**人工智能原理实验**

**实验三 遗传算法求TSP问题**

**组长：**凌翔鸿（学号3119000854）

**组员：**林方泽（学号3119000980）、邱俊文（学号3119001159）、莫嘉楹（学号3219001092）、吴泽鑫（学号3119001082）

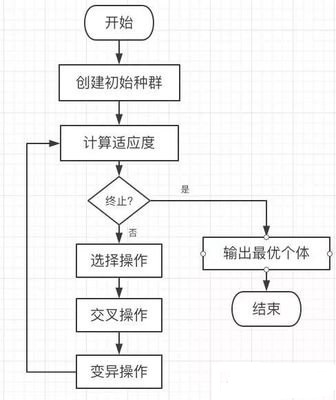
**指导老师：**高怀恩

**一、实验目的：**

熟悉和掌握遗传算法的原理，流程和编码策略，理解求解TSP问题的流程并测试主要参数，对结果的影响，掌握遗传算法的基本实现方法。

**二、实验内容：**

（1）用遗传算法求解不同规模（如10个城市、20个城市、100个城市）的TSP问题，其流程图如下：



（2）需要补充的函数如下：

1. function fitness=calFitness(trajLength) 。%计算各个染色体的适应度

fitness = 1./trajLength;

end

1. function newpop = chooseNewP(pop,fitness) %个体选择函数

n = length(fitness); % 种群个体数

fitness\_sum = sum(fitness); % 适应值之和

p = zeros(1,n); % 个体选择概率

% 个体选择概率分配

for i = 1:n

if i == 1

p(i) = fitness(i)/fitness\_sum\*100;

else

p(i) = fitness(i)/fitness\_sum\*100+p(i-1);

end

end

% 个体选择

newpop = pop;

newpop(:,:) = 0;

s = randi([0,100],1,n);

for i = 1:n

for j = 1:n

if s(i) > p(j)

newpop(j,:) = pop(j,:);

break;

end

end

end

newpop = pop;

end

1. function cpop = crossPop(pop,pc,fitness,SelfAdj) %交叉函数

[N,nc]=size(pop);

list = randperm(N); %产生包含整数1~N的随机矩阵

%\*\*\*\*\*\*\*\*进行随机交叉\*\*\*\*\*\*\*\*%

for i=1:N/2

rval = rand; %产生0~1的随机数，用于判断是否发生交叉

j = i+N/2;

if rval > pc

continue;

end

partner1 = pop(list(i),:);%父代

partner2 = pop(list(j),:);%子代

[child1,child2] = genecross(partner1,partner2); %交叉操作，产生子代child1，child2

pop(list(i),:)=child1;

pop(list(j),:)=child2;

end

cpop = pop;

end

1. function mpop = mutPop(pop,pw,fitness,SelfAdj) %变异函数

[N, nc] = size(pop);

Fmax = max(fitness); % 得到最大适应度

Favg = mean(fitness); % 得到平均适应度

k1 = pw;

for i = 1:N

X = rand; % 随机产生一个数赋给x

if SelfAdj == 1

if fitness(i) > Favg

pw = k1 \* (Fmax-fitness(i))/(Fmax-Favg); % 课本Pm公式

else

pw = k1;

end

end

if X > pw

continue;

end

Z = randi(nc-1)+1;

G = randi(nc-1)+1;

Y = pop(N, Z);

pop(N, Z) = pop(N, G);

pop(N, G) = Y;

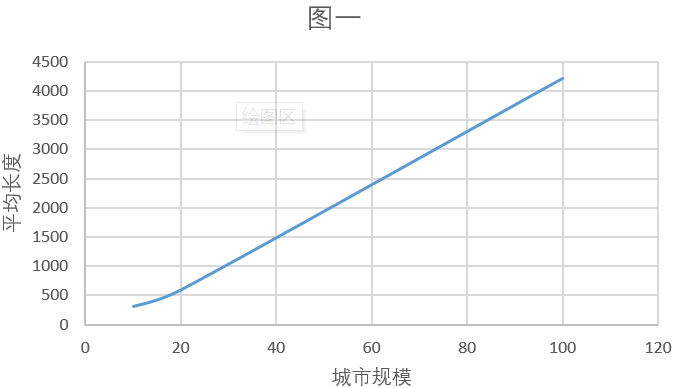
end

mpop = pop;

end

（3）实验结果（运行10次取平均值）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表格一** | | | | |
| **城市规模** | **最短长度** | **最长长度** | **平均长度** | **平均运行时间（秒）** |
| **10** | **298.8325** | **307.4098** | **300.5480** | **4.039** |
| **20** | **557.0279** | **649.7967** | **586.858** | **10.023** |
| **100** | **4054.8808** | **4313.2815** | **4217.0068** | **12.9394** |



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **表格二** | | | | | | |
| **城市规模** | **交叉概率** | **变异概率** | **最短长度** | **最长长度** | **平均长度** | **平均运行时间（秒）** |
| **10** | **0.85** | **0.15** | **298.8325** | **307.4098** | **300.5480** | **4.039** |
| **20** | **0.85** | **0.15** | **557.0279** | **649.7967** | **586.858** | **10.023** |
| **100** | **0.85** | **0.15** | **4054.8808** | **4313.2815** | **4217.0068** | **12.9394** |
| **100** | **0** | **0.15** | **4555.9216** | **4692.6046** | **4625.4563** | **0.3966** |
| **100** | **0.5** | **0.15** | **4031.6271** | **4393.7073** | **4201.3776** | **4.3626** |
| **100** | **1** | **0.15** | **4033.3385** | **4186.8961** | **4091.4163** | **8.4753** |
| **100** | **0.85** | **0** | **4080.0101** | **4251.7018** | **4161.265** | **6.9253** |
| **100** | **0.85** | **0.5** | **4011.7532** | **4217.8747** | **4110.2352** | **7.2372** |
| **100** | **0.85** | **1** | **3935.3301** | **4169.4323** | **4024.3598** | **7.451** |

**三、实验总结：**

（1）心得

通过本次实验，我们对遗传算法有更近一步的认识，随着城市规模的增大，最优路径长度及算法所需时间也会延长，由表一跟图一可以直观地看出。

另外，由图二可看出，在城市规模一定的前提下，分别改变交叉概率及变异概率，所得到的最优路径长度相差不大，可得出初步结论：交叉概率和变异概率对于本题的影响不大。

总之，完成本次实验后，我们把理论上的东西落地到代码上实现，提高了我们的编程能力，更加熟练的掌握了遗传算法，并且理解到遗传算法的优点所在。

除此之外，我们还把遗传算法在python上实现了，所得数据如下表，具体代码见附带文件。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **表格三** | | | | |
| **城市规模** | **最短长度** | **最长长度** | **平均长度** | **平均运行时间（秒）** |
| **10** | **298.83** | **309.42** | **302.664** | **6.012** |
| **20** | **611.2** | **657.0** | **635.586** | **7.32** |
| **100** | **4138.47** | **4351.78** | **4275.098** | **19.42** |