ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.21, No.12, December 2010, pp.3068−3081
doi: 10.3724/SP.J.1001.2010.03713

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种求解度约束最小生成树问题的优化算法*

王竹荣+, 张九龙, 崔杜武

(西安理工大学 计算机科学与工程学院,陕西 西安 710048)

Optimization Algorithm for Solving Degree-Constrained Minimum Spanning Tree Problem

WANG Zhu-Rong+, ZHANG Jiu-Long, CUI Du-Wu

(School of Computer Science and Engineering, Xi'an University of Technology, Xi'an 710048, China)

+ Corresponding author: E-mail: wangzhurong@xaut.edu.cn

Wang ZR, Zhang JL, Cui DW. Optimization algorithm for solving degree-constrained minimum spanning tree problem. *Journal of Software*, 2010,21(12):3068−3081. http://www.jos.org.cn/1000-9825/3713.htm

Abstract: To solve the degree-constrained spanning minimum tree (DCMST) problems with a large scale of nodes, an optimization algorithm based on grafting and pruning operator is proposed. Learning from the flower planting techniques, this paper establishes, an evolutionary computation framework containing accelerating and adjusting operators based on conventional genetic operators. The grafting and pruning are performed by a greedy strategy and gain maximization respectively. The collision caused by possible local minima is analyzed and detected, and several methods dealing with the collision are discussed. To tackle the complexity of DCMST problems, some strategies of grafting and pruning are proposed. The convergence of the proposed algorithm and the computation complexity are analyzed. For DCMST problems of Euclidean and uniform random non-Euclidean instances from 50 to 500 nodes, the experiments show that the quality and convergence rate of the proposed method are the best compared with the known results.

Key words: DCMST; genetic algorithm; grafting; pruning

摘 要: 为求解大规模结点度约束最小生成树问题,提出一种带有嫁接和剪接算子操作的优化算法.通过借鉴花草果树种植技术,建立一种以基本遗传算子为基础、带有加速和调节算子作为激励的进化计算体系;嫁接以一种贪婪的思想加速搜索,按收益最大化原则进行剪接.对可能陷入局部极值引起冲突的现象及冲突检测的方法进行分析,并提出了冲突的若干解决方法.针对 DCMST 问题求解中的复杂性,提出了几种有效的嫁接和剪接的策略,并对算法的收敛性和计算复杂度进行了分析.通过该算法对结点数为 50~500 之间的 Euclidean 问题和按均匀随机方式产生的 non-Euclidean 度约束最小生成树问题进行求解.与现有文献的实验结果对比表明,该方法在求解最好解的精度和收敛速度上均有一定的优势.

关键词: 度约束最小生成树;遗传算法;嫁接;剪接

中图法分类号: TP301 **文献标识码**: A

度约束最小生成树(degree-constrained minimum spanning tree,简称DCMST)问题是一类难解的NP-hard问

* Supported by the National Natural Science Foundation of China under Grant No.60873035 (国家自然科学基金)
Received 2009-02-12; Revised 2009-04-27; Accepted 2009-07-29

题[1],它在通信网络、电力线网络、计算机网络、大规模集成电路等方面都有重要的应用[2,3].一直以来,对它的研究引起国内外许多学者的关注.Narula与Ho最早提出一种分枝限界法(a branch and bound algorithm),该算法可以得到较小规模DCMST问题的最佳解.Caccetta等人提出了一种求解DCMST问题的分枝和剪切算法(a branch and cut algorithm)[4].Andrade等人提出了基于双重解信息的Lagrangian算法[5].Behle等人在标准分枝和剪切算法的基础上,利用 0/1 整型规划中的问题分离原则,提出了一种优先的分枝和剪切算法(a primal branch-and-cut algorithm)[6].近些年来,许多学者对以遗传算法、免疫算法[7]为代表的进化计算方法进行研究并取得广泛的应用.其中,一些学者利用进化计算方法开展对DCMST问题求解.例如,Zhou和Gen首先提出利用遗传算法的途径求解DCMST问题,文中采用一种prüfer计数来表示一棵生成树的编码方法[8].Knowles等人设计了一种包含边结点权重信息的动态表结构的编码方法(the randomized primal method,简称RPM),是将其与多点爬山(multistart hillclimbing,简称MHC)、模拟退火(simulated annealing,简称SA)和遗传算法(GA)相结合的优化方法[3].Raidl等人对最优解中包含低权重边的频次进行统计分析,推导出结点数为 20~100 的DCMST问题边的权重及其在最优解中出现的概率公式,并以该概率作为执行变异操作边的选取依据的(1+1)-EA算法[9].Soak等人提出一种由子图边的结点组成的数字序列的编码方式(the edge-window-decoder),设计了几种按不同策略遍历子图边结点的算子(TCR)来构建生成树的算法[10].其他的方法如蚁群算法[11]、粒子流算法[12]等,均从不同方面取得了一定成效.

不同的DCMST问题可分为两大类,即non-Euclidean DCMST和Euclidean DCMST完全图问题.对这两大类约束最小生成树问题求解的复杂性,不同的文献有不同的看法.多数文献作者认为,按均匀随机方式产生的non-Euclidean问题要比Euclidean问题的求解更为复杂;与之相反,文献[10]的作者倾向Euclidean问题的求解更具有挑战性.通过大量的实验测试及分析,我们认为,以均匀随机方法生成non-Euclidean图例的DCMST问题具有更大的变化性和多样性,使得这类问题解的结构及求解的难易性具有更大的变化性.一方面,这类问题的Prim解对应的度值通常大于5,且在度为3时问题的最好解与Prim算法解的比值随不同的问题或分布范围有所不同.一些文献证实,Euclidean图例的DCMST问题对应Prim解的度最大为 5,且在度值为 3 或 4 时最好解与Prim算法解的比值在一定的范围内.当度为3时,该指标的界限值在不同的文献分别为5/3,1.5 和 1.402[13].文献[14]作者猜测性地认为,该指标界限值能进一步减小到 1.103.另一方面,对均匀随机方式产生分布在某一区间的non-Euclidean问题,随着结点数增加,其解的复杂度逐渐降低.主要表现在:当结点数增大到某一值后,该类问题不再是一个难求的NP-hard问题,较多情形下能以几乎 1 的概率构造出一棵度值为 3 的最小生成树,使其与Prim算法得到的无约束生成树(对应的度值一般大于 10)具有相同目标值.

通过反复的编程测试及对存在问题的分析,我们发展了基于度排列的编码方法[1],通过利用度维关系,不需经过完全解码就能求出及利用待考察结点的关联结点及构成边的权重信息;结合花草果树的人工种植和培育技术,提出一种带有嫁接及剪接算子的遗传算法(a genetic algorithm with grafting and pruning operators,简称GPOGA).通过将该算法用于对DCMST问题进行求解,与现有文献提供的实验数据的对比说明,本文算法在得到最好解精度以及算法的诸多性能指标均有很大提高.

