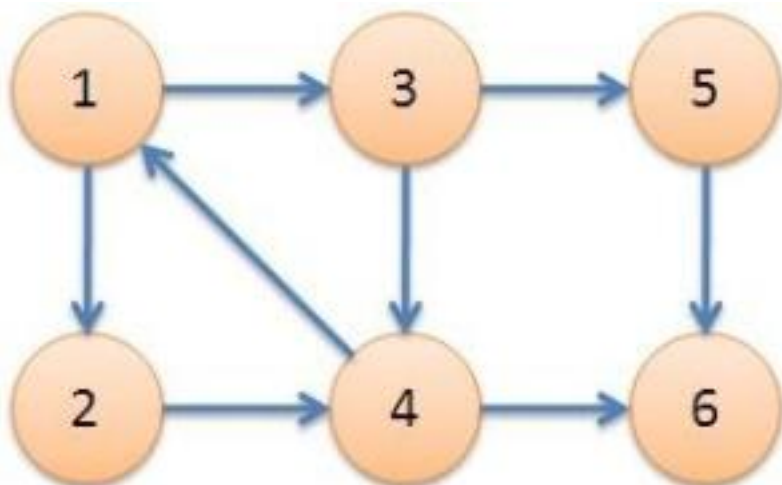


问题概述

Tarjan算法是一种由Robert·Tarjan(罗伯特·塔扬)发明的在有向图中求强连通分量的算法。

概念描述

如果在一个有向图中，任意两个顶点都可以相互到达，则称这个有向图为强连通图，非强连通图的极大强连通子图称为强连通分量。



例如上图中，由点1、2、3、4构成的子图就是一个强连通分量。

算法流程

首先引入两个数组dfn[]和low[]其中dfn[i]表示点i第一次被搜索到的时间，与dfn序类似low[i]表示在点i为根的dfs子树中，能够到达的点中dfn值的最小值。在整个算法结束后low[]值相同的点就在同一强连通分量中。注意在初始化时low[i]=dfn[i]

首先我们对所有dfn[i]=0的点i进行dfs将搜到过的点压入一个栈中，如果下一个点已经在栈中，则更新low值。如果在某一个点回溯时发现这个点有dfn[x]=low[x]说明以这个点为根的dfs子树处于同一个强连通分量中，我们把栈顶元素弹出直到x被弹出，这些被弹出的点组成一个强连通分量。

以上面的图作为一个例子Tarjan算法的流程如下：

dfs到1 dfn[1]=low[1]=1;

dfs到2 dfn[2]=low[2]=2;

dfs到4 dfn[4]=low[4]=3 4可以到1，1在栈中 low[4]=1;

dfs到6 dfn[6]=low[6]=4 6无法继续dfs 有dfn[6]=low[6] 从栈中弹出6，其自己作为一个强连通分量。


从4回溯到2 low[2]=1;

从2回溯到1，然后dfs到3 $\text{dfn}[3]=\text{low}[3]=5$ 3可以到4，4在栈中 $\text{low}[3]=1$

dfs到5 $\text{dfn}[5]=\text{low}[5]=6$ 由于6已经dfs过，所以无法继续dfs $\text{dfn}[5]=\text{low}[5]$ 从栈中弹出5，其自己作为一个强连通分量。

回溯到1，有 $\text{dfn}[1]=\text{low}[1]$ 从栈中弹出3、4、2、1，它们构成一个强连通分量。

From:
<https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link:
<https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:hotpot:tarjan&rev=1589362411> 

Last update: **2020/05/13 17:33**