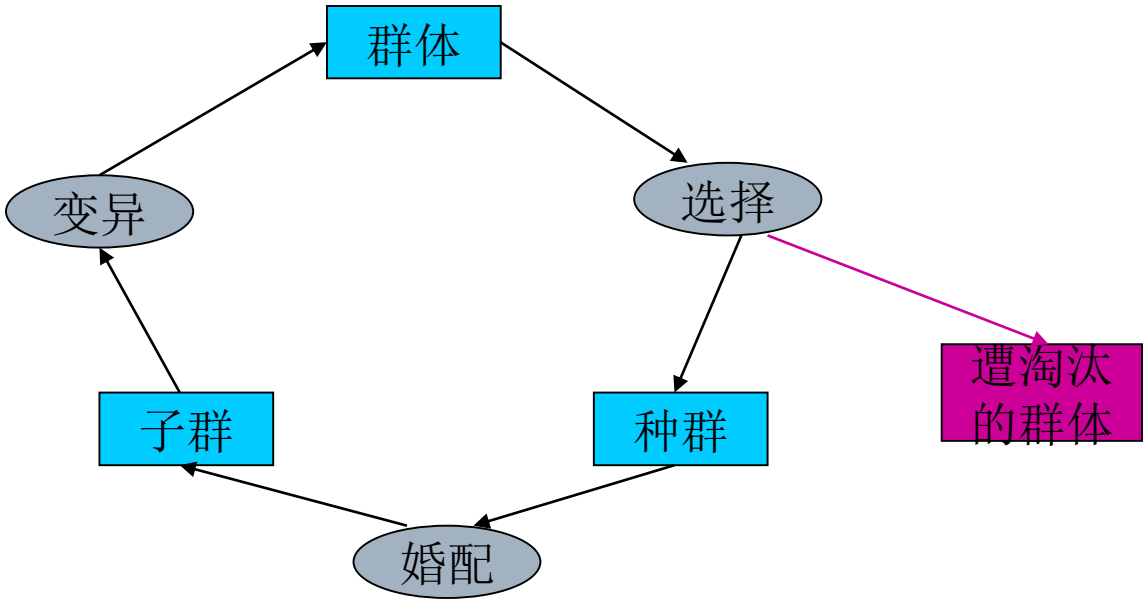
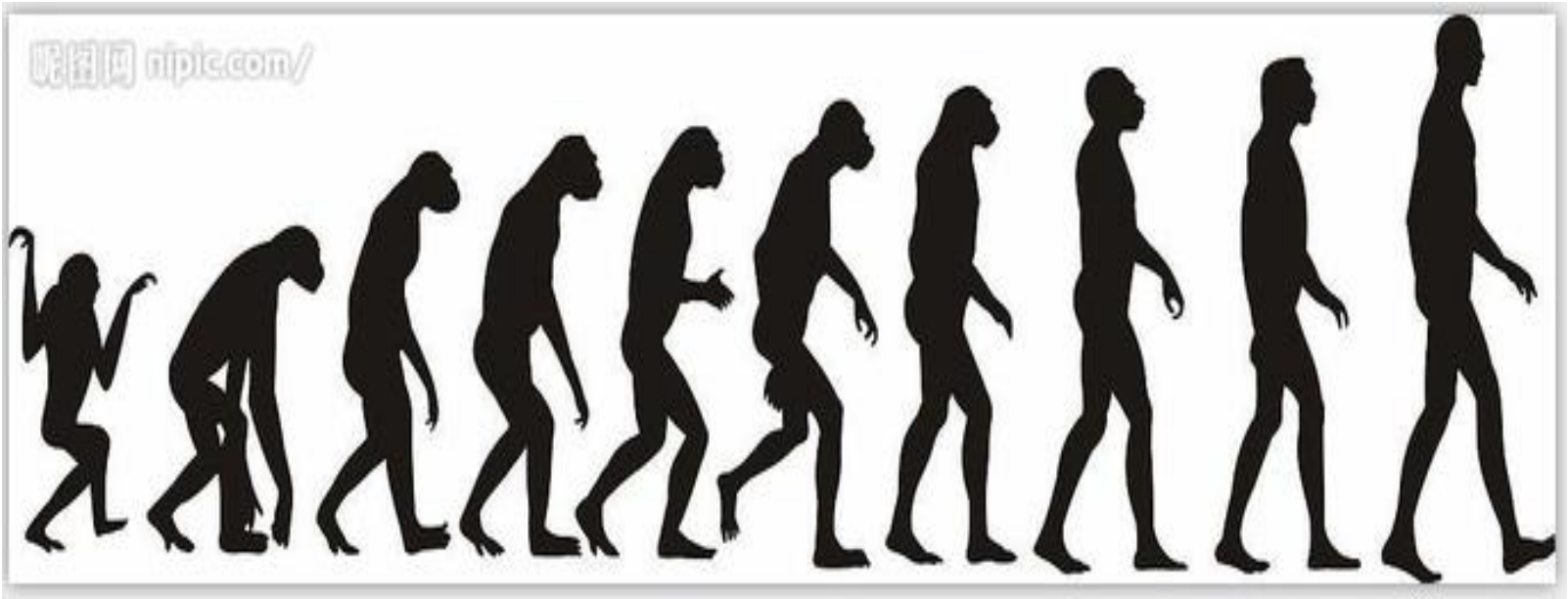
遗传算法的提出

