常见的最小生成树算法及其应用

陈佩琳

（北京理工大学计算机学院 人工智能专业）

**摘 要**：图论在生活中有许多应用场景。图分为有向图和无向图，通过最小生成树的定义和性质可以得出求无向连通图的最小生成树的一般算法，进而得出求最小生成树的常见算法，比如Prim算法和Kruskal算法，并分析其时间复杂度。这些算法有不同的适用场景。最后简要地说明最小生成树在解决实际问题当中的应用。

**关键词**：图论；最小生成树；实际应用

## **Common minimum spanning tree algorithms and their application**

CHEN Peilin

(Artificial intelligence, School of computer science,

Beijing University of Technology)

**Abstract**: Graph theory has many application scenarios in our daily life. Graph are divided into directed graphs and undirected graphs. Through the definition and nature of the minimum generating tree, we can get the general algorithm for finding the minimum generating tree of the undirected connected graph, and then get the common algorithm for finding the minimum generating tree, such as Prim algorithm and Kruskal algorithm, and analyze their time complexity. These algorithms have different applicable scenarios. Finally, the application of the minimum generating tree in solving practical problems is briefly explained.

**Key word:** graph theory; minimum spanning tree; practical application

在生活中，我们经常碰到这样的一类问题，例如：有n个电子元件，用导线把它们连接起来，使得每两个元件之间都有通路，并且所用的导线总长度最短。这类问题转化成图论就是求一个无向连通图中一个极小连通子图，使这个极小连通子图中所有边的权值之和最小。这个子图就是一个图的最小生成树。

**1 最小生成树的定义与一般求解算法**

**1.1 定义**

设一个无向连通图G（V,E），其中V是点的集合，E是边的集合。给每条边一个权值w，需要找到一个无环的边集T，使得T能连接G中所有的点，且总权值最小。因为T是从G中生成的无环边集，那么它一定是一棵树，称为生成树。一个连通图的生成树是一个极小的连通子图，包含全部|V|个顶点和|V| - 1条边。总权值最小的生成树就是最小生成树。

**1.2 一般求解算法**

一般的求解无向连通图的最小生成树的算法都有共同点。一般先设置一个边的集合A，开始时先把A初始化为空集。在算法执行的整个过程中A始终是最小生成树的子集。这个算法执行的每一步都要判断一条边加入A后能否保证A仍然是最小生成树的子集。这个算法实际上是一种贪心算法，流程图如下图所示。

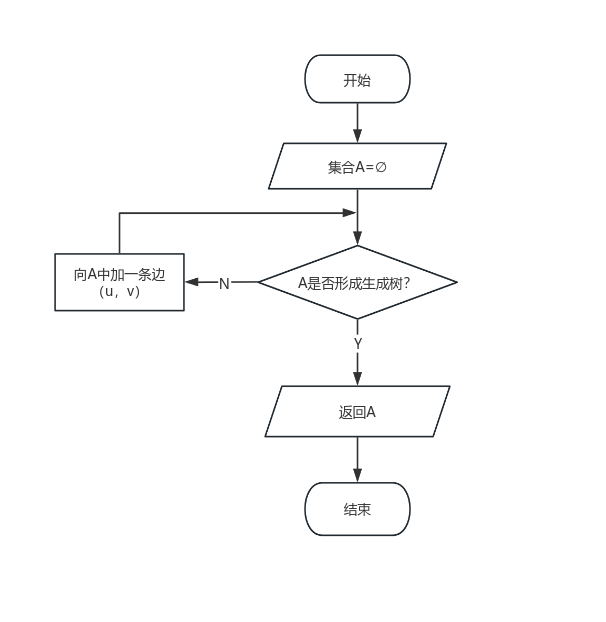


图1 一般的求解无向连通图的最小生成树的算法流程图

Fig.1 Flow chart of the Generic-MST algorithm

该算法每循环一次确定一条最小生成树的边，对于图G（V，E）一共执行|V|-1次；初始状态下A=∅，若边（u，v）满足如下所述的性质，则将（u，v）加入集合A中。

**MST性质**：设连通图G = (V, E)，U是V的非空真子集，边(u, v)是所有一端在U中，另一端在V-U的边中，代价最小的；则在G的所有最小生成树中，一定有一棵包含(u, v)。[1]

**2 最小生成树的常见求解算法**

上一节讨论了最小生成树的基本算法，但是为了在不同的实际场景应用最小生成树算法，还需要一些具体的实现方式。基于上一节所述的思想的常用算法主要有Prim算法和Kruskal算法，它们有不同的特点和适用场景。

