几类特殊图中的最小最大多路割

李曙光[1,2] 辛 晓[3]
(山东工商学院计算机科学与技术学院 烟台 264005)[1] (山东大学计算机科学与技术学院 济南 250101)[2]
(山东工商学院外国语学院 烟台 264005)[3]

摘 要 给定边具有正权的无向图,并指定若干个称为终端的顶点,最小最大多路割问题是要得到所有顶点的一个聚类,要求每个子类恰好包含一个终端,并使得所有子类的最大费用最小。子类的费用定义为该子类边界上所有边的权之和。最小最大多路割问题源于对等网络中的数据放置,是传统多路割问题的一个变形。当给定无向图是树图时,这一问题已经是强NP难解的。对于链图和环图,给出了线性时间的精确算法,该算法同时也使得所有子类的总费用最小。对于树图和限制树宽图,给出了$(2-\frac{1}{2k^2})$-近似算法,k表示终端的数目。

关键词 最小最大多路割,链图,环图,树图,限制树宽图
中图法分类号 TP301 **文献标识码** A

Min-Max Multiway Cuts in Several Special Graphs

LI Shu-guang[1,2] XIN Xiao[3]
(College of Computer Science and Technology, Shandong Institute of Business and Technology, Yantai 264005, China)[1]
(School of Computer Science and Technology, Shandong University, Jinan 250101, China)[2]
(College of Foreign Studies, Shandong Institute of Business and Technology, Yantai 264005, China)[3]

Abstract Given an undirected graph with positive edge weights and a subset of verticescalled terminals, the min-max multiway cut problem is to partition the vertices into clusters such that each cluster contains exactly one terminal and the maximum cost among the clusters is minimized, where the cost of a cluster is defined as the total edge weight at the boundary of the cluster. This problem is motivated by data plcacement in Peer-to-Peer networks and is a variant of the traditional multiway cut problem. It is strongly NP-hard even when the graph is a tree. For chains and rings, linear time exact algorithms were presented which minimize simultaneously both maximum cost and total cost of the clusters. For trees and graphs of bounded treewidth, a $(2-\frac{1}{2k^2})$-approximation algorithm was presented where k is the number of terminals.

Keywords Min-max multiway cuts, Chains, Rings, Trees, Graphs of bounded treewidth

1 引言

Svitkina和Tardos[1]研究了如下的最小最大多路割问题。给定有n个顶点的无向图$G=(V,E)$,G中边e的权为$w_e>0$。指定终端的集合为$X=\{x_1,x_2,\cdots,x_k\}\subseteq V$。$G$中的多路割定义为顶点集合$V$的一个聚类$S_1,S_2,\cdots,S_k$,使得$S_1,S_2,\cdots,S_k$互不相交,$\cup_i S_i=V$,并且$x_i\in S_i$,$i\in\{1,2,\cdots,k\}$,称$S_i$是这一聚类的第$i$个子类。对子类$S_i$,令$\delta(S_i)$表示$S_i$边界上的边(即恰好有一个端点在$S_i$中的边)的集合。记$w(\delta(S_i))=\sum_{e\in\delta(S_i)}w_e$。子类$S_i$边界上所有边的权之和称为$S_i$的费用,记为$C(S_i)$,即$C(S_i)=w(\delta(S_i))$。最小最大多路割问题的目标是要得到$V(G)$的一个聚类,使得所有子类的最大费用$\max_i C(S_i)$最小。

对等系统(Peer-to-Peer system)中的信息资源(数据)放置问题可以抽象为最小最大多路割问题。在对等系统中,信息资源存储在很多服务器上。当用户提交查询请求时,请求将被定向到最适宜的服务器上。对某数据项的查询请求可能会产生对其他数据项的查询请求。在同一服务器上实现的查询请求并不产生通讯费用,只有当查询请求从一个服务器转到另一服务器时才会产生通讯费用。因此,如何将数据合理放置使得服务器负载均衡,是对等系统中的一个重要问题。最小最大多路割问题中,V中的非终端顶点代表数据项,k个终端$x_1,x_2,\cdots,x_k$代表k个服务器。若图G中有边(x_i,v),$v\notin X$,则该边的权代表服务器i上的用户查询数据项v的次数。若图G中有边(u,v),$u,v\notin X$,则该边的权代表对数据项u的查询会产生对数据项v的查询的次数。顶点集合V的一

