

文章编号:1001-9081(2008)04-0954-03

用改进的多智能体遗传算法求解旅行商问题

张继军¹, 田宝国¹, 李 萧²

(1. 海军航空工程学院 基础部, 山东 烟台 264001; 2. 63891 部队, 河南 洛阳 471003)

(igi_pp@126.com)

摘 要:多智能体遗传算法是基于智能体对环境感知与反作用的能力提出的一种新的函数优化方法, 具有很快的收敛速度, 尤其是在优化超高维函数时更显示出了它的优越性。针对这一特点对该算法进行了适当的改进, 在邻域正交叉算子中采用精英保留策略, 在自学习算子中引入邻域正交叉算子并采用小变异概率以加快收敛速度。求解 TSP 的实验结果显示, 改进后算法的性能有了较大的提高。

关键词:智能体; 遗传算法; 多智能体遗传算法; 旅行商问题

中图分类号: TP301.6 **文献标志码:** A

Solving TSP with improved multi-Agent genetic algorithm

ZHANG Ji-jun¹, TIAN Bao-guo¹, LI Xiao²

(1. Basic Department, Naval Aeronautical and Astronautical University, Yantai Shandong 264001, China;

2. 63891 Army, Luoyang Henan 471003, China)

Abstract: Based on agent's capability of perceiving and reacting on environment, Multi-Agent Genetic Algorithm (MAGA) was proposed as a new method of function optimization. MAGA had a rapid convergence velocity especially when it optimized super-high dimensional functions. This algorithm was improved properly based on its characteristics: elitist reservation strategy was adopted in neighborhood orthogonal crossover operator, and neighborhood orthogonal crossover operator was introduced into self-learning operator and small mutation probability was adopted to quicken the convergence speed. The results of solving Traveling Salesman Problem (TSP) show that the performance of improved MAGA is enhanced greatly.

Key words: Agent; genetic algorithm; multi-agent genetic algorithm; Traveling Salesman Problem (TSP)

0 引言

多智能体遗传算法^[1]是基于智能体对环境的感知和反作用的能力提出的一种新的函数优化方法, 它的主要特点是种群规模小, 收敛速度快, 算法的稳定性高。利用该算法求解上千、上万维的函数时, 能快速找到高质量的解, 充分显示了该算法的使用价值, 同时也显示出智能体与遗传算法相结合函数优化问题的巨大潜力。

旅行商问题 (Traveling Salesman Problem, TSP) 是近代组合优化领域中的一个典型 NP 难题, 它具有广泛的应用背景和重要的理论价值, 现在已逐渐成为验证算法性能好坏的重要的测试对象。问题本身在概念上很简单: 有 n 个城市, 一个旅行商从某个城市出发, 遍历所有城市后回到出发点, 求一条经过所有城市仅一次的最短遍历路径。为解决该问题出现了许多近似优化算法, 如模拟退火方法、遗传算法、蚁群算法等, 都能获得较好的近似最优解。针对多智能体遗传算法求解效率和稳定性高的特点, 我们对其邻域正交叉算子和自学习算子进行了适当的改进, 并用不同城市数量的 TSP 进行测试, 结果显示改进后的算法具有明显的优势。

1 多智能体遗传算法

1.1 智能体

智能体可以理解为由能够感知环境并反作用于环境的实体, 多个自治的智能体便组成了多智能体系统。一般情况下, 在多智能体系统中每个智能体只有局部感知能力, 数据是分

散的, 没有系统的全局控制。所有智能体都被固定在一个规模为 $L_{size} \times L_{size}$ 的网格 L 上, 每个智能体占一个格点位置, 它通过与邻域中智能体之间的竞争、合作以及自学习来提高自身的能量。如图 1 所示。

