第 22卷 第 3期 黑龙江大学自然科学学报 Vol. 22 No. 3
2005年 6月 JOURNAL OF NATURAL SCIENCE OF HEILONGJIANG UNIVERSITY June, 2005

流量约束最小生成树问题的分枝定界算法

杨亚碧[1], 黄亚玲[2]
(1. 贵州民族学院 物理与电子信息科学系,贵州 贵阳 550025; 2. 北京航空航天大学 计算机学院,北京 100083)

摘 要: 研究流量约束最小生成树问题 (CMST),它是通讯和网络优化设计中最为基础和重要的问题之一. 给出一种分枝定界算法,详细阐述了算法的原理、搜索过程,数值结果表明,该算法是有效的,并且有较好的计算性能.

关键词:最小生成树; 流量约束; 分枝定界
中图分类号: TU313 **文献标识码**: A **文章编号**: 1001 - 7011(2005)03 - 0353 - 06

1 问题与假定

流量约束最小生成树问题是指在一定的流量约束条件下,找到所有由已知节点集组成的展开树中成本最低的问题. 考虑一个由节点集 $V = \{0, 1, \cdots, n\}$和弧集 A 组成的连通图 $G = (V, A, b, c)$. V中每个节点 i有一个单位节点质量 $b_i = 1$和 $b_0 = 0$. 节点质量可理解为流量需求,而一个弧质量 c_{ij}表示使用 A中弧 (i, j)的成本. 节点 0具有其特殊性,被称为中心节点并将成为树根. 一个根子树 (或组件)定义为通过弧 $(0, i)$ (该弧可被视为中心弧)连接到中心的最大子树. 为了满足流量约束,每个弧上的流量不得超过已知的流量约束 K 借助这些定义, CMST问题便等同于寻找一个连通节点集 V的最低成本树,其中所有弧满足流量约束.

假设当 $i = 1, \cdots, n$时 $b_i = 1$;当 $i = 0$时 $b_i = 0$, 那么 CMST问题可以由一个混合整数线性规划公式来描述,如文献[1]所示. 如果解中包括 arc(i, j),定义 $x_{ij} = 1$;否则 $x_{ij} = 0$. 当 $i = 0, \cdots, n$,且 $j = 1, \cdots, n$时,设 y_{ij}表示弧 $arc(i, j)$上的流量, 下面的公式给出了一个以中心节点 0为根的受流量限制的最低成本树:

$$\text{Minimize} \sum_{i=0}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \tag{1}$$

$s.t.$

$$\sum_{i=0}^{n} x_{ij} = 1 \quad i = 1, \cdots, n \quad j = 1, \cdots, n \tag{2}$$

$$\sum_{i=0}^{n} y_{ij} - \sum_{i=1}^{n} y_{ji} = 1 \quad j = 1, \cdots, n \tag{3}$$

$$x_{ij} \leq y_{ij} \leq (k - b_i) \cdot x_{ij} \quad i = 1, \cdots, n \quad j = 1, \cdots, n \tag{4}$$

$$x_{ij} \in \{0, 1\} \quad y_{ij} \geq 0 \quad i = 1, \cdots, n \quad j = 1, \cdots, n \tag{5}$$

CMST在计算机网络、通信网络、交通运输网络、航空航天等领域还有着广泛的应用.

由文献[2]知,当 $2 < k < n/2$时,流量约束最小生成树是 $NP - hard$问题.

关于 CMST问题,已有很多启发式算法,目前已取得的成果包括构造方法[3-4]、储蓄算法[5-7]、二重方法[5]、分解还原算法[8]、定误差边界方法[9]、本地交换方法[5][10]、二次方法[7][11]和 节点交换方法[12]等.

