Department of Computer Science and Engineering
National Sun Yat-sen University
Design and Analysis of Algorithms - Final Exam., Jan. 5, 2021

1. Explain each of the following terms. (20%)
   (a) NP, NP-complete
   (b) a biconnected graph
   (c) the 1-center problem
   (d) the 2-D ranking problem
   (e) an AVL tree
2. Present an algorithm for solving the *minimum spanning tree* problem on graphs. Analyze the time complexity of your algorithm. (12%)
3. Suppose that n planar points with their *Voronoi diagram* are given. Please present an efficient algorithm to construct the *convex hull* of these n points based on the Voronoi diagram. The time complexity of your algorithm should be O(n). (12%)
4. What data structures are used for *depth-first search*, *breadth-first search* and *best-first search*, respectively? (9%)
5. In the 0/1 *knapsack* problem, there are n objects with knapsack capacity M, where the profit of each object i is p_i and the weight is w_i,$1 \le i \le n$. Please give the *dynamic programming* formula for solving the 0/1 knapsack problem. In the formula, let $f_i(Q)$ be the maximum profit obtained by objects 1,2,3,…,i with capacity Q.(12%)
6. Both 1023 and 9765624 can be divided by 11. There is a simple rule to decide whether a given number N can be divided by 11 or not. The rule is based on the sum of some digits and further calculation. Please give the rule. (8%)
7. Prove that the *partition decision* problem polynomially reduces to the *bin packing decision* problem. (12%)
8. n numbers a_1, a_2, …, a_n, are input sequentially. When each a_i is input, a_i can be appended into the front of a list if a_i is less than the current front, or appended into the tail of the list if a_i is larger than the current tail, or discarded for any reason. The resulting list should be as long as possible. Note that the initial list is empty.
   (a) Suppose the numbers are2, 5, 4, 8, 6, 3, 9, 7, 1. What is the length of the longest resulting list? Write down the content of one of the longest lists.

(5%)

(b) Design an algorithm to find the content of one of the longest lists. Your algorithm should be of polynomial time. And analyze the time complexity of your algorithm. (10%)

Design and Analysis of Algorithms

Final Exam., Jan. 5, 2021 參考解答

1.

(a).

NP：給予一個解答，可以在多項式時間驗證其正確性的decision problem。

NPC：NP與NP-Hard的交集；NP-Hard：所有的NP問題均可reduce至某問題A，則稱A為NP-hard。

(b).

在Biconnected graph中，任意兩個節點均至少有兩條不相交的路徑。

(c).

找出一個最小半徑的圓，使得所有點可被此圓包含。

(d).

rank(A): A可以dominate的點個數。

Dominate: P1(x1,y1) dominates P2(x2,y2) iff x1>x2 and y1>y2.

(e).

平衡的二元樹，其中每一node的左子樹樹高與右子樹樹高相差不超過1。

2.

用prim's algorithm

Step 1：假設一個無向的完全圖 G={V,E}，令A集合為{x}，x為V中任意一點，B=V-{x}。不存在於E的edge之權重件設為無限大。

Step 2：選取一條edge(u,v)，u屬於A集合，v屬於B集合，且(u,v)權重為最小。

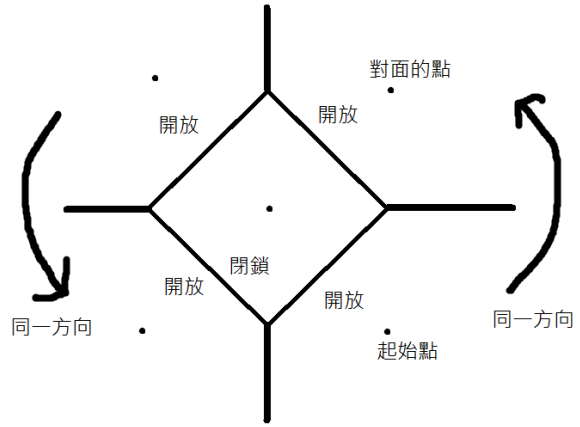
Step 3：選到(u,v)後，將v加入A，而且將v從B刪除。接著更新B中每一個點y與A之連線權重，若(v,y)比較小，則更新之；否則不更新。

Step 4: 重複Step 2~Step 3，直到B集合為空，代表結束。

假設|V|=n，亦即有n個點。每次選一個點，以及更新，需O(n)時間。迴圈需執行n-1次，所以時間複雜度為O(n*(n-1)) => $O(n^2)$。

3.

從有開放勢力範圍的點中任意取一點當起點，沿著同一方向，連向將兩個開放勢力範圍切開的邊對面之點，直到連回自己，形成convex hull。最多共有n個開放勢力範圍，時間複雜度為O(n)。



4.

depth-first search: stack

breadth-first search: queue

best-first search: priority queue

5.

$f_i(Q) = \max\{f_{i-1}(Q-w_i), f_{i-1}(Q) + p_i\}$

$f_0(0) = f_i(0) = f_0(Q) = 0 \quad \text{for } 1 \le i \le n,\ 0 < Q \le M$

6.

奇數位各數字相加以及偶數位各數字相加，如兩者的差為11的倍數或為0，則該整數可被11整除。

7.

partition decision：有A={a_1,a_2,....,a_n}，判斷是否能將此集合分成兩個子集合C,D，使得C的元素之總和等於D的元素之總和。

bin packing decision:有{b_1,b_2,....,b_n}，決定是否能將所有b_i裝入k個bin。

轉換方式：令 $a_i=b_i$, $1 \leqq i \leqq n$, bin packing要將所有b_i裝入k=2個bin，且bin的大小為$y=(a_1+a_2+....+a_n)/2$。

(1)若partition decision有解，則bin packing decision亦有解：

若partition decision有解，則存在兩個子集合C,D，C的元素總和等於y(亦即A總和之一半)，且D的元素總和亦等於y。由於$a_i=b_i$，因此可將集合C的元素裝入一個bin，將集合D中的元素裝入另一個bin，因為bin的大小正好為y。故，bin packing decision亦有解。

(2)若bin packing decision有解，則partition decision亦有解：

若bin packing decision有解，則代表可將b_i分成兩個子集合，且其子集合總和各為y。由於$a_i=b_i$，因此可將第一個bin的內容放入C，其餘則放入D，此時C的總和與D的總和相等(均為y)。故，partition decision亦有解。

以上轉換步驟可以在多項式時間內完成。

8.

(a) 134589 or 134569 or 134567

(b)

input = a_1,a_2,a_3,...,a_n

max = 0, ans = 0

for i = 1 to n ------O(n)

找出LIS(i,n)=以a_i為起點a_i~a_n的最長遞增子序列之長度 ------O(n log n)

找出LDS(i,n)=以a_i為起點a_i~a_n的最長遞減子序列之長度 ------O(n log n)

length= LIS(i,n) + LDS (i,n) ----O(1)

if length > max: ------O(1)

max = length

在找尋LIS或LDS過程中，亦可保留LIS或LDS序列內容。

時間複雜度 : O(n)*O(n log n) = O(n^2 log n)