week3

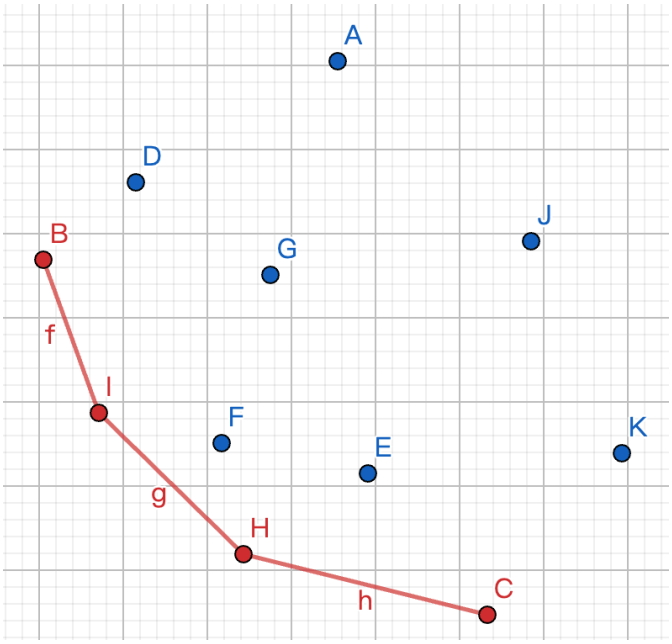
SmpaelFx

2024 年 10 月 17 日

1 最小乘积的寻找

1.1 递归处理

对于一个方案，我们可以认为其对应平面直角坐标系中的一个点 (w,e)，则最优方案一定在这些点的左下凸包上。

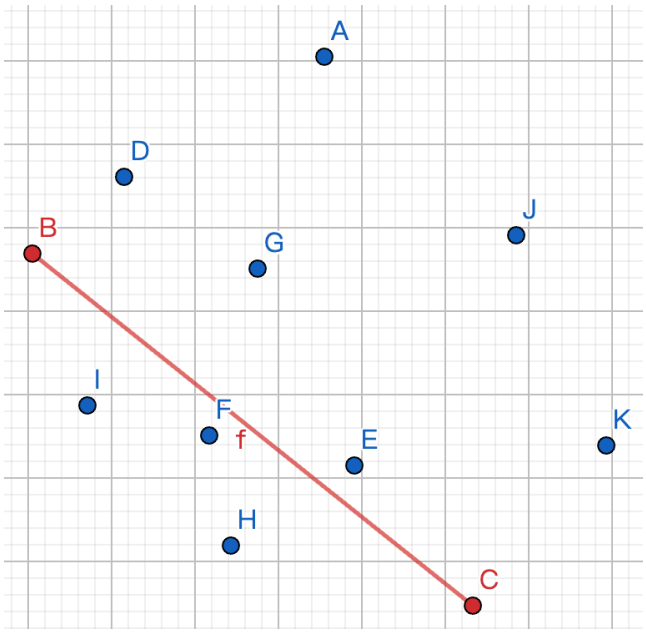


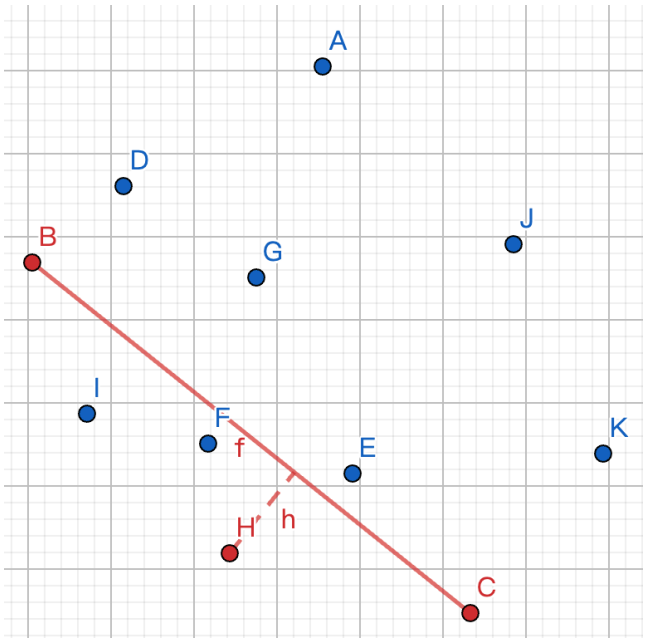
怎么找到凸包呢？我们考虑先找到横坐标最小的点（图中点 B ）和纵坐标最小的点（图中点 C ）。