本文提出一种解决受流量约束的最小生成树问题的确定性算法. 显然,枚举随节点数的增加而计算量指数增长,对于大规模问题是无法应用的,而启发式算法更具有可行性. 但是,开发精确的算法仍然是非常重要的,因为精确算法能够证明解的最优性,而启发式算法却做不到. 对于中、小规模的问题,精确算法依然是切

收稿日期: 2004 - 11 - 25
基金项目: 国家自然科学基金资助项目 (60473010)
作者简介: 杨亚碧 (1962 -),女 (苗族),讲师,主要研究方向:计算物理与高性能计算, E - mail: zttgzcn@yahoo.com.cn
黄亚玲 (1967 -),女,高级工程师,硕士

实可行的,尤其在网络工程的预设计中,它们的运行时间在实践中是往往可以被接受的. 此外,精确算法当被用来评估启发式算法的功效时就显得极为重要. 它们提供的最优解是检验处于设计阶段的启发式算法最具说服力的参考.

Chandy和 Rusell [13] 提出了一种分枝定界的算法,它是根据一种可行的子树的由中心算起已建立的或非第一段尚未建立的弧来对子问题进行分枝,然后根据附加约束条件修改成本权值. 运用这种算法可以考虑多至 40个节点的问题. Chandy和 Lu [14] 根据考查一个子树的两个节点是否超出将该子树将要发生的流量来进行分枝. 可以考虑多至 50个节点的问题. Kershenbaum 和 Boorstyn [15] 提出了一种分枝和界限的算法,它是根据一个特定的子树包括或者排除某个节点来对子问题进行分枝,如此可以给出 20个节点的问题的计算结果. Gouveia和 Paixao [8]提出了一种动态规划算法,该算法可应用于多至约一打节点的例题. Malik和 Yu [16]提出的是另一种运用拉伯朗日次陡度法分枝和定界的算法. 目前已有多至 50个节点的计算结果的报道.

2 分枝定界算法

分枝定界算法是一个通用的概念,而这种办法的具体的实施策略在不同情况下会呈现很大的变化. 即便是针对同一个问题,也往往存在着数种分枝和定界的策略. 因此,某一种算法是否更有效取决于它是否更恰当地对应问题的特殊性. 此外,人们经常希望获得更为好的界限以便更快地删节搜寻树,但是一个能提供更加好的界限的公式并非总是更好的选择,例如计算量过大等弊端.

2.1 搜索树

本文将有 n个点的 CMST问题的解视作一个从所有候选弧中选择出的 $n-1$个弧的集合,因此将引入一种针对弧的分枝定界方法,根据是否使用某一弧来进行分枝. 如图 1所示构造一种二进制搜索树.

有 m个候选弧的 CMST问题,定义一个 m级的二进制搜索树. 由于考虑完全连通图,对于 n个节点问题,m总是等于 $n(n-1)/2$ 其中每一级的分枝对应于是否在解级中增加某一弧 (ordered in ascendant cost)的选择. 举例来说,在阶段 3,要决定是否选择弧 3 这里同时也定义沿着左分枝搜索表示选择一段弧,而沿着右分枝搜索表示不选择一段弧. 例如在图 1中,在阶段 1和 2,从点 1推进到点 2之后到点 5表示选择弧 1和不选择弧 2 从点 1推进到点 3之后到点 6表示选择弧 2而不选择弧 1.

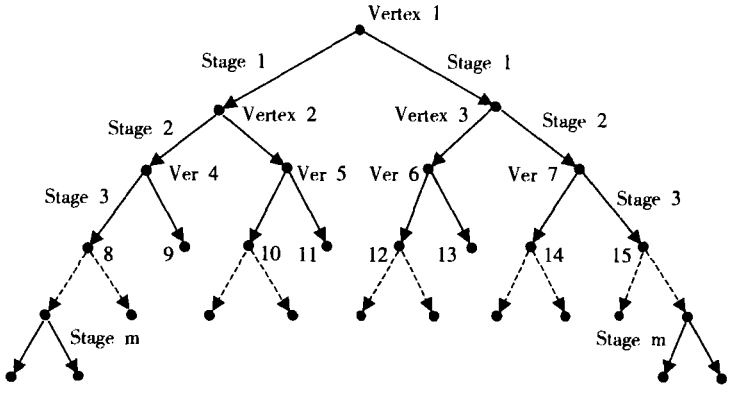

图 1 搜索树

Fig.1 Search tree

显而易见,最优解决可以在访问该二进制搜索树的所有结点后找到.

