CMPT307: Divide-and-Conquer

Week 4-1

Xian Qiu

Simon Fraser University

xianq@sfu.ca

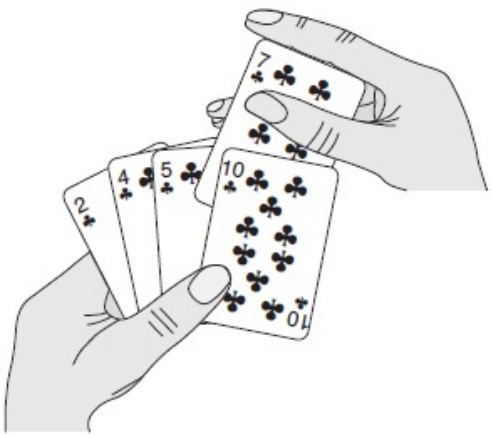
Outline

- sorting
- maximum subarray
- matrix multiplication

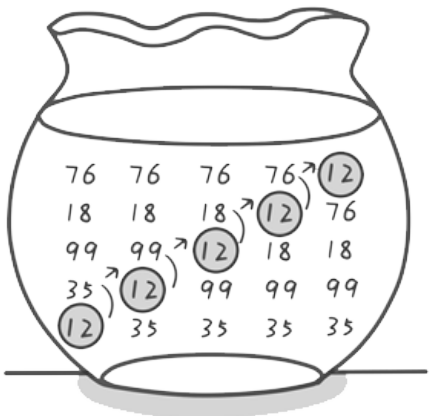
Sorting

Sorting problem

- input: a sequence of n numbers $a_1, \ldots, a_n$
- output: a permutation $a'_1, \ldots, a'_n$ such that $a'_1 \leq \ldots \leq a'_n$



insertion sort



bubble sort

Divide-and-Conquer

Divide-and-conquer approach

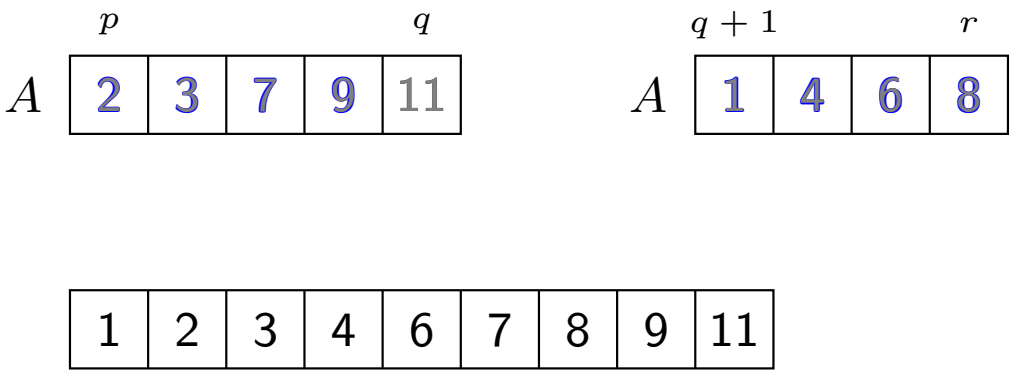
1. divide the problem into subproblems of smaller size each
2. conquer the subproblems by solving them recursively
3. combine the subsolutions correctly

merge sort for the sorting problem

- ▷ divide a sequence into two, of equal size
- ▷ sort the two subsequences recursively using merge sort
- ▷ merge the two sorted subsequences to yield the sorted answer

Merge

MERGE(A,p,q,r)

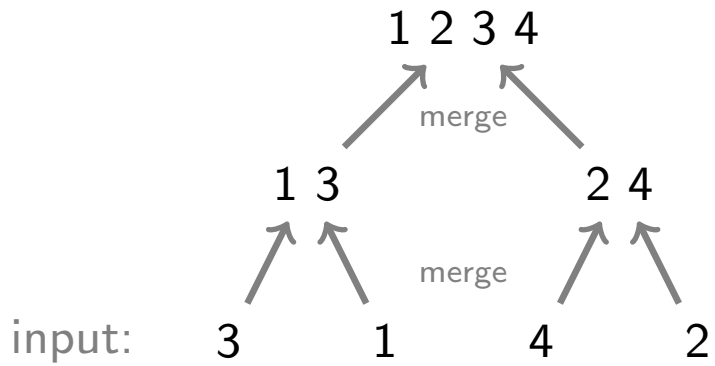


running time?

