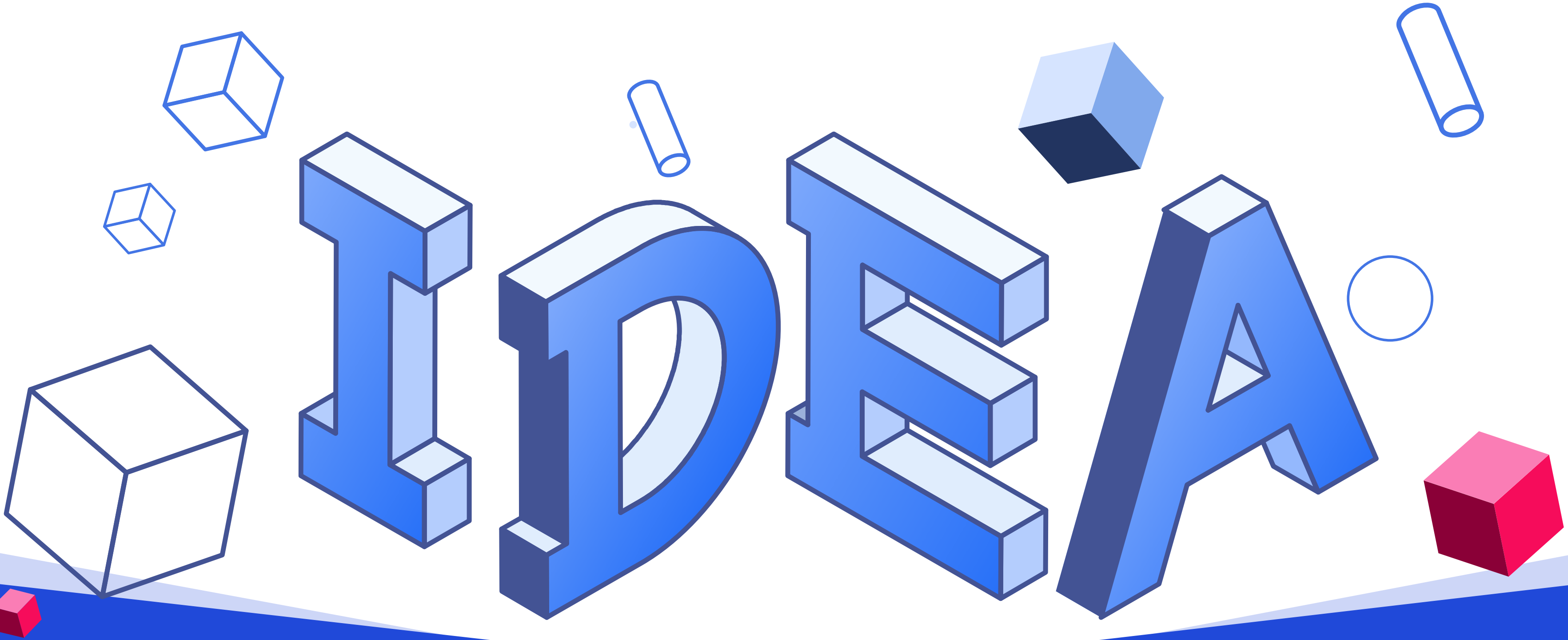


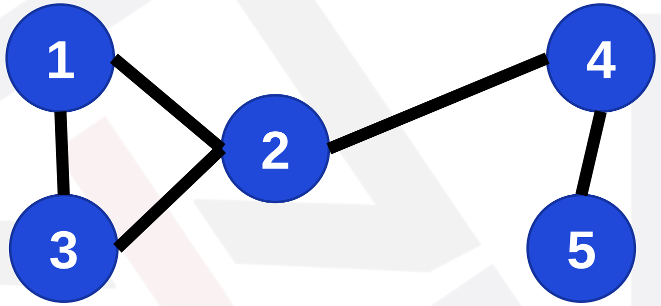
信息学暑期

割点与桥 (割边)

割点与桥(无向图)

