比赛项目排序问题的贪婪算子遗传算法

许盛强[1],屈小波[2],谢国富[3]

(1.厦门大学 数学科学学院信息与计算数学系,福建 厦门 361005;
2.厦门大学 信息科学与计算机学院电子工程系,福建 厦门 361005;
3.厦门大学 软件学院,福建 厦门 361005)

摘 要:将比赛项目的排序问题转化为图论问题中的货郎担问题(TSP),利用 TSP 较为成熟的遗传算法进行求解。这样防止了搜索过程陷入局部最优。针对遗传算法收敛速度慢的特点,对遗传算法进行了改进,引入贪婪交叉算子来加快算法的收敛速度,得到冲突总人次数为 8 的优良结果。在对算法进行合理性分析时,从理论上论证了算法的优劣。

关键词:比赛项目;排序问题;图论;TSP;遗传算法;贪婪算法

中图分类号:TP312 **文献标识码**:A **文章编号**:1672-7800(2007)02-0115-03

1 问题重述

在各种运动比赛中,为了使比赛公平、公正、合理地举行,一个基本要求是:在比赛项目排序过程中,尽可能使每个运动员不连续参加两项比赛,以便运动员恢复体力,发挥正常水平。

假设共有 61 个比赛项目,1050 人参加比赛,要求使连续参加两项比赛的运动员人次尽可能的少。建立此问题的数学模型,给出算法及结果;表中"#"号位置表示运动员参加此项比赛。例表只列出了 14 个比赛项目,15 人参加比赛(原始数据表格见 2005 年"中国电气工程协会杯"B 赛题),见表 1。

2 建立模型及求解

2.1 问题分析

此题可转化为图论问题中求最小哈密尔顿通路,这与货郎担问题(TSP)有很大相似之处,我们将进行如下分解:

表 1 某小型运动会的比赛报名

运动员 \ 项目	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1		#	#						#				#	
2								#			#	#		
3		#		#						#				
4			#					#				#		
5											#		#	#
6					#	#								
7												#	#	
8										#				#
9		#		#						#	#			
10	#	#		#			#							
11		#		#									#	#
12								#		#				
13					#					#				#
14			#	#				#						

把 61 个项目分别看成无向图中 G 的 61 个点(Vertix),其中每两个项目之间都可能连续排列,所以,无向图中任意两点之间有连线。假设第 i 项目与第 j 项目连续排列在一起,无向图 G 中点 v_i 与点 v_j 有一条边 $e[v_i,v_j]$使它们相连。由于每个项目都可能和其它项目连续排列在一起,这样得到一个 61×61 的权矩阵 W_{ij}。

2.2 模型建立

由图论的知识,知即为寻找一条不重复经历点且遍历 61 个点的路径,使所经历的边的权之和最小,即搜索整数子集 x=(1,2,…,n)的一个排列 $\pi(x=v_1,v_2,\cdots,v_{61})$使 $T_e=\sum_{i=1}^{n-1} e[v_i,v_{i+1}]$达到最小值,即求得

$$\min T_e=\sum_{i=1}^{n-1} e[v_i,v_{i+1}]$$

借鉴 TSP 成熟的理论和算法,我们进行转化:先按照 TSP 问题的求解一个圈的总权最小,然后在这个总权最小的圈中截断这个圈中边的权最大的边,定义目标函数为:

$$f(e[vi,vi+1])=\min\sum_{i=1}^{n-1} e[v_i,v_{i+1}]-\max(e'[v_k,v_{k+1}])$$

其中 $\max(e'[v_k,v_{k+1}])$代表在最优哈密尔顿回路中权最大的边的权值。

由此可得模型 II: $F=\max(e'[v_i,v_{i+1}])$

2.3 模型的求解

2.3.1 遗传算法引入

(1)总体思想:借鉴了达尔文的物竞天演、优胜劣汰、适者生存的自然选择和

作者简介:许盛强(1985-),男,福建人,厦门大学本科生,专业为信息科学计算学数学;屈小波(1984-),男,四川人,厦门大学硕士研究生,研究方向为光电通信;谢国富(1985-),男,江西人,厦门大学硕士研究生,研究方向为软件开发与应用。

