Asymptotics, Recurrences, and Divide and Conqure

Hengfeng Wei

hfwei@nju.edu.cn

March 29, 2017

Algorithm analysis

- Given a problem P
- design an alg. A
- input space $\mathcal{X}_n$: inputs of size n

Algorithm analysis

- Given a problem P
- design an alg. A
- input space $\mathcal{X}_n$: inputs of size n

$$W(n) = T_{\text{worst-case}}(n) = \max_{X \in \mathcal{X}_n} T(X)$$

Algorithm analysis

- Given a problem P
- design an alg. A
- input space $\mathcal{X}_n$: inputs of size n

$$W(n) = T_{\text{worst-case}}(n) = \max_{X \in \mathcal{X}_n} T(X)$$

$$B(n) = T_{\text{best-case}}(n) = \min_{X \in \mathcal{X}_n} T(X)$$

Algorithm analysis

- Given a problem P
- design an alg. A
- input space $\mathcal{X}_n$: inputs of size n

$$W(n) = T_{\text{worst-case}}(n) = \max_{X \in \mathcal{X}_n} T(X)$$

$$B(n) = T_{\text{best-case}}(n) = \min_{X \in \mathcal{X}_n} T(X)$$

$$A(n) = T_{\text{average-case}}(n) = \sum_{X \in \mathcal{X}_n} T(X) \cdot Pr(X)$$

Algorithm analysis

- Given a problem P
- design an alg. A
- input space $\mathcal{X}_n$: inputs of size n

$$W(n) = T_{\text{worst-case}}(n) = \max_{X \in \mathcal{X}_n} T(X)$$

$$B(n) = T_{\text{best-case}}(n) = \min_{X \in \mathcal{X}_n} T(X)$$

$$A(n) = T_{\text{average-case}}(n) = \sum_{X \in \mathcal{X}_n} T(X) \cdot Pr(X) = E_{X \in \mathcal{X}_n}[T(X)]$$

(Problem 1.1.8)

$$
\begin{aligned}
A &= \sum_{X \in \mathcal{X}} T(X) \cdot Pr(X) \\
&= T(1)Pr(1) + T(2)Pr(2) + \cdots + T(n)Pr(n) \\
&= \cdots \\
&= \frac{n}{8} + \frac{55}{2}
\end{aligned}
$$

Average-case analysis of Quicksort

$$A(n) = n - 1 + \frac{1}{n} \sum_{i=0}^{i=n-1} \left(A(i) + A(n-i-1) \right)$$

$$A(n) = E_{X \in \mathcal{X}_n}[T(X)] = \sum_{X \in \mathcal{X}_n} T(X) \cdot Pr(X)$$

Average-case analysis of Quicksort

$$A(n) = n - 1 + \frac{1}{n} \sum_{i=0}^{i=n-1} (A(i) + A(n-i-1))$$

$$A(n) = E_{X \in \mathcal{X}_n}[T(X)] = \sum_{X \in \mathcal{X}_n} T(X) \cdot Pr(X)$$

$$\begin{aligned} A(n) &= E[T(X)] \\ &= E[E[T(X)|I]] \\ &= \sum_{i=0}^{i=n-1} Pr(I=i) E[T(X) \mid I = i] \\ &= \sum_{i=0}^{i=n-1} \frac{1}{n}[n - 1 + A(i) + A(n-i-1)] \end{aligned}$$

$\Omega\ (\omega), \Theta, O\ (o)$

$$O(g(n)) = \{f(n) \mid \exists c > 0, \exists n_0, \forall n \geq n_0 : 0 \leq f(n) \leq cg(n)\}$$

$\Omega\ (\omega), \Theta, O\ (o)$

$$O(g(n)) = \{f(n) \mid \exists c > 0, \exists n_0, \forall n \geq n_0 : 0 \leq f(n) \leq cg(n)\}$$

$$\Omega(g(n)) = \{f(n) \mid \exists c > 0, \exists n_0, \forall n \geq n_0 : 0 \leq cg(n) \leq f(n)\}$$