1 DCMST 的数学模型

假设$G=(V,E)$为网图,其中:$V=\{v_1,v_2,\ldots,v_n\}$是结点的有限集合,$n=|V|$为结点总数;$E=\{e_1,e_2,\ldots,e_m\}$是G中边的集合,m 为边的总数.设 $S=\{v'_1,\ldots,v'_i,\ldots,v'_t \mid v'_i \in V\} \subseteq V$ 且$|S|\geqslant 1$;设 T 为图 G 的所有满足度约束的生成树的集合,因此,对DCMST问题求解就是寻找G的一棵生成树,这里用一个向量$\boldsymbol{x}$表示,$\boldsymbol{x}\in T$,使其满足给定的度约束并使对应边权重w_{ij}(一般指边长或费用,本文以下均指边长)之和为最小值.DCMST的数学模型可描述如下:

$$\begin{cases} \min\{z(\boldsymbol{x}) = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} w_{ij} \times x_{ij} | x_{ij} \in \boldsymbol{x}\} \\ \text{subject to} \quad \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} x_{ij} = n-1 \\ 1 \leqslant d_i \leqslant d_c,\ i = 1,2,...,n \\ \sum_{i=1}^{|S|-1}\sum_{j=i+1}^{|S|} x_{ij} \leqslant |S|-1,\ i,j \in S \\ \sum_{i=1}^{n} d_i = 2 \times (n-1) \\ x_{ij} \in \{0,1\},\ i,j = 1,2,...,n \end{cases} \tag{1}$$

上述数学模型由DCMST问题的优化目标函数及约束构成,其中:第 1 个约束为生成树包含不同结点的$n-1$条边;第 2 个约束为每个结点应满足的度约束条件,d_c为给定的度约束值;第 3 个约束表示生成树无环路;第 4 个约束为所有结点的度值和;第 5 个约束当$x_{ij}=1$时,由结点$\langle v_i,v_j\rangle$构成的边是生成树的一条边.

2 GPOGA 算法体系

2.1 GPOGA算法思想

在对与排序或路径长度有关的组合优化问题的求解中,广泛使用一项被称为邻域搜索(local search)的策略[15].其基本思想是,在算子的搜索过程中,应充分利用相邻结点的路径长度信息,使结点或边交换后的收益最大.文献[15]指出,邻域搜索为获取NP-hard问题的高精度解提供一种有效和可靠的途径.在类似问题的求解中,邻域搜索策略成为检验一个算子或算法是否具有高效性的基本前提.但应看到,它的基本出发点是利用一种贪婪思想,极有可能加速算法陷入局部最优.文献[9]的作者也指出,在对DCMST问题的求解中,较小权重的边以较大的概率被选取参与变异操作很可能使算法以更大的概率陷入局部最优.我们发现,当使用local search策略使算法陷入局部最优或消除非正常态时,一般会出现一种称为冲突的现象.而对冲突的检测和处理,成为本文算法要解决的一个关键问题.因此,从算子的作用看,如果将带local search的算子归结为一种加速算子,则算法应包含另一种算子,本文称它为调整算子.其作用是双重的:一方面,它对出现的非正常态进行及时处理,使得当算法陷入局部最优时,能对这种情形进行有效的检测,并使其以较大的几率跳出;另一方面,它同样包含local search,使算子的搜索具有更高的效率.

为便于叙述,以下先给出与本文算法相关的几个概念.

本文所指的树(生成树)是指一棵非空树($n>1$,除根结点之外的元素被分为$m(m>0)$个不相交的集合$T_1,T_2,\ldots,T_m$,其中每一个集合$T_i(1\leqslant i\leqslant m)$又是一棵树,称为根结点的子树,文中称每棵子树为树的一条分支.

在本文算法的嫁接和剪接操作中,当树发生变化时,指以待考察结点为根结点的整条分支的移动.

定义 1(收益和代价). 对一棵(生成)树T_1,若将某结点的一条分枝移至另一结点作为其一条分枝后产生的生成树为T_2,考察分枝移动前后生成树的边长和的变化,则定义收益(gain)和代价(cost)分别为

$$gain = f(T_1) - f(T_2) = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} w_{ij} \times x_{ij}^1 - \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} w_{ij} \times x_{ij}^2 \tag{2}$$

$$cost = f(T_2) - f(T_1) = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} w_{ij} \times x_{ij}^2 - \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} w_{ij} \times x_{ij}^1 \tag{3}$$

公式(2)与公式(3)中, $x_{ij}^1 \in T_1, x_{ij}^2 \in T_2$.

定义 2(嫁接). 从(生成)树中选取具有某种优良特性的分枝(嫁接枝)接入到待考察结点中,以形成更好生成树个体(指收益大于 0)的过程称为嫁接.

定义 3(剪接). 将(生成)树中的某一分枝(剪接枝)接入到另一位置,以形成可行个体或更好生成树的过程称为剪接.

2.2 嫁接算子及策略

2.2.1 嫁接算子的构造

嫁接算子操作可分为两步:1) 根据结点及其关联结点的边长信息,选择具有优良品质的嫁接枝;2) 将选择的嫁接枝重新接入,以形成更好的生成树个体,即嫁接后收益大于 0.

如何选取具有优良品质的嫁接枝,是嫁接操作的关键所在.要选取一条有效的嫁接枝,需解决以下两个关键问题:1) 有效利用边结点及其关联结点的边长信息.为了使嫁接及剪接操作更为有效,对各结点按其关联结点构成边的长度进行排序,并将排序结果保存在指针变量 *pnodesortdis* 中.2) 求任意结点的父结点及子结点关系.本文采用基于度的排列的编码方法,有关编码方法见文献[1].为此,设计了通过利用度维关系查找某一结点的父结点函数 *FindPareNode*(*par*1,*par*2)及其子结点的函数 *FindChildNode*(*par*1,*par*2),*par*1,*par*2 为算法所需参数,它们在较好情形下的时间复杂度均为 $O(1)$,在最坏情形下的时间复杂度为 $O(n)$.

