Assignment 1 Supplement
Algorithm Design and Analysis

bitjoy.net

January 21, 2016

2 Divide and Conquer

假设平面上有 2n 个点，每个点 p 有一个 flag 属性，flag=1 为 Ghostbuster，反之为 ghost。首先找到纵坐标最小的点 O（可能是 Ghostbuster 或者 ghost），以 O 为原点建立坐标系，计算剩余 2n-1 个点和点 O 的夹角，将夹角从小到大排序。根据夹角遍历这 2n-1 个点，遍历的同时累加计数已遍历的 Ghosterbuster 和 ghost 的数量，当加上原点使得 Ghosterbuster 和 ghost 的数量相等时，此时遍历的点 p 和原点 O 的连线将 2n-2 个点划分成两半，使得每一半中 Ghostbusters 和 ghosts 的数量相等。然后在每一半中递归划分，最终将 2n 个点配对成功。

PAIRING(A, i, j)

```
O = i
for k = i to j
    if A[k].y < A[O].y
        O=k
for k = i to j && k != O
    B[k] = angle(O,k)
Sort B ascendingly with QuickSort
same = 0; diff = 0;
for k = i to j && k != O
    if B[k].flag == A[O].flag
        same++
    else
        diff++
    if same + 1 == diff
        Pair(O,k)
        break
PAIRING(B,i,k-1)
PAIRING(B,k+1,j)
```

在找划分线的过程中，首先要找 y 值最小的点，用时 O(n)；然后对剩余 2n-1 个点计算和 O 的夹角，用时 O(n)；然后对 2n-1 个夹角排序并遍历，用时 O(nlogn)。所以找划分线总的时间复杂度为 O(nlogn)。

配对是递归进行的，所以用时为 $T(n) = 2T(n/2) + O(nlogn)$，归纳法可证 $T(n) \leq n^2logn$，所以 $T(n) = O(n^2logn)$。

5 Divide and Conquer

Here is the solution excerpted from UCSD[1].

Consider the following algorithm: find the minimum value x_v in the middle column of nodes by probing the n elements in the column. Next check the left and right neighbors l, r of v and see which of the two x_l, x_r is smaller. If x_v is smaller than both, return x_v as a local minima. If not, without loss of generality say that $x_l < x_v$ and we may **discard** the right half the graph. Now in the left half of the graph(which is now a grid of size $n \times n/2$), find the minimum value x_v in the middle row of nodes by probing the $n/2$ elements in the row. Now as before, probe and compare the values of the top and bottom neighbors of v. If both values are less than v_v, return x_v. Otherwise recurse the entire procedure on the top left quadrant(of the full graph) if the top neighbor was smaller, or the bottom left quadrant(of the full graph) if the bottom neighbor was smaller.

The recurrence relation of this procedure is summarized by $T(n) = n + n/2 + T(\frac{n}{2}) = O(n) + T(\frac{n}{2})$, resulting in a total of $O(n)$ probes.

As every time we choose the smaller node, we will get a local minimum node at last.

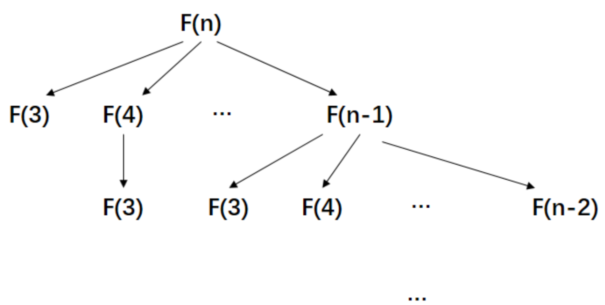
6 Divide and Conquer

令 f(n) 为 n 边形切成若干个三角形的切法数量。

要将一个 n 边形切成三角形，选择一个点 v 作为下刀之处，另一个点 u 可以是除 v 的邻居外的其余 n-3 个点。根据 u 点的不同，可以得到不同的两个多边形，所以固定点 v，我们有 $G = f(3)f(n+2-3) + f(4)f(n+2-4) ... + f(n-1)f(3) = \sum_{i=3}^{n-1} f(i)f(n+2-i)$ 种切法。n 边形有 n 个点，所以共有 nG 种切法，但是有一半是重复的，所以总的切法数为 $f(n) = nG/2 = n\sum_{i=3}^{n-1} f(i)f(n+2-i)/2$。

```
FUNCTION(n)
if n == 3
    return 1
else
    ans = 0
    for i = 3 to n - 1
        if F[i] == NULL
            F[i] = FUNCTION(i)
        if F[n + 2 - i] == NULL
            F[n + 2 - i] = FUNCTION(n + 2 - i)
        ans += F[i] * F[n + 2 - i]
    return ans * n / 2
```

递归树如下：



[1] http://cseweb.ucsd.edu/classes/sp15/cse202-a/midtermsol.pdf

由 $f(n)$ 的计算公式可知，$f(n)$ 只会用到 $k < n$ 的那些 $f(k)$，而这些 $f(k)$ 在之前计算时已经保存下来了。所以求 $f(k)$ 时只需要循环 $\sum_{i=3}^{k-1}$，求 $f(n)$ 时只需要循环 $\sum_{k=3}^{n-1}$，总的时间复杂度为 $O(n^2)$。如果没有保存 $f(k)$，导致每层重复计算，则时间复杂度为 $O(2^n)$。