- 遗传算法是根据自然界“物竞天择、适者生存”的现象而提出的一种随机搜索算法；
- 20世纪70年代由美国密执根大学的霍兰德（Holland）首先提出的；
- 遗传算法把优化问题看作是自然界中的生物进化过程，通过模拟大自然中生物进化过程的遗传规律，来达到寻优的目的。
- 生物进化圈示意图



人类进化图



生物进化的特性

- 进化过程发生在染色体上，而不是发生在他们所编码的生物体上；
- 自然选择把染色体以及由他们所译成的结构的表现联系在一起，适应性好的个体染色体比差的染色体有更多的繁殖机会；
- 繁殖过程发生在进化那一刻。变异可以使生物体子代的染色体不同于他们的父代染色体；通过结合两个父代染色体中的物质，重组过程可以产生在子代中有很大差异的染色体；
- 生物进化没有记忆。

遗传算法的基本思想

- 遗传算法通过自然选择中“优胜劣汰”的策略在每次搜索中利用各种随机的遗传算子生成问题一些新的解，淘汰较差的解，保留较好的以及有希望的解，从而不断对搜索空间中新的未知区域进行探索。它有效地利用了许多历史信息，使得搜索每次都向着最好的方向前进。
- 对优化问题的解进行编码，编码后的一个解称为染色体，组成染色体的元素称为基因；
- 一个群体由若干染色体组成，染色体的个数称为群体的规模；
- 用适应度函数表示环境，它是已编码的解的函数，是一个解适应环境程度的评价；适应函数的构造一般与优化问题的指标函数相关；

遗传算法的基本思想

- 适应函数值大表示所对应的染色体适应环境的能力强，自然选择规律将以适应函数值的大小来决定染色体是否继续生存下去的概率；
- 生存下来的染色体称为种群，他们中的部分或全部以一定的概率进行交配、繁衍，从而得到下一代群体；
- 交配是一个生殖过程，发生在两个染色体之间，作为双亲的染色体，交换部分基因后，生殖出两个新的染色体，即问题的新解；
- 在进化过程中，染色体的某些基因可能会发生变异，即染色体的编码发生了某些变化。一个群体的进化需要染色体的多样性，变异对于保持群体的多样性具有一定的作用。

生物进化与遗传算法之间的对应关系

生物进化中的概念	遗传算法中的作用
环境	适应函数
适应性	适应函数值
适者生存	适应值大的解被保留的概率大
个体	问题的一个解
染色体	解的编码
基因	编码的元素
群体	被选定的一组解
种群	按适应函数选择的一组解（编码表示）
交配	以一定的方式由双亲产生后代的过程
变异	编码的某些分量发生变化的过程