FindPareNode 的基本思想是,从当前位置向前扫描,记录扫描过的结点的度值,根据扫描过的结点与度值的关系计算出其父结点位置,其伪码描述如下:

算法 1(检索某一结点位置 ***nodepos*** 的父结点位置 ***parvpos*** 算法).

```
Begin
    初始化参数:degreesum=0, counter=0, parvpos=nodepos;
    根据输入的个体 individual,得到其结点维 individual.chrom1 和度维 individual.chrom2;
    if parvpos=1 then
        return 1;
    end if
    parvpos=parcpos−1, counter=counter+1;
    degreesum=degreesum+individual.chrom2[parvpos];
    if parv>1 then degreesum=degreesum−1;
    while degreesum<counter do
        parvpos=parcpos−1,counter=counter+1;
        degreesum=degreesum+individual.chrom2[parvpos];
        if parv>1 then degreesum=degreesum−1;
    end while
    返回 parvpos;
End
```

按收益优先及度约束控制嫁接策略执行嫁接操作,算法的伪码描述如下:

算法 2(嫁接算法).

```
Begin
    设置所有结点使用标志变量 pflagnodeuse 为 0;
    初始化参数contrpara=d_c, i=1, k=1, bflaggrafting=false;
    while i≤n do
        设置 counter=0,结点 i 对应结点序号的标志 pflagnodeuse 为 1;
        由 pnodesortdis 得到与 i 关联的第 k 个结点序号 inodeassono,并计算关联边长 nodedis;
        while counter<contrpara && k≤n do
            求出 inodeassono 的父结点 assoparenodeno 及边长 assonodedis;
            计算边交换后收益 gain=nodedis−assonodedis;
            if inodeassono 不是 i 的子结点且 gain>0 then
                选择以 inodeassono 代表的分枝选,bflaggrafting=true;
            end if
            if bflaggrafting=true then
```

执行嫁接操作;

counter=counter+1,将 *inodeassono* 的标志 *pflagnodeuse* 变为 1,*bflaggrafting=flase*;

end if

k=k+1;

求得与 *i* 关联的第 *k* 个结点序号 *inodeassono* 及位置 *inodeassopos* 及关联边长 *nodedis*;

end while

i=i+1;

end while

End

2.2.2 嫁接策略

为使嫁接操作能够处理不同情形的 DCMST 问题,本文提出以下几种嫁接策略,它们分别是:

1) 收益优先嫁接策略

对考察结点,若所选关联结点分枝移至考察结点前后收益大于 0,则该分枝可选为嫁接枝.按该策略执行嫁接,操作简便,且总体效果较好.其缺点是,嫁接后某些结点的分枝数可能大大超过给定的度约束值,从而增加剪接的负担.

2) 无度约束最优嫁接策略

对考察结点,选取与其具有最小边长的关联结点作为嫁接枝.按该策略执行嫁接,在理想的情形下,通过该策略可生成一棵无度约束的最小生成树.其缺点主要有:1) 针对给定某一度约束值d_c,若通过Prim算法得到的无约束生成树对应的度值与d_c相差较大,该控制策略往往不能取得好的效果;2) 增加算法陷入局部最优解的概率.

3) 度约束控制嫁接策略

对考察结点新加入的分枝进行控制,一般不超过给定的度约束值d_c.按该策略执行嫁接,可使嫁接后的生成树基本满足度约束要求.

4) 双重信息嫁接策略

对考察结点,在判断某一结点代表的分枝是否作为嫁接枝时,应将边长信息(决定收益)和位置次序关系信息(决定优先关系)同时进行分析.也就是说,某一结点代表的分枝移动到考察结点后,即使收益不如其他的分枝,但考虑其位置次序关系,应优先将该分枝选为嫁接枝.由于在嫁接中需要同时考察边长及位置次序关系,如果处理恰当,则可有效提高算法的求解精度和收敛速度.

5) 概率选择嫁接策略

文献[9]给出了当最大结点数为 100 时,第 r 条边选择的概率值近似为 $q_r=((n-1)/n)^{r/2}(\sqrt{n/(n-1)}-1)$,$n$ 为结点数,$r(1\leqslant r\leqslant m)$为按边长排序分配的序号,$m$ 为边的总数.考虑到本文算法中嫁接算子的作用与文献[9]变异算子作用的差异,本文的处理方法是按边长排序:针对每一结点的边长信息设定一个阈值,当边长大于或等于该阈值时,其分配概率为 0;当边长小于给定的阈值时,按边长或以边长排列分配位置次序,然后分别计算关联结点代表的分枝对应的概率值.以下分别对它们的概率进行分析.

当按边长分配概率时,对考察结点i=1,2,…,n,设给定的阈值为r_i,设与i关联的某一结点j构成的边长为$len(i,j)$,定义$len(i,j)$与r_i的距离$dis(len(i,j),r_i)$为

$$dis(len(i,j),r_i)=r_i-len(i,j) \tag{4}$$

因此,结点 j 代表的分枝的选择概率为

$$p_{ij}=\begin{cases}0, & len(i,j)\geqslant r_i\\ \dfrac{r_i-len(i,j)}{\sum_{k=1,k\neq i}^{n}len(i,k)} & |\ len(i,k)<r_i,\ len(i,j)<r_i\end{cases} \tag{5}$$

当按以边长排列分配位置次序时,对考察结点i=1,2,…,n,设给定的阈值为r_i.由公式(4),计算出与i关联的结

点j构成的边长为$len(i,j)$与r_i的距离$dis(len(i,j),r_i)$,对$dis(len(i,j),r_i)$按递减排序,以确定与i关联的结点j构成的边的位置次序$rank(i,j)$,分别对它们分配 1,2,…,m值.则结点j代表的分枝的选择概率为

$$p_{ij}=\begin{cases}0, & 当len(i,j)\geqslant r_i\\ \dfrac{m+1-rank(i,j)}{\sum\limits_{k=1,k\neq i}^{m}(m+1-rank(i,k))}, & 当len(i,j)<r_i\end{cases} \tag{6}$$

公式(5)和公式(6)中,j=1,2,…,n 且 $j\neq i$.

在上述策略中,策略 1)~策略 4)属于稳定控制策略,策略 5)属于非稳定控制策略.在算法设计中,嫁接操作在一般情形采用上述策略 1)和策略 3)就能取得较好的效果;当求解一些复杂及特殊情形的 DCMST 问题,选择交替使用策略 2)、策略 4)或策略 5).

2.3 剪接算子及策略

2.3.1 剪接算子构造

嫁接时产生的生成树可能包含某些结点不满足度约束以及具有较差属性分枝的情形,均要进行剪接操作.

判断一条分枝是否在当前位置具有最(较)差属性的依据为下列两种情形之一:1) 若该分枝移动到另一关联结点的收益最(较)大;2) 若所有考察的分枝分别移动收益均小于 0,则指移动后代价最小的分枝.

