



《高级人工智能》

实 验 报 告

实验名称：

学 号：

姓 名：

班 级：

**实验一 智能优化算法求解TSP问题--遗传算法**

1. **实验目的**
2. 掌握遗传算法和模拟退火算法的基本原理和程序流程。
3. 理解TSP问题的基本概念。
4. 能利用遗传算法或模拟退火算法求解TSP问题（实验时，学生任选一种完成）。

**二、实验环境与设备**

实验由1个学生独立完成，实验环境：已安装VC++6.0以上版本的计算机1台。

**三、预备知识**

**1、TSP问题基本概念**

TSP问题即旅行商问题（Traveling Salesperson Problem）。该问题给定n个城市和两两城市之间的距离，要求确定一条经过各城市当且仅当一次的最短路线。其图论描述为：给定图G=(V, A)，其中V为顶点集，A为各顶点相互连接组成的边集，已知各顶点间的连接距离，要求确定一条长度最短的Hamilton回路，即遍历所有顶点当且仅当一次的最短回路。

**2、遗传算法的基本原理**

遗传算法是一类随机优化算法，但它不是简单的随机比较搜索，而是通过对染色体的评价和对染色体中基因的作用，有效地利用已有信息来指导搜索有希望改善优化质量的状态。

标准遗传算法主要步骤可描述如下：

1. 随机产生一组初始个体构成初始种群。
2. 计算每一个体的适配值（fitness value，也称为适应度）。适应度值是对染色体(个体)进行评价的一种指标，是GA进行优化所用的主要信息，它与个体的目标值存在一种对应关系。
3. 判断算法收敛准则是否满足，若满足，则输出搜索结果；否则执行以下步骤。
4. 根据适应度值大小以一定方式执行复制操作（也称为选择操作）。
5. 按交叉概率*p*c执行交叉操作。
6. 按变异概率*p*m执行变异操作。
7. 返回步骤②。

标准遗传算法流程图下图所示。

N

N

计算各个体的适配值（适应度）

算法收敛准则满足？

Y

random[0,1]<*P****c***?

复制（选择）

输出搜索结果

交叉

Y

random[0,1]<*P****m***?

变异

N

随机产生初始种群

Y

**图1 标准遗传算法流程图**

1. **实验内容**

**1、设计算法的编码方式**

路径编码是描述TSP解的最常用的一种策略。所谓路径编码，即直接采用城市在路径中的位置来构造用于优化的状态。

例如：设九城市TSP问题的路径为5-4-1-7-9-8-6-2-3，

对应的路径编码为：(5 4 1 7 9 8 6 2 3)。

这种编码形式自然直观，易于加入启发式信息，也有利于优化操作的设计。

**2、设计遗传算法的适应度函数**

对个体*i*，计算与路径编码相对应的距离，设为*di*。显然距离值*di*越大，适应度值应越小。因此，适应度函数可定义为：。

**3、设计遗传算法的选择操作**

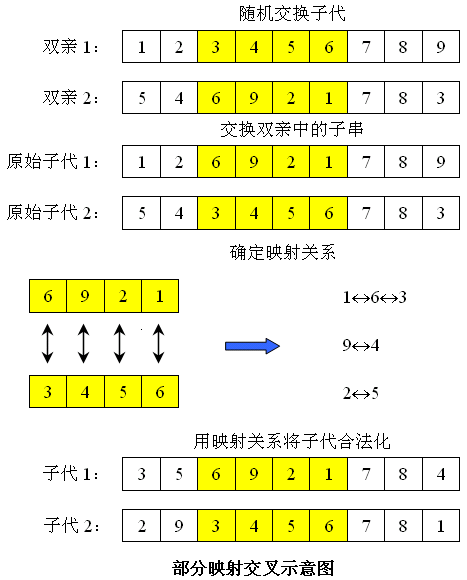
选择是用来确定交叉个体，以及被选个体将产生多少个子代个体。它是基于适应度值计算基础上进行的。在被选集中，每个个体都有一个选择概率，这个概率由种群中个体的适应度及其分布决定。若某个个体*i*，其适应度为*fi*，则其被选取的概率表示为：。

为了选择交叉个体，需要进行多轮选择，每一轮产生一个[0, 1]均匀随机数，将该随机数作为选择指针来确定被选个体。

**4、设计遗传算法的交叉操作**

在选择操作的基础上，根据一定的概率（称为交叉概率）进行交叉操作。交叉的目的是为了能够在下一代产生新的个体，它是遗传算法获取新的优良个体的最重要的手段。交叉操作中，把两个父个体的部分结构进行替换重组，生成新个体。根据个体编码方法的不同可以有不同的算法。

