第 34 卷 第 5 期 计 算 机 工 程 2008 年 3 月
Vol.34 No.5 Computer Engineering March 2008
·人工智能及识别技术· 文章编号：1000—3428(2008)05—0174—02 文献标识码：A 中图分类号：TP18

快速求解巡回施行商问题的遗传算法策略

贾丽媛，周翠红

(1. 湖南省城市学院计算机系，益阳 413000；2. 益阳市委党校，益阳 413000)

摘 要：传统的遗传算法收敛速度与问题解的质量是影响算法寻优性能的一对矛盾。该文提出一种新的遗传算法的控制策略——精英子空间算子、变交叉概率P_c和变异概率P_m算子和变维子空间算子。实例计算表明该算法收敛速度快，可以进一步改善遗传算法的性能。

关键词：遗传算法；巡回施行商问题；控制策略

Strategy for Generic Algorithms to Solve TSP Problem Quickly

JIA Li-yuan, ZHOU Cui-hong

(1. Dept. of Computer, Hunan City University, Yiyang 413000; 2. School of Communist Party, Yiyang City, Yiyang 413000)

【Abstract】 The convergence speed of Genetic Algorithm(GA) and the quality of problem result are the main inconsistency which affects the performance of GA. This paper proposes the control strategies of improved GA, which are the good subspace operator, the variable crossover probability and variable mutation probability operator and the variable dimension subspace operator. Experimental results show that the convergence speed of this algorithm is so fast to improve GA.

【Key words】 Genetic Algorithm(GA); TSP problem; control strategy

遗传算法(Genetic Algorithm, GA)[1-2]是基于生物进化中自然选择、适者生存和物种遗传思想的搜索算法，有自组织、自适应和智能性和并行性等特点，不同于传统搜索和优化方法。近年来，遗传算法已经在组合优化、图形处理、通信等很多领域得到了越来越广泛的应用[2]。遗传算法在寻优过程中，存在收敛速度与问题解的质量的矛盾，如何提高遗传算法的性能引起了人们的关注。本文提出了一种改进现有遗传算法寻优性能的综合控制策略。

1 TSP 问题的描述

巡回施行商问题(TSP)，也称货郎担问题，是一个较古老的问题。用图语言来描述 TSP，给出一个图 $G=(V,E)$，每边$e\in E$上有非负权值$W(e)$，寻找G的Hamilton圈C，要使C的总权$W(C)=\sum_{e\in E} w(e)$最小。这是个典型的优化组合问题，已被证明属于NP(Nondeterministic Polynomial)完全问题，即没有确定的算法能在多项式时间内得到问题的解。对于n个城市，可能的路径总数为$n!/2n$。随n的增加，路径数将按数率急剧增长，即“指数爆炸”。以n=50 为例，即使 1×10^8次/s的计算机按穷举法，也需 5×10^{48}年。目前比较好的求解方法有反馈神经网络法和遗传算法。本文用改进遗传算法的综合控制策略求解TSP问题。

2 改进的遗传算法的控制策略

效果好的算法可以使计算效率和解的质量相互平衡。而对收敛速度与解的质量间关系的理解可借助 GA 的选择操作。一方面在人体选择上，高的收敛速度通常通过采用高的选择压力，即给予当前代中的、具有较高适应度值的个体较高的再生概率来实现；另一方面，由于群体规模不变，使得当前代中的具有较低适应值的个体降低了群体中个体的信息含量，而使算法搜索新解区域的能力降低，即降低了算法搜索全局最优解的能力。因此，容易导致早熟收敛。为使算法既有较低的收敛速度，又获得令人满意的解，本文对传统 GA 进行了改进。

2.1 精英子空间算子

在本算子中，群体的K个最好精英个体直接进入M个个体参与下一代的产生，使解的好的信息被充分利用，使算法更快地收敛到最优解。实验表明，采用这种精英保存策略，收敛速度明显加快。对K的选取并不是越大越好，K大时虽然可以更好地利用解的信息，但K越大杂交子空间的基的自由度就越小，容易使解的搜索空间在某个局部空间徘徊，经过试验证明K的取值范围应该是$K\leq M/2$，这里定义$K=M/2$。否则，$T=1$；δ 是正比于 $1/f_{\max}$的常数。