自然遗传的机理，本质是求解问题的高效并行全局搜索方法。

(2) 个体编码：采用以遍历比赛项目的次序排列编码的方法，每一个体 P_i 的码串形如 $C_1C_2\cdots C_n$，其中 C_i 表示遍历项目的序号，程序中个体定义为一维数组。如码串 1、2、3、...、61，表示从项目 1 开始，依次进行项目 2、3、...、61，最后遍历所有的点。

(3) 适应度函数：我们构造基于序的适应度函数。它的特点是个体被选择的概率与目标函数的具体值无关。将种群中的所有个体按其目标函数值的大小进行降序排列，设参数 $\alpha\in(0,1)$，定义基于序的适应度函数为：

$eval(k_i)=\alpha(1-\alpha)^{i-1}, i=1,2,\cdots,P_{size}$

式中 k_i 为种群排序后的第 i 个体，p_{siae} 为种群个体总数，α 取为 0.1 到 0.3 有利于保持群体的多样性。

(4) 选择机制：采用比例选择算子。该算子是一种回放式随机采样的方法，以旋转赌轮 p_{siae} 为基础，每次旋转都可选择一个体进入子代种群。父代个体 k_i 被选择的概率 p_s 为：

$$p_s=\frac{eval(k_i)}{\sum_{i=1}^{P_{size}} eval(k_i)}$$

(5) 种群交叉。①常规交叉方式。随机选择两个不相同的交配位，后代在这两个交配位之间继承双亲在这两个交配位之间的基因：

如父 A 1 2 3 | 4 5 6 7 | 8 9 10
父 B 4 7 8 | 3 2 5 9 | 1 6 10
子 A 8 3 2 | 4 5 6 7 | 9 1 10
子 B 1 4 6 | 3 2 5 9 | 7 8 10

通过这种方式编程发现，该算法收敛速度较慢，短时间能难以得到比较优的解，这成为贪婪交叉方式考虑的出发点。

②贪婪交叉方式(Greedy Crossover)。贪心交叉算子选择父代的第一个点，然后在双方父代中对比剩下的点，选择距离较近的点，继续搜索路径，如果该点已在经历过，则选择另一父代的点，如果两个点都出现过，则随机生成未选择过的点作为下一个目标点。

贪心交叉算子是为了充分利用染色体的局部信息指导遗传进化搜索。对 TSP 编码是一个循环圈，因此，从任意一个点选择开始贪心交叉操作都是可行的，经过贪心交叉的个体局部性能会有所改善，但个体性能不一定能得到提高，如果个体性能提高就替换父代个体，否则，个体不进行替换。

(6) 种群变异：变异操作的主要目的是改善算法的局部搜索能力，并维持体的多样性，防止出现早熟现象。①方法一：取两个不同的随机数，对这两个数确定的基因区间进行随机排序；②方法二：在染色体的 2- opt 邻域中随机取出一个染色体，2 - opt 邻域搜索优化算法是求解 TSP 问题的常用启发式算法，能有效消除边交叉现象。对 Hamilton 回路中的顶点按经过的顺序以自然数编号，任意两点 v_i、v_j 间边的权重记为 w_{ij}。对于 2- opt，如果 $w_{ij}+w_{i+1,j+1}<w_{i,i+1}+w_{jj+1}$，则以边 $e[v_i,v_j]$、$e[v_{i+1},v_{j+1}]$ 代替边 $e[v_i,v_{i+1}]$、$e[v_j,v_{j+1}]$。

(7) 适应度评估检测：主要针对适应度函数进行检测，限制遗传代数 GEN NUMBER≤GEN_STOP_NUMBER，即遗传代数大于 GEN_STOP_NUMBER 时，遗传结束。

