计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2015.20131508
52(3):769-776,2015

带权图的均衡 k 划分

郑丽丽　武继刚　陈　勇　朱梅霞
(天津工业大学计算机科学与软件学院　天津　300387)
(计算机系统结构国家重点实验室(中国科学院计算技术研究所)　北京　100190)
(lilizheng2013@163.com)

Balanced k-Way Partitioning for Weighted Graphs

Zheng Lili, Wu Jigang, Chen Yong, and Zhu Meixia
(*School of Computer Science and Software Engineering, Tianjin Polytechnic University, Tianjin* 300387)
(*State Key Laboratory of Computer Architecture* (*Institute of Computing Technology, Chinese Academy of Sciences*), *Beijing* 100190)

Abstract　Balanced k-way partitioning for a weighted graph is to divide the vertex set of the graph into k disjoint subsets, in order to minimize the difference of the sums of the vertex-weights between two subsets, together with minimizing the sum of the edge-weights, whose incident vertices belong to different subsets. This k-way partitioning problem is widely applied in the areas such as hardware/software co-design, VLSI design, data partitioning, etc., and it has been proved to be NP-complete. An efficient heuristic algorithm is proposed to generate a good approximate k-way partition by maximizing the sum of the weights associated with the inner edges of the subsets, together with a relatively balanced partition. In detail, the proposed heuristic algorithm constructs a subset S by selecting a group of neighboring vertices with the highest gain from its candidate subset for inclusion S until the sum of vertex-weights of S meets the demand. Moreover, we customize an approach based on tabu search to refine the heuristic partition. Experimental results show that, the proposed algorithm works more efficiently for average solution than the state-of-the-art on 86%, 81% and 68% graphs among the public benchmark, for the cases $k=2,4,8$, respectively, with the improvement up to 60%.

Key words　weighted graph; k-way partitioning; heuristic algorithm; tabu search; algorithm design

摘　要　带权图的均衡 k 划分是把一个图的顶点集分成 k 个不相交的子集,使得任意 2 个子集中顶点的权值之和的差异达到极小,并且连接不同子集的边权之和也达到极小.这种图的 k 划分问题已被应用在软硬件协同设计、大规模集成电路设计和数据划分等领域,它已被证明是 NP 完全问题.首先针对带权图的均衡 k 划分问题提出了能够生成优质近似解的启发式算法.该算法在保证子集均衡的条件下,采用最大化同一子集内部边权之和的策略来构造每一个顶点子集;构建子集 S 的思想是每次从候选集中选择与子集 S 相连的具有最大增益的顶点放入子集 S 中,直到子集 S 的顶点权值之和满足要求.此外,采用了定制的禁忌搜索算法对生成的初始近似解实施进一步优化.实验结果表明,当 k 分别取值为 2,4,8 时所提算法分别在 86%,81%,68%的基准图上求得的平均解优于当前最新算法求得的平均解;解的最大改进幅度可达 60%以上.

收稿日期:2013-11-21;**修回日期**:2014-01-20
基金项目:国家自然科学基金项目(61173032);中国科学院计算机系统结构国家重点实验室开放课题(CARCH201303)
通信作者:武继刚(asjgwu@gmail.com)

关键词 带权图;k 划分;启发式算法;禁忌搜索;算法设计

中图法分类号 TP301.6

图的 k 划分问题是一种组合优化问题,它已被应用在任务调度[1]、大规模集成电路设计[2]和数据划分[3]等众多领域.图的 k 划分问题的基本思想是将一个图的顶点集分成 k 个不相交的子集,且子集之间满足某些限制,如每个子集的顶点数之和有极小的差异,且连接不同子集的边数之和最小.从划分所得的节点子集的大小或权值来讲可分为均衡 k 划分和非均衡 k 划分,现有的研究大多集中在均衡 k 划分方面.

图的 k 划分问题是经典的 NP 完全问题[4],大多实际可行的解决方案都是基于启发式规则的调整策略.一种典型的求解策略是文献[5]提出的 KL (Kernighan-Lin)算法.其主要思想是先将图随机划成两等分,然后对交换任意 2 个顶点能导致的收益值进行估价,再高效地从中选择收益最高的点进行交换.该算法通常处理一万个顶点以内的图.文献[6]的 FM(Fiduccia-Mattheyses)算法对 KL 算法进行了改进.FM 算法采用单点移动,并引入了桶列表(bucket data structure)数据结构,减少了时间复杂度.

