CH5、Computational Geometry

計算幾何

考題重點(目錄)：

[寫 code]

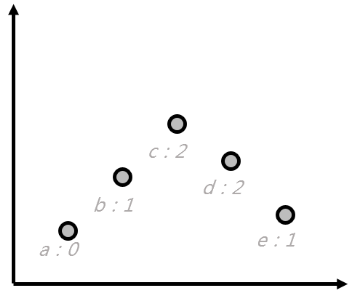
1. 找點的 Rank
2. 找 Maximal Points
3. 找 Cloest Pairs
4. 找 Convex Hull

找點的 Rank

一、Dominate：

P1=(x1, y1) dominates P2=(x2, y2)，則 x1>x2 且 y1>y2

Rank(P1)：P1 dominate 幾個點：



二、找點的 Rank

Input：2D 平面的點集 S

Output：每個點的 Rank

三、Native Approach

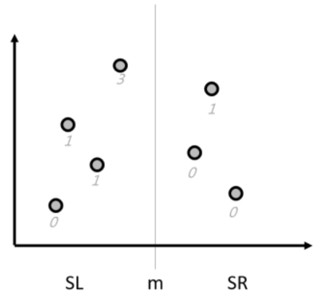
對於任一點而言，去檢查其他點，看是否可 domainate(yes+1；no 下一個)

=> Theta(n^2)

四、Divide-and-Conquer 解

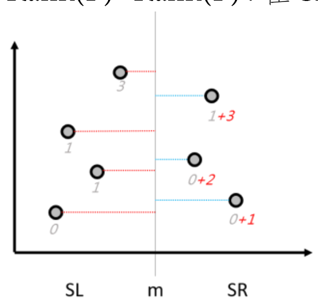
1. 令 m 是 S 中點 x 座標的 Median(中位數，要分成均勻的兩半)，將 S 分成：SL 與 SR(SL 為 S 中 x 座標<m 者；SR 為 S 中 x 座標>m 者)
2. 將 SL 和 SR 中每個點的 Rank 求出(遞迴)

   終止條件：若平面上只有 1 點，則 Rank 為 0



3. 對於每一個在 SR 中的點 P，修正其 Rank(P)如下：

   Rank(P)=Rank(P)+在 SL 中 y 座標比 P 小的點數



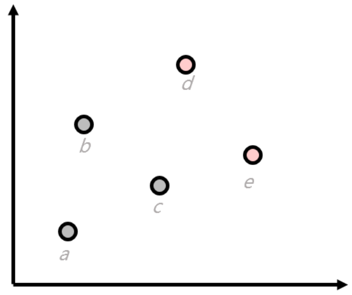
Ans：SL 中點的 Rank(2.)，SR 中更正後點的 Rank(3.)
Time Comlexity：Theta(n lg n)

1. Theta(n)　//(see p8-11)
2. 2T(n/2)
3. Theta(n)

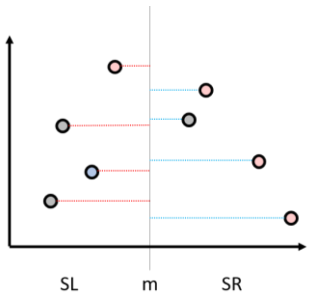
T(n) = 2T(n/2) + Theta(n)　=> Theta(n lg n)

求 Maximal Points

一、Maximal Points 即沒有被任何人 dominate 的點

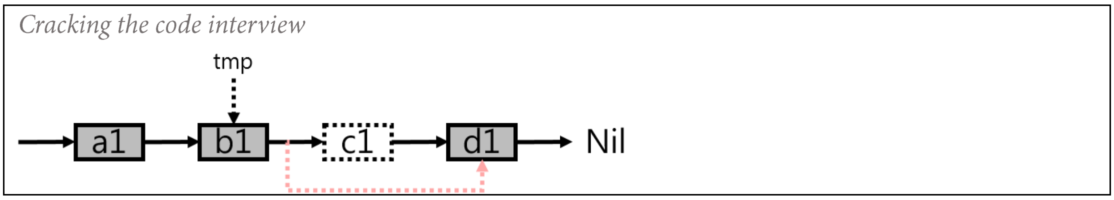


1. 令 m 是 S 中點 x 座標的 Median(中位數，要分成均勻的兩半)，將 S 分成：SL 與 SR(SL 為 S 中 x 座標<m 者；SR 為 S 中 x 座標>m 者)
2. 將 SL 和 SR 中每個點的 Maximal Point 求出(遞迴)
   終止條件：若平面上只有 1 點，則此點即為 Maximal Point



3. 對於每一個在 SL 中的 Maximal Point P，若有一個在 SR 中的 Maximal Point 的 y 座標比 P 大 => 將 P 移除

Ans：SR 中 Maximal Point(2.)及 SL 中剩下的 Maximal Point(3.)