Pseudocode

MERGE-SORT(A,p,r)

```
1 if p < r then
2     q = ⌊(p + r)/2⌋;
3     MERGE-SORT(A,p,q);
4     MERGE-SORT(A,q + 1,r);
5     MERGE(A,p,q,r);
```



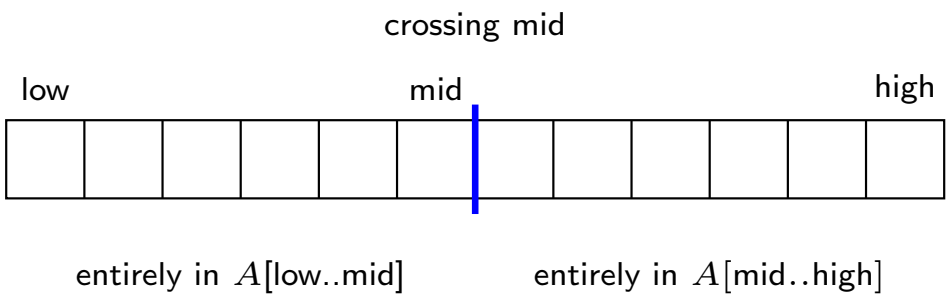
$T(n) = 2T(n/2) + \Theta(n) \quad \Rightarrow \quad T(n) = \Theta(n \log n)$ (master theorem)

Maximum Subarray

Maximum Subarray problem

- given a sequence of numbers $a_1, \ldots, a_n \in \mathbb{R}$
- find a contiguous subsequence with maximum sum

- example: $-1, 3, -1, 5, 1, 6, -3, 1$
- brute force?
- divide-and-conquer?



Combining Solutions

MAX-CROSS(A,low,mid,high)

- ▷ find i s.t. $A[i..\text{mid}]$ has the max sum
- ▷ find j s.t. $A[\text{mid}+1..j]$ have max sum
- ▷ output (i, j, sum), where $sum =$ total sum of $A[i..j]$

to return the maximum of

- ▷ $\max A[\text{low..mid}]$
- ▷ $\max A[\text{mid+1..high}]$
- ▷ MAX-CROSS(A,low,mid,high)

Pseudocode

FIND-MAXIMUM-SUBARRAY(A,low,high)

```
1 if high == low then
2     return (low, high, A[low]);
3 else
4     mid = ⌊(low + high)/2⌋;
5     L = FIND-MAXIMUM-SUBARRAY(A,low,mid);
6     R =FIND-MAXIMUM-SUBARRAY(A,mid+1,high);
7     C = MAX-CROSS(A,low,mid,high);
8     return max {L, R, C} w.r.t. sum property;
```

running time: $T(n) = 2T(n/2) + \Theta(n)$

Matrix Multiplication

given $A = (a_{ij}), B = (b_{ij}) \in \mathbb{R}^{n \times n}$, compute $C = A \cdot B$

$$C = (c_{ij}) = \begin{bmatrix} & & \\ a_{i,1} & \dots & a_{i,k} \\ & & \end{bmatrix} \begin{bmatrix} & b_{1,j} & \\ & \vdots & \\ & b_{k,j} & \end{bmatrix}$$

$$c_{ij} = \sum_{k=1}^{n} a_{i,k} b_{k,j}$$

running time?

Divide-and-Conquer

$$\begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \cdot \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}$$

$$\begin{aligned} C_{11} &= A_{11}B_{11} + A_{12}B_{21} \\ C_{12} &= A_{11}B_{12} + A_{12}B_{22} \\ C_{21} &= A_{21}B_{11} + A_{22}B_{21} \\ C_{22} &= A_{21}B_{12} + A_{22}B_{22} \end{aligned}$$

Pseudocode

MATRIX-MULTIPLY(A, B)

```
1 if n == 1 then
2     c_11 = a_11 · b_11;
3 else
4     partition A, B into 2 × 2 blocks;
5     C_11 = MATRIX-MULTIPLY(A_11, B_11) + MATRIX-MULTIPLY(A_12, B_21);
6     C_12 = MATRIX-MULTIPLY(A_11, B_12) + MATRIX-MULTIPLY(A_12, B_22);
7     C_21 = MATRIX-MULTIPLY(A_21, B_11) + MATRIX-MULTIPLY(A_22, B_21);
8     C_22 = MATRIX-MULTIPLY(A_21, B_12) + MATRIX-MULTIPLY(A_22, B_22);
9 return C;
```

running time: $T(n) = \Theta(1) + 8T(n/2) + \Theta(n^2)$

$$T(n) = O(n^3)$$

Strassen's Method

define new matrices $S_1, \ldots, S_{10}$ (*cf.* textbook), where

$$S_i = A_{*,*} \pm A_{*,*} \text{ or } B_{*,*} \pm B_{*,*} \quad \in \mathbb{R}^{\frac{n}{2} \times \frac{n}{2}}$$

define new matrices $P_1, \ldots, P_7$ (subproblems)

$$P_1 = A_{11}S_1, \quad P_2 = S_2B_{22}, \quad P_3 = S_3B_{11}, \quad P_4 = A_{22}S_4$$
$$P_5 = S_5S_6, \quad P_6 = S_7S_8, \quad P_7 = S_9S_{10}$$

C can be calculated by $P_1, \ldots, P_7$

$$C_{11} = P_5 + P_4 - P_2 + P_6$$
$$C_{12} = P_1 + P_2$$
$$C_{21} = P_3 + P_4$$
$$C_{22} = P_5 + P_1 - P_3 - P_7$$

running time: $T(n) = 7T(n/2) + \Theta(n^2)$