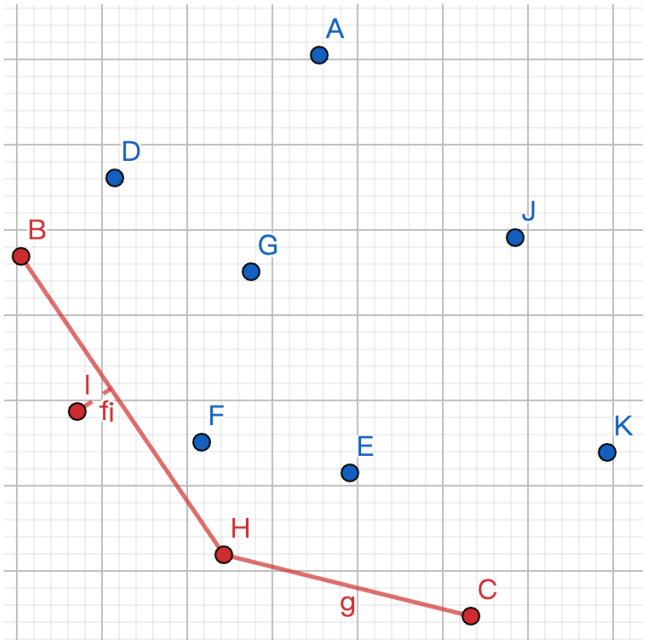
找到左下方离线段 BC 最远的点 H，过 H 做 BC 的平行线，发现与凸包不可能存在其他交点（H 离 BC 最远），因此 H 在凸包上，分别以 B,H，H,C 为新的端点递归处理，找不到新的点则结束。

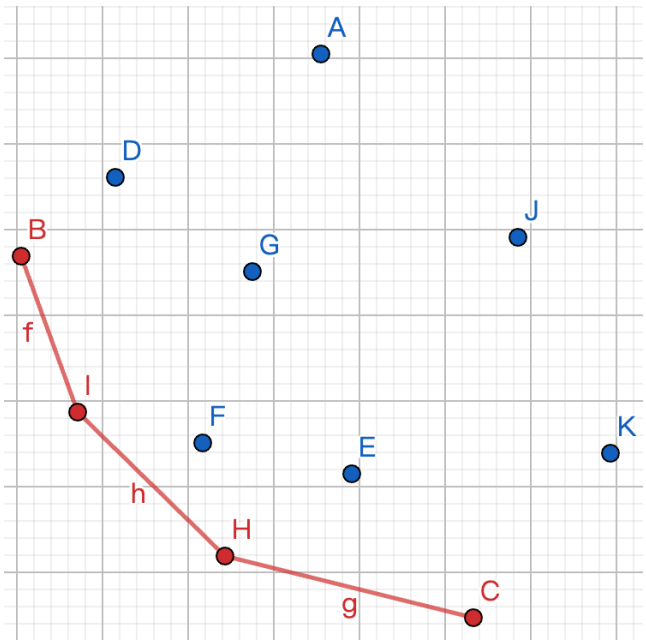
1.2 如何找到线段左下最远点

利用向量叉乘，设 A,B 为正在处理的两端点，其中 A 在 B 左上方，设 C 为任意一点，根据向量叉乘，$k=\overrightarrow{AB}\times\overrightarrow{AC}=|\overrightarrow{AB}|\cdot|\overrightarrow{AC}|\sin<\overrightarrow{AB},\overrightarrow{AC}>$，因为 AB 长为定值，$AC\sin<\overrightarrow{AB},\overrightarrow{AC}>$ 为 C 到 $\overrightarrow{AB}$ 的距离，当 C 在 $\overrightarrow{AB}$ 顺时针方向时，距离为负数，则 k 也为负数，因为 $\overrightarrow{AB}$ 向右下，因此 k 最小时，C 为线段 AB 左下最远点。