割点与桥

- 割点：无向图G，结点x∈G，删去x以及与x相关联的边之后，G分裂成两个或者两个以上不相连的子图，称x为G的割点。

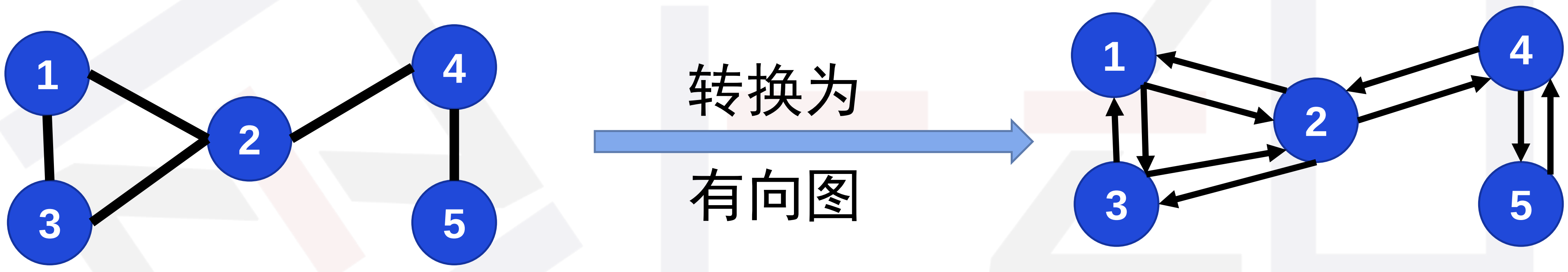


2和4都是割点

边(2, 4)，(4, 5)是割边

- 桥（割边）：无向图G，边e∈G，删去e之后，分裂成两个或者两个以上不相连的子图，称x为G的割边。

无向图Tarjan算法



无向图Tarjan算法

无向图的Tarjan算法，DFS的三种边：

- 树枝边：DFS时经过的边，即DFS搜索树上的边。
- 前向边：与DFS方向一致，从某个结点指向其子孙的边。
- 后向边：与DFS方向相反，从某个结点指向其祖先的边。

（无向图无横叉边）

dfn[u]：u在搜索树中的时间戳。

low[u]：u或u的子树中结点经过一条后向边（**不经过父结点**）能够追溯到的最小的dfn值。

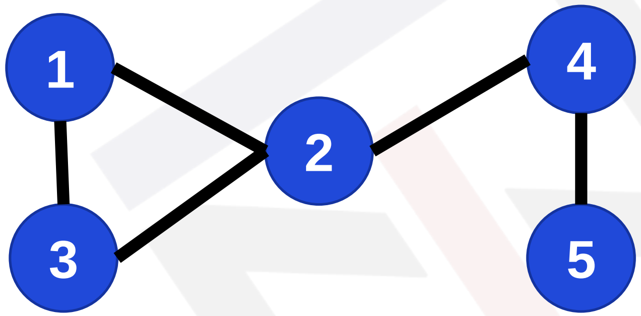
如果(u,v)是树枝边，u为v的父结点，则：

low[u]=min(low[u],low[v])