智能优化算法也被广泛用来解决图划分问题.如禁忌搜索算法(tabu search)[7]、模拟退火算法[8]和遗传算法[9-11].为了能更有效地处理规模较大的图,文献[12]提出了多层划分算法(multilevel approach).该算法包括 3 个阶段:1)通过粗糙化(coarsen-ing)技术将大图归约到可接受的小图;2)将第 1 阶段获得的小图进行随机划分,并进行优化;3)通过反粗糙化(uncoarsening)技术以及优化技术将小图的划分还原为原图的划分.该算法广泛地应用在各类大图的划分,对于百万规模以内的图通常具有较好的实际效果.为了提高该算法的性能,文献[13-15]分别对多层划分算法进行了不同程度的改进.

现有的图的 k 划分算法大部分是解决不带权图(顶点/边的权值相等)的划分问题.然而在某些实际的应用场景中,每个顶点/边的权值可能不相等.例如在多处理机系统中将一组关联的任务分配到 k 个处理机并行执行时,其中这一组关联的任务对应着一个任务图.任务图中的顶点代表计算任务,边代表数据交换关系.每个任务都会执行一定的计算量,因而顶点被赋予一定的权值.同样,边上的权值就反映了任务之间的数据交换量.在实际中顶点(边)的权值可能大不相同,此时不带权图的划分问题模型便不能满足实际需求.因而带权图的划分问题模型可以满足更为复杂的现实需求.为了加速任务的完成,每个处理器需执行大致相同的任务量,通常称之为负载均衡.故对多处理机系统而言,需要将任务(顶点)均衡地分配到 k 个处理器(子集)中,同时使处理器之间的数据交换量(边权之和)最小.可见,研究带权图的均衡 k 划分有着重要的现实意义.

本文研究的是带权图的均衡 k 划分问题,这不同于之前被广泛研究的不带权图的均衡 k 划分问题.本文首先提出了求解带权图的均衡 k 划分问题的启发式算法,该启发式算法能求得一个近似均衡的启发解.然后用特定的禁忌搜索算法优化求得启发解.实验结果表明,本文提出的算法在大多数情况下能明显优于文献[16]的算法.

1 带权图的均衡 k 划分模型

1.1 带权图的定义

定义 1. 给定一个无向带权图 $G=(V,E)$,其中 V 为图 G 的顶点集;E 为其边集;(u,v) 代表了连接顶点 u 和顶点 v 的边.给定顶点 v,$w(v)=m$ 表示顶点 v 上的权值为 m.同样,给定顶点 v 和 u,$w(u,v)=n$ 表示连接顶点 v 和 u 的边的权值为 n.

1.2 带权图的均衡 k 划分

定义 2. 给定一个无向带权图 $G=(V,E)$ 和划分的子集数 k,k 为正整数,图 G 的均衡 k 划分定义为一个映射函数 $\pi:V\rightarrow\{1,2,\cdots,k\}$,即把顶点集 V 均衡(每个子集的顶点的权值之和具有极小化差异)划分到 k 个不相交的子集 $V_1,V_2,\cdots,V_k$ 中,且 $\bigcup_i V_i=V$;使连接不同子集的边权之和最小.

连接不同子集的边权之和在文中称为割权(cut-weights).子集 V_i 的顶点权值 $\omega(V_i)$ 等于子集 V_i 所包含的顶点的权值之和,即

$$\omega(V_i)=\sum_{v\in V_i}w(v),$$

则所有顶点的权值之和为

$$\omega_{\text{sum}}=\sum_{i=1}^{k}\omega(V_i).$$

划分时,希望子集 V_i 的顶点的权值 $\omega(V_i)$ 满足

如下条件:$\omega(V_i)\leqslant\omega_e,\omega_e=\lceil\omega_{\text{sum}}/k\rceil$,其中$|x|$是不小于 x 的最小正整数.

图 1 是带权图的均衡划分问题的一个简单举例.图 1 中共有 5 个顶点,顶点和边均有权值,例如顶点 1 的权值为 6,连接顶点 1 和顶点 2 的边的权值为 6.现将图 1 中的 5 个顶点集划分到子集 V_1 和 V_2 中,即 $k=2$.经计算求得 $\omega_e=\lceil\omega_{\text{sum}}/k\rceil=9$.如图 1 所示,将顶点集划分后的结果为:顶点 1,2,3 属于子集 V_1,顶点 4,5 属于子集 V_2,$\omega(V_1)=\omega(V_2)=9$,割权为 4.