A
D
B
G
J
I
F
f
E
K
H
C


A
D
B
G
J
I
F
f
E
K
H'
h
C


A
D
B
G
J
I
fi
F
E
K
H
g
C


A
D
B
G
J
f
I
F
E
K
h
H
g
C

根据向量叉乘坐标公式，$\overrightarrow{AB} \times \overrightarrow{AC} = (x_B - x_A)(y_C - y_A) - (y_B - y_A)(x_C - x_A) = x_B y_C - x_B y_A - x_A y_C + x_A y_A - y_B x_C + y_B x_A + y_A x_C - y_A x_A = (y_A - y_B)x_C + (x_B - x_A)y_C +$ 常数，此时二维的问题便转化为了一维的最值问题。

2 平面图最小割转对偶图最短路

割的定义详见题目描述。

根据最大流最小割定理，存在一个最小割 S, T 满足 S 构成的子图连通，并且我们可以把 T 子图中与汇点不相连的部分扔进 S 集合中，可以证明这样的操作对割的容量是不劣的。进而我们可以找到一个最小割 S, T 使 S 和 T 构成的子图分别连通，S 与 T 之间的边界也连续（如 **Fig. 1**）。

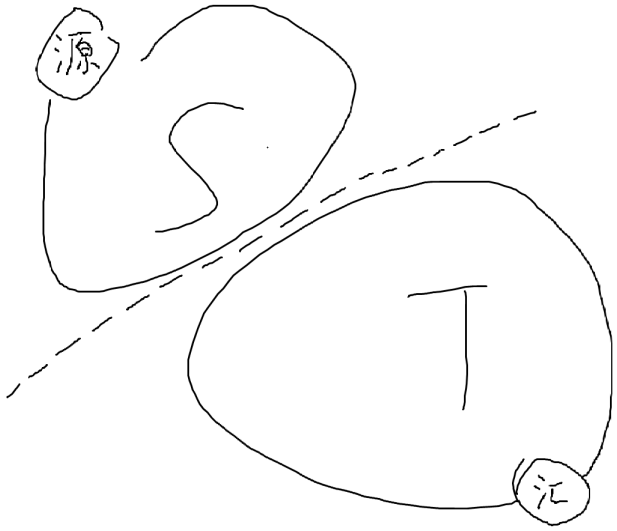


Fig. 1

平面图上的每一条边都被对偶图上唯一的一条边割过，因此对于前文我们构造出的最小割 S, T，我们可以找到对偶图上的一条路，割过所有从 S 到 T 的边，如果设对偶图边权为这条边割过的边的边权，则这条路的长度即为最小割。因为这个割是最小割，这条路是最短路。

因此平面图最小割等于对偶图最短路。

3 整体思路

找对偶图上 $aw + be$ 最小的路径，当从点 u 转移到 v 时，设 w 加了 Δw，e 加了 Δe，则

$$dis[v] \leftarrow dis[u] + a\Delta w + b\Delta e$$

找初始两端点时，a, b 分别取 $1, 0$ 和 $0, 1$，递归过程中，a, b 取 $y_A - y_B, x_B - x_A$ 其中 A 在 B 左上。因此 $a, b \geq 0$ 且 $ab \neq 0$，最短路可以使用 dijkstra。