**2.1 Prim算法（加点法）及其优化方法**

本算法的基本思想是：如果（u,v）是一条权值最小的边，那么这条边一定包含在最小生成树里。在这个算法执行的过程中，表示最小生成树的边的子集A中的边始终只形成一棵树。这棵树从任意根节点r开始形成，每一步向树中添加一条使树的权值尽可能小的边，直到该树包含了图G中所有的节点。由此可见，Prim算法是基于“贪心选择”策略的算法。由第1节中的MST性质，每次加入到树中的边都属于图G的最小生成树。因此当算法终止时，集合A中的边就形成了一棵最小生成树。

下面是用Prim算法求最小生成树的代码。我们把图G（V，E）中的点分为两个集合U和V-U，其中U包含已经属于生成树的点，V-U则包含剩余的点。对于每一个点，dist[i]表示点i到已经在树中的节点的最小权值，若不存在这样的路径则dist[i]=。Pre[i]表示点i的“父节点”。在算法执行的过程中，V-U中的点都在以dist[i]为优先级排序的优先队列Q里，当算法终止时，队列Q为空。

1 //这种实现方式使用邻接矩阵存储图

2 const int maxn = 100;

3 void Prim (int n,int dist[maxn],int map[maxn][maxn],int pre[maxn])

4 //n个点，dist[i]表示向外延伸的最短边长，map记录图的邻接矩阵，pre[]记录每个点连接的前一个点（父节点）的信息，

5 //dist之和最最小权值

6 {

7 int i,j,k;

8 int min;

9 bool p[maxn];//记录该点是否属于U

10 for (i = 2;i <= n;i++)

11 {

12 p[i] = false;

13 dist[i] = map[1][i];

14 pre[i]=1;

15 }

16 dist[1]=0;

17 p[1]=true; //设置根节点

18 for (i=1;i<=n-1;i++) //循环n-1次，每次加入一个点

19 {

20 min=INT\_MAX;

21 k=0;

22 for (j =1;j <= n ;j ++)

23 {

24 if (!p[j] && dist[j] < min)

25 {

26 min = dist[j];

27 k = j;

28 }

29 }

30 if (k == 0)return; //如果没有点可以扩展，即图G不连通，返回

31 p[k] = true;

32 for (j = 1; j <= n; j++)

33 {

34 if (!p[j]&&map[k][j] != INT\_MAX && dist[j] > map[k][j])

35 {

36 dist[j] = map[k][j]; //更新每个定点所连接的权值最小的边

37 pre[j] = k; //更新每个顶点的父节点信息

38 }

39 }

40 }

41 }

可以注意到，Prim算法的依靠邻接矩阵存储图的实现方式包含一个2层嵌套for循环。因此这种实现方法的时间复杂度是。由于Prim算法运算过程中只包含搜索点的操作，不搜索边，因此实际运用时的运算时间只与图的定点数量有关，与边数无关。由此可见，常规的Prim算法更适合求解稠密图的最小生成树。

那么如何优化Prim算法？可以使用堆来存储优先队列Q。设图G有n个顶点，m条边。如果使用堆来保存V-U中每一点到U中所有点的最小权值，则初始化堆的部分运行时间为O（n）。每次从堆中找到最小权值所需的时间为O（），且对于有n个顶点的图G，这个过程一共要进行n次，故“找到最小权值”的总时间为O（）。由于所有邻接表的长度之和为2m，更新顶点的邻接信息的总用时为O（）。最后对dist[i]更新赋值后需要调整优先队列，因此需要对堆进行调整。这个过程用时为。优化后的Prim算法的时间复杂度为，相比于有明显提高，更适合处理稀疏图。

**2.2 Kruskal算法（加边法）**

Kruskal算法也是基于贪心选择策略的算法。对于图G（V,E），先构造G’=（V，∅），然后每一轮向G’中添加剩余的边里权值最小的边。如果加入一条边后G’中存在环，那么移除这条边。重复以上过程直到G’构成一棵树。判断新加入G’的边是否会在G’中产生环可以用并查集。并查集中每一个等价类代表G’的一个连通分支（即森林里的一棵树）。如果要添加的边的两端在同一个等价类里，说明加入l这条边就会产生环，因此不应该加入这条边；否则保留这条边，然后合并这条边两端所在的等价类。

Kruskal算法的步骤用伪代码的表示方式如下所示，其中使用了并查集的初始化（initSet）、查找（findSet）、合并（Union）操作。

Kruskal（G（V，E），权w）

1. A=∅
2. for（v in V）
3. initSet（v）//初始化并查集的等价类，首先为每一个点都单独设置一个等价类
4. 按权w对E中的边进行非递减的排序
5. for（边（u，v）in E）
6. if（findSet（u）！= findSet（v））
7. A=A∪{（u，v）}
8. Union（u，v）
9. return A

如果选择合适的排序算法，步骤4中的排序最短用时为，对于有n个点的图而言，并查集的合并需要进行n-1次。并查集的单次合并的时间复杂度是，其中是Ackermann函数