遗传算法的基本模型

（1）遗传算法可以形式化地描述如下：

$$GA=(P(0),N,l,s,g,p,f,t)$$

（2） $P(0)=(y_1(0),y_2(0),\ldots,y_N(0))\in I^N$ 表示初始群体；

（3） $I=\Sigma^l$ 表示长度为l 的符号串全体，Σ为字母表；
若使用二进制编码，则 $\Sigma=\{0,1\}$ ；

（4）N 为一个正整数，表示种群中含有个体的个数；

（5）l 为一个正整数，表示符号串的长度；

（6） $s:I^N\to I^N$ 表示选择策略；

（7）g 表示遗传算子，通常包括复制算子 $O_r:I\to I$ 、交叉算子 $O_c:I\times I\to I\times I$ 和变异算子 $O_m:I\to I$ ；

（8）p 表示操作概率，包括复制概率p_r 、交叉概率 p_c和变异概率 p_m ；

（9） $f:I\to R^+$ 是适应度函数；

（10） $t:I^N\to\{0,1\}$ 是终止准则。

遗传算法的基本操作

- 简单遗传算法的遗传操作主要有有三种:选择(selection)、交叉(crossover)、变异(mutation)。改进的遗传算法大量扩充了遗传操作，以达到更高的效率。
- **选择操作**实现从群体中选择存活的个体(染色体)。根据个体的适应度函数值所度量的优劣程度决定它在下一代是被淘汰还是被遗传。
- **交叉操作**的简单方式是将被选择出的两个个体*P*1和*P*2作为父母个体，将两者的基因进行交换，产生新个体的染色体。
- **变异操作**的简单方式是改变数码串的某个位置上的数码。二进制编码表示的简单变异操作是将0与1互换：0变异为1，1变异为0。

选择操作

- 选择概率：
  设群体规模为N，$F(x_i)(i=1,\ldots,N)$是N个染色体的适应值，则第i个染色体被选中的概率由下式计算：

$$P(x_i)= F(x_i)/ \sum F(x_j)$$

- “轮盘赌”选择法：
  一个转盘划分为N个扇区，每个x_i在转盘上占有一个扇区，扇区的大小与选择概率$P(x_i)$成正比。在选择一个染色体时，先转动轮盘，等轮盘停下后，指针指向的区所对应的x_i应是被选中的染色体。
- “分组淘汰”选择法：
  设群体规模为N，按 $P(x_i)$由大到小对染色体进行排序，再依次分为相等数目的k组，每组按淘汰概率$P_i(i=1,..,k)$淘汰本组染色体，各组剩余者合并为种群。淘汰概率满足：$P_1< P_2<\ldots < P_k$。

选择操作

- “确定性”选择法：
  设群体规模为N，一个选择概率为$P(x_i)$的染色体被选中的次数的期望值$e(x_i)= P(x_i)$ N。
  对于群体中的每一个x_i，首先选择$[e(x_i)]$次，这样共得$\sum[e(x_j)]$ 个染色体。再按$e(x_i)- [e(x_i)]$由小到大对染色体进行排序，依次取出N- $\sum[e(x_j)]$ 个染色体，这样共得到N个染色体组成种群。

交配操作

- 交配操作发生在两个父代染色体之间，经过杂交产生两个具有双亲的部分基因的新染色体。

- 单点交配：

$$\begin{matrix} a_1\,a_2 \ldots a_i\,a_{i+1} \ldots a_n \\ b_1\,b_2 \ldots b_i\,b_{i+1} \ldots b_n \end{matrix} \Rightarrow \begin{matrix} a_1\,a_2 \ldots a_i\,b_{i+1} \ldots b_n \\ b_1\,b_2 \ldots b_i\,a_{i+1} \ldots a_n \end{matrix}$$

- 多点交配：

$$\begin{matrix} a_1 \ldots a_i\,a_{i+1} \ldots a_j\,a_{j+1} \ldots a_n \\ b_1 \ldots b_i\,b_{i+1} \ldots b_j\,b_{j+1} \ldots b_n \end{matrix} \Rightarrow \begin{matrix} a_1 \ldots a_i\,b_{i+1} \ldots b_j\,a_{j+1} \ldots a_n \\ b_1 \ldots b_i\,a_{i+1} \ldots a_j\,b_{j+1} \ldots b_n \end{matrix}$$