把递归过程中的端点横坐标与纵坐标之积与 ans 取 min，即为答案。

4 代码

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
const ll inf=0x3f3f3f3f3f3f3f3f;
int n;
pair<ll,ll> L[405][405],D[405][405];
ll l[405][405],d[405][405];
bool vis[405][405];
struct Node{
    ll x,y,d;
    Node(){}
    Node(ll x,ll y,ll d):x(x),y(y),d(d){}
    Node(pair<ll,ll> p,ll d):x(p.first),y(p.second),d(d){}
};
Node dis[405][405];
bool operator<(Node u,Node v){
    return u.d<v.d;
}
bool operator>(Node u,Node v){
    return u.d>v.d;
}
Node add(Node x,Node y){
    return Node(x.x+y.x,x.y+y.y,x.d+y.d);
}
void calcDis(){
    for(int i=0;i<=n;i++) for(int j=0;j<=n;j++) dis[i][j]=Node(inf,inf,inf);
    memset(vis,0,sizeof(vis));
    priority_queue<pair<Node,pair<int,int>>,vector<pair<Node,pair<int,int>>>,
        greater<pair<Node,pair<int,int>>>> pq;
    dis[0][n]=Node(0,0,0);
    pq.push({dis[0][n],{0,n}});
    while(!pq.empty()){
```

```
        int x=pq.top().second.first,y=pq.top().second.second;
        pq.pop();
        if(vis[x][y]||(x==0&&y==0)||(x==n&&y==n)) continue;
        vis[x][y]=1;
        if(y>0&&dis[x][y-1]>add(dis[x][y],Node(L[x][y],l[x][y]))){
            dis[x][y-1]=add(dis[x][y],Node(L[x][y],l[x][y]));
            pq.push({dis[x][y-1],{x,y-1}});
        }
        if(y<n&&dis[x][y+1]>add(dis[x][y],Node(L[x][y+1],l[x][y+1]))){
            dis[x][y+1]=add(dis[x][y],Node(L[x][y+1],l[x][y+1]));
            pq.push({dis[x][y+1],{x,y+1}});
        }
        if(x>0&&dis[x-1][y]>add(dis[x][y],Node(D[x-1][y],d[x-1][y]))){
            dis[x-1][y]=add(dis[x][y],Node(D[x-1][y],d[x-1][y]));
            pq.push({dis[x-1][y],{x-1,y}});
        }
        if(x<n&&dis[x+1][y]>add(dis[x][y],Node(D[x][y],d[x][y]))){
            dis[x+1][y]=add(dis[x][y],Node(D[x][y],d[x][y]));
            pq.push({dis[x+1][y],{x+1,y}});
        }
    }
}
ll ans=inf;
void solve(Node A,Node B){
    ans=min(ans,A.x*A.y),ans=min(ans,B.x*B.y);
    for(int i=0;i<=n;i++){
        for(int j=0;j<=n;j++){
            l[i][j]=(B.x-A.x)*L[i][j].second+(A.y-B.y)*L[i][j].first;
            d[i][j]=(B.x-A.x)*D[i][j].second+(A.y-B.y)*D[i][j].first;
        }
    }
    calcDis();
    Node C=dis[n][0];
    if((B.x-A.x)*(C.y-A.y)-(C.x-A.x)*(B.y-A.y)<0) solve(A,C),solve(C,B);
}
int main(){
    scanf("%d",&n);
    for(int i=1;i<n;i++){
        for(int j=1;j<=n;j++) scanf("%lld%lld",&L[i][j].first,&L[i][j].second);
    }
    for(int i=0;i<n;i++){
        for(int j=1;j<n;j++) scanf("%lld%lld",&D[i][j].first,&D[i][j].second);
    }
```

```
    Node A,B;
    for(int i=0;i<=n;i++){
        for(int j=0;j<=n;j++) l[i][j]=L[i][j].first,d[i][j]=D[i][j].first;
    }
    calcDis();
    A=dis[n][0];
    for(int i=0;i<=n;i++){
        for(int j=0;j<=n;j++) l[i][j]=L[i][j].second,d[i][j]=D[i][j].second;
    }
    calcDis();
    B=dis[n][0];
    solve(A,B);
    printf("%lld\n",ans);
    return 0;
}
```