are you here]

Q: why this course? [why am I here]

motivation: google is really useful
- maps [path A → B]
- flights [path A → B - cheapest - shortest - multiple carriers]

[how to relate ideas]
- search ["right" answer is subjective 'apple' vs 'apple']

[lecture not sponsored by google, please consider duckduckgo]

Q: how does google do it?

A: algorithms!

Q: can algorithms do everything?

[not magic]

A: no

[well-defined answer]

fact (cs374): exist computational problems that cannot be solved by computers

↳ undecidable problems, eg the "halting problem"

[we care about solving now]

fact (cs579): exist ________ " ________ solvable ________ " ________ efficiently

Q: which problems can be solved efficiently?

[cs579]

A: no idea

this course: fundamental algorithmic paradigms for designing efficient algo [just do it]
- divide and conquer [recursively break into smaller problems, and recombine] [cs374 ++]
- dynamic programming [divide and conquer, plus memoization]
- cuts and flows [how quickly can information flow in a network?] [what are the bottlenecks?]
- linear programming [optimize linear function subject to linear constraints]
- NP-completeness [classifying problems as suspected of being intractable]
- approximation algo [relaxing success criteria, to cope w intractability]

[surprises] [lots to learn]

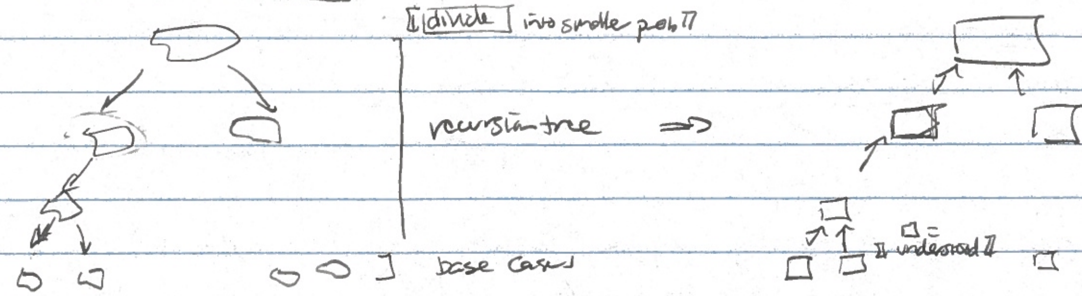
theme: road to efficient algo is winding, long, and filled w/ math

= [Q]

Michael A. Forbes
miforbes@illinois.edu
2024-01-15

⟦ seen in cs374 ⟧

idea (divide and conquer):



recall - grade school multiplication of two n-bit numbers takes $O(n^2)$ time ⟦ binary vs decimal ⟧ ⟦ asymptotics ⟧ ⟦ worst case time as function of input size ⟧

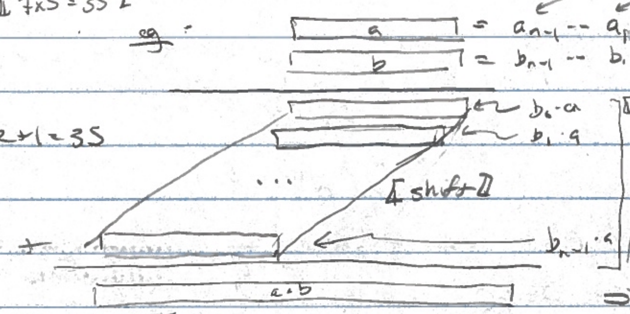
eg:

$$\begin{array}{r} 111 = 7 \\ \times\ 101 = 5 \\ \hline 111 \\ 000 \\ 111 \\ \hline 100011 = 32+2+1 = 35 \end{array}$$