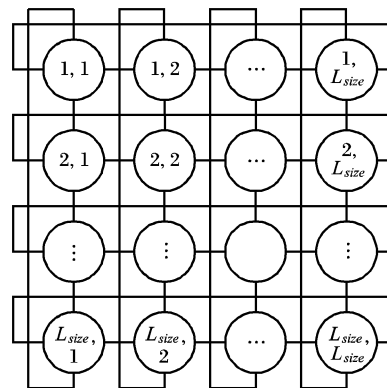


图 1 智能体网格

根据文献^[1,2], 智能体的邻域定义如下: 将位置 (i, j) 的智能体表示成 $L_{i,j}$, $i, j = 1, 2, \dots, L_{size}$, $L_{i,j}$ 的邻域为 $Nbs_{i,j}$, $Nbs_{i,j} = \{L_{i_1,j}, L_{i,j_1}, L_{i_2,j}, L_{i,j_2}\}$, 其中:

$$i_1 = \begin{cases} i-1, & i \neq 1 \\ L_{size}, & i = 1 \end{cases}, j_1 = \begin{cases} j-1, & j \neq 1 \\ L_{size}, & j = 1 \end{cases}$$
$$i_2 = \begin{cases} i+1, & i \neq L_{size} \\ 1, & i = L_{size} \end{cases}, j_2 = \begin{cases} j+1, & j \neq L_{size} \\ 1, & j = L_{size} \end{cases}$$

多智能体遗传算法用于函数优化时基本上都采用实数编码方式对智能体进行编码^[1-3], 而本文将该算法用于求解

收稿日期: 2007-10-26; 修回日期: 2007-12-27。

作者简介: 张继军 (1979-), 男, 山东安丘人, 硕士研究生, 主要研究方向: 系统分析与集成; 田宝国 (1968-), 男, 河北故城人, 教授, 博士, 主要研究方向: 系统分析与集成、人工神经网络、信息融合; 李萧 (1976-), 男, 四川南充人, 硕士, 主要研究方向: 系统对抗。

TSP,采用的是基于路径的编码方式^[4],一个智能体就代表一条路径解。由于一个智能体的染色体编码中不允许有重复的路径符号,所以需要原算法各个算子的细节进行改进。

1.2 智能体进化算子

1) 邻域竞争算子

邻域竞争算子利用邻域的信息实现智能体间的竞争。设算子作用在智能体 $L_{i,j} = (l_1, l_2, \dots, l_n)$ 上, $Max_{i,j} = (m_1, m_2, \dots, m_n)$ 为 $L_{i,j}$ 邻域内能量最大的智能体(l_i, m_i 均为城市代码),若 $L_{i,j}$ 比 $Max_{i,j}$ 的能量大,则可继续存活在网络上;否则必须死亡,空出的格点被按式(1)产生的新智能体占据。其中 p, q 为两个随机位置且 $1 < p < q < n, n$ 为城市个数, $U(0, 1)$ 为均匀分布的随机数, P_o 为占据概率。如果 $U(0, 1) < P_o$, 将 $Max_{i,j}$ 中 p, q 位置上对应的城市代号交换得到新的智能体,放在空出的格点上;如果 $U(0, 1) \geq P_o$, 则将 $Max_{i,j}$ 中 p, q 之间的城市代号反序排列得到新的智能体以取代 $L_{i,j}$ 。

$$New_{i,j} = \begin{cases} (\dots, m_q, m_{p+1}, \dots, m_{q-1}, m_p, \dots), & U(0, 1) < P_o \\ (\dots, m_q, m_{q-1}, \dots, m_{p+1}, m_p, \dots), & U(0, 1) \geq P_o \end{cases} \quad (1)$$

2) 邻域正交叉算子

邻域正交叉算子利用邻域的信息来实现智能体之间的合作。与文献[1]中的邻域正交叉算子有所不同,本文中的邻域交叉算子采用精英保留策略,目的是使种群更快地向最优解空间进化,减少无谓的随机性退化,而由此造成的种群多样性的减少可以通过变异算子进行补偿,后面的实验很好地证明了该策略的正确性。该算子同样作用于 $L_{i,j}$ 和 $Max_{i,j}$ 上,交叉时采用顺序交叉算子,通过该算子的作用产生出两个新的智能体,保留能量值大的智能体;重复上述过程 m ($m < 5$) 次,从中选出能量值最大的智能体 Max 。如果 Max 的能量比 $L_{i,j}$ 的大,则用它替代 $L_{i,j}$;否则不进行替代。顺序交叉算子是通过选择父个体1的一部分并保存父个体2中城市代码的相对顺序生成子个体的,具体操作方法见文献[4]。