变异操作

- 变异操作是发生在某一个基因上的随机变化，模拟基因突变现象，有利于保持群体的多样性，但也有很强的破坏作用。
- 二进制基因的变异操作，可以是简单地按变异概率翻转某一个位，即某位由0变1，或由1变0。
- 字符基因的变异操作将某一个基因字符上的随机地换为任一其他可能基因字符。

序号	种群	是否变异	变异位	新群体	适应值
1	11011	N		11011	729
2	11001	Y	3	11101	841
3	10000	N		10000	256

遗传算法的基本流程

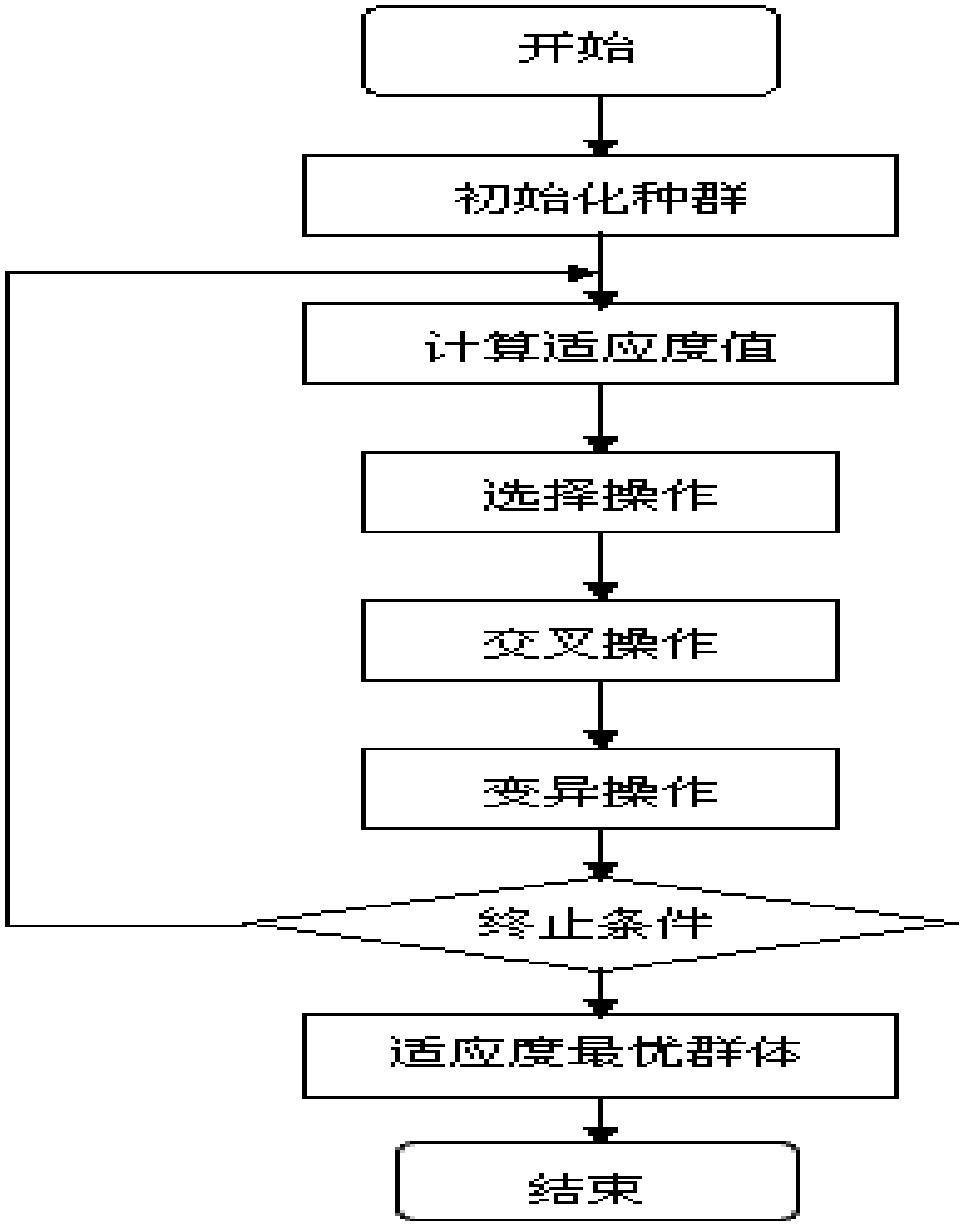


图5.2 简单遗传算法框图

遗传算法的描述

1. 给定群体规模N，交配概率p_c和变异概率p_m
2. 随机生成一个N个初始解组成的初始群体；
3. 计算当前初始群体各染色体x_i的适应度函数值$F(x_i)$；
4. 如果满足停止准则，则转10；
5. 对群体中的每一个染色体x_i计算概率$p(x_i)$；
6. 依据概率值从群体中随机选择N个染色体，得到种群；
7. 依交配概率p_c按交叉算子O_c进行交配，其子代进入新的群体，未进行交配的染色体直接复制到新群体中；
8. 依变异概率p_m从种群中选择染色体按变异算子O_m进行变异，用变异后的染色体代替新群体中的原染色体；
9. 用新群体代替旧群体，t=t+1，转3
10. 进化过程中适应值最大的染色体，经解码后作为最优解输出；
11. 结束。

遗传算法的应用

函数优化问题一般可以直接应用遗传算法进行求解，但是，更多的现实问题中应用遗传算法，首先需要转换为函数优化问题。这些应用中有多个共同问题：

- 编码问题
- 适应值函数
- 交配规则
- 变异规则
- 性能评价

用于求解函数优化问题

例如：求函数$f(x)=x^2$的最大值，其中x为[0，31]间的整数。

利用遗传算法进行求解，关键要解决如下问题：

- 编码与解码：染色体由五个二进制位组成，可能的染色体有：00000~11111共32个。
- 适应度函数：$F(x)= f(x)$
- 选择操作：轮盘赌
- 交配操作：单点交配
- 变异操作：位翻转
- 控制参数：N=4，P_c=100%,P_m=1%
- 求解过程：