Ω $(\omega), \Theta, O$ (o)

$$O(g(n)) = \{f(n) \mid \exists c > 0, \exists n_0, \forall n \geq n_0 : 0 \leq f(n) \leq cg(n)\}$$

$$\Omega(g(n)) = \{f(n) \mid \exists c > 0, \exists n_0, \forall n \geq n_0 : 0 \leq cg(n) \leq f(n)\}$$

$$\begin{aligned} \Theta(g(n)) = \{f(n) \mid \exists c_1 > 0, \exists c_2 > 0, \exists n_0, \forall n \geq n_0 : \\ 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)\} \end{aligned}$$

$\Omega\ (\omega), \Theta, O\ (o)$

$$O(g(n)) = \{f(n) \mid \exists c > 0, \exists n_0, \forall n \geq n_0 : 0 \leq f(n) \leq cg(n)\}$$

$$\Omega(g(n)) = \{f(n) \mid \exists c > 0, \exists n_0, \forall n \geq n_0 : 0 \leq cg(n) \leq f(n)\}$$

$$\Theta(g(n)) = \{f(n) \mid \exists c_1 > 0, \exists c_2 > 0, \exists n_0, \forall n \geq n_0 : \\ 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)\}$$

$$o(g(n)) = \{f(n) \mid \forall c > 0, \exists n_0, \forall n \geq n_0 : 0 \leq f(n) \leq cg(n)\}$$

$\Omega\ (\omega), \Theta, O\ (o)$

$$O(g(n)) = \{f(n) \mid \exists c > 0, \exists n_0, \forall n \geq n_0 : 0 \leq f(n) \leq cg(n)\}$$

$$\Omega(g(n)) = \{f(n) \mid \exists c > 0, \exists n_0, \forall n \geq n_0 : 0 \leq cg(n) \leq f(n)\}$$

$$\Theta(g(n)) = \{f(n) \mid \exists c_1 > 0, \exists c_2 > 0, \exists n_0, \forall n \geq n_0 : \\ 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)\}$$

$$o(g(n)) = \{f(n) \mid \forall c > 0, \exists n_0, \forall n \geq n_0 : 0 \leq f(n) \leq cg(n)\}$$

$$\omega(g(n)) = \{f(n) \mid \forall c > 0, \exists n_0, \forall n \geq n_0 : 0 \leq cg(n) \leq f(n)\}$$

(Problem 1.2.6)

Problem 1.2.6 (4)

$$f(n) = \Theta(g(n)) \iff f(n) = O(g(n)) \wedge f(n) = \Omega(g(n))$$

(Problem 1.2.6)

Problem 1.2.6 (4)

$$f(n) = \Theta(g(n)) \iff f(n) = O(g(n)) \land f(n) = \Omega(g(n))$$

Problem 1.2.6 (5)

$$f(n) = O(g(n)) \iff g(n) = \Omega(f(n))$$
$$f(n) = o(g(n)) \iff g(n) = \omega(f(n))$$

(Problem 1.2.6)

Problem 1.2.6 (4)

$$f(n) = \Theta(g(n)) \iff f(n) = O(g(n)) \land f(n) = \Omega(g(n))$$

Problem 1.2.6 (5)

$$f(n) = O(g(n)) \iff g(n) = \Omega(f(n))$$
$$f(n) = o(g(n)) \iff g(n) = \omega(f(n))$$

$f(n) = O(g(n)) \lor g(n) = \Omega(f(n))$?

(Problem 1.2.6)

Problem 1.2.6 (4)

$$f(n) = \Theta(g(n)) \iff f(n) = O(g(n)) \land f(n) = \Omega(g(n))$$

Problem 1.2.6 (5)

$$f(n) = O(g(n)) \iff g(n) = \Omega(f(n))$$
$$f(n) = o(g(n)) \iff g(n) = \omega(f(n))$$

$f(n) = O(g(n)) \lor g(n) = \Omega(f(n))$?

$$f(n) = n, \quad g(n) = n^{1+\sin n}$$

(Problem 1.2.6)

Problem 1.2.6 (4)

$$f(n) = \Theta(g(n)) \iff f(n) = O(g(n)) \land f(n) = \Omega(g(n))$$

Problem 1.2.6 (5)

$$f(n) = O(g(n)) \iff g(n) = \Omega(f(n))$$
$$f(n) = o(g(n)) \iff g(n) = \omega(f(n))$$

$f(n) = O(g(n)) \lor g(n) = \Omega(f(n))$?

$$f(n) = n, \quad g(n) = n^{1+\sin n}$$

Problem 1.2.6 (6)

$$\Theta(g(n)) \cap o(g(n)) = \emptyset$$

$\Omega\ (\omega), \Theta, O\ (o)$

Reference

"Big Omicron and Big Omega and Big Theta" by Donald E. Knuth, 1976.