2.2 变维子空间

传统的遗传算法在寻优过程中在当代子空间 M 中随机找一个候选解，然后用最好的个体替代该个体。由于它只是随机地从子空间中找 1 个个体，因此有时会遗漏子空间中比较好的解，而影响搜索的效果和效率。如果随机地从子空间中选取多个个体，用其中最好的个体替代当代群体中最差的个体，其搜索效果会更好。因此，可将传统的遗传算法中的代码：“从当代子空间 M 中随机选取 1 个点 X'”替换为以下代码：

“从当代子空间M中随机选取s 个点X_1', X_2', X_3',…, X_s', $(M/3\leq s\leq M/2)$”

$X'=\arg\min_{1\leq i\leq s} f(X_i')$，然后用最好的个体替代 X'。

2.3 变交叉概率P_c和变异概率P_m算子

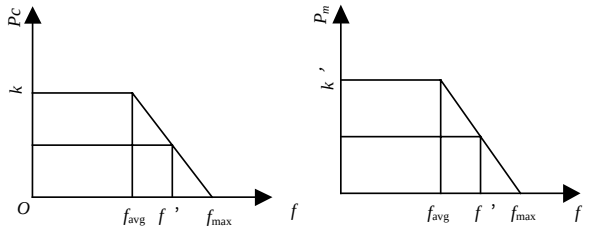
遗传算法的参数中交叉概率P_c和变异概率P_m的选择是影响遗传算法行为和性能的关键所在，直接影响算法的收敛性，

作者简介：贾丽媛(1972－)，女，副教授、硕士，主研方向：遗传算法，人工智能；周翠红，讲师

收稿日期：2007-05-21 **E-mail：**jia_721008@sohu.com

P_c越大，新个体产生的速度就越快。如果P_c过大，遗传算法模式被破坏的可能性就大，从而导致具有高度适应度的个体结构很快被破坏；如果P_c过小，会使搜索过程缓慢，以至停滞不前。如果P_m取值过大，那么遗传算法就变成了纯粹的随机搜索算法；如果P_m过小，就不易产生新的个体结构。针对不同的优化问题需要反复实验来确定P_c和P_m，这一过程很繁琐，且难以找到适于每个问题的最佳值。

本文提出新策略：P_c和P_m能够随适应度自动改变。当种群各个体适应度趋于一致或者趋于局部最优时，使P_c和P_m增加，而当群体适应度比较分散时，使P_c和P_m减少。同时，对于适应值高于群体平均适应值的个体，对应于较低的P_c和P_m，使该解得以保护进入下一代；而低于平均适应值的个体，相对应于较高的P_c和P_m，使该解被淘汰掉。因此，自适应的P_c和P_m能够提供相对某个体解的最佳P_c和P_m。此策略在保持群体多样性的同时，保证遗传算法的收敛性。自适应的交叉概率和变异概率如图 1 所示。



(a)交叉概率 ($k=k_1=k_2$)　　(b)变异概率($k'=k_3=k_4$)

图 1　自适应的交叉概率和变异概率

在自适应遗传算法中，P_C和P_m按如下公式自适应调整：

$$P_c=\begin{cases}k_1(f_{\max}-f')/f_{\max}-f_{\text{avg}} & f\geq f_{\text{avg}}\\ k_2 & f<f_{\text{avg}}\end{cases}$$

$$P_m=\begin{cases}k_3(f_{\max}-f)/f_{\max}-f_{\text{avg}} & f\geq f_{\text{avg}}\\ k_4 & f<f_{\text{avg}}\end{cases}$$

其中，$f_{\max}$是群体中最大的适应度值； f_{avg}是每代群体的平均适应度值；f'是要交叉的两个个体中较大的适应度值；f是要变异个体的适应度值。只要设定k_1, k_2, k_3, k_4的取值在(0,1)区间内，就可以自适应调整以上两个概率。