3) 变异算子

针对 TSP 的变异算子主要有逆转变异、对换变异和插入变异等,本文采用效率相对较高的对换变异算子。在需要变异的智能体 $L_{i,j}$ 的染色体中随机选择两个位置,交换对应的城市代码产生新的智能体,如式(2)所示,其中 P_m 为变异概率。

$$New_{i,j} = \begin{cases} (\dots, l_q, l_{p+1}, \dots, l_{q-1}, l_p, \dots), & U(0, 1) < P_m \\ L_{i,j}, & U(0, 1) \geq P_m \end{cases} \quad (2)$$

4) 智能体自学习算子

智能体自学习算子可以看成是一个小规模的多智能体遗传算法,它作用于每一代中能量值最大的智能体上,通过学习进一步提高其能量值。为了减少计算量,自学习过程的最大进化代数很小,一般不超过 10 代^[1],所以为了提高自学习过程的收敛速度^[4],我们将邻域正交叉算子也加入到自学习算子中,并取较大的交叉概率($sP_c \geq 0.8$)和较小的变异概率($sP_m \leq 0.1$)。该算子的流程如图2所示,其中, $sGen$ 为智能体自学习最大代数, sL 为自学习网格且 $sL_{size} < L_{size}$, $Eng(\cdot)$ 表示智能体的能量; $sBest^l$ 和 $sCBest^l$ 分别表示到第 l 代为止产生的最优智能体和第 l 代中的最优智能体; sP_m 为自学习变异概率。

在初始化自学习网格时按式(3)进行,其中, $New_{i,j}$ 的产生采用两种策略:设定某一概率值 $0 < sP_r < 1$, 如果 $U(0, 1) < sP_r$, 对 $L_{i,j}$ 进行变异产生 $New_{i,j}$; 否则,随机选择某一城市为起始点,然后按最短路径法产生智能体的染色体编码。这样做是为了让 $L_{i,j}$ 在自学习的过程中能够充分利用自身的相关信息。

$$sL_{i',j'} = \begin{cases} L_{i,j}, & i' = 1 \text{ 且 } j' = 1 \\ New_{i,j}, & \text{其他} \end{cases} \quad (3)$$