(Problem 1.2.10)

$$\log(n!) = \Theta(n \log n)$$

(Problem 1.2.10)

$$\log(n!) = \Theta(n \log n)$$

Prove by definition.

(Problem 1.2.10)

$$\log(n!) = \Theta(n \log n)$$

Prove by definition.

Exercise: Prove it by Mathematical Induction.

Horner's rule (Problem 1.1.6)

$$P(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1} + a_n x^n$$

Horner's rule (Problem 1.1.6)

$$P(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1} + a_n x^n$$

Loop invariant (after the k-th loop):

$$\sum_{i=n}^{i=n-k} a_i x^{k-(n-i)}$$

Recurrences

$$T(n) = aT(n/b) + f(n) \quad (a > 0, b > 1)$$

$$\left.\begin{array}{r} f(n) \\ af(\frac{n}{b}) \\ a^2 f(\frac{n}{b^2}) \\ \vdots \\ a^{\log_b^n} f(1) = n^{\log_b^a} \end{array}\right\}$$

Recurrences

$$T(n) = aT(n/b) + f(n) \quad (a > 0, b > 1)$$

$$\left.\begin{array}{r} f(n) \\ af(\frac{n}{b}) \\ a^2 f(\frac{n}{b^2}) \\ \vdots \\ a^{\log_b^n} f(1) = n^{\log_b^a} \end{array}\right\} \sum = \begin{cases} n^{\log_b^a} & q > 1 \\ n^{\log_b^a} \log n & q = 1 \\ f(n) & q < 1 \end{cases}$$

Recurrences

$$T(n) = aT(n/b) + f(n) \quad (a > 0, b > 1)$$

$$\left.\begin{array}{r} f(n) \\ af(\frac{n}{b}) \\ a^2 f(\frac{n}{b^2}) \\ \vdots \\ a^{\log_b^n} f(1) = n^{\log_b^a} \end{array}\right\} \sum = \begin{cases} n^{\log_b^a} & q > 1 & f(n) = O(n^{E-\epsilon}) \\ n^{\log_b^a} \log n & q = 1 & f(n) = \Theta(n^E) \\ f(n) & q < 1 & f(n) = \Omega(n^{E+\epsilon}) \end{cases}$$

Solving recurrences (Problem 1.2.13, 1.2.16)

1. $\Theta(n^{\log_3^2})$
2. $\Theta(\log^2 n)$
3. $\Theta(n)$
4. $\Theta(n \log n)$
5. $\Theta(n \log^2 n)$
6. $\Theta(n^2)$
7. $\Theta(n^{\frac{3}{2}} \log n)$
8. $\Theta(n)$
9. $\Theta(n^{c+1})$
10. $\Theta(c^{n+1})$
11. $\Theta(n)$

$$T(n) = T(n/2) + \log n$$

$$T(n) = 2T(n/2) + n \log n$$

Solving recurrences (Problem 1.2.13, 1.2.16)

1. $\Theta(n^{\log_3^2})$
2. $\Theta(\log^2 n)$
3. $\Theta(n)$
4. $\Theta(n \log n)$
5. $\Theta(n \log^2 n)$
6. $\Theta(n^2)$
7. $\Theta(n^{\frac{3}{2}} \log n)$
8. $\Theta(n)$
9. $\Theta(n^{c+1})$
10. $\Theta(c^{n+1})$
11. $\Theta(n)$

$$T(n) = T(n/2) + \log n$$

$$T(n) = 2T(n/2) + n \log n$$

Reference

$$f(n) = \Theta(n^{\log_b^a} \lg^k n) \Rightarrow \Theta(n^{\log_b^a} \lg^{k+1} n)$$

Solving recurrences (Problem 1.2.13, 1.2.16)

$$T(n) = T(n/2) + T(n/4) + T(n/8) + n$$

Solving recurrences (Problem 1.2.13, 1.2.16)

$$T(n) = T(n/2) + T(n/4) + T(n/8) + n$$

By recursion-tree.