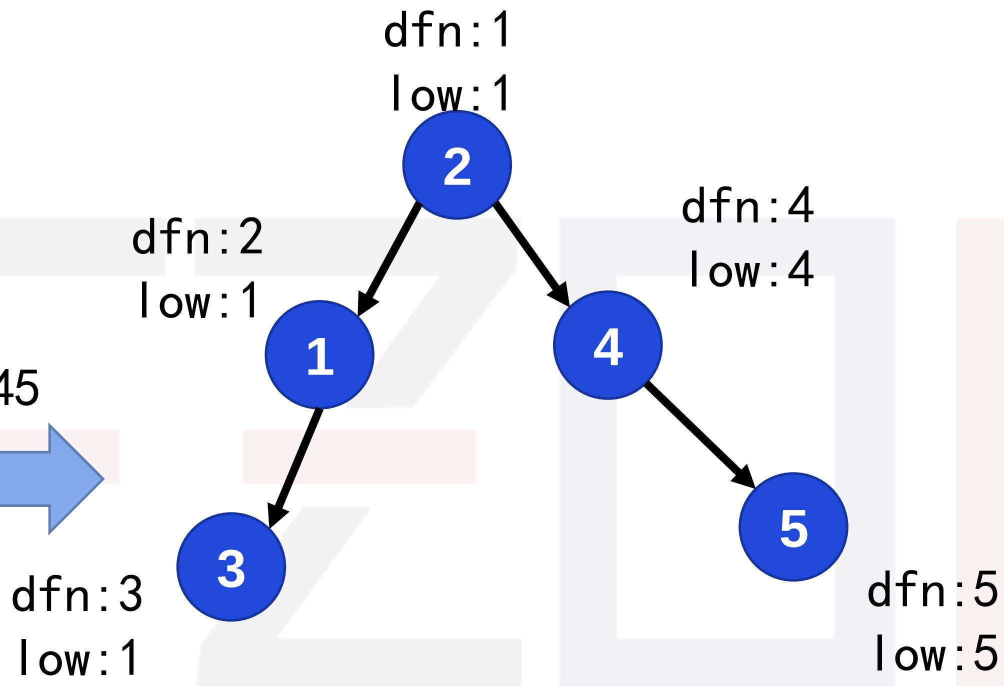
如果(u,v)是后向边，且v不是u的父亲结点，则：

low[u]=min(low[u],dfn[v])

Tarjan求割点



搜索顺序：21345

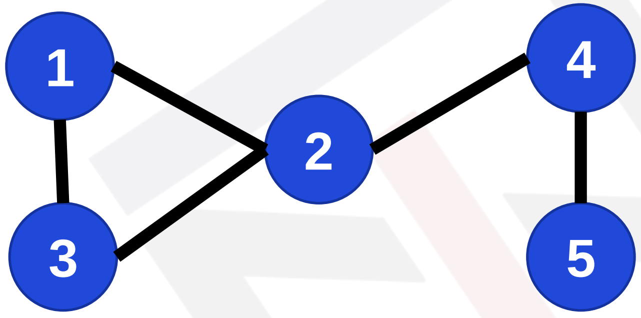


存在割点条件：

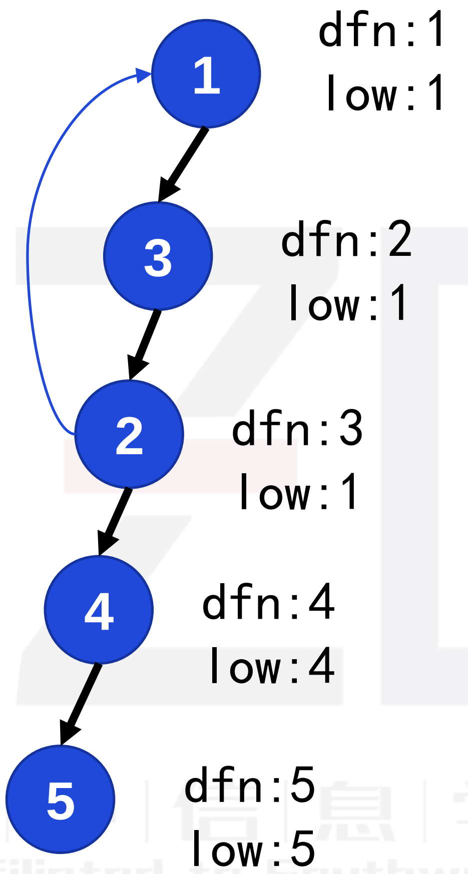
u为树根，且至少存在两个以上的子树。

因为删去u后，子树间没有边相连。

Tarjan求割点



搜索顺序：13245



存在割点条件：

u不为树根，存在树枝边(u, v)（即u为v的父亲），v必须通过u才能访问到祖先节点，那么去掉u之后，u的祖先节点和孩子节点就不连通了。满足：**dfn[u]<=low[v]**，u就是割点。

Tarjan求割点

割点判定:

1.根结点：当且仅当根节点至少有两个儿子时，是割点
2.其他点：对于其他点 v，当仅有一个儿子 u 时，从 u 或 u 的后代出发，没有指向 v 祖先(不含 v)的边，则删除 v后，u 和 v 的父亲不连通时，v 是割点

