# 概念

定义：

强连通分量是对**有向图**来说的，如果两个顶点可以相互通达，则称两个顶点强连通(strongly connected)。如果有向图G的每两个顶点都强连通，称G是一个强连通图。有向图的极大强连通子图，称为强连通分量(strongly connected components)。

下面设图是(V\*E),V代表点数量，E代表边数量

# 强联通分量算法

## Korasaju

这个算法是比较好理解的求图强联通分量的算法，时间复杂度也较好，但比起tarjan，常数时间略慢，且需要额外空间多于tarjan;

**思路：**

1构建一个和原图G相似的图GT，这个图每条边方向和原图相反。

2深度优先遍历G，算出每个结点u的结束时间f[u],起点如何选择无所谓。遍历所有点一次，对于每个到达的点，分别记录进入的时间和结束的时间在begintiem[]和endtime[]这2个数组中。其中begintiem[]用来判断当前点有没有走过，endtime[]接下来用到

3深度优先遍历G的转置图GT，对于GT不断选择起点做dfs,

选择遍历的起点时，按照上面步骤2的dfs结束时间(endtime)从大到小进行。

每进行一次dfs就找到一个强联通分量，对于不同的题，再进行处理

## 分析：

时间复杂度V+E，dfs是用了点数2\*V的次数，而合计尝试次数是边数2\*E。

* 上面的解释已经包含了带有孤立点的情况，重边用脚也能想到，在联通分量问题上不必考虑。
* 自环，说明它自己就是一个强联通的，其实也不需要考虑。

## Tarjan

Tarjan算法一种由Robert Tarjan提出的求解有向图强连通分量的算法，它能做到线性时间的复杂度。

**用到数据结构：**

**(1)、dfn[ ]，表示这个点在dfs时是第几个被搜到的。**

**(2)、low[ ]，表示这个点以及其子孙节点连的所有点中dfn最小的值**

**(3)、stack[ ]（栈），表示当前所有可能能构成是强连通分量的点。**

**(4)、inSt[ ]，表示一个点是否在stack中。**

**大概过程：**

做一遍无回溯的dfs(需要V的次数)，用dfn[i]表示编号为i的节点在DFS过程中的访问序号(也可以叫做开始时间）。在DFS过程中会形成一搜索树。在搜索树上越先遍历到的节点，显然dfn的值就越小。dfn值越小的节点，就称为越“早” 。实时更新low，通过low的值来判断环，给每个环有一个起点，更新low[]之后依然low==dfn的点视为某个强联通分量的起点

**算法步骤：**

任选一个点作为起始点，做递归的dfs,函数体外部初始化len=1;len代表第几个被搜到

1. 每次dfs到一个点now把当点入栈，并且标记dfn值以及low值都是len,标记当前点now在栈isSt里面,然后len++
2. 按照常规dfs方法递归，当前点now周围的点next，用邻接表很容易迭代，每次迭代判断：next是不是走过了(通过inSt的值 或 者dfn是不是0)，没走过就去走；不论next在不在栈中，（也就是不论next有没有，走么走过），都要更新当前点的low，low[now]=min(low[now], low[next])
3. 当前函数结束，要寻找一个环，怎么寻找，dfn[now]==low[now] ？，如果满足，说明这个点的low没被更新，那么它一定是某个环的开始节点( 这是因为每次做了low[now]=min(low[now], low[next]的步骤 ) ，那么出栈直到栈顶是你当前的点now被出栈为止，出栈的所有点构成一组环（强联通分量）。

## 分析：

时间复杂度V+E，dfs是用了点数V的次数，而合计尝试次数是边数E。

* 这个算法虽然是任意选点，但是如果你选了不能到达全部点的点，就不能找到这个图集全部都的 强联通分量，所以对于常规寻找图里，有几个强联通分量的问题，应该再开一个vis数组纪录有没被访问过，来对每一个没被访问的点为起点，尝试做这个算法。然后纪录个数
* 上面的解释已经包含了带有孤立点的情况，重边用脚也能想到，在联通分量问题上不必考虑。
* 自环，说明它自己就是一个强联通的，其实也不需要考虑。

# 连通分量的扩展

## 缩点：

有这样一类问题，给你一个图，问你有哪些点能到达这个图的所有点，

思路就是求强联通分量把他们每组缩成一个点建立新的图，找出新图入度是0的点，如果恰好一个，这个点对其他点一定是可达的，对应的强联通分量的集合就是原图能到达所有点的点的集合；如果不止有一个入度是0的点，那么这些点互相一定不可达，无解

**代码如何设计：**

在原题强联通分量基础上，加一个数组kind[]代表当前点属于哪个集合加在找联通分量代码里，之后做完Tarjan去遍历每条边通过边的两端点属于不同集合时来给新建的图构建邻接表。有些问题中不一定非得建新图，只需要kind数组就够了，kind数组的值由大到小满足拓扑序，所以tarjan算法缩点后不用拓扑排序。

## 无向图强连通分量和仙人掌图找环

在有向图强连通分量的tarjan算法基础上，稍加修改，判断每次找的时候不能向后退即可。可以用来求：仙人图的每个环具体包含哪些边。