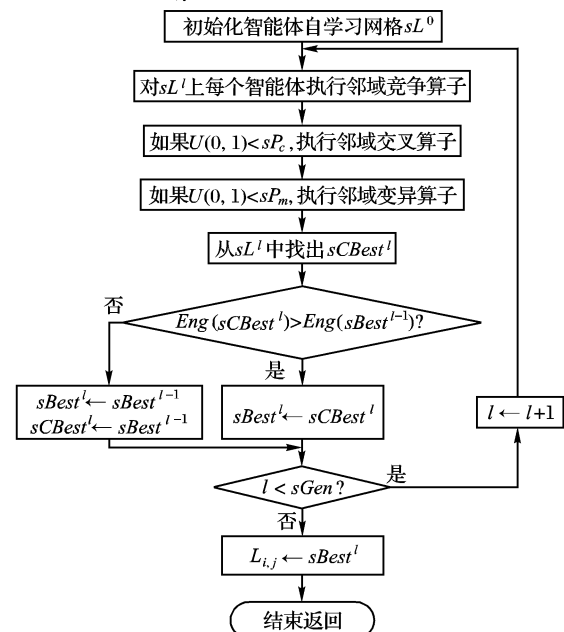


图2 智能体自学习流程

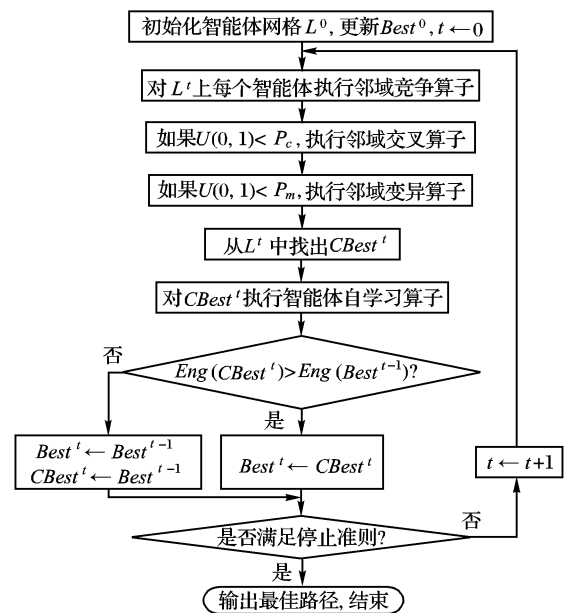


图3 多智能体遗传算法流程

1.3 多智能体遗传算法

将上述四个算子依次作用在智能体上,使每个智能体利用其邻域或自身的信息不断进化以提高自身的能量,从而逐步提高整个网络的能量,使搜索空间快速向最优解附近靠拢。该算法的流程如图3所示。其中, $Best^t$ 和 $CBest^t$ 分别表示到第 t 代为止产生的最优智能体和第 t 代中的最优智能体; P_c 为邻域交叉概率。

2 实验与结果分析

实验采用城市个数分别为 20、31 和 72,考察改进前后算法的求解效率以及与其他算法求解效率的对比。程序运行软件平台为 VC++ 6.0,数据分析采用 Matlab 7.0。

2.1 改进前后算法的求解效率

算法改进前后各参数设置如表1,种群最大进化代数取为 200,程序运行 20 次取平均结果,如表2所示。为方便讨

论,我们将改进前后的算法分别称为算法 1 和算法 2。当城市个数为 20 时,两种算法每次都能求得最优解,但算法 2 的收敛速度明显快于算法 1 的收敛速度;城市个数为 31 时,算法 2 有 3 次求得最优解 15 377.711 333,9 次求得次优解 15 380.515 324,相比之下,算法 1 分别只有 2 次和 5 次;对于 72 个城市的 TSP,算法 2 也明显优于算法 1,平均收敛代数不到算法 1 的二分之一。

表 1 算法改进前后各参数的设置

阶段	L_{size}	P_o	P_c	sL_{size}	$sGen$	sPr	sP_c	sP_m
改进前	8	0.15	0.7	3	10	0.15	0.0	0.15
改进后	8	0.15	0.7	3	10	0.15	0.9	0.05

表 2 算法改进前后求解效率的对比

城市个数	改进前			改进后		
	最优解	平均最优解	平均收敛代数	最优解	平均最优解	平均收敛代数
20	24.522 234	24.522 234 0	21	24.522 234	24.522 234 0	10
31	15 377.711 333	15 428.600 270 0	90	15 377.711 333	15 418.100 340 0	61
72	498.902 757	500.004 821 4	62	498.902 757	499.173 221 8	27

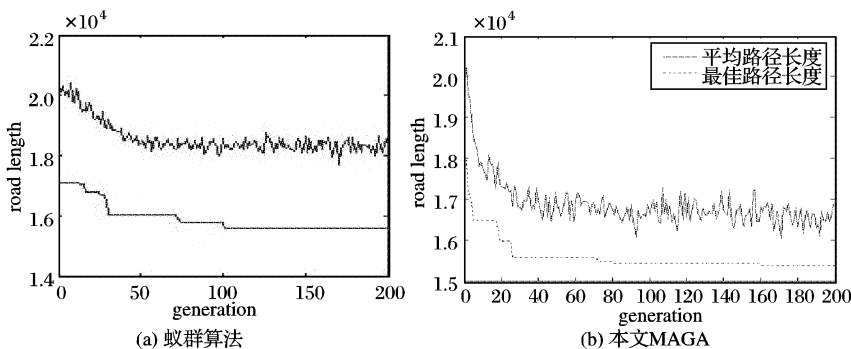


图 4 蚁群算法与本文 MAGA 收敛曲线对比

从结果可以看出,与原算法相比,本文中的多智能体遗传算法在收敛速度和稳定性上都有较大提高,说明在邻域正交叉算子中采用精英保留策略以及在自学习算子中引入交叉算子加快收敛速度的做法是行之有效的。