辅助数组:

dfn[ ]
low[ ]
cut[u]:表示u是否为割点

Tarjan求割点代码

西南大学附属中学
High School Affiliated to Southwest University

```
void tarjan(int u,int dad)  //u的父亲为dad
{
    dfn[u]=low[u]=++t;
    int v,k=0;
    for(int i=first[u];i!=-1;i=nex[i])
    {
        v=to[i];
        if(!dfn[v])
        {
            k++;          //统计子树个数
            tarjan(v,u);
            low[u]=min(low[u],low[v]);
            if( (u==root&&k>=2) || ( u!=root&&dfn[u]<=low[v] ) ) //割点判定

                cnt[u]=1;         //标记为割点
        }
        else if(v!=dad)                 //不经过父结点的后向边
            low[u]=min(low[u],dfn[v]);      //无向图，不需要栈
    }
}
```

避免多个独立的无向图出现：

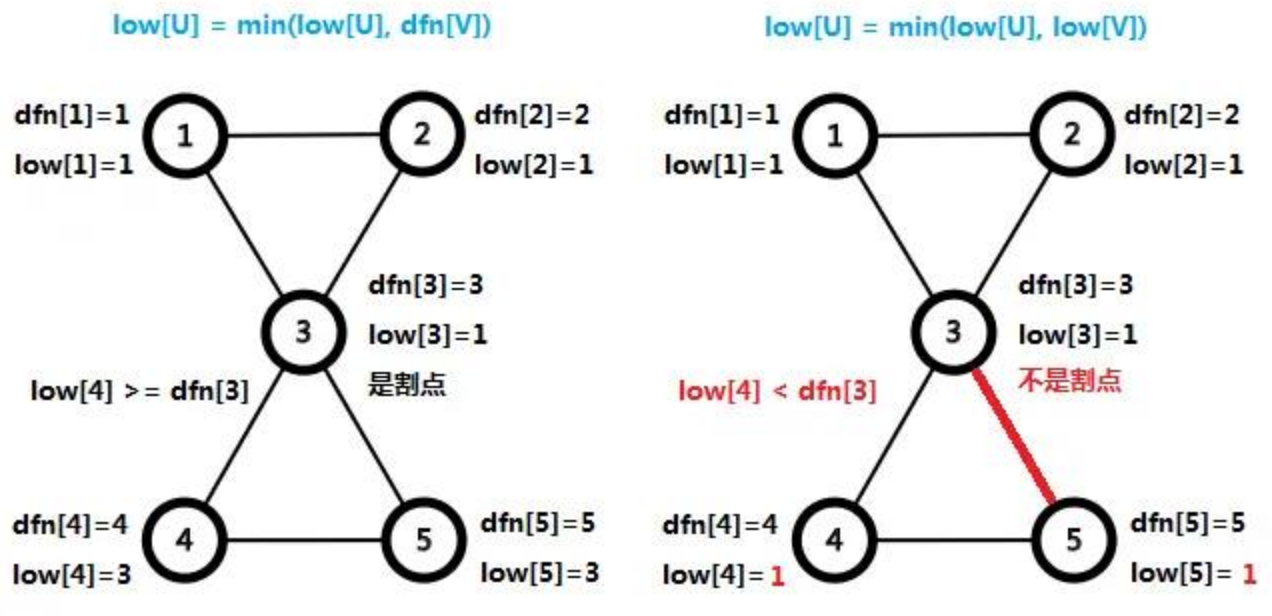
```
for(int i=1;i<=n;i++)
    if(!dfn[i])  {root=i;tarjan(i,i);}
```

关于low[v]与dfn[v]

关于为什么是low[u]=min(low[u],dfn[v])而不能是low[u]=min(low[u],low[v])?

