Department of Computer Science and Engineering
National Sun Yat-sen University
Design and Analysis of Algorithms - Final Exam., Jan. 16, 2018

1. Explain each of the following terms. (24%)
   (a) NP, NP-complete
   (b) breadth-first search
   (c) G^2 of a graph G
   (d) rectilinear m-center problem
   (e) AVL tree
2. Design an algorithm for finding the *minimum spanning tree* of a given graph. And analyze the time complexity of your algorithm. (12%)
3. For solving the convex hull problem, please describe the *Graham scan* method. (10%)
4. (a) Present the *dynamic programming* method for calculating the LCS (*longest common subsequence*) length of two given sequences. (6%)

   (b) Given one sequence of distinct positive integers, the LIS (*longest increasing subsequence*) problem is to find an increasing subsequence with the maximum length. For example, for 6, 3, 5, 9, 7, 8, its LIS solution is 3, 5, 7, 8, whose length is 4. Please design an algorithm with polynomial time for obtaining the LIS length of a given sequence. (Hints: You can utilize the LCS algorithm as a base.) (6%)
5. In the 1-SAT problem, each clause has exactly one literal. Please present an algorithm with polynomial time to solve the 1-SAT problem. Analyze the time complexity of your algorithm. (10%)
6. Given a binary number (with n bits) $B=b_1b_2\ldots b_n$, let $f(i)=(b_ib_{i+1}\ldots b_{i+w-1}) \bmod p$, where p is a positive integer and w is the window size. Suppose $f(i)$ has been calculated and its value is r. Please derive the method to calculate $f(i+1)$ by using r. (10%)
7. The *quadratic selection sort* is described as follows: Divide the n input elements into k groups of k elements each, where $k = \sqrt{n}$. Find the largest element of each group and insert it into an auxiliary array a. Find the largest of the elements in a. This is the largest element of the file. Then replace this largest element in a by the next largest element of the group from which it came. Again, find the largest

element in *a*. This is the second largest element of the file. Repeat the process until the file has been sorted. Use the following 16 elements to explain how this algorithm works: 14, 5, 3, 8, 13, 15, 2, 9, 10, 4, 1, 6, 16, 7, 11, 12. (10%)

8. Prove that the *partition* decision problem polynomially reduces to the *bin packing* decision problem. (12%)

Design and Analysis of Algorithms

Final Exam., Jan. 16, 2018 參考解答

1(a).

NP：可以用non-deterministic polynomial algorithm解決的decision problem

NP-complete：同時為NP與NP-hard的問題

1(b).

breadth-first search(BFS)：針對tree的搜尋，先搜尋root node，接著搜尋child node時同一層的node搜尋過後，才會往下一層搜尋。所需使用的資料結構是queue。

1(c).

G^2 of a graph G：與G有相同點數，G中兩點之間有長度不超過2的路徑相連時(一個點透過中間點，連到另一個點，謂之路徑長度為2)，在G^2將此兩點以edge相連起來。

1(d).

rectilinear *m*-center problem: 給定n個點，找到包含所有點的m個正方形，其邊長為最短，且這些正方形的邊與x軸或y軸平行。

1(e).

AVL tree: 一棵高度平衡的binary search tree，所有點的hb值不可差超過1(即可以為1, 0, -1)，其中hb(v)= (height of right subtree) – (height of left subtree)

2.

利用Prim's algorithm 解法。

Input: A undirected connected graph G = (V, E).
Output: The minimum spanning tree of G.
Step 1: $x \in V$. Let $A = \{x\}$, $B = V - \{x\}$.
Step 2: Select $(u, v) \in E$, $u \in A$, $v \in B$ such that (u, v) has the smallest weight between A and B.
Step 3: Put (u, v) in the tree. $A = A \cup \{v\}$, $B = B - \{v\}$.
Step 4: If $B = \emptyset$, stop; otherwise, go to Step 2.

實作上，Step 2，集合B中的每一個node u，只要保留一條連到A的edge。
每次選取連接A與B的edge e之後，都要更新u所保留的edge。
因此Step 2需要O(n) 時間，其中 n = |V| 。
Step 2~Step 4的iteration次數為n-1。
故Time complexity: $O(n^2)$, n = |V|

3.

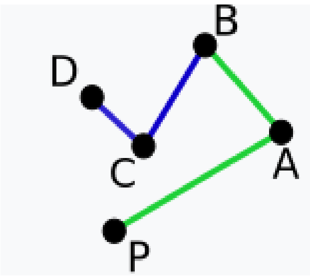
Step 1: 選取某一個內部點 u (將所有點的 x 座標取平均，y 座標取平均)，當作原點。

Step 2: 求出其他點相對於原點的角度，再根據所得的角度逆時針排序。

Step 3: 依照排序的順序，每三個點形成一個角度，若該角度為凹角(大於 180 度)，則此三點的中間點不會出現在 Convex Hull，刪除該點。

Step 4: 當所有點都檢查過後，回到起始點，剩餘的點就是 Convex Hull 的點，而且點已經逆時針排序。

如下圖，角 DCB 是凹角，將 C 點刪除。



4(a).

Let $A = a_1\, a_2 \ldots a_m$ and $B = b_1\, b_2 \ldots b_n$

Let $L_{i,j}$ denote the length of the longest common subsequence of $a_1\, a_2 \ldots a_i$ and $b_1\, b_2 \ldots b_j$

$$L_{i,j} = \begin{cases} L_{i-1,j-1} + 1 & \text{if } a_i = b_j \\ \max\{ L_{i-1,j}, L_{i,j-1} \} & \text{if } a_i \neq b_j \end{cases}$$

$L_{0,0} = L_{0,j} = L_{i,0} = 0$ for $1 \le i \le m$, $1 \le j \le n$.