2.2 与其他算法的对比

在这个实验中,分别将本文中的多智能体遗传算法与 PBIL 算法(城市个数为 72)^[5]和蚁群算法(城市个数为

31)^[6]进行对比,各参数设置同上,结果如表 3 和图 4 所示。从表 3 可以看出,多智能体遗传算法在稳定性上明显好于 PBIL 算法,几乎每次都能找到最优解;而图 4 表明多智能体遗传算法的收敛速度要快于蚁群算法,而且得到的最优解(15 377.711 333)也好于蚁群算法的最优解(15 602)。

表 3 PBIL 算法与本文 MAGA 求解效率对比(城市个数为 72)

指标	PBIL 算法	本文 MAGA
最优解	498.902 9	498.902 757 0
平均最优解	502.783 1	499.173 221 8

3 结语

通过上面的实验分析可以看出,由于多智能体遗传算法借鉴了多智能体系统中智能体之间的竞争、合作和自学习行为,使得它具有很好的稳定性和很高的寻优效率,而改进后的多智能体遗传算法在性能上又有进一步的提高,在求解 TSP 等组合优化问题时明显优于原算法,具有很高的实用价值。

参考文献:

- [1] 钟伟才,薛明志,刘静,等.多智能体遗传算法用于超高维函数优化[J].自然科学进展,2003,13(10):1078-1083.
- [2] 薛志明,钟伟才,刘静,等.用于函数优化的正交 Multi-Agent 遗传算法[J].系统工程与电子技术,2004,26(9):1305-1311.
- [3] 王小平,曹立明.遗传算法——理论、应用与软件实现[M].西安:西安交通大学出版社,2002:123-129.

- [4] 李程俊,张求明.求解 TSP 问题的多线程演化算法[J].计算机工程与设计,2005,26(7):1744-1746.
- [5] 金炳尧.最优化计算中的若干新技术[J].科技通报,2000,16(2):119-124.
- [6] aiwa. 蚁群算法 TSP(旅行商问题)通用 matlab 程序[EB/OL]. [2007-06-20]. <http://bbs.matwav.com/post/view?bid=7&id=529956>.

(上接第 953 页)

参考文献:

- [1] ZWICK L R. Replacement paths and k simple shortest paths in unweighted directed graphs[C]// LNCS 3580. Berlin: Springer, 2005: 249-260.
- [2] COFFMAN T, GREENBLATT S, MARCUS S. Graph based technologies for intelligence analysis[J]. Communications of ACM, 2004, 47(3):45-47.
- [3] CHENY-L, YANG H-H. Finding the first k shortest paths in a time-window network[J]. Computers & Operations Research, 2004, 31(4): 499-513.
- [4] SUBRAMANIAN S, MUTHUKUMAR V. Alternate path routing algorithm for traffic engineering in the Internet[C]// Proceedings of ITCC. Washington, DC, USA: IEEE Computer Society, 2003: 367-372.
- [5] TAKAYUKI G, TAKESHI K, HIROSHI N. On the heuristics of a or A* algorithm in ITS and robot path-planning[C]// Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems. Las Vegas, Nevada: IEEE Press, 2003:1159-1166.

- [6] LI JUN-JIE, ZHANG HAN-YI. A new solution to the k-shortest paths problem and its application in wavelength routed optical networks[J]. Photonic Network Communications, 2006,12(3): 269-284.
- [7] HOCEINI S, MELLOUK A, AMIRAT Y. K-shortest paths Q-routing: A new QoS routing algorithm in telecommunication networks[C]// LNCS 3421. Berlin: Springer, 2005:164-172.
- [8] DIAS L C, CLIMACO J. Shortest path problems with partial information: Models and algorithms for detecting dominance[J]. European Journal of Operation Research, 2000,121(1): 16-31.
- [9] HOFFMAN W, PAVLEY R. A method for the solution of the nth best path problem[J]. ACM, 1959,6(4):506-514.
- [10] 袁红涛,朱美正. K 优路径的一种求解算法与实现[J]. 计算机工程与应用, 2004,40(6):51-53.
- [11] 王明中,谢剑英,陈应麟. 一种新的 Kth 最短路径搜索算法[J]. 计算机工程与应用, 2004,40(30):49-50.
- [12] 张永梅,韩焱,陈立潮. 城市公交查询系统的设计与实现[J]. 计算机应用, 2005,25(2):422-425.