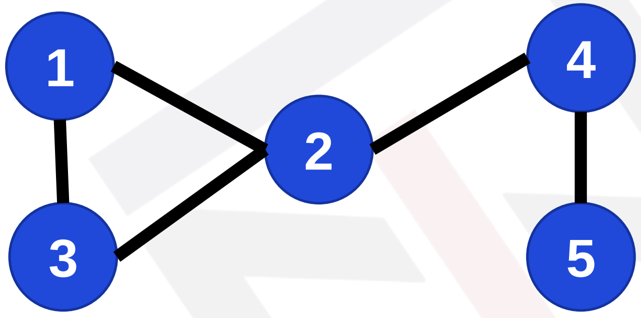
在求强连通分量时，如果v已经在栈中，那么说明u，v一定在同一个强连通分量中，所以到最后low[u]=low[v]是必然的，提前将low[u]更新为low[v]，和最后更新没有什么区别。

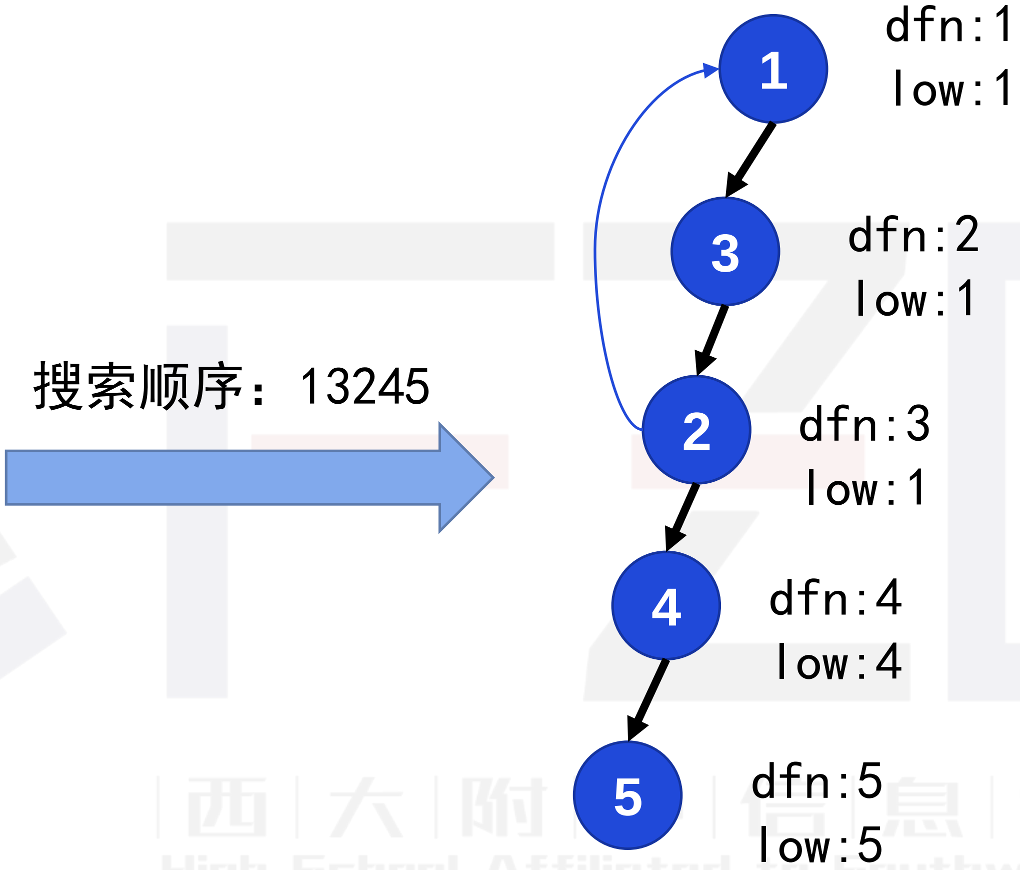
但在求割点时，由于是无向图，所以我们一个边存了两遍，而我们必须要绕开连接u及其父节点的边，此时如果v恰好是u的父节点（u,v之间有边相连）且只有这么一条路径，那么我们更换后无法绕开那条边，所以就会出现问题。



如果到达3号点后先走向了1号点更新low[3]，再走向4号点；会漏判割点

Tarjan求割边





边（u,v)是桥，必须为树枝边，且满足dfn[u]<low[v]