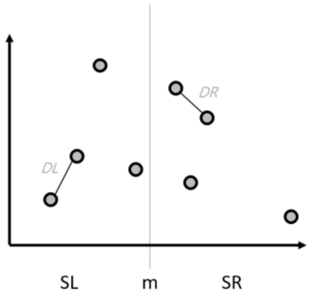
找 Cloest Pairs

一、給定 2D 平面上的點集 S，找 S 中距離 sqrt[(x1-x2)2+(y1-y2)]最小的 2 點的距離

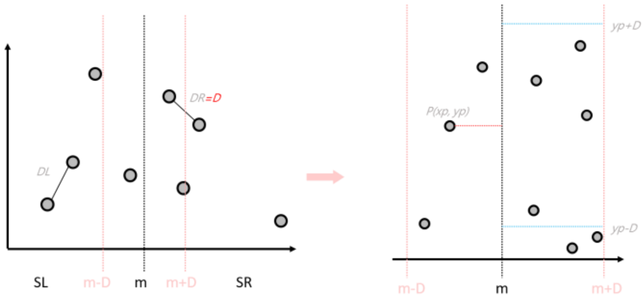
二、Concept：只計算有可能為 Closest Pair 的兩點距離

三、Divide-and-Conquer 解：

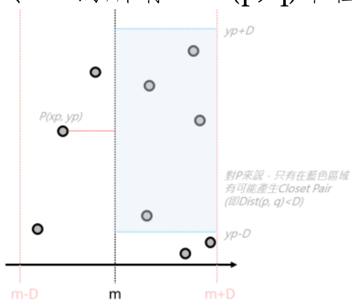
1. 令 m 是 S 中點 x 座標的 Median(中位數，要分成均勻的兩半)，將 S 分成：SL 與 SR(SL 為 S 中 x 座標<m 者；SR 為 S 中 x 座標>m 者)



2. 將 SL 和 SR 中每個點的 Closest Pair 的距離：DL、DR 求出(遞迴)
   終止條件：若平面上只有 1 點，則 Closest Pair 的距離為無限



3. 令 D=min(DL, DR) //不跨平面最短
   對於在 SL 中且 x 座標在 m-D~m 中的點 P=(xp, yp)與在 SR 中 x 座標在 m~m+D 與在 SR 中 x 座標在 yp-D~yp+D 的點 q 算距離
   令 D’為所有 Dist(p, q)中值最小者



Ans：Min(D, D’)

Time Complexity：

1. Theta(n)
2. 2T(n/2)
3. n/2 * Theta(1) *//在口/(◯)最多 6 個點* = Theta(n)

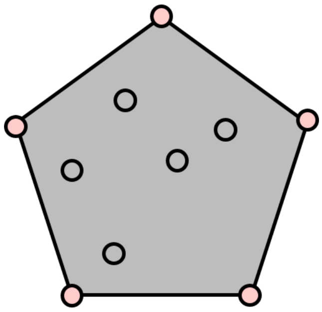
T(n) = 2T(n/2) + Theta(n) = Theta(n lg n)

為何不畫半圓即可？

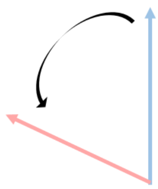
因為都一樣是計算長度，較大的長方形依然能包含半圓形的區域，故直接以長方形來計算，較為省事、且依然能包含半圓的區域

Convex Hull

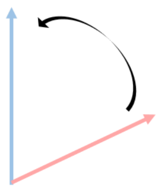
一、Convex Hull 為包含 2D 平面上所有點的最小凸多邊型



二、順(逆)時針方向



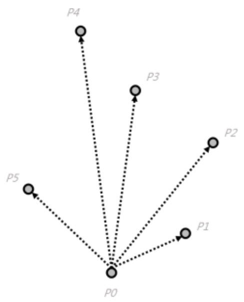
-> 到 -> ：逆時針方向
(Left-turn)



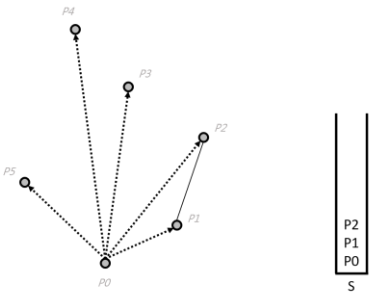
-> 到 -> ：順時針方向
(Right-turn)