2.3.2 标准遗传算法流程

(1) 编码。

(2) 初始群体的生成。

(3) 适应度评估检测。

(4) WHILE<未满足迭代终止条件> DO。①选择；②交叉；③变异；④适应度评估检测。

(5) END DO。

通过 C++编程得到 minT=8。

重合度最小为 8，按照所得项目排序方式，将有 8 人次出现冲突。

3 算法的合理性分析

为评价算法的合理性，我们引入信息学中的熵，熵的定义如下：

已知单符号离散无记忆信源的数学模型

$$\begin{pmatrix} X \\ P(X) \end{pmatrix}=\begin{Bmatrix} x_1 & x_2 & \cdots & x_i & \cdots & x_n \\ p(x_1) & p(x_2) & \cdots & p(x_i) & \cdots & p(x_n) \end{Bmatrix}$$

其中，$0\le p(x_i)\le 1(i=0,1,2,\cdots,n)$，且 $k=\sum_{k=1,2,3} p(x_i)=1$，信源各个离散消息的自信息量的数学期望(即概率加权的统计平均值)为信源的平均信息量，一般称为信源的信息熵，简称熵，记为 H(X)。

$$H(X)=-\sum_{k=1,2,3} p(x_i)\log_2 p(x_i)$$

在遗传算法的搜索过程中，每一代的种群可以看作一个单符号离散无记忆信源.通过统计计算出每个个体在群体中所占的百份数，根据信息熵公式可以计算出当前代种群的信息熵.信息熵一旦确定则相应的信源随之确定.当信源的信息熵为零时，表明随机变量已经失去了随机性变成了确定量.换句话说，信源虽然有很多消息，只有一个消息必然出现.此时种群中的个体具有唯一性，遗传算法不可能再搜索到其他解.对于任意初始的无限种群，按适应值选择算子重复进行，最终可以实现最大的 i 个体有较高的概率. 即只在选择算子的作用下，遗传算法最终收敛到初始种群中的最优个体.不管初始种群给出什么样的分布，在变异算子的重复作用下，其极限分布都是均匀的，即变异把整个个体空间作为搜索空间.当种群的信息熵很小时，遗传算法已经不具备进化能力，在这个阶段并不期望算法找到更优解.由于变异算子的存在，使种群的信息熵为 0 具有随机性，当种群的信息熵小于某一极小值时，同样可以判断遗传算法的截止代数。

求出 4 个表格中各自的信息熵如表 2 所示

表 2 信息熵

重合度(n)	9	9	8	8
信息熵(H(X))	2.4%	2.7%	0.65%	0.43%

(计算结果基于原始数据表)

由以上信息熵作为截止代数，定义重合度的置信度为：

$$1-\overline{H_n(X)}$$

其中 $\overline{H_n(X)}$，表示重合度为 n 的信息熵的平均值，所以

重合度为 9 的置信度为：

$$1-\overline{H_9(X)}=1-\frac{2.4\%+2.7\%}{2}=97.45\%$$

重合度为 8 的置信度为：

$$1-\overline{H_8(X)}=1-\frac{0.65\%+0.43\%}{2}=99.46\%$$

重合度为 8 的置信度比重合度为 9 时大，基于此，将重合度为 8 作为最优解。由于置信度大于 99%，所以，采用遗传算

法求解结果是“最好解”，这也说明用遗传算法是合理的。

4 模型评价与改进

在交叉算子中引入贪婪算法，既避免了贪婪算法导致局部最优而全局不是最优，又加快了遗传算法的收敛速度，快速求到较优解。由于遗传算法是基于计算机运算的随机性，所以，可能有时得到较优解时间会稍长。

