

BAJim_H

为天地立心，为生民立命，为往圣继绝学，为万世开太平。

【学习小记】一般图最大匹配——带花树算法

Text

一般图的最大匹配仍然是基于寻找增广路的

增广路的定义是这样的一条路径，它不经过重复的点，并且路径两端均没有匹配，且整条路径是非匹配边-匹配边-非匹配边这样交错的。

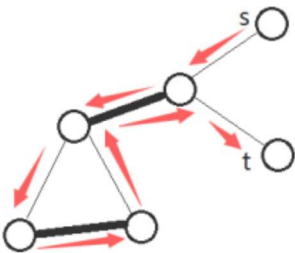
类比二分图最大匹配的增广路算法，如果我们找到了一条增广路，那么将这条增广路的边取反（匹配的变成非匹配，非匹配的变成匹配），那么匹配数会恰好+1，如果全图不存在增广路，也就说明当前已经是一个最大匹配了。（证明略）

一般图和二分图的区别就在于有没有边数为奇数的环

那为什么一般图最大匹配不能直接像二分图那样做呢？

考虑我们寻找增广路的过程

（图片引自 陈胤伯《浅谈图的匹配算法及其应用》，2015年国家集训队论文集）



我们在寻找增广路的时候，会将路径上的点黑白染色，匹配只存在黑点与白点之间。

如果没有环或者只有偶环，那么每个点的颜色（或者说奇偶性）是确定的

但如果出现了奇环，那么点的颜色就不再确定，因为奇环顺时针走一圈和逆时针走一圈的结果是不同的。

怎么办呢？

这就有了我们的带花树算法（名字我觉得很迷）

我们按照算法流程来一步步说明

仍然是从每个未匹配的端点开始寻找增广路，不过我们采用BFS的方式，每个点均设为无色，端点染成黑色

设当前点为u（黑色），枚举与它相邻的点v

考虑v是否已经被访问过

若v尚未访问过（v为无色）

如果v尚未匹配，说明我们找到了一条增广路，直接返回修改。

如果v已经匹配，那么将v染成白色，v的匹配点x加入队列，继续寻找增广路，x染成黑色。

容易看出，根据上面的过程，我们只会对于黑点枚举出边，正确性是显然的，并且我们访问的路径形成了一棵黑白点交错的树。

公告



CN 124

JP 1

HK 5

AU 1

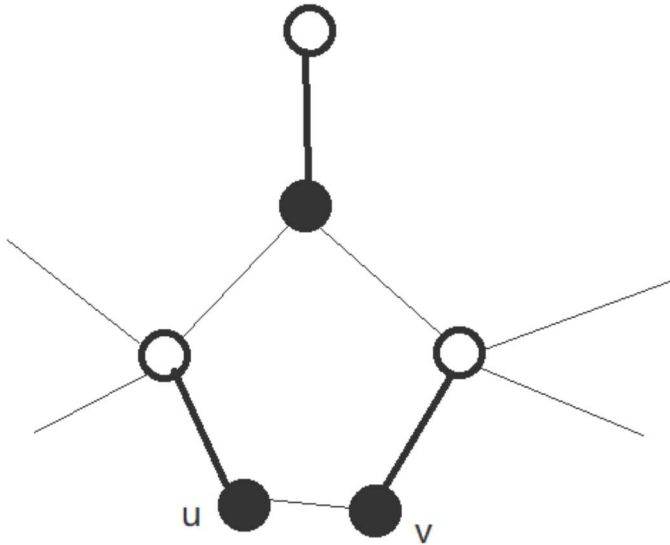
US 2

Pageviews: 326

FLAG counter

欢迎各位交流
QQ:1054689699
旧博客链接
-----Friends-----
Alan_cty
LYD
XHM
HowarLi
Jacky35
LZH
-----Orz-----
YMW
DOFYPXY
Samjia2000
WerKeyTom_FTD
Crazy_czy
Worldwide_D
昵称：BAJim_H
园龄：3个月
粉丝：1
关注：4
+加关注

若v已经访问过（有颜色），说明我们找到了一个环。
如果它是一个偶环（v为白色），那么v显然已经被找过了，无需再找一次。
如果它是一个奇环（v为黑色），这就出锅了
怎么办呢？



<https://blog.csdn.net/hzj1054689699>

如上图，粗边为匹配边。
对于一个奇环，我们一定是从一个黑点进入，然后也在黑点碰头(u,v)，（这个可以画图理解一下，如果不是从黑点进入，我们在之前访问这个奇环时一定还没有绕一圈就找到增广路增广了）
问题在于，此时对于整个奇环的颜色都不确定了，我们令这个奇环的顶点为最顶上的那一个黑点，那么考虑u上方的这一个白点，我们既可以走从顶点走一条非匹配边到它，它作为一个黑点，也可以从另一边转一圈到它，此时这个它变成了黑点，它是需要加入队列继续走的
也就是说，对于一个奇环，它上面的点都可以成为黑点。
继续观察可以发现，整个奇环的匹配状态只与顶点的匹配状态有关，如果在后来的某一次寻找时奇环上的匹配被改变了，那么顶点的颜色唯一决定了整个环的匹配边是如何走的。
也就是说，整个环就可以用一个顶点表示了，也就意味着我们可以将这个环缩掉，缩掉的环就称为“花”，缩环就是开花
我们不妨对于每一个白点x，记pre[x]表示x是由哪一个黑点走过来的，也就是记录了增广路上的非匹配边。
对于每一个点，记match[x]表示x的匹配点是谁。
缩环具体怎么缩呢？