三、Graham Scan

1. P0 <- 2D 平面上所有點中，y 座標最小者*(若有多個最小，則找其中最左者)*
2. <P1, P2, …, Pn> <- 將 P0 與每個點做向量，依角度將點排序



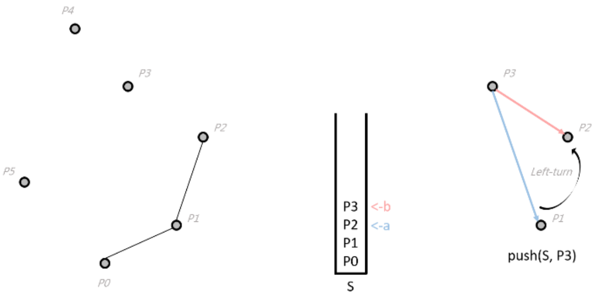
3. S <- entry stack;
   push(S, P0);
   push(S, p1);
   push(S, p2)



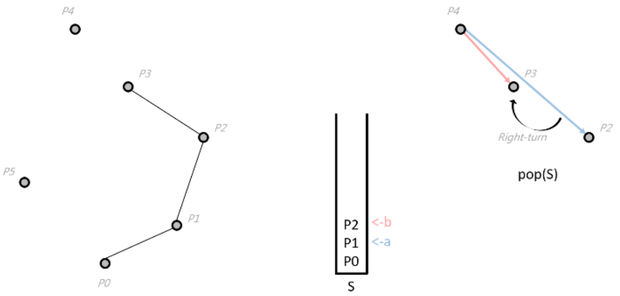
4. 程式：

```
for i<-3 to n
{
        while(pia 到 pib 不為 left-turn)
                pop(s);
        push(s, pi);
}
```

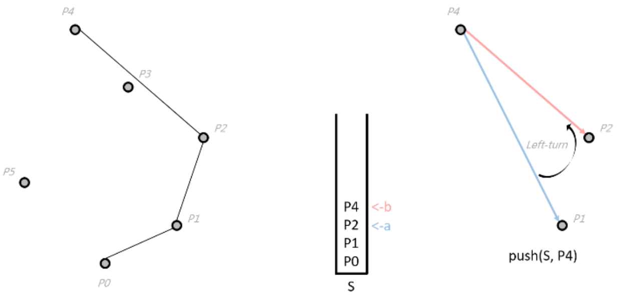
已選 3 點，加入第 4 點時：



已選 4 點，加入第 5 點時：



但因為為 Right-turn，故 pop，重新選點：

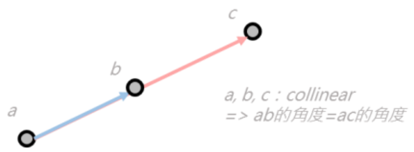


//判斷：『角 Piba』是否為凹角

Time Complexity：
1. Theta(n)
2. Theta(n lg n)
3. Theta(1)
4. Theta(1) *//因為每個點必被 push 一次，又最多被 pop 一次，因此 n 個點最多有 2n 次 push/pop*

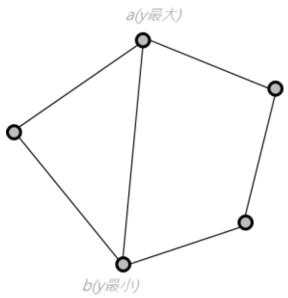
四、應用

1. 判斷是否為 Collainer(92 台大)，p5-18.1
   $O(n^2 \log n)$ -> $n\,O(n \log n)$ *//排 n 個東西*

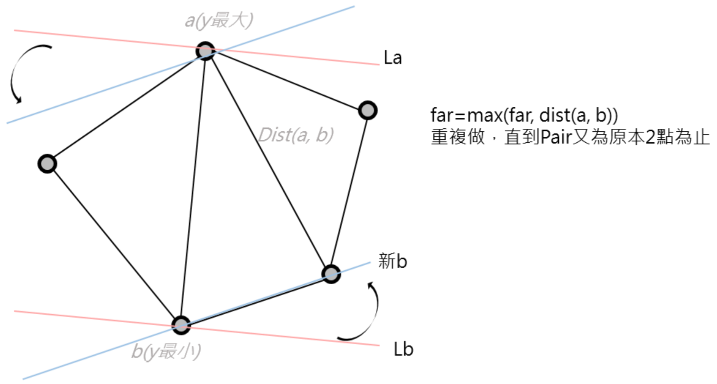


   為了方便判斷此條件是否成立

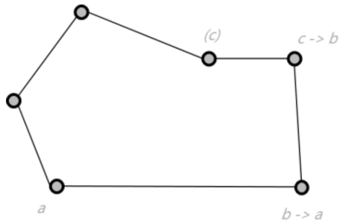
2. 找 farthest pair(台大)
   觀察：2D 平面上最遠的 2 點必為在 Convex Hull 上的某 2 點



   概念：



3. 判斷多邊形是 Convex 或 Concave(100 政大)
   用 Step 4 中的 while 判斷連續 3 點是否形成凹角即可
   例：



判斷 ca 到 cb 是否為 left-tum ? yes 則往下面 test；no 則 concave。若全為 yes 則 Convex