当适应度值低于平均适应度值时，说明该个体的性能不好，对它就采用较大的交叉率和变异率；当适应度值高于平均适应值时，说明该个体性能优良，对它就根根据其适应度值取相应的交叉率和变异率。可以看出，当适应度值越接近最大适应度值时，交叉率和变异率就越小；当等于最大适应度值时，交叉率和变异率的值为 0。这种调整方法对于群体处于进化后期比较合适，但对于进化初期不利，因为进化初期群体中的较优的个体几乎处于一种不发生变化的状态，而此时的优良个体不一定是优化的全局最优解，这容易使进化走向局部最优解的可能性增加。为此，可以做出进一步的改进，使群体中最大适应度值的个体的交叉率和变异率不为 0，分别提高到P_{c1}和P_{c2}，这就相应地提高了群体中表现中优良的个体交叉率和变异率，使得它们不会处于一种近似停滞不前的状态。为了保证每一代的优良个体不被破坏，采用精英选择策略，使它们直接复制到下一代中。

经过上述改进，P_C和P_m计算表达式如下：

$$P_c=\begin{cases}P_{c1}-(P_{c1}-P_{c2})(f'-f_{\text{avg}})/(f_{\max}-f_{\text{avg}}) & f'\geq f_{\text{avg}}\\ P_{c1} & f'<f_{\text{avg}}\end{cases}\qquad(1)$$

$$P_m=\begin{cases}P_{m1}-(P_{m1}-P_{m2})(f_{\max}-f)/(f_{\max}-f_{\text{avg}}) & f\geq f_{\text{avg}}\\ P_{m1} & f<f_{\text{avg}}\end{cases}\qquad(2)$$

其中，P_{c1}=0.9；P_{c2}=0.6；P_{m1}=0.1；P_{m2}=0.001。

3　旅行商问题(TSP)的改进遗传算法求解

3.1　算法描述

以中国施行商问题(CTSP)[1]作为需解决的问题，按以下步骤进行求解：

(1)参数选择，种群大小*POPSIZE*，杂交概率P_C和变异概率P_m根据式(1)、式(2)获得。

(2)编码，直接采用城市在路径中的相对位置来表示。例如，染色体 *R*=0534678912，它表示从城市 0 出发，先达到城市 5，再到城市 3，……，然后到城市 1，最后从城市 2 返回。

(3)确定适应值函数 $f(x)$，取每条旅行路线的总行程为适应值，则适应值越大，方案越优。

(4)确定初始化群体，先用贪婪法确定一部分个体，经过贪婪法产生的个体在一定程度上使得初始群体具有有效的基因模式，有利于算法提高寻优能力，剩余的个体则随时机产生。

(5)杂交算子，采用刘海等提出的改进的遗传交叉算子[3]，此交叉算子可以使交叉后的子代保证可行的同时，很好地继承父代的优秀基因。

(6)变异算子，从种群中随机选择一个染色体，按一定的变异概率P_m进行基因变异。

3.2　算法流程

算法流程如下：

BEGIN

Initialize Population P={ $X_1, X_2, \cdots, X_M$ }, where $X_i \in D$;

generation = 0;

$X_{best} = \arg\min_{1\le i\le M} f(X_i)$

$X_{worst} = \arg\min_{1\le i\le M} f(X_i)$

WHILE $f(X_{best})>f(X_{worst})$do

{

Sort the M individuals by their fitness

Select K individuals with the best fitness, and select M-K individuals randomly from P, to form the M parents for next generation: $X_1', X_2', \cdots, X_M'$;

Adjust their P_C and P_m according to individuals' fitness, Then crossover and mutate individuals to generate a new population for this generation:

$x_1'', x_2'', \cdots, x_M''$,

$X = \arg\max_{1\le i\le M} f(X_i'')$

Select s individuals randomly from $x_1'', x_2'', \cdots, x_M''$, $X' = \arg\min_{1\le i\ge s} f(X_{i'})$

IF better(X, X') THEN X' = X;

generation = generation + 1;

$X_{best} = \arg\min_{1\le i\le N} f(X_i)$;

$X_{worst} = \arg\max_{1\le i\le N} f(X_i)$;}

Output the result X_{best};

END

Output the result X_{best};

3.3　仿真结果

本文用VC＋＋对中国旅行商问题CTSP[1](城市规模为

（下转第 181 页）