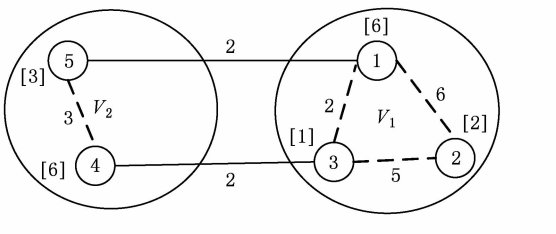


Fig. 1 An example for weighted graph partitioning.

图 1 带权图划分的简单举例

然而对于某些图而言,顶点的权值可能差异较大,因而不能得到非常均衡的 k 划分.如文献[16]所述,为有效地反映集合之间顶点权值的差异,定义参数 e 来反映划分的均衡情况:

$$e=\max_{i\in\{1,2,\cdots,k\}}\frac{\omega(V_i)}{\omega_e}.$$

不难理解,如果划分后任意子集的顶点的权值相等,那么 $e=1$.然而在实际中顶点的权值可能差异较大,所以理论上就得不到非常均衡的 k 划分,即在大多数情况下 $e>1$.因而,带权图的均衡 k 划分是在近似均衡(e 比 1 稍微大一点)的前提下最小化割权.如果 2 个不同的划分有相同的割权,那么 e 相对小的划分要优于 e 相对较大的划分.

2 启发式算法

针对带权图的均衡 k 划分问题,本节提出了启发式算法 PART(partition)来寻找一个初始近似解.因为带权图的均衡 k 划分是在寻找近似均衡的 k 划分的前提下最小化割权.故所提算法 PART 在保证子集近似均衡的条件下,采用最大化同一子集的内部边权之和的策略来实现最小化割权.假设 $V_1,V_2,\cdots,V_k$ 是要构建的 k 个子集.算法 PART 首先将所有的顶点都放在子集 V_k 中;同时求出每个子集所期望的顶点的权值之和 ω_e.然后逐个构建 k 个子集,构建子集 V_i 的基本思想是每次从子集 V_k 中选择最大增益的顶点 v(v 和子集 V_i 相连)放入子集 V_i 中,直到子集 V_i 的顶点的权值之和满足要求.

构建每个子集 V_i($1\leqslant i\leqslant k-1$,$i$ 是正整数)的具体过程如下:初始子集 V_i 为空,为了保证解的多样性,先随机地从子集 V_k 中选择顶点 v_i 移动到子集 V_i;此时子集 V_i 便包含了顶点 v_i.顶点 v_i 的邻接顶点构成了候选解 S.然后将从候选解 S 中选择最大增益的顶点 v_j 移动到子集 V_i;同时,与顶点 v_j 邻接的且仍属于子集 V_k 的顶点被添加到候选解 S.为了保证任意 2 个子集的顶点的权值之和有极小化差异,上面的选择操作一直重复直到子集 V_i 的顶点的权值之和不少于 ω_e.当子集 $V_1,V_2,\cdots,V_k$ 都建立完后,子集 V_k 中剩余的顶点集都属于子集 V_k.从候选解 S 中选择顶点是根据顶点移动到该子集 V_i 的增益($gain$)大小确定的,如给定顶点 v,则:

$$gain(v)=\sum_{u\in V_i}w(u,v)-\sum_{u\in V_k}w(u,v).$$

为了使选中的顶点 v 和子集 V_i 相连的边权之和尽可能大且顶点 v 和子集 V_k 相连的边权之和尽可能小,所以具有最大增益的顶点 $v_{\max}$ 被选择移动到子集 V_i.如果顶点 $v_{\max}$ 此时有最大的增益,说明顶点 $v_{\max}$ 和子集 V_i 相连的边权之和与 $v_{\max}$ 和子集 V_k 相连的边权之和的差值最大,将顶点 $v_{\max}$ 移动到子集 V_i 可以最大化子集 V_i 的内部边权之和.有一些顶点可能同时有最大增益,那么此时将选择先进入候选解 S 的顶点.为了能有效地减少寻找最大增益的时间,本文采用了文献[16]的桶结构,它按照顶点的增益大小将顶点排序.算法 PART 的形式化描述见算法 1.

算法 1. $PART(G,k)$.

/* 寻找图 G 的近似均衡 k 划分 */

输入:一个带权有向图 G 和 k 的值;

输出:图 G 的一个均衡 k 划分($X=(x_1,x_2,\cdots,x_{|V|})$,$x_i\in\{1,2,\cdots,k\}$,$1\leqslant i\leqslant|V|$).