算法 3 以一棵生成树子结点分枝进行修剪为例说明剪接操作的程序流程,算法的伪码描述如下:

算法 3(剪接算法).

```
Begin
    初始化剪接参数,设置 iprunflag,iprunneedflag 为 false,设置初始位置 i=1;
    while i≤n do
        取当前结点度值 nodegree;
        if nodegree 大于给定的度约束值 degreecons then iprunflag=true;
        求解 i 的子结点,结果保存在指针变量 pichild 中;
        while pichild 非空 do
            寻找具有最差属性的子结点 selectchildnode 分枝
            if 收益为大于 0 且 iprunflag 为 false then
                if 检测无冲突 then iprunneedflag=true;
                else if 冲突能有效解决 iprunneedflag=true;
            end if
            if iprunflag or iprunneedflag 为 true then 执行剪接操作,改变剪接标志;
            else break;
            if 剪接插入位置小于当前位置 then 指针回溯;
            取当前位置度值 nodedegree;
            if nodedegree≤degreecons then iprunflag=flase;
        end while
        i=i+1;
    end while
End
```

在剪接过程中,可能遇到的一个关键问题是冲突.在图 1 和图 2 的示例中,不失一般性,假定给定度约束值为 3.由于序号为 6 的结点有 4 条分枝,必须剪去结点 6 的一条分枝.假定已检测到结点 7 代表的分枝具有最差的属性,而结点 7 与除结点 6 以外的最佳关联结点是结点 3,那么将结点 7 代表的分枝插入到结点 3 之后就可以.这一插入过程未引起新的不满足度约束的情形,即不会引起冲突,其操作如图 1 所示.但是,如果结点 7 与除结点 6 之外只与结点 4 结合具有最好的特性,这时将引起新的冲突,因为结点 4 已有 3 条分枝,不能再插入新的分枝,否则引起新的不满足约束的情形发生,此时称插入后引起了新的冲突.其操作如图 2 所示.

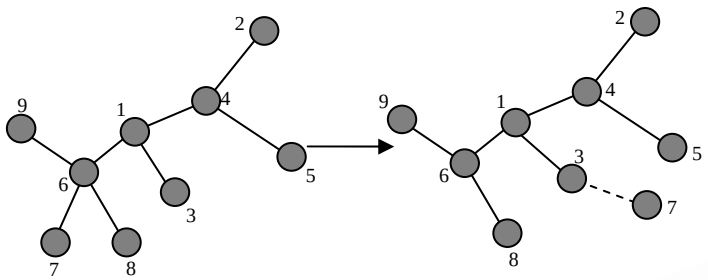

Fig.1 No collision emerges after inserting the branch denoted by node 7 as a new branch of node 3, the new associated edge is denoted by the dash line

图 1 将结点 7 代表的分枝插入到结点 3 之后未引起冲突,虚线为新构建的一条关联边

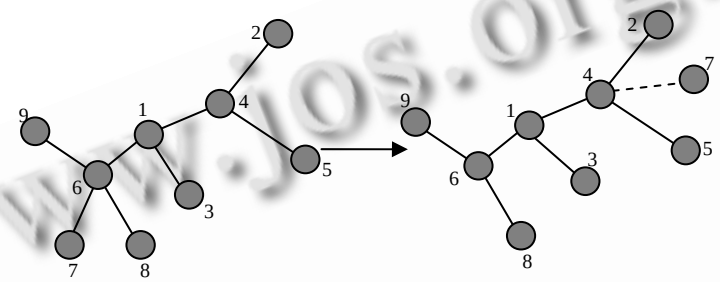

Fig.2 A collision emerges after inserting the branch denoted by node 7 as a new branch of node 4, the new associated edge is denoted by the dash line

图 2 将结点 7 代表的分枝插入到结点 4 之后引起冲突,虚线为新构建的一条关联边

定义 4(**冲突及冲突检测**). 在对消除不满足约束情形进行剪接或对具有较差属性的分枝进行剪接操作,若引起新的不满足约束的情形称为冲突;将发现这一冲突现象的过程称为冲突检测.

定义 5(**冲突解决**).

1) 当冲突出现时,若冲突能在遍历一个或若干个结点后按收益最大化原则完全解决,称冲突能在与问题规模无关的常数内解决,此时冲突解决的时间复杂度为 $O(1)$.

2) 当冲突出现时,若冲突能在遍历所有结点分枝后可按收益最大化原则完全解决,称冲突能在与问题规模相关的线性关系内解决,此时冲突解决的最坏时间复杂度为 $O(n)$.

3) 当冲突出现时,若冲突在遍历所有结点后仍无法按收益最大化原则完全解决,称冲突只能在与问题规模相关的非线性关系内解决,此时冲突解决的时间复杂度为多项式时间或指数时间.

由定义 5 可知,当冲突出现时,若冲突能在 $O(1)$和 $O(n)$时间内解决,称冲突能完全解决.尽管冲突的多项式时间解决和指数时间解决有很大的区别,但冲突的多项式时间解决的循环递归将导致冲突的指数时间解决情形出现,为简化处理过程,称这两种情形为冲突不能完全解决.

定义 6(**冲突的简化解决**). 当冲突出现时,按收益较大化或代价较小化原则寻求在至多与问题规模相关的线性关系时间内解决该冲突的过程,称为冲突的简化解决.

定义 7(**冲突的有效解决**). 当冲突出现时,按收益较大化原则寻求在至多与问题规模相关的线性关系时间内解决该冲突的过程,称为冲突的有效解决.

定义 8(**冲突的退化解决**). 当冲突出现时,按代价最(较)小化的原则寻求在至多与问题规模相关的线性关系时间内解决该冲突的过程,称为冲突的退化解决.

在冲突的解决过程中,使用了一种向前搜索的机制,这种机制能对已确定的边在考虑冲突存在及冲突解决中根据收益和代价进行重新评价,以确定相应的冲突解决方案.

2.3.2 剪接策略

在剪接过程中,由于某些策略与嫁接策略的思想类似,因而对它们仅作简要说明,分别如下:

1) 收益优先剪接策略

对考察结点的所有分枝,若所选分枝移至另一结点位置后收益大于 0 且未引起不能有效解决的冲突,则该分枝可选为剪接枝,执行剪接操作.

2) 双重信息剪接策略

对考察结点的所有分枝,在判断某一分枝是否作为剪接枝时,应将边长信息和位置次序关系信息同时进行分析.

3) 退化剪接策略

当不满足度约束时,若在剪接中出现不能有效解决的冲突,只能按代价最(较)小化原则进行剪接.