到稿日期:2010-08-12 返修日期:2010-11-29 本文受国家自然科学基金(60970105),山东省信息产业发展专项资金项目(2008X00039),山东省软科学研究计划(2010RKE20029)资助。
李曙光(1970—),男,博士,副教授,CCF会员,主要研究方向为图论、组合最优化等,E-mail:sgliytu@hotmail.com;辛 晓(1970—),女,主要研究方向为计算机网络。

个聚类对应一种数据放置方案，其中，子类 S_i 中的顶点对应存储在服务器 i 上的数据，$i\in\{1,2,\cdots,k\}$。目标是找到最优聚类，从而确定如何在服务器上放置数据，使得所有服务器实际产生的最大通讯费用最低。

最小最大多路割问题类似于传统的多路割问题。二者的区别在于目标函数不同。多路割问题是要得到 $V(G)$ 的一个聚类 $S_1,S_2,\cdots,S_k$，使得每个子类恰好包含一个终端，并且所有子类的费用之和 $\sum_i C(S_i)$ 最小。多路割问题在无向图和有向图中都是 APX 难解的[2,3]。无向图中的多路割问题有 $(1.5-\frac{1}{k})$-近似算法[4]和 $(1.3438-\epsilon_k)$-近似算法[5]；有向图中的多路割问题有 2-近似算法[6]。当 $k=2$ 时，多路割问题等价于最小最大多路割问题，也就是著名的最大流最小割问题。该问题在组合最优化领域中占据重要位置，因为它有相当多的应用，并且可在多项式时间之内求解[7,8]。与多路割问题有关的国内研究成果见文献[9-11]，目标函数均为最小化所有子类的费用之和。

传统多路割问题所得解中，通讯费用可能不是在各子类间平均分布，有可能出现某几个子类负载过大或过小的问题。显然，最小化所有子类的最大费用这个目标函数有助于实现各子类间的负载均衡。因此，Svitkina 和 Tardos 提出了最小最大多路割问题[1]。他们给出了求解任意图中最小最大多路割问题的 $O(\log^3 n)$-近似算法，并证明了当 k 为不小于 4 的常数且所有边都具有单位权时这一问题的 NP 难解性。还证明了树图上这一问题的强 NP 难解性，并给出求解树图和限制树宽图的 $(2+\varepsilon)$-近似算法，ε 是任意小的正数。该算法的运行时间随 $\frac{1}{\varepsilon}$ 的增长而增长，但由于 ε 被视为常数，因此运行时间始终是多项式的。

本文研究特殊图中的最小最大多路割问题，以期得到更好的算法。第 2 节介绍相关术语。第 3 节研究链图和环图，给出线性时间精确算法，该算法在最小化最大子类费用的同时，也使得所有子类的总费用最小。第 4 节研究树图和限制树宽图，给出 $(2-\frac{1}{2k^2})$-近似算法，算法的运行时间不依赖于 $\frac{1}{\varepsilon}$。最后指出可供进一步研究的工作。

2 预备知识

给定一个最小化问题，若对于该问题的任意实例，算法都能在多项式时间之内找到一个目标值不大于 ρ 倍最优值的近似解，则称该算法是该最小化问题的一个 ρ-近似算法，ρ 称为该算法的近似比[8]。

若无向图 $G=(V,E)$ 由一列连续的边构成，则此图称为链图。将链图的两个端点粘合在一起，就得到了环图（又称为圈）。若对任意一对顶点 $u,v\in V$，G 中有一条从 u 到 v 的路，则 G 是连通的。若 G 是连通的并且不含圈，则称为树。链、环、树是基本的图结构，很多其他的图都是这些基本图的组合或者派生。

无向图 $G=(V,E)$ 的树分解是指一棵树 T，T 的每个节点 t 对应一个子集 $V_t\subseteq V$。为表述清楚，称树 T 的顶点为节点，称这些子集 V_t 为树 T 的袋。可以将 $G=(V,E)$ 的树分解表示为一个有序对 $(T,\{V_t:t\in T\})$。树 T 和它的袋集合 $\{V_t:t\in T\}$ 必须满足如下 3 个条件：