TSP问题中，交叉操作可设计如下：

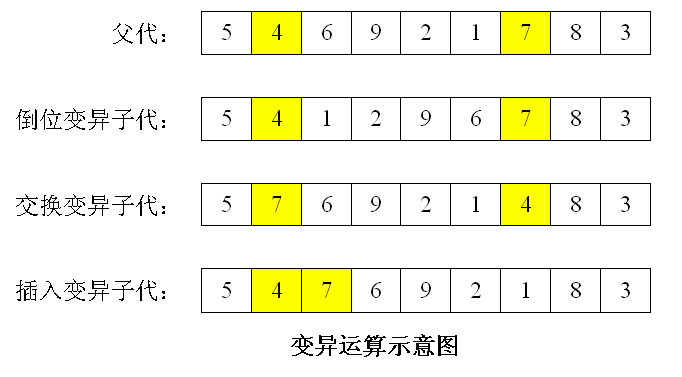


遗传算法中并不是所有被选择的个体，都要进行交叉操作。交叉概率用于控制交叉操作的频率。概率太大时，种群中串的更新很快，使高适应度值的个体很快被破坏掉。概率太小时，交叉操作很少进行，使搜索停滞不前。

**5、设计遗传算法的变异操作**

同交叉操作一样，并不是所有被选择的个体，都要进行变异操作。变异概率是加大种群多样性的重要因素，但是概率太小就很难产生新个体，概率太大会使GA成为随机搜索。基于二进制编码的GA中，通常一个较低的变异率足以防止整个群体中任一位置的基因一直保持不变。

TSP问题中，变异操作可设计如下：



**6、编写基于遗传算法的TSP问题求解程序**

**五、实验结果**

采用遗传算法解决30城市TSP问题。算法运行过程中，始终以城市(41 94)为起始城市。30城市的坐标为：41 94；37 84；53 67；25 62；7 64；2 99；68 58；71 44；54 62；83 69；64 60；18 54；22 60；83 46；91 38；25 38；24 42；58 69；71 71；74 78；87 76；18 40；13 40；82 7；62 32；58 35；45 21；41 26；44 35；4 50。