2.2 搜索规则

寻找最优解的过程事实上是沿着搜索树的所有分枝遍历的过程. 本文定出如下遍历规则:

1. 总是从树的高顶点开始 (图 1中的顶点 1);

2. 在遍历过程中,总是先沿左侧分枝推进;

3. 一直向前推进,并在如下所列情况发生时回溯:

*a)*当前选取的弧使得解不可行;

*b)*一个可行解已经被找到;

*c)*当前选取的所有弧的总成本已经超出了一个已知解的成本.

4. 在沿着其左侧分枝退回到某个点之后,再沿其右侧分枝继续推进. 例如,在图 1中,如果从点 4退回到了点 2,则接下来向点 5推进.

5. 在沿着其右侧分枝退回到某个点之后,继续退回到它的父结点. 例如,在图 1中如果从点 5退回到了点 2,则继续退回到点 1.

如果按照这些规则在搜索树上推进,最终将沿着其右侧分枝退回到起始点而无法再进一步推进 (在图 1中,从点 3到点 1),那么便已完成了对所有可行解的考查并确保找到了最优解.

2.3 搜索树的剪枝

除了在 2.2节中所述的规则外,现在引进搜索树的其它剪枝方法,从而减少搜索步骤. 首先,如果想要一棵预先假定某些弧必须使用 (假定这些弧不构成圆)且某些弧一定不使用的最小生成树,依然可以应用 Kruskal算法. 只需首先选择那些必须使用的弧并从整个候选弧集合中删去那些必须被排除的弧,然后再按正常的 Kruskal算法对余下的部分进行处理 (证明略). 这里将这种情况下的最小生成树命名为 PEMST(Partially Established Minimum Spanning Tree).

由于在搜索树中的某一点,代表着某些弧被选取而某些弧被排除,因而可以将流量约束松弛,并使用 PEMST作为一个界. 在每一阶段,除了进行了可行性检验之外,还构造当前的 PEMST并计算其总成本. 如果不可取 (成本高于现有解决办法),则回溯. 这样,便对搜索树上植根于当前点的子树进行了删剪.

接下来进行有效的可行性检查,这也是快速删减搜索树的重要手段之一. 对于流量约束检查,这里不是直接计算这些弧上的流量,而是保留一些节点集,每个集合包含由选取弧连接起来的节点. 这样一来,在每个节点集中的节点总数就反映了将发生在每个中心弧上的流量量. 这将预告某个子树是否会超负荷,即便它还未被连接到中心节点上.

2.4 算法

本节介绍算法的主程序. 所使用的程序语言是 Pascal

2.4.1 数据结构和功能

作者认为对于一个面向弧的分枝和定界搜索,以下是最佳的数据结构.

设 n为节点数, $m = n(n-1)/2$位候选弧的数量. 首先,将弧按照成本进行升序排列,并为每一段弧编号,从 1到 m, 作为一种标识. 这些候选弧被存入一个数组中 $[1..m]$, 第一个元素 $[1]$存放编号为"1"的最低成本弧,第二元素存放编号为"2"的次最低成本弧的记录,依此类推.

下一步,定义"SolutionBuffer",它存放着以编号识别的被选择建立一潜在可行解的若干弧,作为一个数组 $[1..n-1]$. 例如,如果在某一步骤中, SolutionBuffer等于 [1, 3, 7, 0, …, 0], 那么这就意味这弧 1,弧 3和弧 7已被选中. "Solution"存放所有已找到的可行解中最好的一个. 整数"Flagsolve"用来存放在当前搜索步骤下所有被添加到"SolutionBuffer"中去的弧的总数,例如,如果解决办法缓冲区为 [1, 3, 7, 0, …, 0],那么 Flagsolve等于 3.

