使用面向离散搜索空间的蛙跳算法求解 TSP

陈嶷瑛[1],李文斌[1],王 舵[2],朱群英[1]
CHEN Yi-ying[1],LI Wen-bin[1],WANG Duo[2],ZHU Qun-ying[1]

1.石家庄经济学院 信息工程学院,石家庄 050031
2.石家庄铁道学院,石家庄 050000
1.Department of Computer Science,Shijiazhuang University of Economics,Shijiazhuang 050031,China
2.Shijiazhuang University of Railway,Shijiazhuang 050000,China
E-mail:mr.cyy@163.com

CHEN Yi-ying,LI Wen-bin,WANG Duo,et al.Using discrete search space-oriented leaping frog algorithm to solve TSP. Computer Engineering and Applications,2009,45(27):50-52.

Abstract: In order to search in discrete search space,a leap-frog algorithm named DSSLFA is proposed.Detailed flow and implementation of this method are discussed in this paper.How to use DSSLFA to solve Traveling Salesman Problem(TSP) is given. The experiment results show that this algorithm is effective and practicable.

Key words: traveling salesman problem;artificial intelligence;leap-frog algorithm

摘 要:针对搜索空间是离散的问题近似求解,提出了一种名为 DSSLFA 的蛙跳算法;给出了该算法的具体流程和实现细节;探讨了将该算法用于求解旅行商(TSP)问题的过程。在若干公用数据集上的实验结果表明,该文算法是有效、可行的。

关键词:旅行商问题;人工智能;蛙跳算法
DOI:10.3778/j.issn.1002-8331.2009.27.016 **文章编号:**1002-8331(2009)27-0050-03 **文献标识码:**A **中图分类号:**TP18

TSP(Traveling Salesman Problem)问题最早可追溯到 1759 年 Euler 提出的骑士旅行问题,目前已成为近代组合优化领域的一个经典的 NP 难题[1]。该问题的有效求解算法对交通运输、线路板设计、物流配送等领域有着十分重要的现实意义。正因为如此,尽管已存在不少求解 TSP 的有效算法,但人们对它的研究一直没有停止。总体上讲,已有算法可分为两类:精确(如:分枝界限、线性规划法、动态规划法等)和近似算法(如:遗传算法[2]、蚁群算法[3]、模拟退火算法[4]等)。前者无法处理城市数较多的 TSP 问题,而近似算法受问题规模的影响不大。尽管后一类算法无法确切地求得最优解,但能获得近似最优解在许多情况下已经足够。

混合蛙跳算法[5](Shuffled Frog Leaping Algorithm,SFLA)是最近才被提出的一种新的元启发式的搜索算法,被用于解决水管网设计,求解管网设计方程的最优解。即,它的设计本意是在连续解空间中搜索近似解。而 TSP 等问题的搜索空间是离散的,无法直接使用混合蛙跳算法进行求解。提出了一种面向离散搜索空间的蛙跳算法(Discrete Search Space-oriented Leap-Frog Algorithm,DSSLFA),给出了用它求解 TSP 的具体过程。在 TSPLIB 公共数据集上的实验表明,该算法是有效的,在收敛速度和解的优越性方面都比较理想。

1 TSP 问题的形式化描述

经典 TSP 问题是在一个带权完全无向图中,找一个权值最小的哈密顿回路。精确描述如下:$G=(V,E)$为带权图,$V=\{1,2,\cdots,N\}$为顶点集(城市集),$E=\{e_{ij}|i,j\in V,i\neq j\}$为边集。TSP 的数学模型可表示如下。

每一个可能的解表示为一个顶点序列 $v_1,v_2,\cdots,v_N,v_1$。其中,$x,y=1,\cdots,N$,$v_x\neq v_y$,$v_x,v_y\in V$,下同。解空间即是所有可能的顶点序列的集合。求解算法需要在解空间中找到一个顶点序列,使下式最小:

$$D(v_1,v_2,\cdots,v_N,v_1)=d(v_N,v_1)+\sum_{x=1}^{N}d(v_x,v_{x+1}) \quad (1)$$

其中,$d(v_x,v_y)$表示边(v_x,v_y)上的权重(即城市 v_x 与城市 v_y 间的距离)。

经典的 TSP 问题中,$d(v_x,v_y)=d(v_y,v_x)$,因此,经典 TSP 问题又称为对称 TSP。当 $d(v_x,v_y)\neq d(v_y,v_x)$,称为非对称的 TSP。无论是对称的 TSP 还是非对称的,求解过程都可以加上

基金项目:国家自然科学基金(the National Natural Science Foundation of China under Grant No.60673015);河北省科技厅项目(No.2001BA201A12);石家庄经济学院博士科研启动基金资助。

