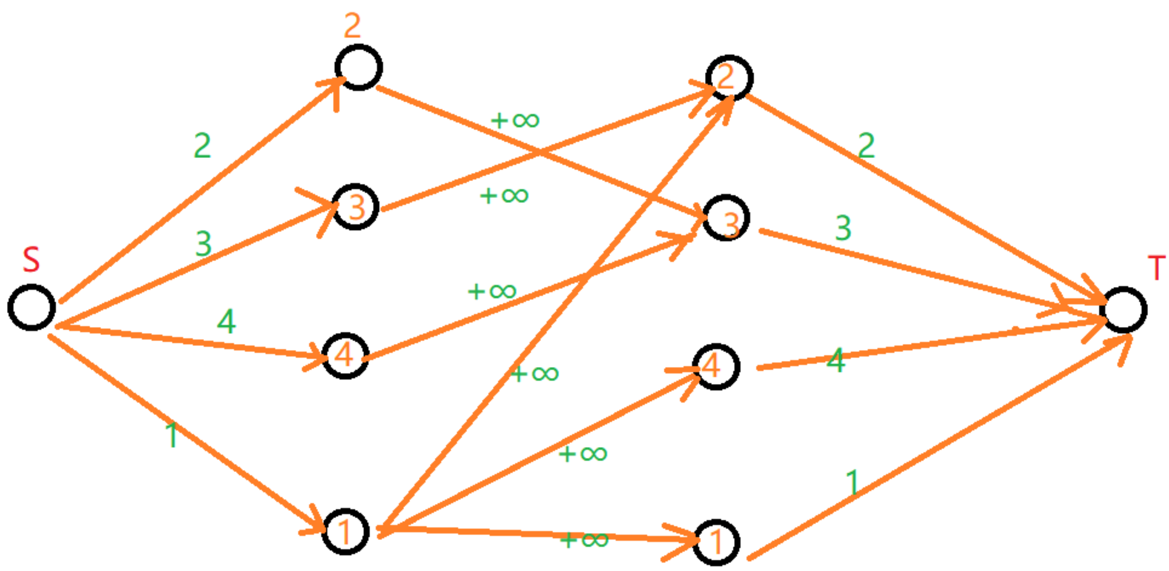
网络流__12 最小割之最小点权覆盖算法与最大独立集算法

由二分图我们知道`最小覆盖集=最大二分图匹配=总点数-最大独立集`，如果每个点都有权值，则最小点权覆盖应该对应`KM`二分图最优匹配算法，那么如果用网络流来求解，我们则需要借助最小割定理来辅助解决。

思路与建图方式：

在一个无向图中



从源点向左集合的所有点连容量为点权的边，从右集向汇点所有点连容量为点权的边，中间相关的点之间连接容量为+∞的边，这样可以使得原图的任意一个割为简单割，且由此，原图的最小割便是原图G的一个最小点全覆盖

例题一道：有向图破坏

> 给定一个有向图，，每次选取一个有权值的点，要么移除这个点的所有入边，要么移除这个点的所有出边，问最小花费

将点拆成移除入边和移除出边之后边能套上最小点权覆盖的板子。然后建好图跑完最大流就是关键的从最小割到最大流的转化：

> 割边的性质：不存在从源S到汇点T的一条路径，因此对应割在网络流上的一条路径：<S,u> <u,v> <v,T>，由于人为规定了<u,v>=INF，因此要么<S,u>=0,要么<u,T>=0;
> 如果<S,u>=0，对应左集合的一条割边
> 如果么<u,T>=0，对应右集合的一条割边

从源点出发，在残留网络中沿着剩余容量大于0的边走，所有遍历到的点构成集合`S{}`，剩余点构成集合`T{}`.注意，最小割中的割边都是正向边，因此`i+=2`进行遍历。

最大权独立集

点独立集定义：

$\forall e(u,v) \in E$满足$u \in V$和$v \in V$不同时成立

解决思路：

类比于二分图所有点权覆盖问题中的结论`最大权独立集=所有点的总权值-最小权点覆盖`

`最大独立集=所有点的总权值-最小点权覆盖集`

证明： 反证法，任给一个覆盖集，其补集为独立集，其中假设有一边两点都没选上，与原集合为覆盖集矛盾。给定一个独立集，若其补集不是覆盖集，说明有一条边的两点都未被选择，说明独立集中有一边存在两个端点，矛盾。