可以看成是割点的一种特殊情况，当结点u的子结点v的后代通过后向边只能连回v，那么删除这条边(u,v)就可以使得图G非联通

Tarjan求割边代码


```
void tarjan(int u,int dad)  //u的父亲为dad
{
  dfn[u]=low[u]=++t;
  int v;
  for(int i=first[u];i!=-1;i=nex[i])
  {
      v=to[i];
      if(!dfn[v])
      {
        tarjan(v,u);
        low[u]=min(low[u],low[v]);
        if( dfn[u]<low[v] )         //(u,v)为桥
          bridge[i]=1;              //第i条边为桥
      }
      else if(v!=dad)               //不经过父结点的后向边
        low[u]=min(low[u],dfn[v]);          //无向图，不需要栈
  }
}
//没有考虑重边的情况
```

避免多个独立的无向图出现：

```
for(int i=1;i<=n;i++)
    if(!dfn[i])  tarjan(i,i);
```

如果出现重边(x,y有多条边相连)怎么求桥?

low[u]：u或u的子树中结点经过一条后向边（**不经过父结点**）能够追溯到的最小的dfn值。

如果存在后向边(u,v)，且没有重边，v为u的父亲，dfn[u]是不能更新low[v]的。

但是如果出现重边，dfn[u]是能够更新low[v]的。

解决方案：

把无向图的每一条边都看作双向边，成对存储在下标“2和3”，“4和5”，“6和7”....。

如果沿着编号为i的边递归进入节点u，则忽略从x出发的编号为i xor 1的边。通过其他边计算low[x]。

(i xor 1，i为偶数，i xor 1为i+1；否则为i-1)

Tarjan求割边代码(重边)


```
void tarjan(int u,int edge)//第edge条边终点为u
{
        dfn[u]=low[u]=++t;
        for(int i=first[u];i!=-1;i=nex[i])
        {
                int v=to[i];
                if(!dfn[v])
                {
                        tarjan(v,i);
                        low[u]=min(low[u],low[v]);
                        if(dfn[u]<low[v])
                                bridge[i]=bridge[i^1]=1;
                }
                else if(i!=(edge^1))
                        low[u]=min(low[u],dfn[v]);
        }
}
```

无向图-双联通分量(DCC)