Solving recurrences (Problem 1.2.13, 1.2.16)

$$T(n) = T(n/2) + T(n/4) + T(n/8) + n$$

By recursion-tree.

Exercise: Prove it by Mathematical Induction.

Solving recurrences (Problem 1.2.13, 1.2.16)

$$T(n) = T(n/2) + T(n/4) + T(n/8) + n$$

By recursion-tree.

Exercise: Prove it by Mathematical Induction.

Reference

"On the Solution of Linear Recurrence Equations" by Akra & Bazzi, 1996.

$$T(n) = \sum_{i=1}^{k} a_i T(n/b_i) + f(n)$$

Gaps (Problem 1.2.16)

$$T(n) = 2T(n/2) + \frac{n}{\log n}$$

Gaps (Problem 1.2.16)

$$T(n) = 2T(n/2) + \frac{n}{\log n} = \Theta(n \log \log n)$$

Gaps (Problem 1.2.16)

$$T(n) = 2T(n/2) + \frac{n}{\log n} = \Theta(n \log \log n)$$

The regularity condition in Case 3:

$$bf(n/c) \le cf(n), \text{ for some } c < 1 \text{ and sufficiently large } n$$

$$T(n) = T(n/2) + n(2 - \cos n)$$

$$n^E = n^0 \quad f(n) = n(2 - \cos n) = \Omega(n^{0+\epsilon})$$

Gaps (Problem 1.2.16)

$$T(n) = 2T(n/2) + \frac{n}{\log n} = \Theta(n \log \log n)$$

The regularity condition in Case 3:

$$bf(n/c) \le cf(n), \text{ for some } c < 1 \text{ and sufficiently large } n$$

$$T(n) = T(n/2) + n(2 - \cos n)$$

$$n^E = n^0 \quad f(n) = n(2 - \cos n) = \Omega(n^{0+\epsilon})$$

$$n = 2\pi k (k \text{ odd}) \Rightarrow c \ge \frac{3}{2}$$

(Problem 1.2.15)

$$
\begin{aligned}
\mathsf{T}(n) &= \sqrt{n}\ \mathsf{T}(\sqrt{n}) + n \\
&= n^{\frac{1}{2}}\ \mathsf{T}\left(n^{\frac{1}{2}}\right) + n \\
&= n^{\frac{1}{2}}\left(n^{\frac{1}{2^2}}\ \mathsf{T}\left(n^{\frac{1}{2^2}}\right) + n^{\frac{1}{2}}\right) + n \\
&= n^{\frac{1}{2}+\frac{1}{2^2}}\ \mathsf{T}\left(n^{\frac{1}{2^2}}\right) + 2n \\
&= n^{\frac{1}{2}+\frac{1}{2^2}}\left(n^{\frac{1}{2^3}}\ \mathsf{T}\left(n^{\frac{1}{2^3}}\right) + n^{\frac{1}{2^2}}\right) + 2n \\
&= n^{\frac{1}{2}+\frac{1}{2^2}+\frac{1}{2^3}}\ \mathsf{T}\left(n^{\frac{1}{2^3}}\right) + 3n \\
&= \cdots \\
&= n^{\sum_{i=1}^{k}\frac{1}{2^i}}\ \mathsf{T}\left(n^{\frac{1}{2^k}}\right) + kn
\end{aligned}
$$

(Problem 1.2.15)

$$n^{\frac{1}{2^k}} = 2 \Rightarrow k = \log \log n$$

(Problem 1.2.15)

$$n^{\frac{1}{2^k}} = 2 \Rightarrow k = \log\log n$$

$$\begin{aligned}
\mathsf{T}(n) &= n^{\sum_{i=1}^{k} \frac{1}{2^i}}\ \mathsf{T}\left(n^{\frac{1}{2^k}}\right) + kn \\
&= n^{\sum_{i=1}^{\log\log n} \frac{1}{2^i}}\ \mathsf{T}(2) + n\log\log n
\end{aligned}$$

(Problem 1.2.15)

$$n^{\frac{1}{2^k}} = 2 \Rightarrow k = \log\log n$$

$$\begin{aligned} \mathsf{T}(n) &= n^{\sum_{i=1}^{k} \frac{1}{2^i}}\ \mathsf{T}\left(n^{\frac{1}{2^k}}\right) + kn \\ &= n^{\sum_{i=1}^{\log\log n} \frac{1}{2^i}}\ \mathsf{T}(2) + n\log\log n \end{aligned}$$

$$\sum_{i=1}^{\log_2 \log_2 (n)} \frac{1}{2^i} < 1 \Rightarrow T(n) = \Theta(n \log\log n)$$