下面是一些函数的说明.

CostCheck(i):布尔值;在添加弧 i到 SolutionBuffer之后,检查当前选取的连接的总成本是否低于已找到的最优解的成本. 如果是,函数反回真值.

TreeCheck(i):布尔值;检查在选择弧 i之后在树解中是否形成了环. 如果选择连接 i不构成一个环,函数反回真值.

CapacityCheck(i):布尔值;检查在选择弧 i后是否满足流量约束. 如果是,函数反回真值.

PEMSTCheck(i):布尔值;检查 PEMST是否比已经找到的解的目标函数值小. 如果是,函数反回真值.

IniSolve:整数;反还由启发式算法找到的初始可行解的目标函数值.

2.4.2 主搜索过程

在搜索树上的主遍历程序见本文后部附录所示. 整数"i"表示当前正在处理中的弧的编号. "TotalCost"指当前已经找到的可行解中最优的目标函数值. 某些特殊事件处理和变量分配在此省略.

3 测试结果

3.1 随机产生的实例

本研究使用了大量随机产生的实例测试提出的分枝定界算法,其中一组 50节点问题 (p1 - p10)的计算结果 (时间单位为秒)见表 1. 表 2是一组 100个节点问题的计算结果. 计算是在安装了奔腾 II 400MHz CPU和 128MB RAM的计算机上完成的,操作系统是 Windows 98. 测试结果显示本文提出的分枝定界算法可以解决多至 100节点的 CMST问题. 迄今为止对使用其它精确算法解决多至 50节点的随即产生的 CMST问题的计算经验尚未有过报道.

3.2 同其它算法的比较

表 1 一组 50节点问题的计算结果

Table 1 Test result of one set of 50 - node problems

	P1	P2	P3	P4	P5	P6	P7	P8	P9	
$k=20$	0.00	0.9	0.00	0.22	0.5	0.03	0.06	0.8	0.1	0.4
$k=15$	0.02	3.4	0.5	0.03	84	9.4	0.08	30	0.1	13
$k=10$	0.2	1294	329	6	3943	351	2	178	532	1060
$k=5$	33331		59527	82332			38133			

表 2 一组 100节点问题的计算结果

Table 2 Test result of one set of 100 - node problems

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12
k = 30	156	27989		6		4367			28077			

表 3 同 Hall和 Gouviea的成果比较

Table 3 Comparison with Hall and Gouviea's work

Problem	Root	B&B CPU	Hall	Gap	Hall	CPU	Gouv	Gap	Gouv	CPU
Tc41 - 1. txt	Center	379	min	0.000	min	1.21	min	0.000	min	177
Tc41 - 2. txt	Center	3617								
Tc41 - 3. txt	Center	12173	avg	0.079	avg	2.60	avg	1.603	avg	1620
Tc41 - 4. txt	Center	56330								
Tc41 - 5. txt	Center	92199	max	0.388	max	6.78	max	2.755	max	3877
Tc41 - 1. txt	Comer	unsolved	min	0.698	min	8.59	min	0.712	min	1332
Tc41 - 2. txt	Comer	unsolved								
Tc41 - 3. txt	Comer	unsolved	avg	2.002	avg	15.24	avg	2.092	avg	3473
Tc41 - 4. txt	Comer	unsolved								
Tc41 - 5. txt	Comer	unsolved	max	3.691	max	28.32	max	4.078	max	9360

必须指出的是,由不同的研究人员使用的不同实验方法,并非在所有情况中都具有可比性. 对于流量约束下最小生成树问题,具有更大流量约束值 K的实例将更易于解决. 此外,如果在二维欧几里德平面设置所有节点,并使用两个节点间的欧几里德距离作为它们之间的弧的成本,当根结点的位置在欧几里德平面上处于中央或角落时,称其"在中央"或者"在角落". 另一个影响问题难度的重要参数是根结点的位置.