用于求解函数优化问题-2

第0代：随机生成，设为01101,11000,01000,10011

序号	群体	$F(x_i)$	$\boldsymbol{P}(x_i)$	$\boldsymbol{e}(x_i)$	选中次数
1	01101	169	0.14	0.58	1
2	11000	576	0.49	1.97	2
3	01000	64	0.05	0.22	0
4	10011	361	0.31	1.23	1

从第0代中选择产生的种群：01101,11000,11000,10011

假定按顺序两两交配，即01101与11000、 11000与10011，
产生4个新个体为：01100，11001，11011，10000

作为第1代群体。

用于求解函数优化问题-2

第1代： 01100，11001，11011，10000

序号	群体	$F(x_i)$	$\boldsymbol{P}(x_i)$	$\boldsymbol{e}(x_i)$	选中次数
1	01101	144	0.08	0.33	0
2	11001	625	0.36	1.42	1
3	11011	729	0.42	1.66	2
4	10000	256	0.15	0.58	1

从第1代中选择产生的种群：11001,11011,11011,10000

假定按顺序两两交配，即11001与11011、 11011与10000，
产生4个新个体为：11011，11101，10000，11011
作为第2代群体。

用于求解函数优化问题-2

第2代： 11011，11101，10000，11011

序号	群体	$F(x_i)$	$\boldsymbol{P}(x_i)$	$\boldsymbol{e}(x_i)$	选中次数
1	11011	729	0.29	1.14	1
2	11101	841	0.33	1.31	1
3	10000	256	0.10	0.40	1
4	11011	729	0.29	1.14	1

从第2代中选择产生的种群: 11011,11101,10000,11011

假定按顺序两两交配，即11011与11101、 10000与11011，
产生4个新个体为：11001，11111，10001，11010
作为第3代群体。(以后过程略)