双联通分量

若一个无向图不存在割点,称它为“点双连通图”。即删除任意一个点都连通。

若一个无向图不存在桥(割边),称它为“边双连通图”。即删除任意一条边都连通。

无向图的极大点双连通子图称为“点双连通分量”，简称v-DCC。

无向图的极大边双连通子图称为“边双连通分量”，简称e-DCC。

边双连通分量

定理：一张无向连通图是“边双连通图”，当且仅当任意一条边都包含在至少一个简单环中。其中简单环指不自交的环。

边双连通分量的求法：

1. 使用Tarjan算法求出无向图的所有桥。
2. 再对无向图进行一次深度优先遍历，遍历过程中不经过桥，在同一棵搜索树上即为一个边连通分量，也是一个连通块。

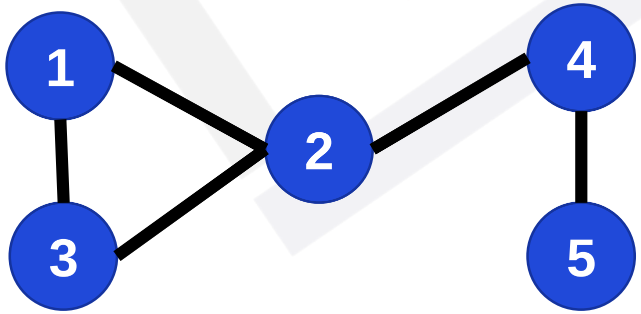
边双连通分量求解代码

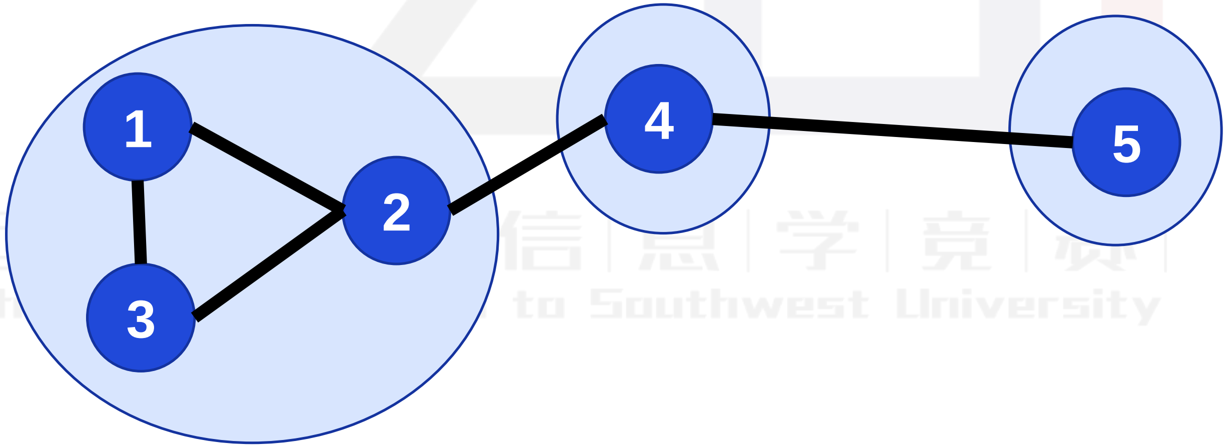
```
int c[N],dcc;
void tarjan(略){//求桥}
void dfs(int u)
{
		c[u]=dcc; //结点u属于第dcc个边连通分量
		for(int i=first[k];i!=-1;i=nex[i])
		{
				int v=to[i];
				if(c[v]||bridg[i]) continue;  //如果求解过或桥
				dfs(v);
		}
 }

 //以下在main函数中
for(int i=1;i<=n;i++)
		if(!c[i])  {dcc++;dfs(i);}
```

缩点：

将边双连通分量看作一个点，把桥边(u, v)看作是连接边双连通分量dcc[u]和dcc[v]的边，会产生一棵树。

可以存在一个新的邻接表中。





一个有桥的连通图，如何通过加边变成边双连通图？

1. 首先求出所有桥，对边双连通分量进行缩点，这些点通过桥连接在一起，形成一棵树。

2. 统计树中度为1的结点，即叶结点，记为leaf。则至少添加ceil（(leaf+1)/2））条边，使得树达到边双连通。当leaf=1时，不需要添加边。

原因：首先把两个最近公共祖先最远的两个叶节点之间连接一条边，这样可以把这两个点到祖先的路径上所有点收缩到一起，因为一个形成的环一定是双连通的。然后再找两个最近公共祖先最远的两个叶节点，这样一对一对找完，恰好是(leaf+1)/2次，把所有点收缩到了一起

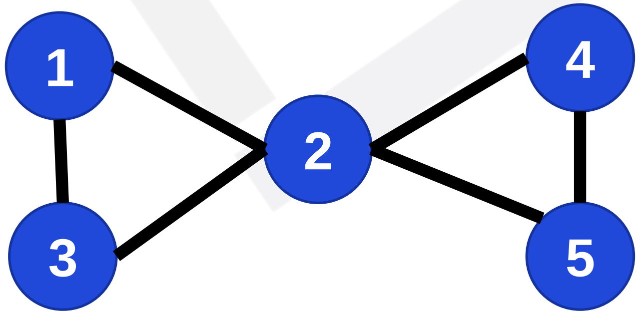
点双连通分量

定理：一张无向连通图是“点双连通图”，当且仅当满足下列条件之一：

1. 图中的顶点数超过2。
2. 图中任意两点都同时包含在至少一个简单环中。

若某个结点为孤立点，只有一个，则它单独构成一个点双连通分量。

割点可能属于多个点双连通分量



求解点双连通分量

当我们找到割点的时候，就已经完成了一次对某个极大点双连通子图的访问，那么我们如果在进行DFS的过程中将遍历过的点保存起来，是不是就可以得到点双连通分量了?