由英国皇家学院运筹学研究中心建立的 OR - Library (http://mscmga.ms.ic.ac.uk/info.html)发布的CMST问题的标准例题集是最权威的 benchmark problem集合, 世界上研究 CMST问题的学者大都以该标准例题集为测试对象. 被报道已经找到保留在 OR - Library里的一些问题实例的最优解的算法包括 Hall [17] and Gouviea [18].

表 3是当 n = 41且 k = 10时,本实验结果同 Hall和 Gouviea's的工作的一个比较. Hall的测试是在一个 Silicon Graphics Indy R4000 SC的工作站上运行的. 根据 Hall和 Gouviea's的结果,"Gap"指 (UB - LB) /LB,这里 UB和 LB指分别产生的上界和下界. "CPU"指 CPU秒数.

从表 3中可以看出本文提出的分枝定界算法找到了更多的中心 41节点问题的最优解,但是速度慢很多. 所有算法都无法找到角落 41节点的最优解. 但事实上, Hall和 Gouviea的算法并非一种精确算法,而是由其他启发式方法发现可行解,并努力寻找更好的下界;当下界和启发式方法发现的可行解之间不存在差距时,可以证明一个解的最优性. 本文提出的分枝定界算法是一种真正的精确算法,能够保证找到最优解,因此,它运行较慢,且对问题的"难度"更为敏感.

在本文之前,既没有能够解 OR - Library中 CMST问题的精确算法,也没有能够解多于 50个节点的随机

产生的 CMST问题的精确算法的报道.所有的测试问题和用本文提出的算法找到的最优解可以参见下列网址: http://www.freewebs.com/xyzdown/

4 结　论

本文提出了一个针对 CMST问题的分枝定界算法,并且进行了同其它算法的比较.该算法可以解决多至100个节点的随机生成的 CMST问题的精确算法,并计算出 OR-Library中部分 CMST标准例题的最优解.计算实验还支持了人对 CMST问题的难度对于根的位置和受限流量的量值极为敏感的推测.

附录:

主编历程序.

```
Procedure Searching;
    Var i, TotalCost: Integer;
    begin
    TotalCost:=IniSolve; FlagSolve:=0; i:=1;
    while (flagsolve>0) or (i<m) do
      begin
        if CostCheck(i) then
          begin
            if TreeCheck(i) then
              begin
                if PEMSTCheck(i) then
                  begin
                    if CapacityCheck(i) then
                      begin
                        Flagsolve:=Flagsolve+1;
                        SolutionBuff[FlagSolve]:=i;
                        if Flagsolve=NN-1 then
                          begin
                            TotalCost:=TempCost;
                            Solution:=SolutionBuff;
                            i:=SolutionBuff[NN-2];
                            i:=i+1;
                            FlagSolve:=NN-3;
                            SolutionBuff[NN-2]:=0;
                            SolutionBuff[NN-1]:=0;
                          end
                        else
                          begin
                            i:=i+1;
                          end;
                      end
                    else
                      begin
                        i:=i+1;
                      end;
                  end
                else
                  begin
                    if Flagsolve=0 then break;
                    i:=SolutionBuff[FlagSolve];
                    SolutionBuff[FlagSolve]:=0;
                    i:=i+1;
                    FlagSolve:=FlagSolve-1;
                  end;
              end
            else
              begin
                i:=i+1;
              end;
          end
        else
          begin
            if Flagsolve=0 then break;
            i:=SolutionBuff[FlagSolve];
            SolutionBuff[FlagSolve]:=0;
            i:=i+1;
            FlagSolve:=FlagSolve-1;
          end;
      end;
    end;
```

参考文献:

[1] AMBERG A, W DOMSCHEKE, BRAUNSCHWEIG S. Capacitated Minimum Spanning Trees: Algorithms using intelligent search[J]. Combinatorial optimization: Theory and Practice, 1996, 1: 9-39.