作者简介:陈嶷瑛(1971-),女,博士,副教授,研究兴趣:软件工程、GIS、人工智能等;李文斌(1974-),男,副教授,博士,硕士生导师,研究兴趣:人工智能、Web 智能、数据挖掘等;王舵(1982-),男,助教,研究兴趣:数据挖掘、人工智能;朱群英(1970-),女,高级工程师,研究兴趣:计算机应用技术。

收稿日期:2008-11-03 **修回日期:**2009-01-19

约束条件,此时称为带约束的 TSP,如:要求城市 i 必须出现在城市 j 之前。

2　DSSLFA

2.1　SFLA 简介

SFLA 模拟青蛙群体在觅食过程中所体现出的协同行为来完成对问题的求解。这种算法按照族群分类进行信息传递,并将局部信息的交换与全局进化搜索相结合,因此,具有较快搜索全局最优解的能力。在 SFLA 中,每只青蛙代表一个解,都拥有自己的"文化"。整个青蛙群体按一定规则划分为若干个子群体。每个子群体有着自己的文化,独立地执行局部搜索和进化策略。每个子群体进化到一定阶段后,各子群体通过青蛙在群体间的"跳跃"进行文化交流。这种交流使局部文化得以互相传播,从而达到整个群体对全局的了解逐渐趋于统一和稳定。

对 N 维函数优化问题,第 i 只青蛙可表示为 $F_i=<F_{i1},F_{i2},\cdots F_{iN}>$,实际上,$F_i$ 即是第 i 只青蛙的坐标,F_{ij} 表示第 j 维坐标。F_i 的适应度用 $f(F_i)$ 表示,即是青蛙 F_i 所在位置的函数值。随机产生的青蛙初始群体 $F=\{F_1,\cdots,F_M\}$。青蛙子群体间进行若干次文化交流,每次称为一次迭代。在每次迭代中,对各子群体,只对其中的最坏青蛙进行进化,以提高它的适应度。进化的基本思路是:产生一个随机步长(步长的产生受本子群体中最好青蛙乃至整个群体中最好青蛙的影响),若最坏青蛙按此步长移动到新位置后,它的适应度得到提升,则更新它的位置。每次迭代后,都混合所有子群体、更新全局最优解、重新划分子群体,然后进入下一次迭代。迭代终止的条件可根据全局最优解在若干次迭代中的变化情况设定,或强制指定迭代次数。关于 SFLA 的详细描述可参考文献[5-6]。

2.2　DSSLFA 算法及 TSP 求解

对经典 TSP 问题,随机产生的青蛙初始群体 $F=\{F_1,\cdots,F_M\}$。其中,第 i 只青蛙可表示为 $F_i=<F_{i1},F_{i2},\cdots F_{iN}>$,$N$ 为城市数,F_{ij} ($j=1,\cdots,N$)代表某城市的编号,$F_{i1}\neq F_{i2}\neq\cdots\neq F_{iN}$。$F_i$ 代表的可能解为:$F_{i1},F_{i2},\cdots F_{iN},F_{i1}$,其适应度用 $f(F_i)$ 表示,计算式如下:

$$f(F_i)=\frac{1}{D(F_{i1},F_{i2},\cdots F_{iN},F_{i1})} \tag{2}$$

其中,D 的计算见式(1)。

DSSLFA 的算法如算法 1。算法的第 #11 行体现了子群体青蛙间的文化交流,因为,最好的青蛙,影响了最坏青蛙,使它产生位置上的变化。第 #23 行则体现了子群体间的文化交流,因为,子群体混合后,原属于某子群体的某青蛙可能会成为另一子群体中的最好青蛙,从而影响它所在的新子群体的文化。问题的关键在于:F_w^j 的随机漫步问题。即,最差个体如何通过随机漫步移动到新位置。首先,需要理解"移动"和"位置"的概念。从 2.1 节的描述来看,个体编码代表了它所在的坐标位置,个体编码的改变即实现了它的移动。同理,DSSLAFA 中,改变 F_w^j 的编码,即可以认为是 F_w^j 的位置发生了移动,或说青蛙发生了跳跃。F_w^j 编码的改变依赖于 F_b^j(如算法 1 的第 #11 行)乃至 F_g(如算法 1 的第 #16 行)。"移动"F_w^j 到新"位置"的方法如算法 2。

