最小生成树

引言

图论的基础内容，例如稀疏图用邻接链表，稠密图用邻接矩阵□DFS□BFS和Dijkstra都已经写过了，一个重要工具并查集也写过了。那么接下来，本文是图论的核心内容：最小生成树。

最小生成树的定义等等，都已经十分清楚了，无需多讲。还需要用到一个工具叫割集（反圈），列在第一部分。“ 割集 ”是离散数学的名词，与任韩图论中“ 反圈 ”是同一个概念。割集也称为反圈，是因为在最小生成树体系下割集与圈是对偶的两个概念，很多性质可以照搬。

接下来会简要阐述算法部分□Prim算法（又称反圈法或割集法），可以使用优先队列优化，适用于稠密图，因此常与邻接矩阵搭配□Kruskal算法（避圈法），必须采用并查集优化，适用于稀疏图，因此常与邻接链表搭配。

其他不常见的算法例如Rosenstiehl算法（破圈法），因为难以优化，不如上两种算法好，已经被弃用了，因此不讲。

割集（反圈）

割集（反圈）：将图的顶点集划分为两部分，连接两部分的全体边构成割集。将图G的顶点集合V分成非空两部分□S和V-S□图G中连接S和V-S两部分的边的全体，称为G的一个反圈。

简单地来讲，反圈刻画了图G两部分顶点间的连通性。如果这两部分顶点之间连通性强，那么对应的反圈的边数多。同样地，如果这两部分顶点之间连通性弱，那么对应的反圈的边数少。

因此有一个简单结论：

对于图G的支撑子图H□如果H连通，那么H与G的任意一个反圈有公共边。如果H不连通，那么存在G的反圈与H无公共边。

这个模型称为“ 反圈 ”的原因，与下面的结论有关。

设T是图G=(V,E)的一个支撑树，则T满足以下特征：

T中没有圈□G-E(T)中没有G的反圈。

T添加任何一条边后，图中的边包含且仅包含一个圈，称为G关于T的一个基本圈。

对于T中任意一条边e□G-E(T)-e中包含且仅包含G的一个反圈，称为G关于T的一个基本反圈。

只需要解释基本反圈的唯一性。事实上这个命题等价于：

从树中任意去掉一条边，恰好包含两个分支。

因此仅当反圈选择的两部分顶点恰好为这两个分支的时候，才能与去掉一条边的树恰好没有公共边。

基本割集（反圈）：生成树的每条树枝对应一个基本割集。

割集与生成树至少有一条公共边。

圈与割集有偶数条公共边。

圈在生成树外至少有一条边。

基本圈中的弦，位于圈中树枝的每一个基本割集，不在其他基本割集中。

最小支撑树T满足以下特征：

对于T中任意一条边e□e是由e决定的基本反圈中的最小权边。

对于不在T中的任意一条边e□e是由e决定的基本圈中的最大权边。

Prim算法（反圈法）

第一步：任取一个节点作为初始点。

第二步：从已选点和未选点构成的反圈中，选择权最小的边。如果有多条边权最小，则任选其中一条。

第三步：若在某一步，反圈为空集，则图中没有支撑树。若在某一步，已经选择了图的所有顶点，则所有被选择的边构成最小树，算法终止。

Kruskal算法（避圈法）

第一步：选取一个无圈支撑子图。初始时选择图的全体节点。

第二步：若图连通，则它是最小支撑树。

第三步：若图不连通，则选择边e□两个端点属于不同分支，并且权最小。重复上述过程。

有趣结论

在本文的最后，介绍一个有趣的结论。

如果存在图G的一个支撑树T□使得所有的基本圈长都是偶数，那么G的所有圈长都是偶数，即G是二部图。

证明这个结论的关键，在于编程中常用的“异或”运算。

异或运算的实质是，将并集中出现偶数次的元素去掉。为了叙述方便，简记每个节点度数都是偶数的子图为“偶度子图”。有以下显然结论：

非空偶度子图一定有圈。

偶度子图的异或还是偶度子图。

不含奇长圈的图（二部图），异或也不含奇长圈。

于是要证结论等价于：

图G中任意一个圈可以写成基本圈的异或。

这近乎于显然。图G中的圈C□必然有边不在支撑树T中，每一条这样的边都对应一个基本圈。这些基本圈的异或全体可以得到图C0□只要证明C0和C完全相同。

由条件□C0是偶度子图。

对于树T之外，圈C中的边，由基本圈的定义，这些边在C0中存在并且仅存在一次。

对于树T之外，不在圈C中的边，显然也不会在选取的基本圈中，所以不在C0中。

因此考虑C0和C的异或，得到的结果必然包含在树T之内。但是它们的异或是偶度子图，而树T当中没有圈，由于非空偶度子图一定有圈，只能说明C0和C的异或是空集。因此C0和C完全相同。