2.4 GPOGA体系

在基本遗传算法体系的基础上,结合嫁接和剪接算子,形成 GPOGA 算法,其伪码描述如下:

算法 4(GPOGA).

```
Begin
    随机初始化种群 P(0),t=0;
    计算 P(0)中个体的适应值并按适应值排序;
    while 计算代数 t≤max(gen) do
        j=0;
        while 新产生个体 j<N do
            根据个体的适应值随机从当代种群中选择两个父个体,设为 oldindi1 和 oldindi2;
            执行杂交操作 CrossOver(oldindi1,oldindi2),
            将产生的新个体保存在 newpop[j],newpop[j+1]中;
            执行变异操作 Mutation(newpop[j]),Mutation(newpop[j+1]),
            同时,将两个新个体复制到临时种群变量数组 temp1[j]和 temp1[j+1]中;
            执行嫁接操作 Grafting(newpop[j]),Grafting (newpop[j+1]);
            执行剪接操作 pruning(newpop[j]),pruning(newpop[j+1]);
            计算并评价 temp1[j],temp1[j+1],newpop[j],newpop[j+1],oldindi1 及 oldindi2 的目标值,
            选择两个较好的个体复制到 newpop[j]及 newpop[j+1]中;
            j=j+2;
        end while
        计算新种群 newpop 的目标值、适应值并排序,同时将 oldpop 最好个体替换 newpop 中最差个体;
        将 newpop 种群中个体复制到 oldpop 中,调整其目标值、适应值及顺序;
        t=t+1;
    end while
    记录及输出结果.
End
```

在算法中采用的选择策略称为$(\mu+\lambda+\lambda)$,即将随机选择的两个父个体、由基本遗传算子——杂交及变异后产生的新个体及经嫁接和剪接后产生的新个体共同参与竞争,以选择两个较好的个体进入下一代种群空间.上述算法,根据情形也可采用基于适应值概率的转盘式选择策略.

3 算法分析

3.1 算法的收敛性分析

本文算法中,若考虑遗传算子的作用及精英选择策略,则算法是以概率 1 收敛到所求问题的最优解.

单独从嫁接和剪接算子的作用来看,它们只是在一定程度上加速或减缓算法的收敛速度.考虑其综合效果,嫁接和剪接操作只能是加速算法的收敛速度,否则,得到的解被原有父个体替换而被抛弃.因此,本文算法以概率 1 收敛到问题的全局最优解.详细的证明过程可参考文献[16]的有关定理和结论.

在对 DCMST 问题求解时,由于嫁接和剪接算子均使用 local search 策略以及嫁接和剪接策略的差异,它们在一定程度上可能使算法陷于局部最优;当陷于局部最优时,要跳出这种状态一般会出现冲突,本算法可以有效检测冲突,并提出了若干有效的解决方法.综合考虑本文算法各遗传算子的作用,与没有加入嫁接和剪接算子的情形相比,算法具有更强的寻优能力,同时算法获得问题最好解的概率增大.

3.2 算法的时间复杂度分析

该算法每一代计算主要由杂交、变异、嫁接、剪接及评价 5 部分组成.假定结点数大小为n,种群规模为N,度约束值为d_c,则杂交操作的时间复杂度为$O(N\times n^2)$,变异操作的时间复杂度为$O(N\times n^2)$,嫁接操作的最坏时间复杂度为$O(N\times n^3)$.而在剪接操作中,考虑到树、回溯和冲突解决的最坏情形,其最坏时间复杂度为$O(N\times(n\times(n-1-d_c)\times n\times n))$,而评价的时间复杂度为$O(N\times n\times d_c)$,因此在一代运算中,算法的最差时间复杂度为

$$T(n)=O(N\times n^2)+O(N\times n^2)+O(N\times n^3)+O(N\times n^3\times(n-1-d_c)+O(N\times n\times d_c)$$
$$=O(2\times N\times n^2+N\times n\times d_c+N\times n^4-N\times n^3\times d_c).$$

在本文所讨论的结点规模范围内,N的取值与n无关,度约束值d_c一般为常数 3,因此,

$$T(n)=O(2\times N\times n^2+N\times n+N\times n^4-N\times n^3\times d_c)\approx O(n^4).$$

通过对测试用例计算时间的曲线拟合及分析,可得出本文算法的平均时间复杂度为$O(n^a)$,$2.5<a<3.0$.

我们认为,在算法中由于增加嫁接和剪接算子操作,虽然算法的最坏计算复杂度理论值可达$O(n^4)$.但这一过程没有破坏模式定理和隐含并行性定理,上述两定理在本文算法中的基本遗传算子(杂交与变异算子)及新算子中均发挥作用.所不同的是,由于增加了知识的作用及冲突的解决,使得新算子由一种近似随机的搜索变成一种在知识指导下按一定方向进行的搜索.另一方面,评价算法性能的一个重要指标是评价次数,通过对non-Euclidean和Euclidean两大类测试问题的实验分析,本文算法在得到所求问题相同解精度解的评价次数均远低于相关文献给出的数据.即使考虑在一代中计算代价(或计算时间)的增加量,本文算法对所求问题的求解精度和收敛速度方面都有较大提高.

4 实验及分析

为了验证本文所提算法GPOGA在求解DCMST问题的有效性和正确性,我们在VC环境下采用C/C++编程,某些图形曲线根据计算数据由Matlab绘制输出.在对实验数据进行比较和分析时,本文采用以下几个指标作为对通过算法得到所求DCMST问题的最好解的精度和算法的性能进行评价的依据,它们分别是:评价(次)数(evalutions)[1];最好解偏差(best-gap);平均最好解偏差(average-best gap).

首先,我们对一个经典的 9 结点完全图DCMST问题进行求解,该问题在度为 3 时的最好解为 2 256[1].遗传操作基本参数设置与对比文献[3,9]所给参数设置基本相同.为避免随机性影响,我们进行了 1 000 轮测试.得到的实验结果为:在度约束为 3 时的最好解为 2 256,最小收敛代数为 1,最大收敛代数为 5,平均收敛代数为 1.52;算法在 1 000 轮测试中均收敛到 2 256,即算法得到该问题最优解的概率为 100%.这一测试结果,无论从算法的收敛速度还是算法收敛到最优解的概率均优于相关文献提供的实验数据.

4.1 Non-Euclidean图例DCMST问题测试

我们参照有关文献所提方法设计了一个专用的随机数生成器,该生成器按均匀随机方式产生[10,100]之间

整数作为生成non-Euclidean测试用例图的边长.在本文的测试实验中,GPOGA算法的基本参数设置相同,只是针对不同的测试问题,最大计算代数和计算轮数依不同问题有所不同.其基本参数设置为:种群大小N为 100,交叉概率p_c为 0.4,变异概率p_m为 0.2,嫁接及剪切概率p_g,p_p分别为 1.

测试问题按上述方法生成结点规模为 50~500 之间、间隔为 50 的 non-Euclidean 图例.对结点数为 50~250,最大计算代数 max(*gen*)为 100;当结点为 300 及以上时,max(*gen*)为 20.计算轮数 max(*run*)为 10.