编码问题

- 利用遗传算法进行问题求解首先是表示问题，即将问题的解以适合于遗传算法求解的形式进行编码。
- 进行编码时，要考虑交配和变异等操作。
- 采用什么样的编码形式与具体的问题有关。
- 可以采用二进制编码，简单，但长度大。
- 也可以采用整数编码、实数编码、符号编码等形式。

例7.7 函数在实数区间上的最优化问题。

例7.8 十杆桁架问题。

例7.9 人工蚁问题。

例7.10 TSP问题。

二进制编码-求函数的最大值

- 求函数$f(x)=x^2$的最大值，其中x为[0，31]间的整数。

  由于x的定义域是[0,31]之间的整数，因此可以用五位二进制数表示该问题的解，如用10101表示x=21，0、1为基因。

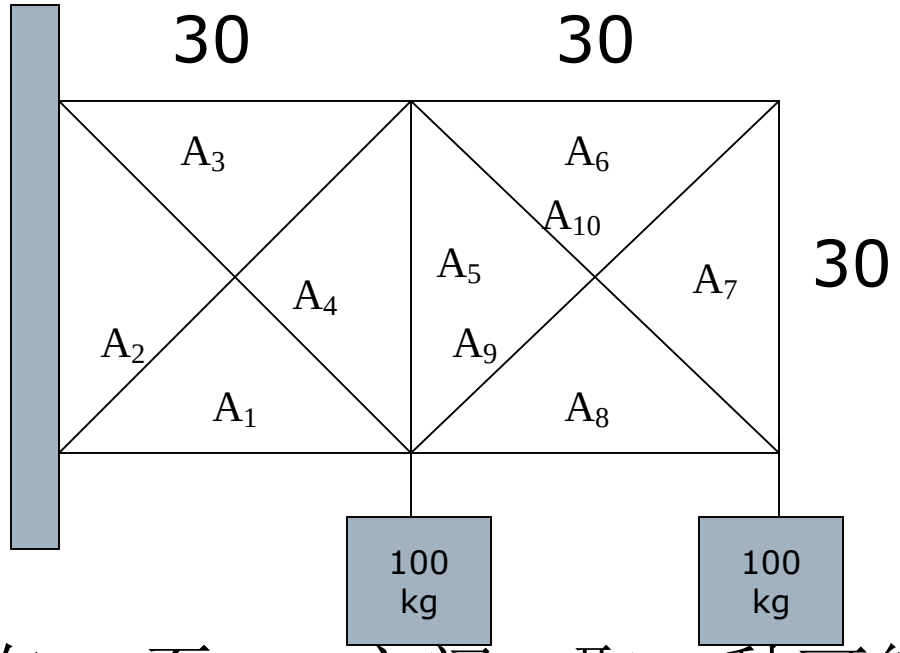
- 求函数$f(x)=x^2$的最大值，其中x为[0，31]间的实数，最大误差为1/8。

  对于区间[a,b]上的实数，当用n位的二进制向量表示时，其最大误差为$(b-a)/(2^n-1)$,所以有$(b-a)/(2^n-1)<1/8$，所以需要用8位二进制向量来表示。x与8位二进制向量之间的关系为：