的反函数，其中j+1表示此函数的迭代次数。

返回使得成立的最小值，它的值很小，不超过4。因此并查集合并的操作可视为常数时间复杂度。因此Kruskal算法的总体时间复杂度只受到排序部分的影响，为。

由于Kruskal算法的运行时间主要取决于对边按权值排序所需的时间，它更适合处理边数较少的稀疏图。

**3 最小生成树的实际应用**

**3.1 经典应用场景**

由无向连通图的最小生成树的定义可以得知，最小生成树算法适合求解城市规划中的道路、管道、通信网络铺设等路径选择问题。n个城市之间想要建立一个互相连通的“网络”，有很多种方法。把城市看作是顶点，城市之间的线路看作是边，把修建一条线路所需的成本当做边的权值，则沿着最小生成树中的边铺设线路就是用最小的成本连接所有城市的方案。类似的最低成本路径问题都适合用最小生成树算法求解。

在工业领域中，仪表回路图可以显示不同仪表之间的硬件连接关系。由于最小生成树一定是无环图，可以用最小生成树算法优化仪表回路图的显示。这样不仅可以用最小的代价得到所有仪表的信息以及其数据反馈的路径，还可以减少存储空间、优化控制软件的性能，使整个仪表回路图简洁明了，便于工程师查看。[2]

**3.2 创新的应用场景**

由于网络上产生的数据大多数都是无标签数据，在挖掘这些数据中的信息时需要对大量数据进行聚类。因此人们越来越重视数据集的聚类算法。然而，传统的密度聚类算法对密度分布不均匀的不平衡数据集的聚类效果不够理想。将数据集中的元素视为顶点，元素之间的相互可达距离视为边，通过Prim算法构建最小生成树，然后指定聚类的簇的个数，再进行剪枝。这种基于最小生成树的聚类算法可以用来改进传统聚类算法。[3]

基于最小生成树的新算法近年来得到了广泛的应用。其应用场景包括数据集聚类、图像分割、图像修复等。[4]

**4 总结与学习心得**

在离散数学课程的学习中，我们学习了图论。图论在生活中有广泛的应用。在这次研究性学习中，我从最小生成树的概念出发，了解了求解最小生成树的基本算法及其原理。同时还学习了从基本算法衍生出的两种常见算法——Prim算法和Kruskal算法的原理及代码实现方法；然后分析了两种算法的时间复杂度及其影响因素，并得出了Prim算法的时间复杂度与图的顶点数有关，适合处理稠密图；Kruskal算法的时间复杂度与边数以及对边按权值排序所用的算法有关，适合处理稀疏图的结论。通过这次研究性学习，我学会了如何评估一个算法的性能以及它的适用场景，并学习了从什么维度比较实现同一功能的不同算法。

最小生成树给了我们在实际情境中“降本增效”的新思路。它是对实际问题进行数学建模的一种实用工具。无论是小到规划旅行时的交通路径的问题，还是大到铺设城市之间的通信线路，都可以用最小生成树算法来解决。

人们提出最小生成树算法是为了解决城市规划、路径规划等经典问题。如今，最小生成树的“总权值最小”的性质使它有了新的应用场景，例如用于大量数据聚类算法的优化以及图像分割、图像修复等，甚至在生物医学工程、经济学等领域都有其应用。[5]最小生成树算法还能用于哪些领域的流程精简化、快速化是一个值得研究的课题。

**参考文献：**

[1] 《算法导论(原书第2版)》Introduction to Algorithms[J].计算机教育,2006,(11):80.

[2] 钟世平,闫婷,张立飞等.基于最小生成树算法构造有向无环图在工业控制的应用[J].石油化工自动化,2023,59(03):13-16+28.

Zhong Shiping, Yan Ting, Zhang Lifei, etc. Application of constructing directed acyclic graph in industrial control based on minimum spanning tree algorithm [J]. Petrochemical industry automation

[3] 蓝欢玉.基于最小生成树的不平衡数据集聚类算法[J].信息与电脑(理论版),2023,35(14):120-122.Lan Huanyu. Unbalanced data clustering algorithm based on the minimum spanning tree [J]. Information and Computer (Theory version)

[4] 王诚,高兴东.基于最小生成树的密度聚类算法研究[J].计算机技术与发展,2022,32(02):45-50.

Wang Cheng, Gao Xingdong. A density clustering algorithm based on the minimum spanning tree [J]. Computer technology and development

[5] 商晓锋,罗志增,史红斐.基于脑肌耦合导联选择和最小生成树网络的脑电特征提取[J].仪器仪表学报,2022,43(07):191-198.DOI:10.19650/j.cnki.cjsi.J2209462

Shang Xiaofeng, Luo Zhizeng, Shi Hongfei. EEG feature extraction based on brain muscle coupled lead selection and minimal generative tree network [J]. Journal of Instrumentation