例题

王者之剑 姚金宇

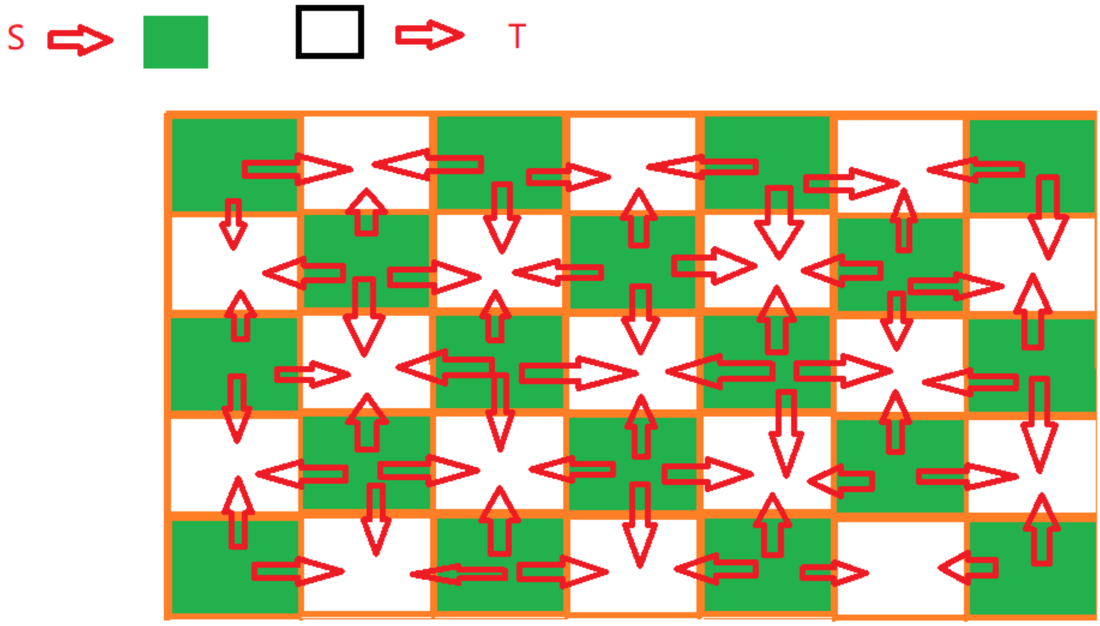
> wyh在一个n*m的网格，每个格子上有一个价值为$v_{i,j}$的宝石。wyh可以自己决定起点。开始时刻为第0秒，后面每秒按照顺序执行：
>
> 1、若第i秒开始时，wyh在(x,y)，则wyh可以拿走(x,y)上的宝石
>
> 2、在偶数秒时，wyh周围上下左右四格的宝石全部消失
>
> 3、若第i秒开始时，wyh在(x,y)，则在第(i+1)秒开始前，wyh可以移动到上下左右相邻的格子内（如果存在的话，移动速度为瞬间）

我们对奇偶秒不同这一条件进行挖掘可以得到1.取宝石必然在偶数秒（因为奇数秒进入的格子必为0），2.相邻两个格子的宝石必然不能同时拿

由2可以直接想到这和大多数二分图的问题的隐含条件类似，因此加入点权(宝石)这一条件后便是一个最大独立集问题。

建图思路：

每个格子进行黑白染色，相邻格子染上不同的颜色。不妨设左集合为坐标值和为偶数的点，右集合为坐标之和为奇数的点。因此从源点向左集合连对应点权的边，从左集合向右集合连容量为正无穷的边，从右集合向汇点连容量为点权的边。然后使用`总点权-最小覆盖集=总点权-最小割=最大独立集`进行求解



证明：

很容易证明每一个方案对应于二分图中的一种划分方式

下证明每一种划分方式对应一个可行方案：

> 每次考虑两行（第一行为主行，第二行只起辅助作用。因为第一行考虑结束后原来的第二行会变成考虑的主行）：
>
> 保证每次到达需要取得的格子时为偶数时间，如果是奇数时间，就在前两格的位置停顿一秒。

建图代码：

```
void solve(){
    LL ans = 0;
    n=read(),m=read(); S=0,T=n*m+1;
    rep(i,1,n) rep(j,1,m) {
        g[i][j] = read();
        ans+=g[i][j];
        if((i+j)%2==0) add(S,get(i,j),g[i][j]);
        else add(get(i,j),T,g[i][j]);
    }
    rep(i,1,n) rep(j,1,m) if((i+j)%2==0) rep(k,0,3) {
        int x = i + mov[k][0], y = j + mov[k][1];
        if(x<1||x>n||y<1||y>m) continue;
        add(get(i,j),get(x,y),INF);
    }
    print(ans-dinic());
}
```