① Begin

② 对所有的顶点 e,计算其权重之和 ω_e;

③ $i:=0$, $X:=(k,k,\cdots,k)$;

④ While $i<k-1$ Do

⑤ $i:=i+1$;

⑥ $S:=\varnothing$, $V_i:=\varnothing$, and $\omega(V_i):=0$;

⑦ While $\omega(V_i)<\omega_e$ Do

⑧ If $S=\varnothing$ Then

⑨ 从 V_k 中随机选择顶点 v 放到 V_i;

⑩ $\omega(V_i):=w(v)+\omega(V_i)$;

⑪ Else

⑫ 选择具有最大 $gain$ 的顶点 $v_{\max}$;
⑬ 将顶点 $v_{\max}$ 从 V_k 移动到 V_i 中;
⑭ $\omega(V_i):=w(v_{\max})+\omega(V_i)$;
⑮ End If;
⑯ 更新 S;
⑰ End While
⑱ End While
⑲ End

在算法 PART 的形式化描述中,行⑦～⑰的 While 循环是建立每个子集的过程,行④～⑱是建立所有子集的过程.算法 $PART$ 的时间复杂度主要是由 2 个部分来主导的:1)将顶点 v 从子集 V_k 移动到目标子集 V_i(行⑬);2)更新候选解 S(行⑯).

因为采用了文献[16]的桶结构,移动顶点 $v\in V_k$ 到目标子集 V_i 的时间复杂度大致为顶点 v 的邻接点数目;寻找最大的 $gain$ 在 $O(1)$ 时间内就可以完成.假设 $n(v)$ 是顶点 v 的邻接点数目,那么构建子集 V_i(通过移动顶点 v 到子集 V_i)的移动操作所消耗的时间为

$$t(V_i)=O\Big(\sum_{v\in V_i}n(v)\Big),$$

则构建所有子集在最坏的情况下的执行时间为

$$O\Big(\sum_{i=1}^{k}t(V_i)\Big)=O(|V|),$$

因而移动操作所消耗时间之和为 $O(|V|)$.更新选候选解 S 只考虑被移动到子集 V_i 的顶点的邻接边.假设 m_i 为顶点 v_i 的邻接边数之和,那么更新选候选解 S 在最坏的情况下执行时间为 $m_1+m_2+\cdots+m_{|V|}=2|E|$.所以构建所有子集时更新候选解 S 所消耗的时间为 $O(|E|)$.综上所述,算法 $PART$ 的时间复杂度为 $O(|V|+|E|)$.

总之,给定带权图 G,所提算法 $PART$ 能高效快速地寻找到图 G 的均衡 k 划分的一个近似解.算法 PART 产生的初始解将作为禁忌搜索算法的输入.

3 禁忌搜索算法

禁忌搜索是一种元启发式优化算法,最早于 1977 年由 Glover 提出,它有效地解决了许多较难的组合优化问题.禁忌搜索算法是对局部邻域搜索的一种扩展,并且是一种全局逐步寻优算法;其基本思想是从一个初始可行解出发,选择一系列的特定搜索方向(移动)作为试探,选择实现让特定的目标函数值变化最多的移动.禁忌搜索算法模仿人类的记忆功能,为了避免迂回搜索,使用禁忌表来禁止重复前面达到的局部最优状态[16];同时赦免禁忌中的一些优良状态,进而保证搜索的多样性,从而达到全局优化.

由于禁忌搜索算法主要是基于邻域搜索的,初始解的好坏对搜索的性能影响很大.特别是一些复杂的问题,如果随机给出初始解是很难找到一个较好的可行解的.因而,一般采用启发式方法找出一个可行解作为禁忌搜索算法的初始解.本文使用禁忌搜索算法对所提算法 $PART$ 产生的初始近似解进行优化,即将所提算法 $PART$ 产生的可行解作为禁忌搜索算法的初始解.本文中禁忌搜索算法的目标是在满足一定的均衡条件下,使得连接不同子集的边权和尽可能小.

3.1 邻域和禁忌对象

不难理解,邻域的选择对结果有重要的影响;正如文献[17]所述,不同的邻域定义将会产生不同质量的解.在该问题中,邻域是基于一种移动操作.移动是从当前解产生新解的途径.从当前解可以进行的所有移动构成邻域.

