# 演算法 PA2

B10901041 林新晨

**Data Structure**

Step 1: Let data[a] = b and data[b] = a for chord $\overline{ab}$.

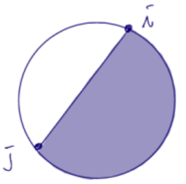
Step 2: Top-down dp store $MPS(i,j)$ in mps_table.

Step 3: Top-down search the mps chords and store them in chord_list.

(Step 3 is similar to step 2.)

In Step 2:

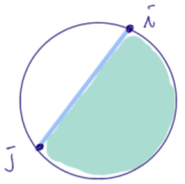
$$MPS(i,j)=\begin{cases}0 & \text{if } i \geq j\\ 1 & \text{if } data[i]=j \text{ and } i+1=j\\ 1+MPS(i+1,j-1) & \text{if } data[i]=j \text{ and not } (i+1=j)\\ 1+MPS(i+2,j) & \text{if } data[i]=i+1 \text{ and } data[i]<j\\ \max(1+MPS(i+1,data[i]-1)+MPS(data[i]+1,j),MPS(i+1,j)) & \text{if } data[i]>i+1 \text{ and } data[i]<j\\ MPS(i+1,j) & \text{otherwise}\end{cases}$$



1. if $i \geq j$, return 0

2. 若已經儲存進 dp table, return 儲存的值

3.

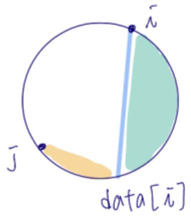


data[i]=j, chord 剛好在範圍上

if(i+1==j) 弦連接隔壁2點.就不用繼續找

else: 取弦再往內找[i+1, j-1]

4.



data[i]<j, chord 在範圍內

if(data[i]=i+1) 弦連接隔壁2點

找[i+2, j]範圍

else

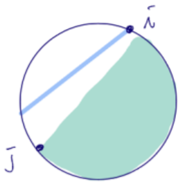
chooseChord = 1+範圍[i+1, data[i]-1]

+範圍[data[i]+1, j]

notChooseChord = 範圍[i+1, j]

選 max{chooseChord, notChooseChord}

5.



data[i]>j, chord 在範圍外

找[i+1, j]

**Findings**

1. Time Complexity is $O(n^2)$.

Because of memorization, table[i, j] is calculated only once.

2. Space Complexity is also $O(n^2)$.

The mps_table is $n \times n$.

3. Note that we should put the current mps chord in vector before we call the next recurring function so that the final chord_list is in increasing order.