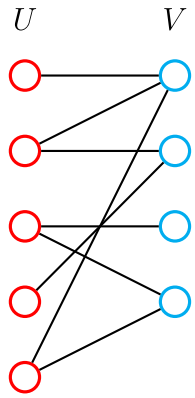
二分图

定义

二分图，又称二部图，英文名叫 Bipartite graph。

二分图是什么？节点由两个集合组成，且两个集合内部没有边的图。

换言之，存在一种方案，将节点划分成满足以上性质的两个集合。



性质

1. 如果两个集合中的点分别染成黑色和白色，可以发现二分图中的每一条边都一定是连接一个黑色点和一个白色点。
2. 二分图不存在长度为奇数的环
   因为每一条边都是从一个集合走到另一个集合，只有走偶数次才可能回到同一个集合。

二分图判定

染色法 $O(m+n)$

题目链接

首先随意选取一个未染色的点进行染色，然后尝试将其相邻的点染成相反的颜色。
如果邻接点已经被染色并且现有的染色与它应该被染的颜色不同，那么就说明不是二分图。
而如果全部顺利染色完毕，则说明是二分图。
染色结束后的情况（记录在数组中）便将图中的所有节点分为了两个部分，即为二分图的两个点集。

```
bool dfs(int x,int c) {
   color[x]=c;
    for(int i=head[x]; i; i=e[i].next) {
        int to=e[i].to;
        if(!color[to]) {
            if(!dfs(to,3-c))
                return 0;
        } else if(color[to]==c)
            return 0;
    }
    return 1;
}
for(int i=1; i<=n; i++)
    if(!color[i]) if(!dfs(i,1)) flag=0;break;}
```

应用

常见问题

1. **二分图最大匹配**
2. 二分图带权匹配
3. 二分图最小覆盖点
4. 二分图最大独立集
5. 有向无环图的最小路径覆盖

二分图最大匹配

题目链接

一些概念

> 二分图的匹配：
> 给定一个二分图 G，在 G 的一个子图 M 中，M 的边集 {E} 中的任意两条边都不依附于同一个顶点，则称 M 是一个匹配。
> 换句话说：任意两条边没有公共端点的边的集合被称为为图的一组匹配
>
> 二分图的最大匹配：
> 所有匹配中包含边数最多的一组匹配被称为二分图的最大匹配，其边数即为最大匹配数。

> 完美匹配：
> 如果一个图的某个匹配中，所有的顶点都是匹配点，那么它就是一个完美匹配。
>
> 交替路：
> 从一个未匹配点出发，依次经过非匹配边、匹配边、非匹配边…形成的路径叫交替路。
>
> 增广路：
> 从一个未匹配点出发，走交替路，如果途径另一个未匹配点（出发的点不算），则这条交替 路称为增广路（agumenting path）。

每次找到一条增广路，把path上的所有边的状态取反（匹配边变成非匹配边，非匹配边变成匹配边），那么得到的新的边集S'还是一组匹配，并且匹配边数增加了1.所以可以得到推论：二分图的一组匹配S是最大匹配，当且仅当图中不存在S的增广路

匈牙利算法

1. 设$S=\varnothing$
2. 寻找增广路path，把路径上所有边的匹配状态取反，得到一个更大的匹配。
3. 重复2，直至图中不存在增广路

复杂度：

1. 时间复杂度：邻接矩阵最坏为$O(n^3)$
   邻接表：$O(mn)$
2. 空间复杂度：邻接矩阵：$O(n^2)$
   邻接表：$O(n+m)$

匈牙利算法基于贪心思想，重要特点是：一个节点成为匹配点之后，最多只会因为找到增广路而更换匹配对象，但是绝不会再变回匹配点。

详细说明链接

```
int dfs(int x) {
```

```
    for(int i=head[x]; i ; i=e[i].next) {
        int to=e[i].to;
        if(!vis[to]) { //如果在这一轮模拟匹配中,这个点尚未被预定
            vis[to]=1;
            //如果to没有匹配，或者它原来的匹配点能够匹配其它的点。配对成功
            if(!match[to]||dfs(match[to])) {
                match[to]=x;
                return 1;
            }
        }
    }
    return 0;
}
for(int i=1; i<=n1 ; i++) {
    memset(vis,0,sizeof vis);
    if(dfs(i))ans++;
}
```

二分图其他常见问题

可转化为求最大匹配问题

1. **二分图带权匹配**
   二分图每条边都有一个权值，求该二分图的一组最大匹配，并且匹配边的权值和最大。
   解法：KM算法（KM必需在满足"带权最大匹配一定是完备匹配"的二分图中求解）或者转换成网络流模型。如果使用 Dinic 算法 求该网络的最大流，可在$O(\sqrt{n}m)$求出。
2. **二分图最小覆盖点**
   给定一张二分图，求出一个最小的点集S,使得图中任意一条边都有至少一个端点属于S。
   即：以最少点覆盖所有的边
   *Konig*定理:二分图最小点覆盖包含的点数等于二分图最大匹配包含的边数。
   例题：泥泞的区域
3. **二分图最大独立集**
   选最多的点，满足两两之间没有边相连。
   结论：n个节点的二分图，最大独立集的大小等于n − 最大匹配数
   例题：骑士放置
4. **有向无环图的最小路径覆盖**
   给定一张有向无环图，要求用尽量少的不相交简单路劲，覆盖有向无环图的所有顶点。
   需要建立拆点二分图，然后求解最大匹配。
   结论：最小路径点覆盖（最小路径覆盖） = 总点数 - 最大匹配
   例题：舞动的夜晚