- 图 G 的每个顶点至少属于某个袋 V_t；
- 对于图 G 的每条边 e，至少有某个袋 V_t 包含边 e 的两个端点；
- 设 t_1,t_2 和 t_3 是树 T 的 3 个节点，t_2 在从 t_1 到 t_3 的路上。若图 G 的某个顶点 v 同时属于 V_{t_1} 和 V_{t_3}，那么它也必属于 V_{t_2}。

分解树 T 的宽度定义为 $\max\{|V_t|:t\in T\}-1$。图 G 的树宽就是图 G 的任何树分解中最小的宽度。若图 G 的树宽限制为某个常数，则称图 G 为限制树宽图。关于树分解和限制树宽图的进一步介绍参阅文献[12,13]。

设图 $G=(V,E)$，非空子集 $V_1\subseteq V$，则称以 V_1 为顶点集、以 G 中两个端点都在 V_1 中的边组成边集 E_1 的图 (V_1,E_1) 为 G 的由 V_1 导出的子图，记作 $G[S]$。若 $G[S]$ 是一个连通子图，则称 S 构成一个连通段，或直接称 S 是一个连通段。

3 链图和环图中的最小最大多路割

本节将研究链图和环图中的最小最大多路割问题，并给出线性时间精确算法。

设 $L=(V,E)$ 是链图，顶点集 $V=\{1,2,\cdots,n\}$，边集 $E=\{e_v=(v,v+1)\mid v=1,2,\cdots,n-1\}$。设 $R=(V,E)$ 是一个环图，其中，顶点集 $V=\{0,1,\cdots,n-1\}$，边集 $E=\{e_v=(v,v+1)\mid v=0,1,\cdots,n-1\}$（取模 n 加法）。

引理 1 链图中的最小最大多路割问题存在一个最优聚类，该聚类中的每个子类都构成一个连通段。

证明：考虑一个最优聚类。若不是每个子类都构成一个连通段，则该聚类中必含有超过 k 个的连通段。取一个不含终端的连通段，设链图上与该连通段相邻的两条边中，权较大的那条边的不在该连通段中的那个端点为 v。将这个不含终端的连通段改放入点 v 所在的子类。显然，各子类费用均不增加，每个子类恰好含一个终端，故所得仍然为一个最优聚类，且连通段的数目减 1。重复这一过程，最后得到的最优聚类恰有 k 个连通段，每个连通段恰含有一个终端，故是满足引理要求的最优聚类。

这个引理保证了我们可以在线性时间内精确求解链图中的最小最大多路割问题。注意到树图中的最小最大多路割问题并不具备引理 1 所描述的性质，这正是它成为强 NP 难解问题的原因。

链图最小最大算法如下。

第一步　删除链图 L 上每对相邻终端之间那一段上的最小权边，得到 k 个连通段；

第二步　将每个连通段作为一个子类，得到 $V(L)$ 的一个聚类；

第三步　输出所得到的聚类。

我们得到如下定理。

定理 1 链图最小最大算法是求解链图中的最小最大多路割问题的线性时间精确算法，该算法同时也使得所有子类的总费用最小。

证明：在任一可行聚类中，每个子类只能包含一个终端，因此链图 L 上每对相邻终端之间那一段上至少要删除一条边。根据引理 1，L 上每对相邻终端之间那一段上只需删除一条边。算法第一步选择删除最小权边，不论目标函数是最

小化最大子类费用还是最小化所有子类费用之和，都是最优的，因此该算法同时使得这两个目标函数最优。

类似地，我们得到关于环图的如下结果。

引理 2 环图中的最小最大多路割问题存在一个最优聚类，该聚类中的每个子类都构成一个连通段。

环图最小最大算法如下。

第一步 删除环图 R 上每对相邻终端之间那一段上的最小权边，得到 k 个连通段；

第二步 将每个连通段作为一个子类，得到 $V(R)$ 的一个聚类；

第三步 输出所得到的聚类。

我们得到如下定理。

定理 2 环图最小最大算法是求解环图中的最小最大多路割问题的线性时间精确算法，该算法同时也使得所有子类的总费用最小。

4 树图和限制树宽图中的最小最大多路割

文献[1]研究了树图和限制树宽图中的最小最大多路割问题，给出了$(2+\varepsilon)$-近似算法。算法分为两阶段：在第一阶段，通过求解一个辅助问题(即稍后定义的切割问题)，将给定图切割为若干个连通分支，每个连通分支至多包含一个终端；在第二阶段，将这些连通分支合并为 k 组，每组恰好含有一个终端。每组的顶点对应一个子类，这样 k 组就对应给定图顶点集合的一个聚类，其中每个子类的费用均不超过最优值的 $2+\varepsilon$ 倍。