(Problem 1.2.15)

$$n^{\frac{1}{2^k}} = 2 \Rightarrow k = \log\log n$$

$$\begin{aligned} \mathsf{T}(n) &= n^{\sum_{i=1}^{k} \frac{1}{2^i}}\ \mathsf{T}\left(n^{\frac{1}{2^k}}\right) + kn \\ &= n^{\sum_{i=1}^{\log\log n} \frac{1}{2^i}}\ \mathsf{T}(2) + n\log\log n \end{aligned}$$

$$\sum_{i=1}^{\log_2 \log_2 (n)} \frac{1}{2^i} < 1 \Rightarrow T(n) = \Theta(n\log\log n)$$

Exercise: Prove it by Mathematical Induction.

(Problem 1.2.15)

$$T(n) = \sqrt{n}T(\sqrt{n}) + n$$

$$n = 2^k \quad \sqrt{n} = 2^{k/2} \quad k = \log n$$

Integer multiplication (Problem 2.15)

Integer Multiplication

Multiplying two n-bit integers in $o(n^2)$ time. (Assuming $n = 2^k$.)

Column multiplication in $\Theta(n^2)$

Elementray operations:

- n-bit $+$ n-bit: $O(n)$
- 1-bit $\times$ 1-bit: $O(1)$
- n-bit shifted by 1-bit: $O(1)$

Integer multiplication (Problem 2.15)

Simple divide and conquer:

$$x = x_L : x_R = 2^{n/2} x_L + x_R$$
$$y = y_L : y_R = 2^{n/2} y_L + y_R$$

$$\begin{aligned} xy &= (2^{n/2} x_L + x_R)(2^{n/2} y_L + y_R) \\ &= 2^n x_L y_L + 2^{n/2}(x_L y_R + x_R y_L) + x_R y_R \end{aligned}$$

$$T(n) = 4T(n/2) + \Theta(n) = \Theta(n^2)$$

Integer multiplication (Problem 2.15)

A little history:

- Kolmogorov (1952) conjecture: $\Omega(n^2)$
- Kolmogorov (1960) seminar
- Karatsuba (*within a week*): $\Theta(n^{1.59})$
- "The Complexity of Computations" by Karatsuba, 1995

Integer multiplication (Problem 2.15)

Karatsuba algorithm:

$$T(n) = 3T(n/2) + \Theta(n) = \Theta(n^{\log_2 3}) = \Theta(n^{1.59})$$

Integer multiplication (Problem 2.15)

Karatsuba algorithm:

$$T(n) = 3T(n/2) + \Theta(n) = \Theta(n^{\log_2 3}) = \Theta(n^{1.59})$$

$$xy = 2^n x_L y_L + 2^{n/2}(x_L y_R + x_R y_L) + x_R y_R$$

$$\underbrace{(x_L + x_R)(y_L + y_R)}_{P_0} = \underbrace{x_L y_L}_{P_1} + (x_L y_R + x_R y_L) + \underbrace{x_R y_R}_{P_2}$$

$$xy = 2^n P_1 + 2^{n/2}(P_0 - P_1 - P_2) + P_2$$

Matrix multiplication (Problem 2.16)

Matrix multiplication

Multiplying two $n \times n$ matrices in $o(n^3)$ time. (Assuming $n = 2^k$.)

$$Z = X \times Y$$

$$Z_{ij}$$

$$T(n) = \Theta(n^2 \cdot n) = \Theta(n^3)$$

Elementrary operations:

- integer addition: $O(1)$
- integer multiplication: $O(1)$

Matrix multiplication (Problem 2.16)

$$X = \begin{bmatrix} A & B \\ C & D \end{bmatrix}, \quad Y = \begin{bmatrix} E & F \\ G & H \end{bmatrix} \qquad (A \dots H \in \mathbb{R}^{n/2} \times \mathbb{R}^{n/2})$$

$$XY = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}$$

$$T(n) = 8T(n/2) + \Theta(n^2) = \Theta(n^3)$$

Matrix multiplication (Problem 2.16)