算法 1　DSSLFA 的迭代过程

```
#0   F=随机产生初始种群
#1   DO
#2   {  将 F 中所有青蛙按适应度从大到小排列
#3      找到全局最优的个体 F_g
#4      FOR j=1:m//M 为青蛙总数,m 为群体数
#5         将第 j,j+m,j+2m, …j+(M/m-1)*m 只青蛙放入第 j 个子群中,记为 F^j
#6      END
#7      FOR j=1:m//每个群体独立执行进化和搜索
#8      找到 F^j 中的最优个体和最差个体,分别记为 F_b^j 、F_w^j
#9         Count=0;//记录最差个体被更新的次数
#10        FOR k=1:K//K 为最差个体的进化次数
#11        受 F_b^j 的影响,F_w^j "随机漫步"以产生 F_w^j(k+1)
#12        IF f(F_w^j(k+1))>f(F_w^j(k)) THEN
#13           F_w^j(k)←F_w^j(k+1)
#14           Count=Count+1
#15        ELSE
#16        F_w^j(k)←F_g//最差个体的进化受全局最优解的影响
#17        END
#18     END
#19     IF Count=0 THEN
#20        产生随机个体替代 F_w^j
#21     END
#22  END
#23  混合所有子群体至 F 中 //子群体文化交流
#24  }WHILE(终止条件)
```

算法 2 中,F_b 对 F_w 的影响体现在:按预设规则,在保证是可能解的前提下,随机变化 F_b 中的若干个编码(如算法 2 第 #6、8、10、12、14、16 行)。如果变化后的 F_b 优于 F_w,则将变化后的 F_b 赋给 F_w。从而,使得 F_w 朝着 F_b(甚至比 F_b 更优的解)的方向跳跃。如果 F_b 无法影响到 F_w 的跳跃,则 F_w 的跳跃受全局最好青蛙 F_g 的影响(如算法 1 第 #15~#17 行)。如果 F_w 的跳跃产生了比 F_g 还好的结果,则用该更好的解取代 F_w。这一操作使得 F_w 朝着 F_g(甚至比 F_g 更优的解)的方向跳跃。如果 F_w 和 F_g 都无法影响到 F_w 的跳跃,则 F_w 随机跳跃到另一个解(如算法 1 第 #20 行)。

算法 2　最差个体的进化过程

```
#1   I=rand(N); //随机产生 I,I∈[1…N]
#2   J=rand(N); //随机产生 J,J∈[1…N]
#3   op=rand(6);//随机产生移动方向,op∈[1…6]
#4   SWITCH(op)
#5      CASE 1:
#6         F_w←对 F_b,交换 F_bI 与 F_bJ
#7      CASE 2:
#8         F_w←对 F_b,翻转 F_bI 至 F_bJ 的城市
#9      CASE 3:
#10        F_w←对 F_b,将 F_b(I+1) 至 F_bJ 间的城市前移一个位置,将 F_bI 放在 F_bJ 的位置
#11     CASE 4:
#12        F_w←对 F_b,将 F_bI 至 F_b(J-1) 间的城市后移一个位
```

置,将 F_{bJ} 放在 F_{bI} 的位置

#13 CASE 5:

#14 F_w←对 F_b,将 F_{b1} 到 F_{bI} 间的城市进行翻转

#15 CASE 6:

#16 F_w←对 F_b,将 F_{bJ} 到 F_{bN} 间的城市进行翻转

3 实验结果与分析

该文进行了一系列模拟实验。硬件平台为:Intel Core 2 Duo Processor T5500、1 G 内存。软件平台:Windows XP+Matlab 7.1。数据集来自于 TSPLIB(http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/)。实验参数选取如下:(1)青蛙总数:100;(2)迭代次数:1 000;(3)青蛙子群体个数:20;(4)每个子群体青蛙的进化次数:10。

图 1 显示了 DSSLFA 在 att48.tsp 数据集上的得到的最优解及收敛情况,图 2 则显示了 DSSLFA 在 berlin52.tsp 数据集上得到的最优解及执行结果。目前,在所见的文献范围内,国外研究人员在 att48.tsp 上最新取得的最优解为:10 628(http://www.informatik.uni-heidelberg.de/groups/comopt/software/TSPLIB95/STSP.html)(注:许多文献报导 att48.tsp 的最优解是在 2006 年取得的,即 33 523。最新取得的最优解向前迈进了一大步);在 berlin52.tsp 上最新取得的最优解为:7 542。国内大多数文献在这两个数据集上取得的最优解分别在 33 523 和 7 544 以上。DSSLFA 在 att48.tsp 上取得的最优解为 33 588,在 berlin52.tsp 上取得的最优解为:7 544。这达到或超过了国内部分文献的水平。在 att48.tsp 上离国际最好结果有一定差距,但在 berlin52.tsp 接近国外报导的最好结果。尽管 DSSLFA 并未取得如国际文献报导那样的好结果,但该算法是一个新的尝试,在结果上确实证明了 DSSLFA 是有效和可行的。由于搜索空间是离散的问题繁多,因此,DSSLFA 推广和应用面非常广泛。