定义 3. 移动操作.先假设 $P=\{V_1,V_2,\cdots,V_k\}$ 为图 G 的一个 k 划分.首先从不同的子集中选择两顶点 $v_1\in V_i,v_2\in V_j$,且 $v_1(v_2)$ 和目标子集 $V_j(V_i)$ 至少有一个顶点相连.然后将顶点 $v_1(v_2)$ 移动到 $V_j(V_i)$.禁忌的对象是顶点 v 移动到其原始的子集 V_i,在本文中禁忌长度为 6.

3.2 选择策略

假设已经产生当前解 X_{cur} 的邻域解.如第 1 节所述,图的均衡 k 划分是在 e 比 1 稍微大一点的前提下,最小化割权.本文是在 $e\leqslant 1.01$ 的前提下寻找最优解.显然在 $e\leqslant 1.01$ 的前提下,割权越小邻域越好.如果不止一个邻域有最小割权,选择策略应由以下 2 点决定:

1) 首先考虑 3.1 节提出的对象是否处于禁忌状态;

2) 考虑移动频数,即每个顶点移动到不同子集的频数.将惩罚移动频数较高的顶点;对移动频数较低的顶点,给予较高的优先选择权.

在每次迭代中,本文根据上述 2 点从当前解的邻域中选择最好的非禁忌状态的邻域解.同时附加一个特赦准则,即全部对象都处于禁忌状态或某个禁忌对象能使目标值有较大的减少时,采用特赦准

则,使这些禁忌对象重新可选.如果该邻域解优于历史最优解 X_{best},那么更新历史最优解 X_{best}.

3.3 扰动机制及停止准则

为了带来搜索的多样性,当迭代 n 次还得不到较好的解时,本文引入扰动机制.在本文中 n 设为150,扰动机制的基本思想是在不考虑禁忌状态和移动频数的前提下随机移动顶点到其他的子集.当算法执行到事先设置的最大迭代次数算法终止,最大迭代次数设为 2 000.算法 TSR(tabu search to refine)的形式化描述如下.

算法 2. $TSR(G,X)$.

/* 改进初始解 X 的禁忌搜索算法 */

输入:算法 $PART$ 所产生 G 的一个均衡 k 划分的初始近似解 X;

输出:图 G 的一个均衡且最好的划分 X_{best}.

① Begin
② 初始化 *tabu tenure*, *move_frequence_list*, 将迭代计数器 *iter* 初始化为 0,*tabulist*=∅;
③ $X_{cur}:=X$, $X_{best}:=X$;
④ Repeat
⑤ 计算 X_{cur} 的邻域;
⑥ 计算邻域中每个节点的割权;
/* 如果基于一种移动操作所产生的邻域解优于历史最优解 X_{best},那么它的禁忌状态将被忽略 */
⑦ 为移动操作选择目标子集 $V_j(V_i)$ 和顶点 $v_1(v_2)$;
⑧ 移动 $v_1(v_2)$ 到 $V_j(V_i)$,产生一个新的划分 X^*, $X_{cur}:=X^*$;
⑨ 更新 *tabulist* 和 *move_frequence_list*;
⑩ If $e(X_{cur})\leqslant 1.01$ and $f(X_{cur})<f(X_{best})$
⑪ Then $X_{best}:=X_{cur}$;
⑫ End If
⑬ If X_{best} 在 n 次迭代之后没有提高 Then
⑭ 在子集中随机移动一些顶点;
⑮ End If
⑯ Until 终止准则得到满足;
⑰ Return X_{best};
⑱ End

4 实验结果及分析

本文实验所用的带权图是在一套广泛用来评估图划分算法的基准图(benchmark graphs)的基础上产生的,其中基准图来自格林威治大学 Walshaw 教授[18]所收集的一系列开放的测试用图.为了能有效地评估本文算法的性能,针对带权图(给定图中任意顶点 u,$w(u)\in[1,10]$,若顶点 u 和 v 相连,$w(u,v)\in[1,10]$)的均衡 k 划分作了大量实验.表 1 描述了实验基准图的属性,其中 $|V|$ 和 $|E|$ 分别为图中的节点数和边数:

Table 1 The Characteristics of Benchmark Graphs

表 1 实验基准图的属性

Graph	$\|V\|$	$\|E\|$	Graph	$\|V\|$	$\|E\|$
add20	2 395	7 462	4elt	15 606	45 878
data	2 851	15 093	cti	16 840	48 232
3elt	4 720	13 722	cs4	22 499	43 858
uk	4 824	6 837	bcsstk30	28 924	1 007 284
add32	4 960	9 462	fe_pwt	36 519	144 794
bcsstk33	8 738	291 583	bcsstk32	44 609	958 046
whitaker3	9 800	28 989	brack2	62 631	366 559
crack	10 240	30 380	fe_tooth	78 136	452 591
wing_nodal	10 937	75 488	fe_rotor	99 617	662 431
fe_4elt2	11 143	32 818	598a	110 971	741 934
bcsstk29	13 922	302 748	fe_ocean	143 437	409 593