  $x=31*(y/255)$，其中y为0-255之间的二进制数。

二进制编码-十杆桁架问题

- 十杆桁架问题如图所示，有10个截面积为$A_1A_2\ldots A_{10}$。如何设计每个杆的截面使建造这个桁架的材料总费用最小？

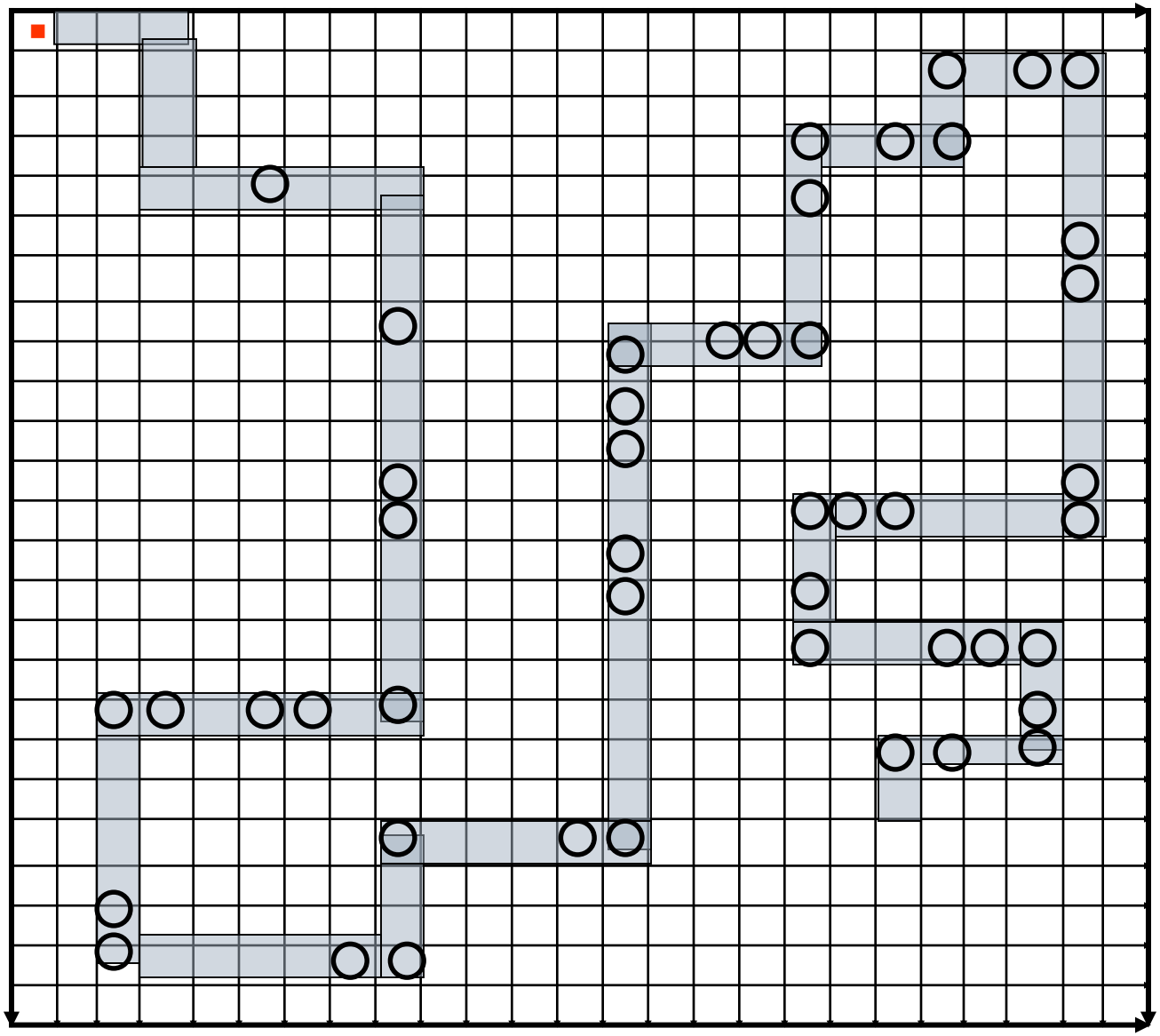


- 假设每个杆的截面积在0.1至10.0之间，取16种可能的值。用4位二进制表示截面的面积，即0000对应0.1,…,1111对应10.0。问题的解是10个杆的面积，要用40位二进制串表示一个解。例如，该问题的一个染色体为：

