ComputationalGeometry \_HW1

**(演算法及證明題並沒有一定解，只要能依題意解出即可)**

**Chap 1**

* 1. The convex hull of a set S is defined to be the intersection of all convex sets that contain S. For the convex hull of a set of points it was indicated that the convex hull is the convex set with smallest perimeter. We want to show that these are equivalent definitions.

1. Prove that the intersection of two convex sets is again convex. This implies that the intersection of a finite family of convex sets is convex as well.
2. Prove that the smallest perimeter polygon P containing a set of points P is convex.
3. Prove that any convex set containing the set of points P contains the smallest perimeter polygon P.

Ans:

a.有兩個convex sets C1,C2交集產生一個的新集合C3，其中含有兩點a, b，假設有一點c為ab連線上的一點，但他並不在兩convex sets C1, C2的交集C3之中，這也表示他不存在於C1或C2集合裡，這也會使C1與C2不屬於convex，前後將互相矛盾，故兩convex sets的交集也必為convex set。

b.在polygon P中任選兩點a, b，a, b兩點相連後，若能產生一個新的polygon(原polygon非convex hull，此線不在P中)，則新的polygon P’的周長必定會小於原本P的周長，由此得證polygon P要有最小的周長，其必為convex。

c.承上，一個convex hull包含所有點的集合，其必為最小周長之多邊形，否則會違背其定義。

* 1. Let E be an unsorted set of n segments that are the edges of a convex polygon. Describe an O(nlogn) algorithm that computes from E a list containing all vertices of the polygon, sorted in clockwise order

Ans:

使用merge sort或quick sort將每個線段之兩點做排序( worst case為O(nlogn) )，從含有最左點之邊開始比較，以順時針方式連接出上半部所有邊( O(n) )，下半部同上，總共的時間複雜度為2n+nlogn-> O(nlogn)。

* 1. The O(nlogn) algorithm to compute the convex hull of a set of n points in the plane that was described in this chapter is based on the paradigm of incremental construction: add the points one by one, and update the convex hull after each addition. In this exercise we shall develop an algorithm based on another paradigm, namely divide-and-conquer
  2. Let P1 and P2 be two disjoint convex polygons with n vertices in total. Give an O(n) time algorithm that computes the convex hull of P1 ∪P2.
  3. Use the algorithm from part a to develop an *O*(*n*log*n*) time divide-andconquer algorithm to compute the convex hull of a set of *n* points in the plane.

Ans:

a. 先找出P1最右邊的點a，以及P2最左邊的點b，相連起來後得到一個line segment，當他不是所有line segment中最底端的那條時，以此線為準，順時針沿著P1點a向下，逆時針沿著P2點b向下找，點與點相連，至相連的線為最底那條切線( O(n) ); 同樣以ab連線為準，逆時針沿P1點a向上，順時針沿P2點b向上找，至相連的線為最頂端的切線( O(n) )，找完後，連接這兩條切線。

b.

根據x軸大小做排序O(nlogn)，將所有點分為左半及右半，做成左右兩個convex hull，在依照上面a的方式合併做出一個完整的convex hull ( O(n) )。

**Chap 2**

1. Let S be a set of n disjoint line segments whose upper endpoints lie on the line y = 1 and whose lower endpoints lie on the line y = 0. These segments partition the horizontal strip [−∞ : ∞]×[0:1] into n+1 regions. Give an O(nlogn) time algorithm to build a binary search tree on the segments in S such that the region containing a query point can be determined in O(logn) time. Also describe the query algorithm in detail.

Ans:

Build a B.S.T.

1. Sort all line segments in S by its x value on y=0

2. Choose n/2 as a root to build a binary search tree

Recursive choose n/4, 3n/4, n/8, 3n/8, 5n/8, 7n/8....... as nodes until each subtree only has one or

two segments as leaves.

=> we can get a binary search tree that containing all of the line segments. Its height will be logn.

∴ time complexity is O(nlogn).

Search the given point.

1. We substitute the give point into the line segment from the root of binary search tree and record

the result.

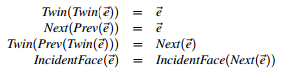
go to right child of root if the result is positive

go to left child of root if the result is negative

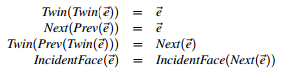
2.And then recursively binary search the given point in the tree.

∴ time complexity is O(logn).

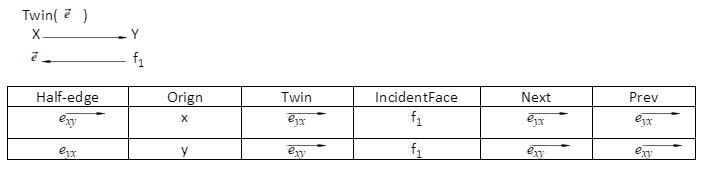
* 1. Which of the following equalities are always true?

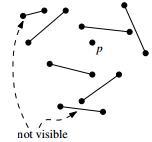


Ans: (1)(2)(4) are always true by definition.

(3) is not always true. (當degree of 的起點 > 2時)

* 1. Give an example of a doubly-connected edge list where for an edge e the faces IncidentFace(e) and IncidentFace(Twin(e)) are the same.  
     Ans:



**2.14** Let S be a set of n disjoint line segments in the plane, and let p be a

point not on any of the line segments of S. We wish to determine all

line segments of S that p can see, that is, all line segments of S that

contain some point q so that the open segment pq doesn’t intersect any

line segment of S. Give an O(nlogn) time algorithm for this problem that

uses a rotating half-line with its endpoint at p.

Ans:

1.對於所有的端點相對 p 做極角排序，並且知道相對應的角度上會存有那些線段的端點。

map<double, vector<int, int> > angle;

for (int i = 0; i < n; i++) {

double v1 = atan2(D[i].s.y - pos.y, D[i].s.x - pos.x);

double v2 = atan2(D[i].e.y - pos.y, D[i].e.x - pos.x);

angle[v1].push\_back(make\_pair(i, 0));

angle[v2].push\_back(make\_pair(i, 1));

}

2.藉由出現的角度，使用極角掃描，一開始必須將碰的線段加入。

double c;

CMP::base = pos;

double ftheta = angle.begin()->first;

pair<int, int> u = angle.begin()->second[0];

Pt fpt;

if (u.second == 0)

fpt = D[u.first].s;

else

fpt = D[u.first].e;

for (int i = 0; i < n; i++) {

if (cross(pos, fpt, D[i].s) \* cross(pos, fpt, D[i].e) < 0)

S.insert(D[i]);

}

CMP::sin = sin(ftheta);

CMP::cos = cos(ftheta);

for (map<double, vector< pair<int, int> >, CMP2>::iterator it = angle.begin();

it != angle.end(); it++) {

CMP::sin = sin(it->first);

CMP::cos = cos(it->first);

for (int i = 0; i < it->second.size(); i++) {

pair<int, int> u = it->second[i];

if (u.second == 0)

c = cross(pos, D[u.first].s, D[u.first].e);

else

c = cross(pos, D[u.first].e, D[u.first].s);

if (fabs(c) > eps) {

if (c > 0)

S.insert(D[u.first]);

else

S.erase(D[u.first]);

}

}

if (S.size()) {

visual[S.begin()->label] = 1;

}

}