# 生成树计数

## Matrix 定理

\*算法思想：

\*(1)G的度数矩阵D[G]是一个n\*n的矩阵,并且满足:当i≠j时,dij=0;当i=j时,dij等于vi的度数;

\*(2)G的邻接矩阵A[G]是一个n\*n的矩阵,并且满足:如果vi,vj之间有边直接相连,则aij=1,否则为0;

\*定义图G的Kirchhoff矩阵C[G]为C[G]=D[G]-A[G];

\*Matrix-Tree定理:G的所有不同的生成树的个数等于其Kirchhoff矩阵C[G]任何一个n-1阶主子式的行列式的绝对值；

\*所谓n-1阶主子式,就是对于r(1≤r≤n),将C[G]的第r行,第r列同时去掉后得到的新矩阵,用Cr[G]表示;

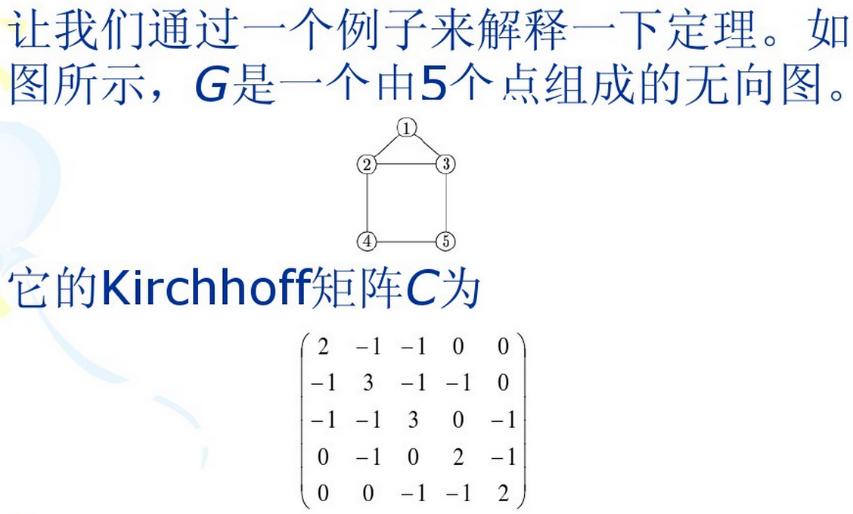
\*

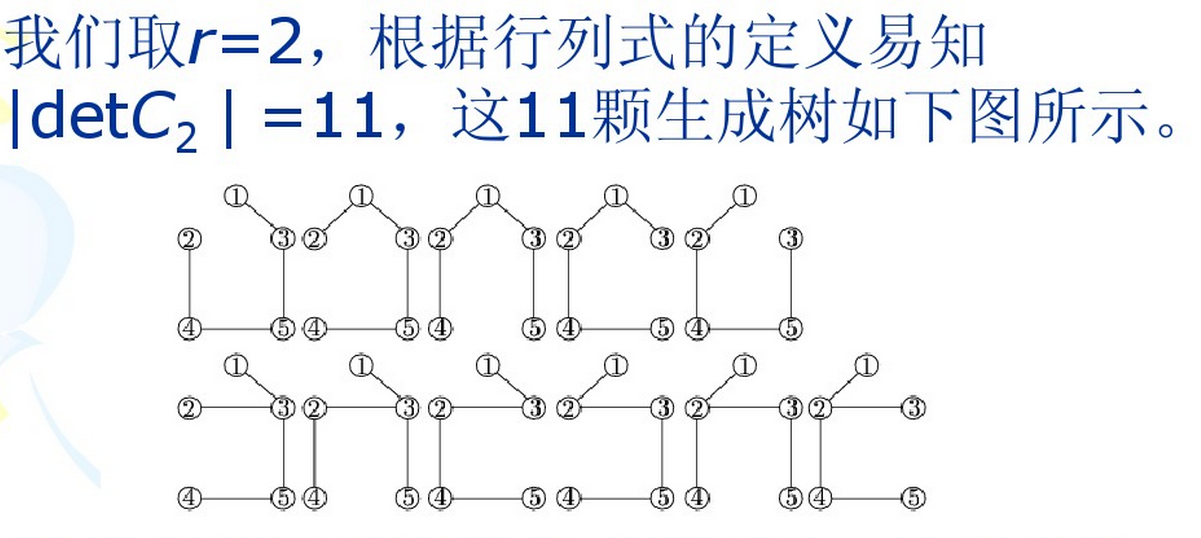
\*Kirchhoff矩阵的特殊性质：

\*(1)对于任何一个图G,它的Kirchhoff矩阵C的行列式总是0,这是因为C每行每列所有元素的和均为0;

\*(2)如果G是不连通的,则它的Kirchhoff矩阵C的任一个主子式的行列式均为0;

\*(3)如果G是一颗树,那么它的Kirchhoff矩阵C的任一个n-1阶主子式的行列式均为1;





## HDU 4408

【题意】

一张无向图，要求求出其中最小生成树的棵树。

【分析】

生成树计数可以使用Matrix-Tree定理解决，本题最主要的区别是有了一个最小生成树的额外条件。

首先考虑一下如何得到最小生成树。

Kruskal算法的基本思想是，按照边长排序，然后不断将短边加入集合，最终一步如果能成功把n-1条边都加入同一个集合，则找到了最小生成树。在维护集合时，可以使用并查集来快速处理。

如果把Kruskal的过程按照边长划分成多个阶段，实际上是处理了所有短边的连通性之后继续处理下一个长度的边的连通性，并依次继续处理剩下边的连通性。然后我们可以发现，不同长度的边之间的连通性互不影响！！！

假设存在n1条长度为c1的边，n2条长度为c2的边...则Kruskal首先处理c1边的连通性，然后处理c2边的连通性，对于c1边的连通性的处理可能有多种方案，即从n1条边中取出一定数量的边构成最大连通图，但是最终处理完之后的结果对于c2来说是完全一样的。因此算法就出来了，在Kruskal的基础上，使用Matrix-Tree定理处理每个阶段生成树的种数，最后将所有阶段的结果相乘即可。

具体实现为：

在Kruskal的基础上，每完成一个阶段（检查完一个长度），就将所有遍历过的点缩成一个点，然后用Matrix-Tree定理计算该点与下一组点组成的连通图中生成树的个数。最终把每一个阶段的结果相乘即可。