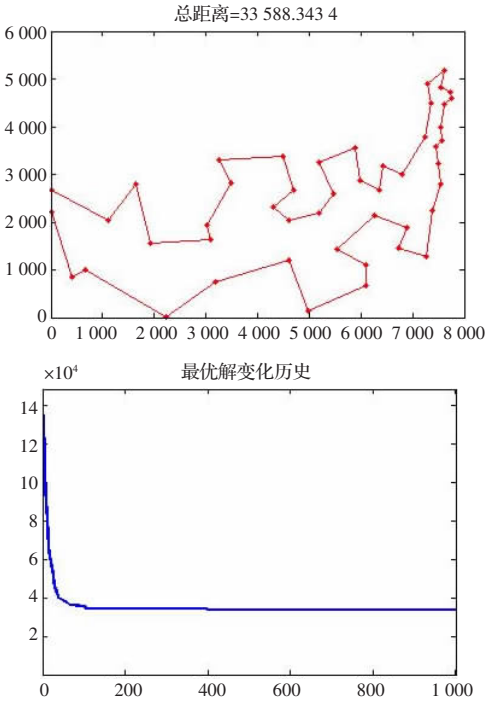

图 1 DSSLFA 在 att48.tsp 上取得的最优解和收敛情况

从图 1 和图 2 可以看出,DSSLFA 的收敛速度很快。基本上在迭代 100 次左右,全局最优解即达到稳定。对每个实验,都

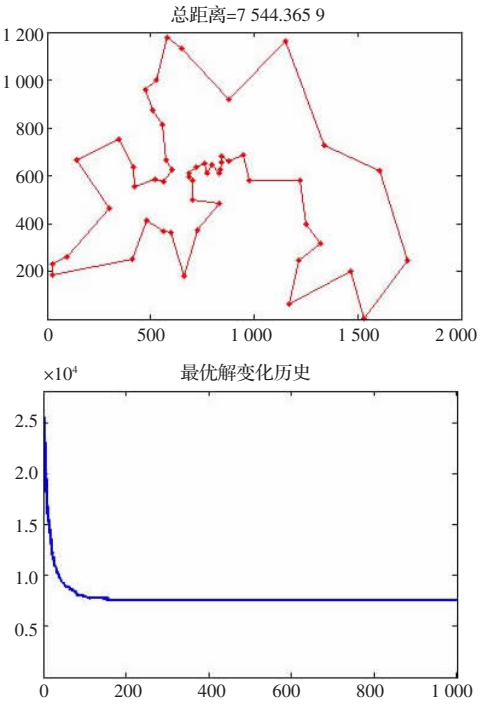

图 2 DSSLFA 在 berlin52.tsp 上取得的最优解和收敛情况

观察了 20 次左右的求解过程,每次取得的最短距离的差距并不是很大。这说明 DSSLFA 的稳定性较好。在 TSPLIB 的其他数据集上也做了实验,基本上,DSSLFA 都取得了接近 TSPLIB 或国内文献报导的结果,充分说明 DSSLFA 是确实可行的。

实验中发现,迭代次数、青蛙子群体的数目对解的影响不大。但子群体中最坏青蛙的"跳跃"次数对解影响较大。此外,算法 2 对解也具有较大影响。可见,局部文化的进化直接影响到了全局文化,即全局解。因此,对算法 2 的改进,是对 DSSLFA 进行性能和效果提升的进一步研究方向之一。参数设置对它的性能变化也是一个研究点。

4 总结

许多经典问题的搜索空间都是离散的,而且是 NP 难问题。近似求解算法是解决此类问题的有力武器。先后诞生的诸如蚁群算法、PSO 粒子群算法、遗传算法等都可用于求解此类问题。混合蛙跳算法于 2003 年首次被提出,并成功用于水管网方程优化求解。受搜索空间是离散的限制,混合蛙跳算法无法直接使用求解诸如 TSP 等问题。提出了一种面向离散搜索空间的蛙跳算法(名为 DSSLFA)。在 TSPLIB 数据集上的验证实验表明,DSSLFA 在解的优越性和收敛性上都具有较理想的表现。将蛙跳算法应用于近似求解是一种新的尝试。因此,论文的意义不仅在于将 DSSLFA 用于求解 TSP 问题上,而是期望研究人员关注这一新的算法。将 DSSLFA 或改进的版本应用于非线性函数优化、下料问题、齿轮问题等是下一步的工作。

参考文献:

[1] 周康,强小利,同小军.求解 TSP 算法[J].计算机工程与应用,2007,43(29):43-37.

[2] 刘烨,倪志伟,刘慧婷.求解旅行商问题的一个改进的遗传算法[J].计算机工程与应用,2007,43(6):65-68.

(下转 122 页)