这一节将对算法的第二阶段加以改进，以得到$(2-\frac{1}{2k^2})$-近似算法，且算法的运行时间不依赖于$\frac{1}{\varepsilon}$。

为不失一般性，设给定图中的最小最大多路割问题存在一个聚类，其中每个子类的费用不超过一个已知整数 B。令 $W=\sum_{e\in E(G)}w_e$，则 W 显然是 B 的一个上界，因此可以用二分搜索的方法来优化 B，需要尝试 $\log W$ 次。

定义 1[1](切割问题) 给定无向图 $G=(V,E)$，终端集合 $X=\{x_1,x_2,\cdots,x_k\}\subseteq V$，以及一个整数界 B。切割问题是要将 G 分为若干个连通分支 $G_1,G_2,\cdots,G_h$，使得每个连通分支至多包含一个终端，且费用 $C(V(G_i))\leqslant B,i=1,2,\cdots,h$。目标是使得所有连通分支的总费用 $\sum_i C(V(G_i))$ 最小。

引理 3[1] 存在求解树图切割问题的拟多项式时间精确算法，其运行时间为 $O(nB^2)$。利用取整技巧经过简单修正之后，该算法可在 $O(\frac{n^3}{\varepsilon^2})$ 时间之内得到$(1+\varepsilon)$-近似解，其中每个子类的费用不超过$(1+\varepsilon)B$，ε 是任意小的正数。

引理 4[1] 存在求解限制树宽图切割问题的拟多项式时间精确算法。利用取整技巧经过简单修正之后，该算法可在多项式时间之内得到$(1+\varepsilon)$-近似解，其中每个子类的费用不超过$(1+\varepsilon)B$，ε 是任意小的正数。

现在给出求解树图和限制树宽图中的最小最大多路割问题的$(2-\frac{1}{2k^2})$-近似算法。算法第三步应用了著名的 LPT(最大加工时间优先)规则[14]。

最小最大多路割算法如下。

第一步 置 $\varepsilon=\frac{1}{2k}$，调用文献[1]中的算法，得到给定图(树图或限制树宽图)切割问题的$(1+\frac{1}{2k})$-近似解，其中每个子类的费用不超过$(1+\frac{1}{2k})B$；

第二步 将每个含有终端的子类作为一组，得 k 个组；

第三步 将所有不含终端的子类按照费用从大到小的顺序依次分配给 k 个组：将当前未分配的费用最大的子类分配给当前费用最小的那个组。

我们有如下定理。

定理 3 最小最大多路割算法是求解树图和限制树宽图中的最小最大多路割问题的$(2-\frac{1}{2k^2})$-近似算法。

证明：设给定切割问题的最优值为 $C_{切}^{*}$，则有 $C_{切}^{*}\leqslant kB$。设算法第一步所得所有连通分支的总费用为 $C_{切}$，其中最大连通分支的费用为 $M_{切}$。由引理 3 和引理 4，有 $C_{切}\leqslant(1+\frac{1}{2k})\cdot C_{切}^{*}\leqslant(1+\frac{1}{2k})kB$，$M_{切}\leqslant(1+\frac{1}{2k})B$。

设算法第三步所得最终聚类为 $S_1,S_2,\cdots,S_k$。为不失一般性，不妨设 $S_1,S_2,\cdots,S_k$ 是按照费用从大到小的顺序排列，即 $C(S_1)\geqslant C(S_2)\geqslant\cdots\geqslant C(S_k)$。我们要证明 $C(S_1)\leqslant(2-\frac{1}{2k^2})B$。

