Sprout 2020 Algorithm - Week 9

Author: 陳楚融

Problem 1

歸約過程中，各步驟依序執行，複雜度互相獨立，因此 P 問題之複雜度，因此 $p(n) = O(f_1(n) + f_2(n) + f_3(n))$

根據 week 2 ， $O(f_1(n) + f_2(n) + f_3(n)) = O(max(f_1(n), f_2(n), f_3(n)))$ ，因此 P 問題存在一個複雜度為 $O(max(f_1(n), f_2(n), f_3(n)))$ 的解

Problem 2

令 $f(n), g(n)$ 與 Q, P 問題之複雜度分別為 $O(f_1(n)), O(f_2(n)), O(f_3(n)), O(p(n))$

由 Problem 1 之結論可得 $p(n) \in O(max(f_1(n), f_2(n), f_3(n)))$ ，因此 P 問題存在一個多項式複雜度的解

Problem 3

假設 Q 問題的複雜度下界為 $\Omega(2^n)$ ， P 問題的複雜度下界為 $\Omega(n^2)$ 且 $f(n), g(n)$ 的複雜度皆為 $O(n)$ ，可得 P 問題至少存在一個複雜度為 $O(2^n)$ 的解，與假設不矛盾，因此造成原命題矛盾

Problem 4

假設 heap 的插入、刪除、查詢極值之複雜度皆為 $O(1)$

給定一組 n 個元素的陣列以及一個空的動態陣列 B ，已知動態陣列的插入複雜度為 $O(1)$

對陣列元素進行排序，可將 n 個陣列元素全部插入 heap 中，然後重複將極值插入 B 、刪除當前極值，共 n 次

總時間複雜度為 $O(n)$ ，然而排序複雜度下界為 $O(n \log n)$ ，與假設矛盾，得證不可能存在一種 heap 結構之插入、刪除、查詢極值之複雜度皆為 $O(1)$

Problem 5

將輸入（ w ）中每個元素 a_i 轉換為平面上的點 $(a_i, 0)$ ，此過程（ $f(w)$ ）複雜度為 $O(n)$

若有元素重複則至少存在兩重合點，其距離為 0 ；若沒有元素重複則任兩點 $(a_i, 0)$, $(a_j, 0)$, $a_i \neq a_j$ 之距離 $\sqrt{(a_i - a_j)^2 + (0-0)^2} = |a_i - a_j| > 0$

答案轉換 （ $g(Q(f(w)))$ ）之複雜度為 $O(1)$

至此將原問題線性歸約到平面最近點對問題

Problem 6

把陣列的每個元素視為一個節點，元素值為指向的節點編號，如第 i 個元素為 $node_i$ ，指向 $node_{a_i}$ 。若從 $node_n$ 作為起點開始遍歷，由於 $node_n$ 不被任何節點指向，因此不會回到 $node_n$ ，若重複走過節點 $node_x$ ，表示 x 為陣列中的重複元素 ，且 $node_x$ 會形成環的起點，因為環形成前遍歷到的節點皆相異，因此圖的大小不超過 n

使用兩指針 s, t ，皆從起點開始，s 每走一步， t 走兩步。當指針第一次相遇時，設 s 已走 k 步 ，令相遇點為 $node_c$ 、 $node_c$ 與起點的距離為 d 、環的長度為 L ：

$$\because d = k - qL = 2 * k - pL,\ q, p \in \mathbb{Z}_0^+$$
$$\therefore L \mid k \implies d = rL, r \in N$$

接著將 s 移回， s, t 同時開始每次走一步，當 s 第一次走到 $node_x$ 時，設 s 已走 k 步 ， t 與起點的距離為 $d_t = d + k - qL = k + (r - q)L$ ：

$$\because \begin{cases} d_t = d + k - qL = k + (r - q)L \\ k \le d_t < k + L \end{cases}$$
$$\therefore r - q = 0 \implies d_t = k$$

因此 t 也在 $node_x$ 上，也就是說當 s, t 第一次相遇時，相遇點即為環的起點，也就是陣列中的其中一個重複元素

共使用兩變數 s, t ，空間複雜度為 $O(1)$

指針第一次遍歷時，當 s 進入環內後，s 每走一步， s, t 之距離皆縮短 1 步，再走最多 $L - 1$ 步會相遇，因此時間複雜度為 $O(n)$

第二次遍歷時， s 第一次走到 $node_x$ 便與 t 相遇，因此時間複雜度為 $O(n)$ ，得總時間複雜度為 $O(n)$