用本文算法 PART 构造出优质启发解,并用禁忌搜索算法对启发解进行优化的整个算法过程在本文中记作 PATS(PART-TSR).如文献[1]所述,不带权图的划分算法通常也用来处理带权图的划分问题.文献[16]所提的算法 MITS(multilevel iterated tabu search)是当前最新的图划分算法.本文是将所提算法 PATS 和算法 MITS 在相同的实验基准图上进行性能比较.本文使用了文献[16]提供的源代码.为解决带权图的均衡 k 划分问题,需要在不改变算法 MITS[16]的时间复杂度和相关参数设置的前提下作少许的改动.例如在算法 MITS 的粗化阶段(coarsening phase)中,如果顶点 $v_1\in V_i$ 和 $v_2\in V_i$ 合并成新的顶点 $v_c\in V_{i+1}$;那么 v_c 的顶点的权值为 v_1 和 v_2 的顶点的权值之和,v_c 的边权等于 v_1 和 v_2 的边权之和减去 2 倍的 $w(v_1,v_2)$.

对每个实验基准图,在 $e\leqslant 1.01$ 的前提下,比较所提算法 PATS 和算法 MITS 运行大致相同时间所求得割权.由划分目标可知,在均衡程度相同的条件下,割权越小越好.如文献[16]所述,算法 MITS 不能保证运行一次就能获得较好的解,因此对每个基准图作 20 次实验.为了公平地比较算法的性能,本文算法 PATS 也运行 20 次.表 2 和表 3 分别是上

述两种算法最好解和平均解的实验结果,其中 k 分别取值为 2,4,8.

表 2 展示了 20 次重复实验中最终获得的可行解的最好值(最好解). 表 2 表明:当 $k=2$ 时,本文所提算法 PATS 在 77%的基准图上明显优于算法 MITS[16];解的改进幅度最大可达 70%. 当 $k=4$ 时,本文所提算法 PATS 在 63%的基准图上优于算法 MITS;解的改进幅度最大可达 29%. 当 $k=8$ 时,本文提出的算法和算法 MITS 在性能上是可比的,解的改进幅度最大可达 12%.

表 3 展示了 20 次实验中最终获得的可行解的平均值(平均解). 实验结果表明,当 $k=2$ 时,本文所提算法 PATS 在 86%的基准图上明显优于算法 MITS. 例如,在图 3elt,add32,4elt,cti 及 598a 上解的质量均提高 64%以上;在图 uk, fe_4elt2, wing_nodal 及 fe_ocean 上解的质量均提高 52%以上;在图 data,whitaker3,crack,cs4,bcsstk30,fe_pwt 及 fe_rotor 上解的质量均提高 22%以上. 如表 3 所示,$k=4,8$ 时分别有 81%,68%的基准图,本文所提算法求得的平均解优于算法 MITS 求得的平均解. $k=4$ 时,解的改进幅度最大可达到 37.7%;$k=8$ 时,解的改进幅度最大可达到 28.9%. 表 3 的实验结果表明,所提算法 PATS 求得的平均解要明显优于算法 MITS 求得的平均解.

表 2 和表 3 综合表明,虽然原算法在某些情况下能取得质量较好的解,但这些解的质量与其平均值相比还是有较大差距. 这说明算法 MITS 是不稳定的. 因为算法 MITS 用随机初始划分作为禁忌搜索算法的初始解;然而本文提出了启发式算法能为禁忌搜索算法提供较好的初始解. 这是因为本文所提的启发式算法 PART 能最大化同一子集内的边权之和,使得连接不同子集的边权之和最小化.