[2] PAPADIMITRIOU C H. The Complexity of the capacitated tree problem[J]. Networks, 1978, 8: 217-230.

[3] SCHNEIDER G M, ZASTROW M N. An algorithm for the design of multilevel concentrator networks[J]. Computer Networks, 1982, 6: 1-11.

[4] SHARMA R L, EL-BARDAIM T. Suboptimal communications network synthesis[A]. Proc 1970 Int Conf Comm[C]. 1970. 19: 11-16.

[5] ELIAS D, FERGUSON M J. Topological design of multipoint teleprocessing networks[A]. IEEE Transactions on Communications[C]. 1974. 22: 1753-1762.

[6] GAVISH B. Topological design of telecommunication networks - local access design methods[J]. Annals of Operations Research, 1991, 33: 17 - 71.

[7] GAVISH B, KALTINKEMER. Parallel savings heuristics for the topological design of local access tree networks[A]. Proc IEEE INFOCOM 86 Conference[C]. 1986. 130 - 139.

[8] GOUVEIA L, PAIXAO J. Dynamic programming based heuristics for the topological design of local access networks[J]. Annals of Operations Research, 1991, 33: 305 - 327.

[9] ALTINKEMER K, GAVISH B. Heuristics with constant error guarantees for the design of tree networks[J]. Management Science, 1988, 32: 331 - 341.

[10] FRANK H, FRISCH I T, VAN SLYKE R, CHOU W S. Optimal design of centralized computer design network[J]. Networks, 1971, 1: 43 - 57.

[11] KARNAUGH, M. 1976. "A new class of algorithms for multipoint network optimization". IEEE Transactions on Communications, 24: 500 - 505.

[12] THANGIAH S R, OSMAN I H, SUN T. Hybrid genetic algorithms, simulated annealing and tabu search methods for vehicle routing problems with time windows[R]. Working Paper, Univ of Kent, Canterbury. 1994.

[13] CHANDY K M, RUSSELL R A. The design of multipoint linkages in a teleprocessing tree network[J]. IEEE Transactions on Computers, 1972, 21: 1062 - 1066.

[14] CHANDY K M, LO T. The capacitated minimum spanning tree[J]. Networks, 1973, 3: 173 - 181.

[15] KERSHENBAUM A, BOORSTYN P R. Centralized teleprocessing network design[J]. Networks, 1983, 13: 279 - 293.

[16] MALIK K, YU G. A branch and bound algorithm for the capacitated minimum spanning tree problem[J]. Networks, 199323: 525 - 532.

[17] HALL L. Experience with a cutting plane algorithm for the capacitated spanning tree problem[J]. INFORMS Journal on Computing, 1996, 8 (3): 219 - 234.

[18] GOUVEIA L, MARTINS P. An extended flow based formulation for the capacitated minimum spanning tree[R]. Presented at the third ORSA Telecommunications Conference, Boca Raton, FL. 1995.

[19] AHUJA R K, ORLIN J B, D SHARMA. Very large - scale neighborhood Search[J]. Operational Research, 2000, 7: 30.

A branch and bound algorithm for the CMST problem

YANG Ya - bi, HUANG Ya - ling

(1. Department of Physics and Information Technology, Guizhou University for Ethnic Minorities, Guiyang 550025, China; 2. School of Computer Science, Beihang University, Beijing 100083, China)

Abstract: The capacitated minimum spanning tree problem (CMST), one of the most fundamental problems in telecommunications and in the optimal design of networks, is studied. A branch and bound algorithm is proposed and the principle and search process of the algorithm is introduced. Test results show that the algorithm is effective and has good computation performance.

Key words: MST; Capacity constraint; Branch and Bound