1.实验步骤：

相关定义如下：

#define POPULATION\_NUM 1000 //种群数量

#define CROSSOVER\_RATE 0.65 //交叉概率

#define ABERRATION\_RAET 0.05 //变异概率

#define ITERATIONS 30 //若连续ITERATIONS次解无改进，则退出遗传算法

struct OneRoad{

int path[city\_num];//路径信息

double length; //总路径长度

double fit; //适应度

double sum\_fit; //累计适应度概率

};//一条路径的信息

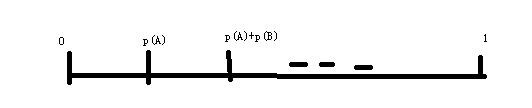
1. 初始化种群

随机分配POPULATION\_NUM条路径

1. 计算适应度和累计适应度概率

采用(1/x)^2函数作为适应度函数，x是一条路径的总长度

1. 复制选择，采用轮盘赌策略



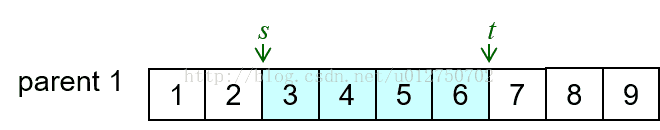
当随机生成的概率大于某一累计适应度概率时，选择该条路径

1. 重新进行评估后进行交叉

当随机生成的0-1之间的数小于交叉率时进行交叉操作

采用Order Crossover (OX)算子

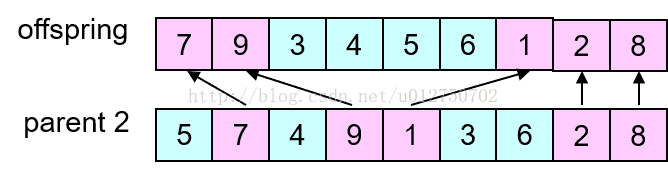
第一步，随机选择染色体（父代）中几个基因的起止位置：



第二步，生成子代，并保证子代中被选中的基因的位置与父代相同：



第三步，先找出第一步选中的基因在另一个父代中的位置，再将其余基因按顺序放入上一步生成的子代中：



与PMX不同的是，不用进行冲突检测工作（实际上也只有PMX需要做冲突检测）。

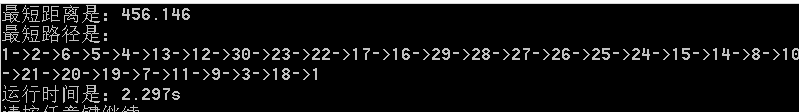
1. 重新进行评估后进行变异

当随机生成的0-1之间的数小于变异率时进行变异操作

随机生成两个位置，交换两个位置的路径信息

1. 若连续ITERATIONS次解无改进，则退出遗传算法

2.实验数据分析



1. 参数的影响

种群规模：

初始种群的信息对于收敛性影响很大，由于固定初始路径点为1，所以一共有2^29种路径选择。当种群规模太小的时候会导致初次选择的结果不是很好，从而影响后面的收敛。

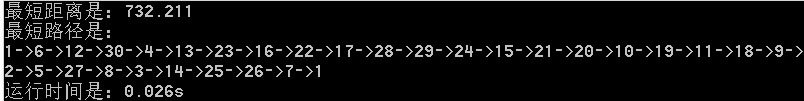
当

#define POPULATION\_NUM 100 //种群数量

#define CROSSOVER\_RATE 0.65 //交叉概率

#define ABERRATION\_RAET 0.05 //变异概率

时，结果如下。我们发现时间虽然短了，但是最终的结果不是很好。



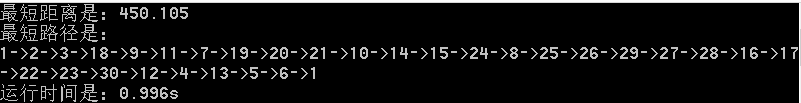
当

#define POPULATION\_NUM 1000 //种群数量

#define CROSSOVER\_RATE 0.65 //交叉概率

#define ABERRATION\_RAET 0.05 //变异概率

时，结果如下。我们发现时间比种群规模为100的要长，但是最终结果已经收敛的很好了。



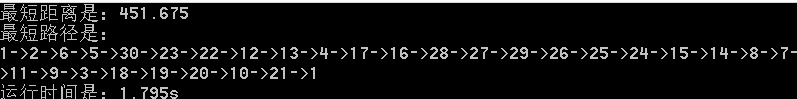
当

#define POPULATION\_NUM 1500 //种群数量

#define CROSSOVER\_RATE 0.65 //交叉概率

#define ABERRATION\_RAET 0.05 //变异概率

时，结果如下。我们发现最终结果已经基本不变了，后面如果再加大种群规模，结果都没有太大变化。



交叉概率：

当交叉概率很小时，由于交叉的可能性很小，会导致收敛的结果不太好，收敛的时间会大大加长。当交叉概率很大时，如果已经逐渐趋于稳定状态了，但是依旧不停的变化会破坏当前的平衡，导致收敛效率降低。

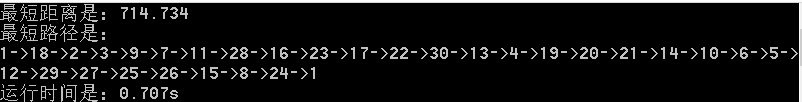
当

#define POPULATION\_NUM 1000 //种群数量

#define CROSSOVER\_RATE 0.05 //交叉概率

#define ABERRATION\_RAET 0.05 //变异概率

时，结果如下。我们发现收敛的结果不太好。



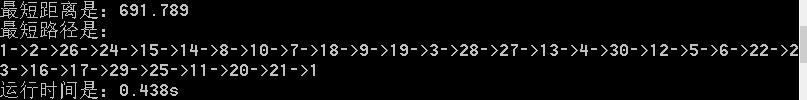
当

#define POPULATION\_NUM 1000 //种群数量

#define CROSSOVER\_RATE 0.95 //交叉概率

#define ABERRATION\_RAET 0.05 //变异概率

时，结果如下。我们发现收敛的结果不太好。



当交叉概率取适中时会达到较好的结果。

变异概率：

由于已经实行了交叉操作，所以不需要太大的变异概率。若变异概率太大，反而会导致收敛性变差。

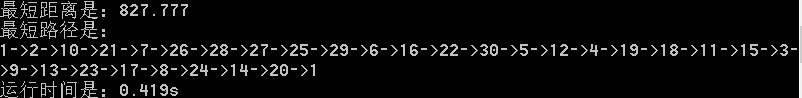
当

#define POPULATION\_NUM 1000 //种群数量

#define CROSSOVER\_RATE 0.65 //交叉概率

#define ABERRATION\_RAET 0.8 //变异概率

时，结果如下。



1. 终止准则

采用若连续ITERATIONS次解无改进，则退出遗传算法，发现虽然有时候收敛结果会有偏差，但是误差较小。

若是通过控制迭代次数来收敛，会不知道迭代多少次后效果会好。

3.用遗传算法解决函数优化问题和组合优化问题中时算法的不同

（1）编码方式不同：

函数优化问题一般采用二进制编码、Gray编码、实数编码、有序编码

组合优化问题，例如TSP采用路径编码

（2）适应度函数不同：

函数优化问题根据相关函数编写适应度函数

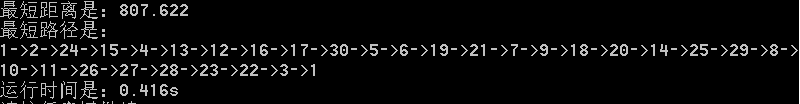
组合优化问题，例如TSP，通过对路径长度的相关操作进行编写适应度函数

4.适应度函数选择

适应度函数一般要求非负，可以将目标函数映射成一个值域为非负的函数，如果目标函数本身就是一个非负的函数，对于求最大值的问题，可以直接使用目标函数作为适应度函数，但是对于求最小值的问题，一般要将目标函数处理一下，即目标函数值越小，适应度越高且非负这样的映射关系。