Strassen algorithm:

$$T(n) = 7T(n/2) + \Theta(n^2) = \Theta(n^{\lg 7}) = \Theta(n^{2.808})$$

$$XY = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

$$\begin{aligned}
P_1 &= A(F - H) \\
P_2 &= (A + B)H \\
P_3 &= (C + D)E \\
P_4 &= D(G - E) \\
P_5 &= (A + D)(E + H) \\
P_6 &= (B - D)(G + H) \\
P_7 &= (A - C)(E + F)
\end{aligned}$$

Matrix multiplication (Problem 2.16)

Strassen algorithm:

$$T(n) = 7T(n/2) + \Theta(n^2) = \Theta(n^{\lg 7}) = \Theta(n^{2.808})$$

$$XY = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

$$\begin{aligned} P_1 &= A(F - H) \\ P_2 &= (A + B)H \\ P_3 &= (C + D)E \\ P_4 &= D(G - E) \\ P_5 &= (A + D)(E + H) \\ P_6 &= (B - D)(G + H) \\ P_7 &= (A - C)(E + F) \end{aligned}$$

- Strassen (1969): $\Theta(n^{2.808})$
  "Gaussian Elimination is Not Optimal"

Matrix multiplication (Problem 2.16)

Strassen algorithm:

$$T(n) = 7T(n/2) + \Theta(n^2) = \Theta(n^{\lg 7}) = \Theta(n^{2.808})$$

$$XY = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

$$\begin{aligned} P_1 &= A(F - H) \\ P_2 &= (A + B)H \\ P_3 &= (C + D)E \\ P_4 &= D(G - E) \\ P_5 &= (A + D)(E + H) \\ P_6 &= (B - D)(G + H) \\ P_7 &= (A - C)(E + F) \end{aligned}$$

- Strassen (1969): $\Theta(n^{2.808})$
  "Gaussian Elimination is Not Optimal"
- (2014): $\Theta(n^{2.373})$

Matrix multiplication (Problem 2.16)

Strassen algorithm:

$$T(n) = 7T(n/2) + \Theta(n^2) = \Theta(n^{\lg 7}) = \Theta(n^{2.808})$$

$$XY = \begin{bmatrix} P_5 + P_4 - P_2 + P_6 & P_1 + P_2 \\ P_3 + P_4 & P_1 + P_5 - P_3 - P_7 \end{bmatrix}$$

$$\begin{aligned} P_1 &= A(F - H) \\ P_2 &= (A + B)H \\ P_3 &= (C + D)E \\ P_4 &= D(G - E) \\ P_5 &= (A + D)(E + H) \\ P_6 &= (B - D)(G + H) \\ P_7 &= (A - C)(E + F) \end{aligned}$$

- Strassen (1969): $\Theta(n^{2.808})$
  "Gaussian Elimination is Not Optimal"
- (2014): $\Theta(n^{2.373})$
- Known lower bound: $\Omega(n^2)$

1-D DP

Maximal sum subarray (Problem 1.3.5)

- array $A[1 \cdots n], a_i >=< 0$
- to find (the sum of) an MS in A

$$A[-2, 1, -3, 4, -1, 2, 1, -5, 4] \Rightarrow [4, -1, 2, 1]$$

1-D DP

Maximal sum subarray (Problem 1.3.5)

- array $A[1 \cdots n], a_i >=< 0$
- to find (the sum of) an MS in A

$$A[-2, 1, -3, 4, -1, 2, 1, -5, 4] \Rightarrow [4, -1, 2, 1]$$

Trial and error.

- try subproblem MSS$[i]$: the sum of the MS (MS[i]) in $A[1 \cdots i]$
- goal: mss $=$ MSS$[n]$
- question: Is $a_i \in$ MS$[i]$?
- recurrence:

$$\text{MSS}[i] = \max\{\text{MSS}[i-1], ???\}$$

1-D DP

Solution.

- subproblem MSS$[i]$: the sum of the MS *ending with* a_i or 0
- goal: mss $= \max_{1 \le i \le n}$ MSS$[i]$

1-D DP

Solution.

- subproblem MSS$[i]$: the sum of the MS *ending with* a_i or 0
- goal: mss $= \max_{1 \le i \le n}$ MSS$[i]$
- question: where does the MS$[i]$ start?
- recurrence:

$$\text{MSS}[i] = \max\{\text{MSS}[i-1] + a_i, 0\} \text{ (prove it!)}$$

1-D DP

Solution.

