Algorithms PA2 Report R09945021 洪怡庭

1. Dynamic Programming:

   Subproblem 的 recursion 關係可分為三種 case:

   (1) Case1 kj ∈ C and k ∉ [i, j]: M[i][j] = M[i][j-1]

   (2) Case2 ij ∈ C: M[i][j] = M[i+1][j-1]+1

   (3) Case3 kj ∈ C and k ∈ [i, j]: M[i][j] = max{M[i][j-1], M[i][k-1]+1+M[k+1][j-1]}

2. Implementation:

   Allocate 兩個 2N*2N 的 array M 與 C，分別紀錄 chord number 及所屬 case。

   (1) Number of chords:

   使用 top-down 方式減少運算量，可不必填完所有表格，且每次呼叫 recursion 前先檢查是否已計算過。boundary condition 為當 i == j 時，M[i][j] = 0。同時在 C 中填入所屬 case。

   (2) Find the chords:

   根據記錄的 case，用 recursion 方式回推源自於哪一條 chord。

   Case1: 不新增 chord

   Case2: 新增 chord kj 並繼續找[i+1, j-1]

   Case3: 若 M[i][k-1]+1+M[k+1][j-1] > M[i][j-1]，新增 chord kj 並繼續找 [k+1, j-1]及[i, k-1]，否則不新增 chord。

3. Findings:

   紀錄 M 與 C，故 Space complexity: $O(N^2)$。

   若以 bottom-up 方式計算，由於要將表格全部填完，Time complexity 也為 $O(N^2)$，但在此題中，不需全部計算，因此使用 top-down 以 recursion 方式僅計算所需位置可省去較多時間。