设算法第一步所得所有连通分支中，F_l 最后一个加入子类 S_1，其费用为 M_l。根据算法第三步的分配规则，当 F_l 分配给 S_1 之前，每个组的费用均不小于 $C(S_1)-M_l$，因此，有 $C_{切}\geqslant(C(S_1)-M_l)\cdot k+M_l$。故得到：

$$\begin{aligned}C(S_1)&\leqslant\frac{C_{切}}{k}+(1-\frac{1}{k})M_l\\&\leqslant(1+\frac{1}{2k})B+(1-\frac{1}{k})(1+\frac{1}{2k})B\\&=(2-\frac{1}{2k^2})B\end{aligned}$$

对于树图来说，最小最大多路割算法的运行时间是 $O(k^2n^3\log W)$。对于限制树宽图，最小最大多路割算法的运行时间也是多项式的。

结束语 本文研究了链图、环图、树图和限制树宽图中的最小最大多路割问题，给出了相应算法。可以继续研究其他图上的这一问题，看是否存在多项式时间精确算法，或找到好的近似算法。

参 考 文 献

[1] Svitkina Z, Tardos E. Min-max multiway cut [C]// Lecture Notes in Computer Science 3122: Proceedings of the 7th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems. Heidelberg: Springer-Verlag, 2004: 207-218

[2] Dahlhaus E, Johnson D S, Papadimitriou C H, et al. The complexity of multiterminal cuts[J]. SIAM Journal on Computing, 1994, 23: 864-894

[3] Garg N, Vazirani V V, Yannakakis M. Multiway cuts in directed and node weighted graphs [C] // Proceedings of ICALP'94. 1994: 487-498

[4] Calinescu G, Karloff H, Rabani Y. An improved approximation algorithm for MULTIWAY CUT[J]. Journal of Computer and System Sciences, 2000, 60(3): 564-574

[5] Karger D R, Klein P N, Stein C, et al. Rounding algorithms for a

geometric embedding of minimum multiway cut[J]. Mathematics of Operations Research,2004,29(3):436-461

[6] Naor J, Zosin L. A 2-approximation algorithm for the directed multiway cut problem[C]// Proceedings of the 38th Annual Symposium on Foundations of Computer Science. New York: IEEE Computer Society,1997:548-553

[7] Ford L R, Fulkerson D R. Flows in networks[M]. New Jersey: Princeton University Press,1962

[8] Papadimitriou C H, Steiglitz K. Combinatorial optimization: algorithms and complexity[M]. New Jersey: Prentice-Hall,1982

[9] 张鹏. 树上推广的 Multicut 问题的近似算法[J]. 计算机研究与发展,2008,45(7):1195-1202

[10] 李曙光,辛晓. 参数为 k 的几乎树中的染色多路割[J]. 计算机科学,2010,37(2):246-249

[11] 孙统风,丁世飞,任子晖. 一种基于 Multiway Cut 的多对象图像分割[J]. 计算机应用研究,2010,27(8):3138-3141

[12] Bodlaender H L, Koster A M C A. Combinatorial optimization on graphs of bounded treewidth[J]. The Computer Journal, 2008,51(3):255-269

[13] Kleinberg J, Tardos E. Algorithm design[M]. Boston: Addison-Wesley,2005

[14] Graham R L. Bounds on multiprocessing timing anomalies[J]. SIAM Journal on Applied Mathematics,1969,17(2):416-429

[15] 李强,闫洗文,梅耀元. 基于 VB 的最小生成树 KRUSKAL 算法的实现[J]. 重庆理工大学学报:自然科学版,2010,24(4):101-104

(上接第 202 页)

由表 1 可见,在保持约束强度不变的情况下,随着问题规模的增大,传统的桶消元算法产生溢出错时,本文的算法仍能在较短的时间内找到问题的所有解。这主要是因为 OBDD 的高紧凑性的特点,加上 OBDD 的操作是以集合方式进行,而不像传统算法那样逐个的进行,为此本文算法在时间和空间上都优于传统的桶消元算法。另外,相比符号直接求解法而言,本文的算法在执行时间上也明显占有优势。这主要是因为本文算法在求解 CSP 的所有解之前,对各 *bucket*[*i*]中约束作了更强的约束,即剔除了一些不可能的变量赋值。为此,在计算中生成的中间积的 OBDD 可能要比符号直接求解法的小。而基于 OBDD 的各种操作,其计算时间主要取决于参与操作的 OBDD 的大小,为此本文的算法要优于符号直接求解算法。