- subproblem MSS$[i]$: the sum of the MS *ending with* a_i or 0
- goal: mss $= \max_{1 \le i \le n}$ MSS$[i]$
- question: where does the MS$[i]$ start?
- recurrence:

$$\text{MSS}[i] = \max\{\text{MSS}[i-1] + a_i, 0\} \text{ (prove it!)}$$

- initialization: MSS$[0] = 0$

1-D DP

Code.

```
MSS[0] = 0
For i = 1 to n
  MSS[i] = max{MSS[i-1] + A[i], 0}
return max_{i = 1 to n} MSS[i]
```

1-D DP

Code.

```
MSS[0] = 0
For i = 1 to n
  MSS[i] = max{MSS[i-1] + A[i], 0}
return max_{i = 1 to n} MSS[i]
```

Simpler code.

```
mss = 0
MSS = 0
For i = 1 to n
  MSS = max{MSS + A[i], 0}
  mss = max{mss, MSS}
return mss
```

Pancake sorting (Problem 1.3.1)

How to bring the biggest pancake to the bottom?

Pancake sorting (Problem 1.3.1)

How to bring the biggest pancake to the bottom?

$$T(n) = 2n - 3$$

Pancake sorting (Problem 1.3.1)

How to bring the biggest pancake to the bottom?

$$T(n) = 2n - 3$$

Reference

- "Sorting by Perfix Reversals" by Bill Gates & Papadimitriou

Pancake sorting (Problem 1.3.1)

How to bring the biggest pancake to the bottom?

$$T(n) = 2n - 3$$

Reference

- "Sorting by Perfix Reversals" by Bill Gates & Papadimitriou
- $T(n) \leq \frac{5n+5}{3}$ (1979)
- $T(n) \leq \frac{18n}{11}$ (2009)

(Problem 1.3.8)

How many Big V's are there at most?

(Problem 1.3.8)

How many Big V's are there at most?

"Does A follow B?"

(Problem 1.3.8)

How many Big V's are there at most?

"Does A follow B?"

Don't forget to check it!

Repeated elements (Problem 2.12)

- $R[1 \ldots n]$
- check$(R[i], R[j])$
- $\# > \frac{n}{13}$
- $n \log n$

Repeated elements (Problem 2.12)

- $R[1 \ldots n]$
- check$(R[i], R[j])$
- $\# > \frac{n}{13}$
- $n \log n$

We will talk about an $O(n \log k)$ algorithm and the lower bound.

Reference

"Finding Repeated Elements" by Misra & Gries, 1982

Bolts and nuts (Problem 2.10)

$$A(n) = O(n \log n)$$

Bolts and nuts (Problem 2.10)

$$A(n) = O(n \log n)$$

Reference

$\Theta(n \log n)$ in the worst case:

- "Matching Nuts and Bolts" by Alon *et al.*, $\Theta(n \log^4 n)$
- "Matching Nuts and Bolts Optimality" by Bradford, 1995

Bolts and nuts (Problem 2.10)

$$A(n) = O(n \log n)$$

Reference

$\Theta(n \log n)$ in the worst case:

- "Matching Nuts and Bolts" by Alon *et al.*, $\Theta(n \log^4 n)$
- "Matching Nuts and Bolts Optimality" by Bradford, 1995

$\Omega(n \log n)$ by decision-tree argument.

Bolts and nuts (Problem 2.10)

$$A(n) = O(n \log n)$$

Reference

$\Theta(n \log n)$ in the worst case:

- "Matching Nuts and Bolts" by Alon *et al.*, $\Theta(n \log^4 n)$
- "Matching Nuts and Bolts Optimality" by Bradford, 1995

$\Omega(n \log n)$ by decision-tree argument.

$$3^H \geq L \geq n! \Rightarrow H \geq \log(n!) \Rightarrow H = \Omega(n \log n)$$

K-sorted (Problem 2.9)

Counting inversions (Problem 2.11)

Maxima-finding (Problem 2.14)

Wrong recursions!

Maxima-finding (Problem 2.14)

Wrong recursions!

3D?

Maxima-finding (Problem 2.14)

Wrong recursions!

3D?

Lower bound $\Omega(n \log n)$!