采用(1/x)^2作为适应度函数，效果比较不错。

若采用1/x，结果如下：



5.遗传算法的优点和缺点

遗传算法的优点：

与问题领域无关切快速随机的搜索能力；

搜索从群体出发，具有潜在的并行性，可以进行多个个体的同时比较；

搜索使用评价函数启发，过程简单；

使用概率机制进行迭代，具有随机性；

具有可扩展性，容易与其他算法结合。

遗传算法的缺点：

遗传算法的编程实现比较复杂,首先需要对问题进行编码,找到最优解之后还需要对问题进行解码；

另外三个算子的实现也有许多参数,如交叉率和变异率,并且这些参数的选择严重影响解的品质,而目前这些参数的选择大部分是依靠经验；

算法对初始种群的选择有一定的依赖性，能够结合一些启发算法进行改进；

算法的并行机制的潜在能力没有得到充分的利用，这也是当前遗传算法的一个研究热点方向。

现在的工作中，遗传算法（1972年提出）已经不能很好的解决大规模计算量问题，它很容易陷入“早熟”。常用混合遗传算法，合作型协同进化算法等来替代，这些算法都是GA的衍生算法。

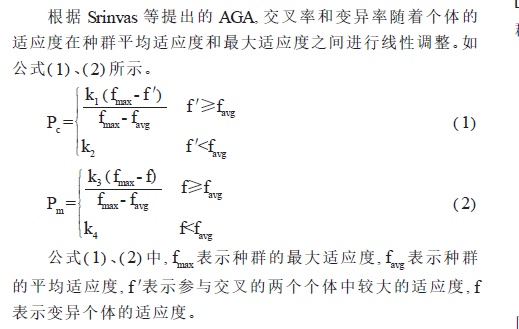
遗传算法具有良好的全局搜索能力，可以快速地将解空间中的全体解搜索出，而不会陷入局部最优解的快速下降陷阱；并且利用它的内在并行性，可以方便地进行分布式计算，加快求解速度。但是遗传算法的局部搜索能力较差，导致单纯的遗传算法比较费时，在进化后期搜索效率较低。在实际应用中，遗传算法容易产生早熟收敛的问题。采用何种选择方法既要使优良个体得以保留，又要维持群体的多样性，一直是遗传算法中较难解决的问题。

模拟退火算法虽具有摆脱局部最优解的能力，能够以随机搜索技术从概率的意义上找出目标函数的全局最小点。但是，由于模拟退火算法对整个搜索空间的状况了解不多，不便于使搜索过程进入最有希望的搜索区域，使得模拟退火算法的运算效率不高。模拟退火算法对参数（如初始温度）的依赖性较强，且进化速度慢。

6.改进措施

选择合适的种群规模。在计算量允许的情况下，尽可能选择较大的群体规模；适中的选择压力，避免选择过程将大多数个体淘汰，保持种群的多样性。

1994年，Srinivas依据遗传算法传统思想建立了一套随当代种群适应度值而动态变化的自适应遗传算法。其中交叉概率和变异概率的值由以下公式确定。



由以上公式得出在遗传过程中，当种群趋于收敛达到局部最优时，遗传过程的Pc和Pm概率会增加，而处于刚开始阶段个体适应度值大小不一时，交叉变异概率会很小。这样保证了当出现“完美”个体时，不会被该类个体迅速占领主要地位而过早收敛。

**六、实验心得**

通过本次实验，我掌握了遗传算法和模拟退火算法的基本原理和程序流程，理解了TSP问题的基本概念，能利用遗传算法或模拟退火算法求解TSP问题。

对我而言，本次实验较大的难点是参数的调节和适应度函数的选择，由于缺乏经验，刚开始参数调节的效果并不太好，但是在不断地改进中，能够获得较好的参数。

从运行结果发现遗传算法的缺点：进化停滞。往往前几代就找到了局部最优解，也就是早熟现象。发现也有很多人对这方面最了改进。比如调整选择算法 ，在最初几代降低优秀个体差异带来的选择概率，后面增加优秀个体被选择的概率。还有变异时简单的单点变异会造成等位基因的缺失，所以采用两条染色体求同或和异或保证等位基因的多样性。这些都能降低收敛速度。