如果直接修改原本的连边比较麻烦，我们考虑采用并查集，记录每个点所在的奇环的顶点，初始时就是它自己。缩环的时候，我们直接将环上的所有点并查集父亲连向奇环的顶点，并将环上的白点都变成黑点，并且加入队列。
此外，由于奇环可以双向走，因此我们的pre边也要变成双向的。
容易发现，我们有可能经过了多次缩环，也就是说某一次缩环的一个点很有可能是缩过的一个环顶，我们在缩环以及找到增广路返回修改的时候是需要走原来缩之前的环的，这个只需要沿着pre和match一直走即可，pre在这里相当于记录了缩掉的环内部的走法。

现在我们来理一理思路
从每个未匹配的点BFS寻找增广路，每个点均设为**无色**，端点染成黑色
枚举与当前点u（黑色）相邻的点v
考虑v是否已经被访问过
若v尚未访问过（v为无色）
如果v尚未匹配，找到了一条增广路，直接返回修改。
如果v已经匹配，将v染成白色，将v的匹配点x加入队列，继续寻找增广路，x染成黑色。

日	一	二	三	四	五	六
30	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	1	2	3
4	5	6	7	8	9	10

搜索

常用链接

我的随笔
我的评论
我的参与
最新评论
我的标签

最新随笔

- 1. [JZOJ6247] 【NOI2019模拟2019.6.27】C 【计数】
- 2. [JZOJ6244] 【NOI2019模拟2019.7.1】islands 【计数】 【图论】
- 3. [JZOJ6244] 【NOI2019模拟2019.7.1】Trominoes 【计数】
- 4. [JZOJ6241] 【NOI2019模拟2019.6.29】字符串 【数据结构】 【字符串】
- 5. 【GDSOI2019】滑稽二乘法 【数据结构】 【LCT】
- 6. 【学习小记】支配树 【图论】
- 7. 【杂题】[CodeForces 1172F] Nauuo and Bug 【数据结构】 【线段树】
- 8. 【杂题】[CodeForces 1172E] Nauuo and ODT 【LCT】 【口胡】
- 9. 【杂题】[CometOJ Contest #5] E：迫真游戏 【概率】 【排列组合】 【多项式】
- 10. 【杂题】[CodeForces 1172D] Nauuo and Portals 【构造】

我的标签

动态规划(1)
其他(1)
数据结构(1)
线段树(1)

随笔分类(102)

A-好题(23)
A-模板(3)
A-其他(2)
A-学习小记(7)
A-杂题(10)
A-总结(4)
白科技
博弈
动态规划(5)

若 v 在当次增广已经访问过，找到环
 v 为白色，是一个偶环，跳过。
 v 为黑色且 u,v 所在的奇环已经缩过了，那么也跳过。

否则， v 为黑色，找到一个新的奇环，那么找到 u,v 所在奇环的环顶（即它们在BFS上跑出来的交错树的lca，称之为最近公共花祖先），将 u 到环顶的路径以及 v 到环顶的路径修改掉，白点染成黑点，加入队列，并将环上的点（或者是某个已经缩了的环顶）并查集父亲指向lca。