表 1 为利用 GPOGA 求解按均匀随机方式生成结点数为 50~500 的 non-Euclidean 图例问题在度约束值为 3 时的实验数据.表 1 中不带*时间是当结点数为 50~250 程序运行所需时间;带*是结点数为 300~500 所对应的运行时间.

Table 1 Experimental results of GPOGA for solving DCMST problems in uniform random graph instances with nodes from 50 to 500 (a degree constraint at 3)

表 1 GPOGA 求解结点数为 50~500 的均匀随机图例的实验数据(度约束为 3)

n	Prim (without DC)	Best (GPOGA)	Best-Gap (%)	Average-Best gap (%)	Times* (s)
50	610 (6)	622	1.97	2.05	8.7
100	1 044 (6)	1 055	1.01	1.21	48.6
150	1 537 (7)	1 546	0.59	0.74	125.8
200	2 018 (8)	2 020	0.10	0.19	278.7
250	2 505 (10)	2 507	0.08	0.20	526.8
300	2 998 (9)	2 999	0.03	0.07	171.8*
350	3 497 (9)	3 497	0	0.04	266.1*
400	3 997 (9)	3 997	0	0	389.2*
450	4 492 (10)	4 492	0	0	529.1*
500	4 992 (12)	4 992	0	0	690.3*

表 2 为 GPOGA 求解上述测试问题在各种度约束时的实验数据,若当度约束为某一值时已得到与 Prim 算法得到的解相同,则不再计算度约束为其他值的情形.

Table 2 Experimental results of GPOGA for solving DCMST problems with nodes from 50 to 500 in uniform random graph instances at various degree constraints (a degree constraint given in brackets)

表 2 GPOGA 求解 50~500 个结点 DCMST 问题在给定度约束下的实验数据(括号内为对应度值)

n	Prim (without DC)	Best (3)	Best (4)	Best (5)	Best (6)
50	610 (6)	622	616	615	610
100	1 044 (6)	1 055	1 047	1 045	1 044
150	1 537 (7)	1 546	1 538	1 537	
200	2 018 (8)	2 020	2 019	2 019	2 018
250	2 505 (10)	2 507	2 505		
300	2 998 (9)	2 999	2 998		
350	3 497 (9)	3 497			
400	3 997 (9)	3 997			
450	4 492 (10)	4 492			
500	4 992 (12)	4 992			

对 non-Euclidean DCMST 问题的实验数据说明及对比分析如下:

1) 在对结点数为 50~250 的测试问题中,由本文算法得到的最好解偏差最大值为 1.97%,最小值为 0.08%;其平均最好解偏差最大值为 2.05%,最小值为 0.19%.在与同类或类似文献提供具有可比性实验的数据对比表明,由本文算法得到的实验数据在所求解问题最好解的精度和算法的收敛速度上均有一定的优势.

2) 对结点数为 400 及以上的情形,随机生成了间隔为 20 的若干组 non-Euclidean 图例,经过反复测试表明,实验结果表明:随机生成 non-Euclidean 图例问题由 Prim 算法得到的无约束时最佳解对应的度值一般在 10 及 10 以上,本文算法总能找到度约束为 3 时的一棵最好生成树,它对应的目标值与通过 Prim 算法计算得到的值相等.主要原因是:(1) 按上述均匀随机产生分布于[10,100]之间的整数作为 non-Euclidean 图例的边长,各边对应的结点位置并不真实地反映在 Euclidean 空间中,从而结点之间的距离只是一种假想的给定值;(2) 对每个结点,均可能产生与其他结点构成含最小权重的边.以结点数 400 为例,按等概率均匀随机数方式分布于[10,100]之间的整数作为测试图例的边长,将分别期望产生约 877 条边长为 10 和 11 的边,考虑边的无向性,每个结点与其他

结点构成边长为 10 和 11 的边各有 4 条之多.从这些具有较小边长的边中选择 399 条边构成一棵满足度约束为 3 的最佳生成树,使其与 Prim 算法得到的无约束生成树具有相同目标值的概率几乎是 1.理论分析和实验测试得出:对分布在其他区间的均匀随机问题,只要结点数大小达到一定规模,将会出现类似的实验现象.

4.2 Euclidean DCMST问题测试

4.2.1 按均匀随机生成 Euclidean 测例问题

参照文献[9]按均匀随机方式产生[1,10000]之间整数作为生成测试问题结点的坐标值,以结点之间的 Euclidean 距离作为测试用例图的边长,生成结点数分别为 50~500,间隔为 50 的随机问题.GPOGA 算法最大计算代数取值为:当结点大小为 50~300 时,最大计算代数取值为 100,计算轮数 max(*run*)为 20;当结点数大小为 350 及以上时,最大计算代数为 200,计算轮数 max(*run*)为 20.

文献[9]给出在度约束为 3 时,由 branch-and-cut 算法、ABACUS 及 CPLEX8.1 得到的最好解作为检验文献[9]算法找到满意解的基准.在我们的实验方案中,通过参考相关文献,实现了 branch-and-cut 算法.同时,在 Prim 算法基础上,通过边边交换设计实现了一种启发式算法(称为 PrimES).将上述两种算法的最好解作为问题的满意解,以此作为检测本文算法成功找到所求问题解的概率,并计算相应解的平均代数及评价次数.

表 3 为利用 GPOGA 求解按文献[9]方式生成结点数为 50~500 的 Euclidean 图例问题在度约束为 3 时的实验数据.由实验数据可以发现,在每轮计算中,由本文算法求解结点数为 50~200 的问题得到最好解与 Prim's 解的相对误差控制在 0.2%以内.

Table 3 Experimental results of GPOGA for solving DCMST problems in uniform random graph instances with nodes from 50 to 500 (a degree constraint at 3)

表 3 GPOGA 求解结点数为 50~500 的均匀随机生成 Euclidean 图例问题的实验数据(度约束为 3)

n	Prim (without DC)	Best (GPOGA)	Best-Gap (%)	Average-Best gap (%)
50	48 840 (4)	48 842	0.004	0.004
100	68 704 (4)	68 725	0.03	0.03
150	83 505 (4)	83 518	0.016	0.017
200	94 986 (4)	95 072	0.097	0.109
250	102 801 (4)	102 891	0.088	0.089
300	113 094 (4)	113 178	0.074	0.152
350	123 462 (4)	123 482	0.016	0.027
400	133 642 (4)	133 666	0.018	0.035
450	141 313 (4)	141 319	0.004	0.017
500	146 494 (4)	146 517	0.016	0.092

表 4 为 GPOGA 与文献[9]所提方法求解结点数为 50~200 在度约束为 3 时的对比数据.从表 4 对比数据看出,由 GPOGA 得到所求问题满意解的概率与文献提供的数据相当;但本文算法收敛到满意解的平均评价次数均小于文献的实验数据,说明本文算法可以快速得到所求问题的(近似)最优解.