在第二组实验中,选取参数为⟨15,5,37,*t*⟩的测试用例,其中约束强度 $t=0.12+0.04z, z=0,1,\cdots,10$。针对 t 的不同取值,3 种算法的求解时间对比曲线图如图 5 所示,其中图 5 中的纵坐标采用对数坐标。

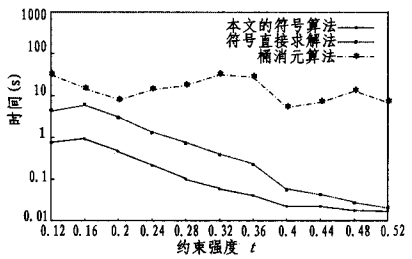

图 5 3 种算法在测试用例⟨15,5,37,t⟩上的实验对比结果

由图 5 可见,在保持问题规模不变的情况下,随着约束强度的增大,两种符号算法都优于传统的桶消元算法。这主要是因为在问题规模不变的情况下,符号算法中的布尔变量数将保持不变,并且随着约束强度的增大,每条约束中不存在冲突的值对逐渐减少,而在表示每条约束的 OBDD 中从根结点到终结点 1 的每一条路径对应的正是不冲突的值对。为此随着 t 的增大,表示每条约束的 OBDD 的大小将逐渐减小,而符号算法的执行时间主要取决于参与操作的 OBDD 的大小,故随着 t 的增大,符号的执行时间逐渐减少。此外,由图 5 可知,本文的符号算法要优于符号直接求解法。其原因同样是因为本文的算法在求解 CSP 的所有解之前,对各 *bucket*[*i*]中约束作了更强的约束,即剔除了一些不可能的变量赋值。

结束语 约束满足问题是实际应用中常见的一类问题。通过对 CSP 问题的符号化描述,将 CSP 问题的求解转化为对 OBDD 图的操作。为了避免在求解过程中生成过大的 OBDD 图,结合了桶消元算法的思想,给出了 CSP 的符号 OBDD 求解算法。以随机 CSP 模型随机生成的大量 CSP 作为测试实例,将本文的算法与桶消元算法和符号直接求解法进行了实验对比,结果表明,本文算法具有较好的时间和空间复杂度。

参 考 文 献

[1] Sadeh N, Sycara K, Xiong Y. Bachtracking techniques for the job shop scheduling constraint satisfaction problem [J]. Artificial Intelligence,1995,76(1/2):455-480

[2] 伍丽华,陈嵩洋,姜云飞. 规划问题编码为约束可满足问题的研究[J]. 计算机科学,2006,33(8):187-292

[3] Pham D N, Thornton J, Sattar A. Modelling and Solving temporal reasoning as propositional satisfiability [J]. Artificial Intelligence,2008,172(15):1725-1782

[4] Miguel I, Shen Q. Solution Techniques for Constraint Satisfaction Problems: Foundations [J]. Artificial Intelligence Review, 2001,15(4):243-267

[5] Dechter R. Bucket Elimination: A Unifying Framework for Reasoning [J]. Artificial Intelligence,1999,113:41-85

[6] Bryant R E. Symbolic Boolean Manipulation with Ordered Binary Decision Diagrams [J]. ACM Computer Survey, 1992, 24(3): 293-318

[7] Burch J, Clarke E, Mcmillan K, et al. Symbolic Model Checking: 10^{20} States and Beyond[J]. Information and Computation,1998, 98(2):142-170

[8] Hachtel G D, Somenzi F. A Symbolic Algorithm for Maximum Flow in 0-1 Networks [J]. Formal Methods in System Design, 1997,10(2/3):207-219

[9] Pan G, Vardi M Y. Symbolic Techniques in Satisfiability Solving [J]. Journal of Automated Reasoning,2005,35(1-3):25-50

[10] Walsh T. SAT v CSP [C] // Proceeding of the 6th international Conference on Principles and Practice of Constraint Programming. 2000,1894:441-456

[11] 徐周波,古天龙. 装配序列规划问题的 CSP 模型及其符号 OBDD 求解技术[J]. 计算机辅助设计与图形学学报,2010,22(5): 803-810