求解方法：

在Tarjan算法中，维护一个栈：

1. 当一个结点第一次被访问时，入栈。

2. 当割点判定法则中的条件dfn[u]<=low[v]成立时，无论u是否为根，都要：

(1)从栈顶不断弹出结点，直至结点v被弹出。

(2)刚才弹出的所有结点与结点u一起构成一个点双连通分量。

```
void tarjan(int u,int dad)
{
        dfn[u]=low[u]=++t;
        sta[++top]=u;
        if(root==u&&first[u]==-1) //处理孤立点
        {
                dcc[++cnt].push_back(u);
                return;
        }
        int k=0;
        for(int i=first[u];i!=-1;i=nex[i])
        {
                int v=to[i];
                if(!dfn[v])
                {
                        tarjan(v,u);
                        low[u]=min(low[u],low[v]);
                        if(dfn[u]<=low[v])  //满足条件说明存在点双连通分量
                        {
                                k++;
                                if(u!=root||k>1) cut[u]=1;//割点标记
                                cnt++;                    //第cnt个点双连通分量
                                int z;
                                do{
                                        z=sta[top--];//弹出
                                        dcc[cnt].push_back(z); //存到dcc里
                                }while(z!=v);
                                dcc[cnt].push_back(u);
                        }
                }
                else if(v!=dad)
                        low[u]=min(low[u],dfn[v]);
        }
}
```

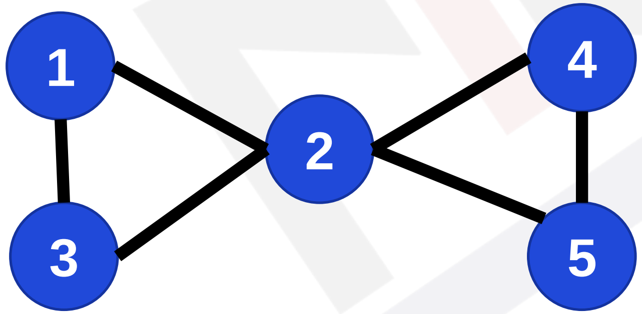
因为割点可能属于多个点连通分量，所有不能用标记的方法，只能标记每个点属于的点双连通分量，这里使用vector数组。

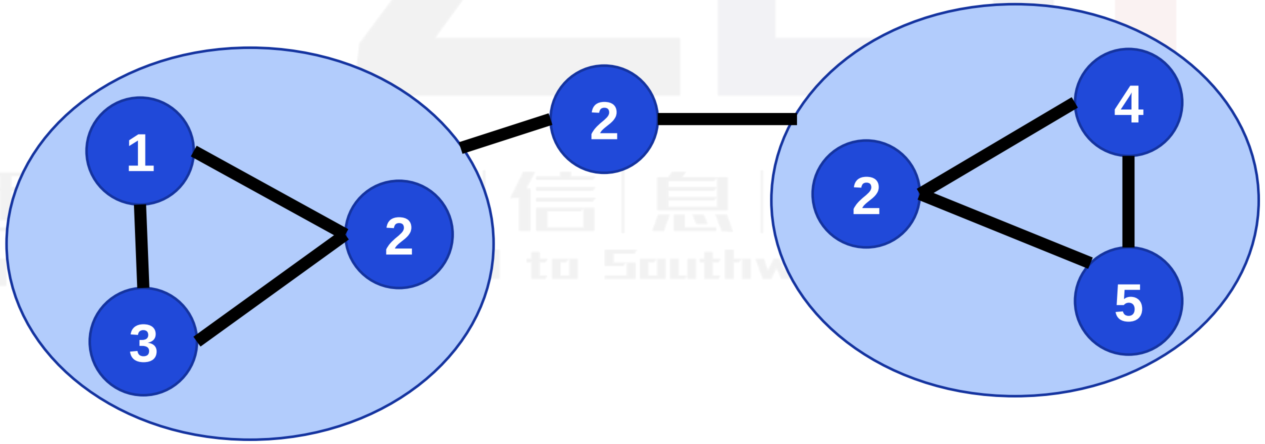
点双连通分量缩点

缩点：

因为一个割点可能属于多个点双连通分量。

如果有p个割点和t个点双连通分量，需要建一个p+t个结点的新图，把点双连通分量后割点作为图中的结点。





Thanks

For Your Watching

FZOI

西大附中信息学竞赛

High School Affilinted to Southwest University