

2017/01/31. Algorithm Analysis (Week 1)

Example) 0, 100, -50, -60, 30, -10, 40, 5, 45, -2, 1

Problem) Given a sequence of integers $\{a_1, a_2, \dots, a_n\}$, find a contiguous subsequence whose sum is the maximum. (where ^{sum} value of empty subsequence is defined as 0)

Alg 1 (Exhaustive Method) ^{very naive!}

```
sol ← 0
for i ← 1, ..., n do
  for j ← i, ..., n do
    calculate sum(i to jth)
return sol;
```

$O(n^3)$

Thm) Alg 1 runs in $O(n^3)$ time.

pt) # of basic operations is proportional to.

$$\left(\sum_{i=1}^n \sum_{j=i}^n (j-i+1) \right) = \sum_{i=1}^n \sum_{j'=1}^{n-i+1} j' = \frac{n(n+1)(n+2)}{6}$$

$$\Rightarrow \underline{O(n^3)}$$

⊗ Proof of Algorithm

① running time analysis

② proof of correctness

partial proof 1: sol is correct if alg terminates
partial proof 2: alg always terminates.

Alg 2

```
sol ← 0
for i ← 1, ..., n do
  sumi,i ← 0
  for j ← i, ..., n do
    sumi,j ← sumi,j-1 + aj
    if sumi,j > sol then
      sol ← sumi,j
return sol
```

$O(n^2)$

Thm) Alg 2 runs in $O(n^2)$ time.

pt) # of basic operations is proportional to

$$\left(\sum_{i=1}^n \sum_{j=i}^n 1 \right) = \sum_{i=1}^n (n-i+1) \Rightarrow \underline{O(n^2)}$$

Thm²) Alg 2 is correct.

pt) The algorithm returns $\max [\{ \text{sum}_{i,j} \mid 1 \leq i \leq j \leq n \} \cup \{0\}]$

hence it suffices to prove that sum_{i,j} is equal to $(\sum_{k=i}^j a_k)$: sum of subsequence ①

Induction

Basic Step) When $j-i=0$ then $\text{sum}_{i,j} = \text{sum}_{i,i} = \frac{\text{sum}_{i,i-1} + a_i}{=0} = a_i$ therefore ① holds.

Induction Step) When $j-i = d_0 + 1$

$$\begin{aligned} \text{then } \text{sum}_{i,j} &= \text{sum}_{i,j-1} + a_j = \sum_{k=i}^{j-1} a_k + a_j \text{ (by induction hypothesis)} \\ &= \sum_{k=i}^j a_k. \text{ therefore ① holds.} \end{aligned}$$

⇒ Since Alg 2 trivially terminates, Alg 2 is correct by thm² and runs in $O(n^2)$ by thm¹

Alg 3 (Idea from merge sort)

function MaxSubseq(s, t)

⇒ return MaxSubseq(1, n)

if $s = t$ then

returns $\max(ds, 0)$

O/W

$$m \leftarrow \left\lfloor \frac{s+t}{2} \right\rfloor$$

$$sol_e \leftarrow \text{MaxSubseq}(s, m)$$

$$sol_r \leftarrow \text{MaxSubseq}(m+1, t)$$

$$subsol_e \leftarrow 0, subsol_r \leftarrow 0$$

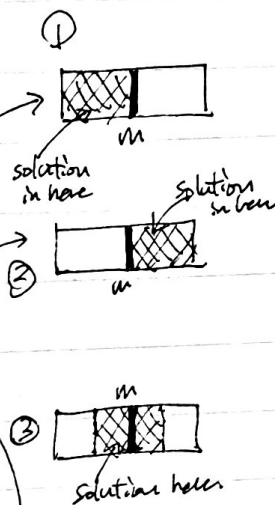
for $i \leftarrow m, m+1, \dots, s$ do

if $\sum_{k=i}^m a_k > subsol_e$ then
 $subsol_e \leftarrow \sum_{k=i}^m a_k$

for $i \leftarrow m+1, \dots, t$ do

if $\sum_{k=i}^{m+1} a_k > subsol_r$ then
 $subsol_r \leftarrow \sum_{k=i}^{m+1} a_k$

return $\max \begin{cases} sol_e \\ sol_r \\ subsol_e + subsol_r \end{cases}$



Thm) Alg 3 runs in $O(n \log n)$ time. (Just like Mergesort)

pt) When $s = t$ Let $T(1) = 1$ for max(a,b) operation.

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

call two recurrences with half range. ↗ subseq operations.

Let $n = 2^k$ and $k = \log_2 n$.

$$T(2^k) = 2T(2^{k-1}) + 2^k$$

$$= 2^k T(1) + k2^k$$

$$\Rightarrow T(n) = nT(1) + (\log_2 n)n$$

$$= n + n \log_2 n = O(n \log n) \quad \square$$

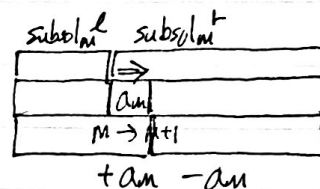
Alg 4 (linear time)

$$\text{subsol}_0^L \leftarrow 0, \text{subsol}_0^R \leftarrow \left(\sum_{i=1}^n a_i \right)$$

for $m \leftarrow 1, 2, \dots, n-1$ do.

$$\begin{aligned} \text{subsol}_m^L &\leftarrow \max \left[\left\{ \sum_{k=i}^m a_k \mid 1 \leq i \leq m \right\} \cup \{0\} \right] \\ \text{subsol}_m^R &\leftarrow \max \left[\left\{ \sum_{k=m+1}^i a_k \mid m+1 \leq i \leq n \right\} \cup \{0\} \right] \end{aligned}$$

$$\begin{aligned} \text{subsol}_m^L &\leftarrow \max \left\{ \begin{array}{l} \text{subsol}_{m-1}^L + a_m \\ 0 \end{array} \right\} \\ \text{subsol}_m^R &\leftarrow \max \left\{ \begin{array}{l} \text{subsol}_{m-1}^R - a_m \\ 0 \end{array} \right\} \end{aligned}$$



"single run of $\Rightarrow \text{subsol}_m^L, \text{subsol}_m^R$ etc."

$$\text{sol} \leftarrow 0$$

for $m \leftarrow 1, 2, 3, \dots, n-1$ do.

if $(\text{subsol}_m^L + \text{subsol}_m^R) \geq \text{sol}$ then

$$\text{sol} \leftarrow (\text{subsol}_m^L + \text{subsol}_m^R)$$

return sol.

⊗ Obs) for $m > 1$, we have

$$\text{subsol}_m^L = \max \left\{ \begin{array}{l} \text{subsol}_{m-1}^L + a_m \\ 0 \end{array} \right\}$$

$$\text{subsol}_m^R = \max \left\{ \begin{array}{l} \text{subsol}_{m-1}^R - a_m \\ 0 \end{array} \right\}$$

pt)

\Rightarrow If $\text{subsol}_m^L \neq 0$, the contiguous subsequence yields subsol_m^L has to be obtained from $\max(\text{subsequences ending at } a_{m-1}) + a_m$

\Rightarrow If $\text{subsol}_m^R \neq 0$, the contiguous subsequence yields subsol_m^R has to be obtained from $\max(\text{subsequences start at } a_{m-1}) - a_m$.
 $\quad \quad \quad = \text{subsol}_{m-1}^R$ □