接下来就是代码部分，我们可以结合代码来理解上面的过程。

Code (UOJ #079)

```
#include <bits/stdc++.h>
#define fo(i,a,b) for(int i=a;i<=b;++i)
#define fod(i,a,b) for(int i=a;i>=b;--i)
#define N 505
#define M 130005
using namespace std;
int n,m,m1,fs[N],nt[2*M],dt[2*M],pre[N],match[N],f[N],bz[N],bp[N],ti,d[N*N];
void link(int x,int y)
{
    nt[++m1]=fs[x];
    dt[fs[x]=m1]=y;
}
int getf(int k)
{
    return (f[k]==k)?k:f[k]=getf(f[k]);
}
int lca(int x,int y) //整个lca实现比较巧妙，由于是BFS，那么这两个点在当前奇环上的深度一定相等，交替暴力寻找lca即可。
{
    ti++;x=getf(x),y=getf(y);
    while(bp[x]!=ti)
    {
        bp[x]=ti; //此处仅仅是一个标记，无其他作用
        x=getf(pre[match[x]]);
        if(y) swap(x,y);
    }
    return x;
}
void make(int x,int y,int w) //缩环（开花）过程
{
    while(getf(x)!=w)
    {
        pre[x]=y,y=match[x]; //x是原本的黑点，y是原本的白点，将原本的pre边变成双向。
        if(bz[y]==2) bz[y]=1,d[++d[0]]=y; //若y还是白点则染黑
        if(getf(x)==x) f[x]=w;
        if(getf(y)==y) f[y]=w;
        x=pre[y];
    }
}
bool find(int st) //主过程
{
    fo(i,1,n) f[i]=i,pre[i]=bz[i]=0;
    d[d[0]=1]=st,bz[st]=1;
    int l=0;
    while(l<d[0])
    {
        int k=d[++l];
        for(int i=fs[k];i;nt[i])
        {
            int p=dt[i];
            if(getf(p)==getf(k)||bz[p]==2) continue; //如果找到一个已经缩过的奇环或者偶环则跳过
            if(!bz[p])
            {
                bz[p]=2,pre[p]=k;
                if(!match[p]) //找到增广路
                {
                    for(int x=p,y;x; x=y) y=match[pre[x]],match[x]=pre[x],match[pre[x]]=x; //返回修改匹配
                    return 1;
                }
                bz[match[p]]=1,d[++d[0]]=match[p]; //否则将其匹配点加入队列
            }
            else
            {
                int w=lca(k,p);
            }
        }
    }
}
```

多项式(11)
非传统题(1)
构造(1)
计算几何(2)
数据结构(9)
数学(16)
搜索
贪心
图论(7)
字符串(1)

随笔档案(41)

2019年7月 (3)
2019年6月 (12)
2019年5月 (9)
2019年4月 (4)
2019年3月 (13)

相册(7)

Wallpaper(6)
Wallpaper2(1)

积分与排名

积分 - 1853
排名 - 132878

最新评论

1. Re:【PKUSC2019】线弦图【计数】
【树形DP】【分治FFT】
@大菜鸡xxxEmmmm好吧我去看看多谢提醒...
--BAJim_H
2. Re:【PKUSC2019】线弦图【计数】
【树形DP】【分治FFT】
博主您这题代码是错的，，，数据范围调大随机一组数据就会输出0
--大菜鸡xxx

阅读排行榜

1. 【数学】稀疏图的随机游走问题(221)
2. 关于我(183)
3. 【学习小记】一般图最大匹配——带花树算法(146)
4. 【PKUSC2019】树染色【线段树合并】
【树形DP】(135)
5. 【NOI2019十二省联合省选】部分题简要题解(133)
6. OI中一些常见实用的套路【更新中】(76)
7. 【PKUSC2019】线弦图【计数】
【树形DP】【分治FFT】(75)
8. 【学习小记】常系数齐次线性递推(74)
9. 【杂题】[LibreOJ 2541]
【PKUWC2018】猎人杀【生成函数】
【概率与期望】(53)
10. 比赛注意事项(50)

评论排行榜


1. 【PKUSC2019】弦图【计数】【树形DP】
【分治FFT】(2)

```
        make(k,p,w);
        make(p,k,w); //以上分别修改k到lca的路径以及p到lca的路径（环的两半）
    }
}
return 0;
}
int main()
{
    cin>>n>>m;
    fo(i,1,m)
    {
        int x,y;
        scanf("%d%d",&x,&y);
        link(x,y),link(y,x);
    }
    int ans=0;
    fo(i,1,n)
        if(!match[i]) ans+=find(i);
    printf("%d\n",ans);
    fo(i,1,n) printf("%d ",match[i]);
}
```

分类: A-学习小记 , 图论

好文要顶 关注我 收藏该文





BAJim_H
关注 - 4
粉丝 - 1

00

+加关注

« 上一篇: [比赛注意事项](#)
» 下一篇: [博客搬家啦!](#)

posted @ 2019-03-21 09:07 BAJim_H 阅读(146) 评论(0) 编辑 收藏

刷新评论 刷新页面 返回顶部

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。

- 【推荐】超50万C++/C#源码: 大型实时仿真组态图形源码
- 【前端】SpreadJS表格控件, 可嵌入系统开发的在线Excel
- 【推荐】码云企业版, 高效的企业级软件协作开发管理平台
- 【推荐】程序员问答平台, 解决您开发中遇到的技术难题

相关博文:

- 一般图最大匹配——带花树
- 带花树总结
- 【Learning】带花树——一般图最大匹配
- 【UOJ】#79.一般图最大匹配
- 带花树学习笔记

最新新闻:

- 他假装病人住进精神病院, 揭开了精神病学诊断的黑历史
- 谷歌进军电信? 拟联手Dish成立美国第四大运营商
- 新思科技助力, 三星5nm/4nm/3nm工艺再加速
- 华为在泰国手机市场份额增长至20%, 鸿蒙有没有具体时间表
- 联发科首发旗舰智能电视芯片S900: 支持8K视频解码
- » 更多新闻...