Table 4 Comparison of experimental statistic results between GPOGA and the methods proposed in Ref.[9] for solving Euclidean DCMST problems with nodes from 50 to 200 (a degree constraint at 3)

表 4 GPOGA 与文献[9]所提方法求解结点数为 50~200 的 Euclidean DCMST 问题的对比数据(度约束为 3)

Methods	*n*=50		*n*=100		*n*=200	
	Hits (%)	Evalutions	Hits (%)	Evalutions	Hits (%)	Evalutions
GPOGA	100	1 350	100	4 866	95	5 850
UNIF*	68	57 900	66	360 199	8	990 999
OPTEX*	100	11 698	100	49 088	96	230 771
PROPP*	100	16 877	88	118 942	78	411 953
*N*0.75*	48	149 643	20	432 160	6	967 138
*N*1*	68	96 968	46	314 193	46	715 735
*N*1.5*	94	25 005	82	166 909	64	503 273
*N*2*	100	15 007	90	92 152	76	404 871
*N*3*	98	16 216	94	87 970	66	468 987
INVW*	98	24 980	86	169 252	46	824 540

4.2.2 取自网络资料 TSPLIB 中的 Euclidean 实例问题

为进一步检测算法,我们随机选取网络资源 TSPLIB 中的 8 个 Euclidean 的测试问题,所选的 8 个用例尽量选取不同类型的数据,以避免同一类型数据的内在规律性.对某些包含小数的测例数据,舍去其小数.对 Prim 算法和本文算法得到的生成树对应边长和,为保证计算精度,只将最后计算结果的小数部分舍去.分别为:eil51, pr107, ch150,kroA200,gil262,pr299,lin318 及 pr439.上述 8 个测例对应 Prim 算法最佳解对应的度值均为 4.

GPOGA 算法的基本参数设置同上,对前 6 个测例问题,最大计算代数 max(*gen*)为 100,计算轮数 max(*run*)为 20;对 lin318 和 pr439 问题,max(*gen*)设为 200,max(*run*)均为 10,实验测试及统计数据见表 5.

Table 5 Experimental results of GPOGA for solving the eight DCMST problems of TSP instances from TSPLIB (a degree constraint at 3)

表 5 GPOGA 求解 TSPLIB 中 8 个 DCMST 问题用例的实验数据(度约束为 3)

n	Prim (without DC)	Best (GPOGA)	Best-Gap (%)	Average-Best gap (%)	Iterations at first convergence
eil51 (51)	376 (4)	378	0.53	0.53	12
pr107 (107)	34 757 (4)	34 895	0.40	0.98	79
ch150 (150)	5 880 (4)	5 881	0.02	0.05	61
kroA200 (200)	29 672 (4)	29 675	0.01	0.01	68
gil262 (262)	2 103 (4)	2 106	0.14	0.16	54
pr299 (299)	44 293 (4)	44 340	0.11	0.26	86
lin318 (318)	37 918 (4)	37 925	0.018	0.018	79
pr439 (439)	92 193 (4)	92 205	0.013	0.032	112

图 3 为采用 GPOGA 求解 ch150 问题最好解偏差与平均值偏差随计算代数变化的曲线图.从图可以看出,由于嫁接及剪接控制策略的交互使用及采用的选择策略,平均值偏差会出现一定的波动.测试表明,这种波动不会影响算法对最优解的搜索,也不会对算法的收敛性产生影响.

图 4 为种群中个体的多样性随代数变化的曲线图.为统计方便,本文将具有相同目标值或适应值的个体认为是同一个体.从曲线图可以看出,种群中个体的多样性基本维持在 25%~45%之间.

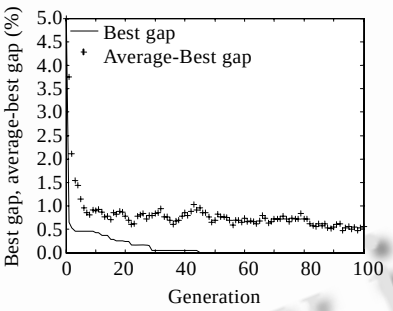

Fig.3 Variation curves of best gap and average gap versus generation using GPOGA to solve the ch150 instance

图 3 GPOGA 求解 ch150 用例其最好解偏差及平均解偏差随代数变化曲线图

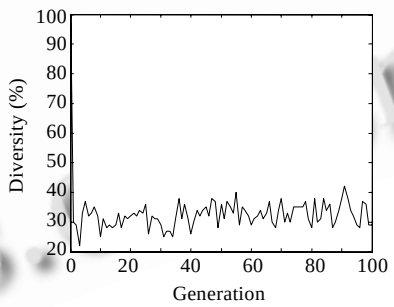

Fig.4 Variation curves of diversity of population individuals versus generation using GPOGA to solve the ch150 instance

图 4 GPOGA 求解 ch150 用例其种群个体的多样性随代数变化曲线图

图 5 为由 GPOGA 得到测例 eil51 在度约束为 3 时的最好解取整后为 378,该值对应的最小生成树并不唯一.图 5 给出两种情形下的最小生成树.图 5(a)对应的边长和为 378.566,图 5(b)对应的边长和为 378.635.

对 Euclidean DCMST 问题的实验数据分析及小结:

1) 最好解偏差被许多文献引用,作为对算法求解 DCMST 问题得到的解的精度进行评价的客观标准,它反映了得到的最好解在给定的度约束时与 Prim 算法得到的无约束解的偏差.文中对随机产生结点数为 50~500 和选取网络资源的 8 个 TSP 数据的 Euclidean 测试问题中,对随机产生的 Euclidean 测试问题,最好解偏差最小值为 0.004%,最大偏差为 0.097%;对随机选取的 TSP 测试数据,最好解偏差最小值为 0.01%,最大偏差为 0.53%.可

见,由本文算法对这两类 Euclidean 问题求解,均能得到较小的下界指标值.

2) 综合算法的基本参数设置可以发现,GPOGA 具有很强的鲁棒性和稳定性:① 对所选取的测试问题,虽然待求 DCMST 问题包含的结点数相差很大,但 GPOGA 的基本控制参数基本保持不变.例如,对种群规模的选取,本文算法对种群大小参数设置一直保持为 100;② 在算法的执行过程中,种群中个体的数量与种群的规模的比值能自适应地保持在一定的范围内.我们曾试图通过控制个体的浓度将多样性维持在 60%左右,但算法的整体性能并未改善甚至有时下降.通过分析我们认为,这种情形的出现与算法中采用的选择策略有一定关系,但更大程度上与嫁接和剪接算子的作用及采用的策略相关.所有这些说明算法具有较强的自身调节和维持平衡的能力,使算法保持了很好的健壮性和稳定性.