为避免运动员连续参加3个项目，即连续产生2次冲突，可对遗传算法做如下改进：

(1)对于连续产生两次冲突的个体，应在遗传算法选择时，使其适应度降到一个很低的数，使其不容易产生后代。

(2)对于不连续的两次冲突，也在遗传算法选择时选择，使其适应度降到一个比较低的数，使其不是很容易产生后代。

5 对引入贪婪算法的遗传算法的几点补充

5.1 权矩阵的求解方法

```
for i=1:14;
    for j=1:14;
        Ai=data(:,i);
        Aj=data(:,j);
        Bi=Ai';
        F(i,j)=Bi*Aj;
    end
end
```

5.2 求出排列顺序之后的重合度测试

```
a=[13 40 0 18 59 7 58 27 16 3 24 52 56 15 50 23 28 26 12 31 43 1 60 57 29 36 21 41 6 44 35 49 2 9 20 11 30 38 32 46 54 4 19 5 14 25 42 53 10 55 48 45 34 8 17 39 51 37 47 33 22];
b=[0];
for i= 1:61;
    a(i)=a(i)+1;
end
for i=1:60;
    Ai=data(:,a(i));
    Bi=data(:,a(i+1));
    Cj=Bi';
    b(1)=b(1)+Cj*Ai
end
```

5.3 引入贪婪算法的遗传算法主程序

```
#include<stdio.h>
#include<vector>
#include<math.h>
#include<string.h>
#include<algorithm>
#include<time.h>
using std::vector;
#define NUMBER_OF_PROJ 61
#define POP_PROJNUMBER_RATE
typedef struct
{
int pop[NUMBER_OF_PROJ];
int total;
double fitness;
}PROJ_POP;
int proj_r[61][61];
vector<PROJ_POP> pop_space;
typedef std::vector<PROJ_POP>::iterator POP_ITER;
#define CROSSOVER_P
0.3                    //交配概率
#define MUTATION_P
0.1                    //变异概率
#define EVAL_BASE
0.5                    //基于序评价基数
#define POP_CITYNUMBER_RATE
5          //群体个数与城市个数之比
#define FITNESS_MODE
1                  //评价函数计算方式
#define CROSSOVER_MODE
2                      //交叉方式
#define MUTATION_MODE
1                      //变异方式
#define TOTAL_GENERATION
5000                   //迭代次数
#define TRUE 1
#define FALSE      0
long g_NowGenNumber = 0;
//当前第几代
void main()
{      int nowiter = 0;
char strbuf[1000];
int min=1000000;
open_file();
form_initial_pop();
FILE * fp=fopen("result.txt","w+");
    printf("计算中...\n");
    while(nowiter >= 0 )
    {
        nowiter=one_iter_GA_compution();
        if((nowiter & 0x3F)==0)
        {
    int r=get_min_from_pop(strbuf);
            if(min>r)
            {
                min=r;
    fprintf(fp,"重合度:%d  第%d代  排列:",r+1,nowiter);
                fputs(strbuf,fp);
                fflush(fp);
            }
            printf ("  计算到%d代 \n", nowiter);
        }
    }
    fclose(fp);
    }
```


参考文献：

[1] Bernard Kolman,Robert C.Busby,Sharon Cutlwr Ross.离散数学结构.第四版[M].北京:高等教育出版社 &Pearson Education 出版集团，2001.

[2] 陈国良，王煦法，庄镇泉，王东生.遗传算法及其应用.第一版[M].北京:人民邮电出版社,1996.

[3] 穆艳玲，李学武，赵杰修.遗传算法中截止代数的判定[J].天津师范大学学报(自然科学版)，2005,(1).

[4] 姜启源，谢金星,叶俊.数学模型[M]. 北京:高等教育出版社,2005.

[5] 吴建国.数学建模案例精编[M].北京:中国水利水电出版社,2005.

[6] 钱颂迪.运筹学.第二版[M].北京.清华大学出版社，1990.

[7] 李贤平.概率论基础[M].北京:高等教育出版，1997.

[8] 施阳，李俊等.MATLAB 语言工具箱.第一版[M].西安.西北工业大学出版社，1998.

(责任编辑:曙　光)