⟦ $7 \times 5 = 35$ ⟧

32 16 8 4 2 1

eg:



n numbers to add, $O(n)$ bits each, $O(n)$ steps

$\Rightarrow n \cdot O(n) = O(n^2)$ steps ⟦ not divide and conquer ⟧

Q - do better? ⟦ in crypto, $n \approx 4000$ ⟧ ⟦ test taking skills say "yes" ⟧ ⟦ first step ⟧

lem: multiplication of two n-bit numbers can be done in $O(n^2)$ steps, ⟦ with ⟧ divide and conquer ⟦ same ⟧ ⟦ new ⟧

pf: a, b n-bit $a = a_1 \cdot 2^{n/2} + a_0$, w/ a_1, a_0 $\frac{n}{2}$ bits ⟦ ignoring rounding issues ⟧

$b = b_1 \cdot 2^{n/2} + b_0$, w/ b_1, b_0 —"—

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

⟦ shift ⟧ mult, $O(n)$ time

$$= a_1 b_1 2^n + (a_0 b_1 + a_1 b_0) 2^{n/2} + a_0 b_0$$

"conquer" - 3 add - 2 shifts $O(n)$

4 multiplications of $n/2$-bit numbers "divide"

algo: ⟦ what is algo ⟧ clear ⟦ webpage policy ⟧ ⟦ do divide and conquer ⟧ ⟦ base cases ⟧ ⟦ not always "clear" ⟧ ⟦ output is correct ⟧

correctness: clear

complexity: $T(n)$ = worst case time of any n-bit multiplication ⟦ solve recurrence ⟧

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

⟦ divide ⟧ ⟦ conquer ⟧

$\le \cdots$ ⟦ solve, eg psets ⟧

$\le O(n^2)$ ∎

⟦ time bound hasn't changed, only the paradigm ⟧

Q: do better?

thm [Karatsuba60]: ⟦ cs374 ⟧ ———— $O(n^{\log_2 3}) = O(n^{1.584\ldots})$ ⟦ cs374 ⟧ yes. approx

pf: $a = a_1 2^{n/2} + a_0$ || $ab = a_1 b_1 2^n + (a_0 b_1 + a_1 b_0) 2^{n/2} + a_0 b_0$

$b = b_1 2^{n/2} + b_0$

Michael A. Forbes
miforbes@illinois.edu
2024-01-15.4

above: use 4 recursive calls to compute 3 numbers a_1b_1, $a_0b_1+a_1b_0$, a_0b_0

idea: use 3

lem: $(a_1-a_0)(b_1-b_0) = a_1b_1 + a_0b_0 - (a_0b_1+a_1b_0)$

another sign — lem: $\in (-2^{n/2}, 2^{n/2})$, so are $n/2$-bit multiplications [renormalize to positive #]

former — algo: - recursively compute a_1b_1, a_0b_0, $(a_1-a_0)(b_1-b_0)$ [$n/2$-bit mult]
- compute $a_0b_1+a_1b_0$ via [add]
- compute $c \cdot r$ via [add]

correctness: clear

complexity: $T(n) \le 3\cdot T(n/2) + O(n)$ [divide] [conquer]
$\le \ldots$ [solve]
$\le O(n^{\log_2 3})$ ∎

rmk: - $\Omega(n^2)$ conjectured necessary by Kolmogorov 60
- Karatsuba 60 disproved this! [do better?]
- Toom 63, Cook 66: split n-bit numbers into $k \ge 2$ parts
$\Rightarrow$ mult in $n^{1+O(1/\lg k)}$ time for $k \le O(1)$ [better?]
[miforbes: my $\Theta_k(n^{\log(2k-1)/\log k})$
$\le \lg 2k/\lg k = 1 + 1/\lg k$

- Gauss 1800's, Cooley Tukey 65, Schonhage Strassen 71:
multiplication via Fast Fourier Transform (FFT) in $O(n \lg n \lg\lg n)$ steps [better?]
- Fürer 07: $O(n \lg n\, 2^{O(\lg^* n)})$ iterated log [very slowly growing] [better?]
- Harvey - van der Hoeven 19 $O(n \lg n)$

Q: do better? [test taking skills say class is over]

A: not believed likely

today: - introduction [motivation and goals] [logistics]
- divide and conquer: integer multiplication: Karatsuba's algo

next lecture: divide and conquer

logistics: - pset 0 out F17
- signup - piazza
- gradescope