由上述公式，最後所求得的 $L_{m,n}$ 就是 LCS 的長度

4(b).

Let $A = a_1\, a_2 \ldots a_m$

<u>Step1:</u> 先對 A 字串做由小至大的排序，排序後存為 B 字串 $B = b_1\, b_2 \ldots b_m$ ($b_1 < b_2 < \ldots < b_m$)。

<u>Step2:</u> 計算 A、B 的 LCS 長度，結果即為 A 字串的 LIS 長度。

Example: sequence A= 6, 3, 5, 9, 7, 8, 排序後 B = 3, 5, 6, 7, 8, 9

LCS(A,B) = LIS(A) = 3, 5, 7, 8, length = 4

5.

假設有 n 個 clauses

(1) n 個 clauses 兩兩配對，可得 n(n-1)/2 個配對。

(2)對於每個配對中的 clause A 跟 clause B，判斷是否為相同變數，而且一個為正(例如 x)，另一個為負(例如-x)。若其中一個配對有此情形，則輸出 unsatisfiable；若均無，則輸出 satisfiable。

Time complexity:共有 n(n-1)/2 個配對，全部需要判斷 n(n-1)/2 次，因此時間複雜度為 $O(n^2)$

6.

$f(i) = (b_i b_{i+1} \dots b_{i+w-1})\ mod\ p\ =\ r$

因此，$f(i) = mp + r$

$f(i+1) = (b_{i+1} b_{i+2} \dots b_{i+w})\ mod\ p$

$\qquad = (2(mp + r) - b_i * 2^w + b_{i+w})\ mod\ p$

$\qquad = (2r - b_i * 2^w + b_{i+w})\ mod\ p$

Answer: $(2r - b_i * 2^w + b_{i+w})\ mod\ p$

7.

n=16, k=4, 分成四組：{14,5,3,8}, {13,15,2,9}, {10,4,1,6}, {16,7,11,12}
每組選最大值，存入 a，得到 a={14,15,10,16}

輸出 a 的最大值 16，並從與 16 同組拿出最大值(16 除外)，為 12，存入 a：
output={16}, a={14,15,10,12},
輸出 a 的最大值 15，並從與 15 同組拿出最大值(15 除外)，為 13，存入 a：
output={16,15},a={14,13,10,12}
以此類推，各步驟所得結果如下：
output={16,15,14}, a={8,13,10,12}
output={16,15,14,13}, a={8,9,10,12}
output={16,15,14,13,12}, a={8,9,10,11}
output={16,15,14,13,12,11}, a={8,9,10,7}
output={16,15,14,13,12,11,10}, a={8,9,6,7}
output={16,15,14,13,12,11,10,9}, a={8,2,6,7}
output={16,15,14,13,12,11,10,9,8}, a={5,2,6,7}
output={16,15,14,13,12,11,10,9,8,7}, a={5,2,6}
output={16,15,14,13,12,11,10,9,8,7,6}, a={5,2,4}
output={16,15,14,13,12,11,10,9,8,7,6,5}, a={3,2,4}
output={16,15,14,13,12,11,10,9,8,7,6,5,4}, a={3,2,1}
output={16,15,14,13,12,11.10,9,8,7,6,5,4,3}, a={2,1}
output={16,15,14,13,12,11.10,9,8,7,6,5,4,3,2}, a={1},
output={16,15,14,13,12,11.10,9,8,7,6,5,4,3,2,1}, a={},

8.

partition $\propto$ bin packing
instance of partition：

給予$S = \{a_1, a_2, \ldots, a_n\}$為 n 個正整數的集合(每個$a_i$皆為正整數)，是否存在P滿足：

$$\sum_{a_i \in P} a_i = \sum_{a_i \notin P} a_i \text{ ，} P \subseteq S$$

instance of bin packing problem：

B為箱子的數量，C為每個箱子的容量，c_i為物件的大小，$1 \le i \le n$。

令$B = 2$、$C = \sum_{1 \le i \le n} a_i/2$、$c_i = a_i$、$1 \le i \le n$

(1) Partition => bin packing

如果存在一個集合P滿足$\sum_{a_i \in P} a_i = \sum_{a_i \notin P} a_i$，則

$$\sum_{1 \le i \le n} a_i/2 = \sum_{a_i \in P} a_i = \sum_{a_i \notin P} a_i = C \ \ (\because \sum_{a_i \in P} a_i + \sum_{a_i \notin P} a_i = \sum_{1 \le i \le n} a_i)$$

所以可以把c_i成兩堆，若$a_i \in P$，則c_i放在箱子B_1；若$a_i \notin P$，則c_i放在箱子 B_2，此時滿足$\sum_{c_i \in B_1} c_i = \sum_{c_i \in B_2} c_i \le C$。因此，若存在 partition，則 bin packing problem 可找到一種組合方法，能夠把n個物件裝在兩個箱子中。

(2) Partition <= bin packing

如果能夠用兩個箱子把n個項目$c_1, c_2, \ldots, c_n$裝起來，其中$\sum_{1 \le i \le n} c_i = 2C$，則在箱子$B_1$中所有物件加總為$\sum_{c_i \in B_1} c_i = C$，同理$\sum_{c_i \in B_2} c_i = C$。

明顯存在 partition，令$P = \{a_i | a_i = c_i, c_i \in B_1\}$, $\bar{P} = \{a_i | a_i = c_i, c_i \in B_2\}$

滿足$\sum_{a_i \in P} a_i = \sum_{a_i \notin P} a_i$，也就是可以 partition。

因此可證得 partition $\propto$ bin packing。