1. **点连通度与边连通度**

在一个无向连通图中，如果有一个顶点集合，删除这个顶点集合，以及这个集合中所有顶点相关联的边以后，原图变成多个连通块，就称这个点集为割点集合。一个图的点连通度的定义为，最小割点集合中的顶点数。

类似的，如果有一个边集合，删除这个边集合以后，原图变成多个连通块，就称这个点集为割边集合。一个图的边连通度的定义为，最小割边集合中的边数。

1. **双连通图、割点与桥**

如果一个无向连通图的点连通度大于1，则称该图是点双连通的(point biconnected)，简称双连通或重连通。一个图有割点，当且仅当这个图的点连通度为1，则割点集合的唯一元素被称为割点(cut point)，又叫关节点(articulation point)。

如果一个无向连通图的边连通度大于1，则称该图是边双连通的(edge biconnected)，简称双连通或重连通。一个图有桥，当且仅当这个图的边连通度为1，则割边集合的唯一元素被称为桥(bridge)，又叫关节边(articulation edge)。

可以看出，点双连通与边双连通都可以简称为双连通，它们之间是有着某种联系的，下文中提到的双连通，均既可指点双连通，又可指边双连通。

1. **双连通分支**

在图G的所有子图G'中，如果G'是双连通的，则称G'为双连通子图。如果一个双连通子图G'它不是任何一个双连通子图的真子集，则G'为极大双连通子图。双连通分支(biconnected component)，或重连通分支，就是图的极大双连通子图。特殊的，点双连通分支又叫做块。

1. **求割点与桥**

该算法是R.Tarjan发明的。对图深度优先搜索，定义DFS(u)为u在搜索树（以下简称为树）中被遍历到的次序号。定义Low(u)为u或u的子树中能通过非父子边追溯到的最早的节点，即DFS序号最小的节点。根据定义，则有：

Low(u)=Min { DFS(u) DFS(v) (u,v)为后向边(返祖边) 等价于 DFS(v)<DFS(u)且v不为u的父亲节点 Low(v) (u,v)为树枝边(父子边) }

一个顶点u是割点，当且仅当满足(1)或(2)：

(1) u为树根，且u有多于一个子树。

(2) u不为树根，且满足存在(u,v)为树枝边(或称父子边，即u为v在搜索树中的父亲)，使得DFS(u)<=Low(v)。

一条无向边(u,v)是桥，当且仅当(u,v)为树枝边，且满足DFS(u)<Low(v)。

1. **求双连通分支**

下面要分开讨论点双连通分支与边双连通分支的求法。

对于点双连通分支，实际上在求割点的过程中就能顺便把每个点双连通分支求出。建立一个栈，存储当前双连通分支，在搜索图时，每找到一条树枝边或后向边(非横叉边)，就把这条边加入栈中。如果遇到某时满足DFS(u)<=Low(v)，说明u是一个割点，同时把边从栈顶一个个取出，直到遇到了边(u,v)，取出的这些边与其关联的点，组成一个点双连通分支。割点可以属于多个点双连通分支，其余点和每条边只属于且属于一个点双连通分支。

对于边双连通分支，求法更为简单。只需在求出所有的桥以后，把桥边删除，原图变成了多个连通块，则每个连通块就是一个边双连通分支。桥不属于任何一个边双连通分支，其余的边和每个顶点都属于且只属于一个边双连通分支。

1. **构造双连通图**

一个有桥的连通图，如何把它通过加边变成边双连通图？方法为首先求出所有的桥，然后删除这些桥边，剩下的每个连通块都是一个双连通子图。把每个双连通子图收缩为一个顶点，再把桥边加回来，最后的这个图一定是一棵树，边连通度为1。

统计出树中度为1的节点的个数，即为叶节点的个数，记为leaf。则至少在树上添加(leaf+1)/2条边，就能使树达到边二连通，所以至少添加的边数就是(leaf+1)/2。具体方法为，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。