Table 2 Performance Comparison between Our Algorithm PATS and MITS for Best Solution

表 2 新算法 PATS 与旧算法 MITS 在最好解上的比较

Graph	k					
	2		4		8	
	PATS	MITS	PATS	MITS	PATS	MITS
add20	3 495	2 885	5 676	5 477	**8 375**	8 539
data	**996**	1 003	**2 310**	2 610	3 767	3 611
3elt	**460**	773	**1 378**	1 809	**2 075**	2 232
uk	**238**	348	495	426	**984**	1 019
add32	**56**	217	**244**	288	**526**	604
bcsstk33	57 553	49 057	**107 342**	108 095	176 491	166 502
whitaker3	**795**	802	2 618	1 921	4 244	3 258
crack	**1 275**	1 895	2 367	1 739	4 351	3 086
wing_nodal	**8 428**	12 672	18 012	16 881	29 163	25 558
fe_4elt2	**667**	1 158	**2 174**	2 194	3 741	3 206
bcsstk29	28 929	13 658	**42 468**	44 876	78 858	72 410
4elt	**742**	2 012	**2 306**	2 589	**4 056**	4 214
cti	**2 136**	3 876	5 793	5 250	**11 920**	12 495
cs4	**2 519**	2 628	7 717	5 396	12 172	9 128
bcsstk30	**35 564**	38 291	**141 444**	192 523	**298 328**	306 519
bcsstk32	41 507	30 968	**82 805**	95 744	174 805	155 683
fe_pwt	**1 883**	3 541	**3 850**	4 999	10 233	8 896
brack2	9 240	8 026	**23 940**	29 011	**46 448**	46 849
fe_tooth	**30 835**	31 575	55 254	49 453	84 559	70 392
fe_rotor	**10 754**	27 567	**46 124**	65 552	**84 746**	85 576
598a	**12 457**	31 150	**44 961**	45 145	95 859	90 042
fe_ocean	**3 457**	7 416	**13 897**	14 325	**24 676**	26 480

Table 3 Performance Comparison between Our Algorithm PATS and MITS for Average Solution

表 3 新算法 PATS 与旧算法 MITS 在平均解上的比较

Graph	k					
	2		4		8	
	PATS	MITS	PATS	MITS	PATS	MITS
add20	3 502.0	3 090.4	5 939.2	5 801.0	**8 476.9**	9 001.4
data	**1 065.6**	1 377.5	**2 487.6**	3 381.1	**4 140.9**	5 028.6
3elt	**597.1**	1 679.6	**1 537.3**	2 233.6	**2 215.1**	3 114.6
uk	**331.3**	747.5	**557.4**	655.9	**1 104.4**	1 284.1
add32	**82.1**	364.4	**269.5**	452.7	**629.4**	648.9
bcsstk33	57 553.0	54 581.2	**109 671.0**	114 821.0	179 830.5	172 555.8
whitaker3	**902.1**	1 278.5	2 766.5	2 686.5	4 381.7	3 605.7
crack	**1 501.4**	3 096.3	2 677.7	2 551.6	4 590.4	4 468.1
wing_nodal	**8 512.3**	18 009.4	**18 811.0**	18 908.5	**30 081.8**	34 428.1
fe_4elt2	**867.3**	1 596.6	**2 474.2**	2 637.5	**4 062.8**	4 100.1
bcsstk29	28 929.0	18 712.3	**44 001.0**	52 633.0	**81 112.2**	91 611.8
4elt	**1 180.3**	3 850.1	**2 612.7**	2 936.9	**4 223.1**	4 507.7
cti	**2 227.1**	7 255.5	**6 866.2**	6 895.9	**12 976.6**	14 287.0
cs4	**2 831.3**	3 694.0	7 918.9	6 818.6	12 391.3	9 402.3
bcsstk30	**51 494.0**	85 603.1	**159 072.9**	232 699.8	**302 293.3**	352 835.3
bcsstk32	**48 283.6**	57 070.6	**93 078.7**	138 032.9	181 920.0	179 692.2
fe_pwt	**2 146.0**	4 768.1	**4 517.3**	7 071.6	**10 620.3**	10 772.5
brack2	**12 076.9**	14 152.1	**27 467.1**	35 064.8	51 048.3	49 531.5
fe_tooth	**32 045.7**	37 209.8	**60 789.9**	60 963.7	85 334.5	77 059.3
fe_rotor	**20 987.8**	41 537.5	**50 263.7**	80 658.0	**85 799.8**	91 069.6
598a	**14 983.6**	50 986.6	**47 229.5**	47 438.0	**99 801.4**	106 634.6
fe_ocean	**4 921.8**	10 535.7	**14 531.8**	15 347.3	**30 853.4**	33 088.5

5 结束语

本文针对带权图的均衡 k 划分问题提出了一种高效的启发式算法 $PART$,并用特定的禁忌搜索算法对算法 $PART$ 求得的优质初始解进行优化.本文所提算法和文献[16]的算法在相同的实验基准图上进行比较,且不改变文献[16]的相关参数设定.实验结果表明,所提启发式算法能为禁忌搜索算法提供较好的初始解.当 k 分别取值为 2,4,8 时,本文所提算法求得的最优解分别在 77%,63%,50%的基准图上,优于文献[16]的算法求得的最优解,所求得的平均解分别在 86%,81%,68%的基准图上,明显优于文献[16]的算法求得的平均解;解的改进幅度最高可达到 60%以上.

