Math 251, Fri 8-Oct-2021 -- Fri 8-Oct-2021
Discrete Mathematics
Fall 2021

Friday, October 8th 2021

Wk 6, Fr
Topic:: Algorithmic complexity
Read:: Rosen 3.3

for - - -

break

Algorithmic Complexity

Basic idea: relate size n of input to, for instance

- time complexity (often assessed by number of comparisons, flops, etc.)
  - worst-case analysis
  - average-case analysis
- space complexity
- terms like
  - linear complexity $\Theta(n)$
  - quadratic complexity $\Theta(n^2)$
  - polynomial complexity $\Theta(n^r)$ some $r \in \mathbb{Z}^+$
    - linear, quadratic are special cases
    - call these problems "tractable", and are of **Class P**
  - exponential complexity
    - call these problems "intractable"
      - may still be of **Class P** if a polynomial-time algorithm exists
      - say it is of **Class NP** if no alg. of polynomial time is known for solving it, but there is a polynomial time alg. for checking a solution
    - **NP-complete** problems, and the **P vs. NP problem**

Algorithm:

1. Seek divisor of $n \in \mathbb{Z}^+$
   Similar to analysis of linear search algorithm
2. binary search - sorted list
3. bubble sort
4. matrix multiplication
5. evaluating an n^{th}-degree polynomial

1. Given positive integer n. Task: Find a divisor

Ex.) $6397 = n$

→ 2 divide 6397 ? No
3 " " ? No
4 divide 6397 ? No
5 " " ? No
⋮
6396 " " ?

If no short-circuit, then # of ~~attempted~~ divisions $6395 = n-2$

$O(n)$

Could stop once reached $\sqrt{n}$: Applying this as a stopping criterion makes it $O(\sqrt{n})$.

2. Binary search

n entries

37, 82, 115, 272, ..., 13509

Split list

37, 82, ..., 2611 — lower half — $\lfloor n/2 \rfloor + 1$

2981, ..., 13509 — upper half — $\lfloor n/2 \rfloor - 1$

Q: How many times splitting list in half (size n) before I have singleton lists

A: $\lfloor \log_2 n \rfloor + 1$ is $O(\log_2 n)$

3. Bubble sort

\# of comparisons: for list size n

$$1+2+3+\dots+(n-1) = \frac{1}{2}n(n-1)$$
$$= \frac{1}{2}n^2 - \frac{1}{2}n, \quad \Theta(n^2).$$

4. Matrix multiplication

$(3)(1)+(-1)(-1)$

$$\underset{2\times 2}{\begin{bmatrix} 3 & -1 \\ 2 & 7 \end{bmatrix}} \underset{2\times 2}{\begin{bmatrix} 1 & 5 \\ -1 & -3 \end{bmatrix}} = \underset{2\times 2}{\begin{bmatrix} \_ & \_ \\ \_ & \_ \end{bmatrix}}$$

$(2)(1)+(7)(-1)$

Same process applied to $n \times n$ matrices

$$\begin{array}{c} \text{row } i \rightarrow \end{array} \underset{n\times n}{\begin{bmatrix} & & & \\ a_{i1} & a_{i2} & a_{i3} \cdots & a_{in} \\ & & & \end{bmatrix}} \underset{n\times n}{\begin{bmatrix} & b_{1j} & \\ & b_{2j} & \\ & \vdots & \\ & b_{nj} & \end{bmatrix}} = \underset{n\times n}{\begin{bmatrix} & & \\ & c_{ij} & \\ & & \end{bmatrix}}$$

j^{th} column

results i^{th} row, j^{th} column

i^{th} row

j^{th} col.

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \dots + a_{in}b_{nj}$$

$\leftarrow$ n multiplications, $(n-1)$ additions to fill one slot in product

$n + (n-1)$, or $2n-1$ flops

$$\underset{\text{slots}}{n^2}\,(\underbrace{2n-1}_{\text{fills one slot}}) = 2n^3 - n^2 \text{ flops} \qquad \Theta(n^3).$$

5. Evaluating n^{th}-degree polynomial

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$$

at $x = c$.

Compute $p(c) = a_n \cdot \underbrace{c \cdot c \cdot c \cdots c}_{n \text{ factors}} + a_{n-1} \cdot \underbrace{c \cdot c \cdots c}_{\to (n-1) \text{ mults.}} + \cdots$

$\to n$ mults.

$$+ \underbrace{a_1 c}_{1 \text{ mult.}} + a_0$$

$$\text{flops} = (\# \text{ of mults.}) + (\# \text{ of addition})$$

$$= \underbrace{(1 + 2 + 3 + \cdots + n)} + n$$

$$= \frac{n}{2}(n+1) + n = \frac{1}{2}n^2 + \frac{3}{2}n \qquad (\mathcal{O}(n^2))$$

Better algorithm, labeled Horner's Algorithm, for evaluating polynomials comes up in the textbook, Exercise 14 of Section 3.3.