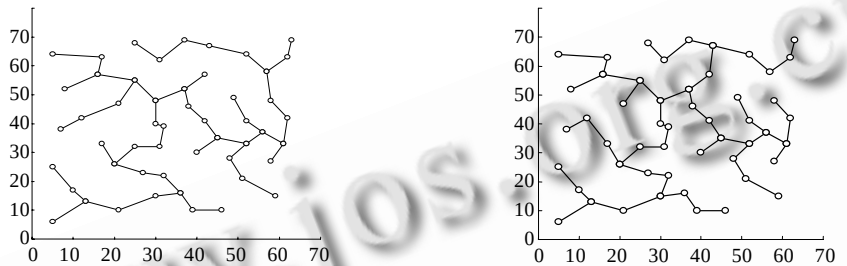

Fig.5 Minimum spanning tree corresponding the best solution using GPOGA for solving the eil51 instance (a degree constraint at 3)

图 5 GPOGA 求解 eil51 的解对应的最小生成树(度约束为 3)

5 总结与展望

度约束最小生成树(DCMST)问题是一类难解的 NP-hard 问题,从对国内外文献的查阅来看,目前尚未有可靠和有效的解决方法.通过对现有方法求解该类问题时存在的不足进行分析,我们发展了基于度排列的编码方法,使其能够充分利用生成树的边权重信息.借鉴花草果树的人工种植和培育技术,设计出从正、反两方面出发加速搜索过程;并对出现的问题及时调整,以激发满足约束的最佳生成树的产生.文中分析了传统 local search 技术的局限性,提出了对 local search 策略由于贪婪导致局部最优情形出现的冲突进行检测,并根据冲突的分类提供若干不同的处理方法.在上述基础上,提出一种新的基于嫁接和剪接算子的遗传算法(GPOGA),通过理论分析和实验数据为算法的正确性和有效性提供了依据.我们下一步的研究重点是针对超大规模 DCMST 问题出现的复杂情形,结合进化计算的最新进展,例如人工免疫进化、知识进化的最新思想,以及结合并行化的算法设计方法设计出更为有效的嫁接及剪接策略,从而为超大规模及复杂情形的 DCMST 问题提供精确和可靠的解决方案.

致谢 在此,谨向对本文评审中提出宝贵建议的匿名审稿专家和对本文中的一些观点提供有参考价值观点的学者表示诚挚的感谢.同时,对编辑部老师的辛勤劳动和默默奉献,这里一并表示诚挚的感谢和敬意.

References:

[1] Gen M, Cheng RW. Genetic Algorithms and Engineering Optimization. Beijing: Tsinghua University Press, 2004 (in Chinese).

[2] Premkumar G, Chu CH, Chou HH. Telecommunications network design-comparison of alternative approaches. Decision Sciences, 2000,31(2):483−506. [doi: 10.1111/j.1540-5915.2000.tb01631.x]

[3] Knowles J, Corne D. A new evolutionary approach to the degree-constrained minimum spanning tree problem. IEEE Trans. on Evolutionary Computation, 2000,4(2):125−134. [doi: 10.1109/4235.850653]

[4] Caccetta L, Hill SP. A branch and cut method for the degree-constrained minimum spanning tree problem. Networks, 2001,37(2): 74−83. [doi: 10.1002/1097-0037(200103)37:2<74::AID-NET2>3.0.CO;2-E]

[5] Andrade R, Lucena A, Maculan N. Using Lagrangian dual information to generate degree constrained spanning trees. Discrete Applied Mathematics, 2006, 154(5):703−717. [doi: 10.1016/j.dam.2005.06.011]

[6] Behle M, Jünger2 M, Liers F. A primal branch-and-cut algorithm for the degree-constrained minimum spanning tree problem. In: Demetrescu C, ed. LNCS 4525. Berlin, Heidelberg, New York: Springer-Verlag, 2007. 379−392.

[7] Jiao LC, Du HF, Liu F, Gong MG. Immunological Computation for Optimization, Learning and Recognition. Bejing: Scienice Press, 2007 (in Chinese).

[8] Zhou GG, Gen M. A Note on Genetic Algorithms for Degree-Constrained Spanning Tree Problems. John Wiley & Sons, Inc., 1997.

[9] Raidl GR, Koller G, Julstrom BA. Biased mutation operators for subgraph-selection problems. IEEE Trans. on Evolutionary Compution, 2006,10(2):145−156. [doi: 10.1109/TEVC.2006.871251]

[10] Soak SM, Corne DW, Ahn BH. The edge-window-decoder representation for tree-based problems. IEEE Trans. on Evolutionary Compution, 2006,10(2):122−144.

[11] Bau YT, Ho CK, Ewe HT. An ant colony optimization approach to the degree-constrained minimum spanning tree problem. In: Hao Y, *et al*., eds. LNCS 3801. Berlin, Heidelberg, New York: Springer-Verlag, 2005. 657−662.

[12] Binh HTT, Nguyen TB. New particle swarm optimization algorithm for solving degree constrained minimum spanning tree problem. In: Ho TB, Zhou ZH, eds. LNCS 5351. Berlin, Heidelberg, New York: Springer-Verlag, 2008. 1077−1085.

[13] Chan TM. Euclidean bounded-degree spanning tree ratios. Discrete Computation Geometry, 2004,32(2):177−194.

[14] Fekete SP, Khuller S, Klemmstein M, Raghavachari B, Young N. A network flow technique for finding low-weight bounded-degree trees. Journal of Algorithms, 1997,24(2):310−324. [doi: 10.1006/jagm.1997.0862]

[15] Aarts EHL, Lenstra JK. Local Search in Combinatorial Optimization. Princeton University Press, 2003.

[16] Baok T. Evolutionary Algorithm in Theory and Practice. New York: Oxford University Press, 1998.

附中文参考文献:

[1] 玄光男,程润伟.遗传算法与工程优化.北京:清华大学出版社,2004.

[7] 焦李成,杜海峰,刘芳,公茂果.免疫优化计算,学习与识别.北京:科学出版社,2007.

王竹荣(1966－),男,湖南衡阳人,博士,副教授,CCF 会员,主要研究领域为智能计算,并行计算,智能优化系统.

张九龙(1974－),男,博士,副教授,CCF 会员,主要研究领域为智能信息处理.

崔杜武(1945－),男,教授,博士生导师,主要研究领域为进化计算,计算机网络.