参 考 文 献

[1] Karypis G, Kumar V. A fast and high quality multilevel scheme for partitioning irregular graphs [J]. SIAM Journal on Scientific Computing, 1998, 20(1): 359-392

[2] Słowik A, Białko M. Partitioning of VLSI circuits on subcircuits with minimal number of connections using evolutionary algorithm [G] //LNCS 4029: Proc of the 8th Conf on Artificial Intelligence and Soft Computing (ICAISC'2006). Berlin: Springer, 2006: 470-478

[3] Lin Jiao, Chen Wenguang, Li Qiang, et al. A new data clustering algorithm for parallel whole-genome shotgun sequence assembly [J]. Journal of Computer Research and Development, 2006, 43(8): 1323-1329 (in Chinese)
(林皎, 陈文光, 栗强, 等. 基于图划分的全基因组并行拼接算法[J]. 计算机研究与发展, 2006, 43(8): 1323-1329)

[4] Garey M R, Johnson D S, Stockmeyer L. Some simplified NP-complete graph problems [J]. Theoretical Computer Science, 1976, 1(3): 237-267

[5] Kernighan B W, Lin S. An efficient heuristic procedure for partitioning graphs [J]. The Bell System Technical Journal, 1970, 49(1): 291-307

[6] Fiduccia C M, Mattheyses R M. A linear-time heuristic for improving network partitions [C] //Proc of IEEE DAC'82. Piscataway, NJ: IEEE, 1982: 175-181

[7] Battiti R, Bertossi A A. A greedy and prohibition-based heuristics for graph partitioning [J]. IEEE Trans on Computers, 1999, 48(4): 361-385

[8] Aarts E, Korst J, Michiels W. Simulated Annealing [M]. Berlin: Springer, 2005: 187-210

[9] Bui T N, Moon B R. Genetic algorithm and graph partitioning [J]. IEEE Trans on Computers, 1996, 45(7): 841-855

[10] Menouar B. Genetic algorithm encoding representations for graph partitioning problems [C] //Proc of IEEE ICMWI'10. Piscataway, NJ: IEEE, 2010: 288-291

[11] Soper A J, Walshaw C, Cross M. A combined evolutionary search and multilevel optimisation approach to graph-partitioning [J]. Journal of Global Optimization, 2004, 29(2): 225-241

[12] Hendrickson B, Leland R. A multilevel algorithm for partitioning graphs [C] //Proc of ACM/IEEE Sumpercomputing'95. New York: ACM, 1995

[13] Walshaw C, Cross M. Mesh partitioning: A multilevel balancing and refinement algorithm [J]. SIAM Journal on Scientific Computing, 2000, 22(1): 63-80

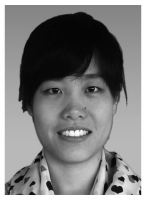
[14] Walshaw C. Multilevel refinement for combinatorial optimisation problems [J]. Annals of Operations Research, 2004, 131(1/2/3/4): 325-372

[15] Benlic U, Hao J K. A multilevel memetic approach for improving graph k-partitions [J]. IEEE Trans on Evolutionary Computation, 2011, 15(5): 624-642

[16] Benlic U, Hao J K. An effective multilevel tabu search approach for balanced graph partitioning [J]. Computers & Operations Research, 2011, 38(7): 1066-1075

[17] Fan W, Machemehl R B. A Tabu Search Based Heuristic Method for the Transit Route Network Design Problem [M]. Berlin: Springer, 2008: 387-408

[18] Walshaw C. The graph partitioning archive [OL]. [2011-03-20]. http://staffweb.cms.gre.ac.uk/~c.walshaw/partition/



Zheng Lili, born in 1988. Master. Her main research interests include graph partitioning and parallel computing (lilizheng2013@163.com).

Wu Jigang, born in 1963. PhD, professor. Member of China Computer Federation. His main research interests include theoretical computer science, and parallel and distributed computing.

Chen Yong, born in 1959. Master, associate professor. His main research interests include computer architecture and parallel computing (chenyong@tjpu.edu.cn).

Zhu Meixia, born in 1981. PhD, lecturer. Her main research interests include software modeling and program verification (zhu@pku.edu.cn).