0010 1110 0001 0011 1011 0011 1111 0011 0011 1010

二进制编码-人工蚁问题

- 人工蚁问题如图所示，有一人工蚁在32×32的网格中移动。

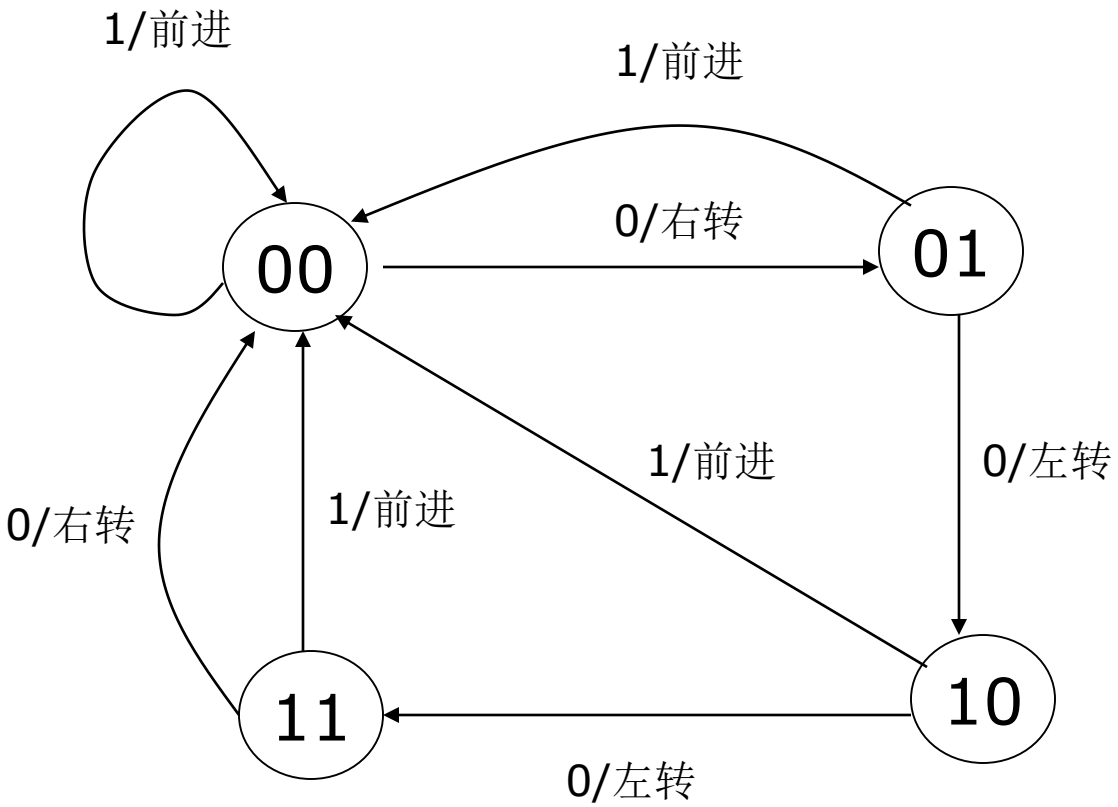


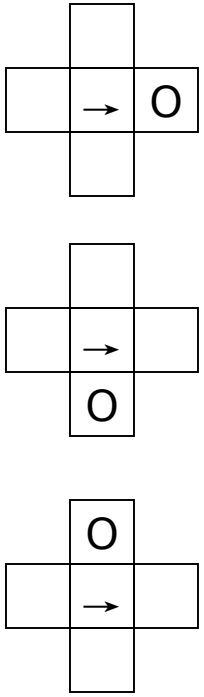
二进制编码-人工蚁问题

- “圣菲轨道”是一条不规则的弯曲轨道，上面放有89块食物。轨道上的食物不是连续排放，有些位置是空的。
- 人工蚁的视力有限，只能看到正面邻格中的是否有食物。它只有四种动作：
  - ✓ 00-不动
  - ✓ 01-左转90度(原地)
  - ✓ 10-右转90度(原地)
  - ✓ 11-走进正面相邻的网格中，如果有食物，就吃到该食物
- 要求设计一个有限状态自动机，该自动机能在有限步内引导人工蚁吃到所有的食物。

二进制编码-人工蚁问题

- 四状态自动机，容易验证：如果食物是不间断排列，该自动机可以在有限步引导人工蚁吃到所有食物。





二进制编码-人工蚁问题

- 自动机可以用状态转换表等价表示。前面的四状态自动机可等价地表示为下表，实际上只要给定了初始状态和表的最后两列， 就可以确定该自动机，因此，只要34个二制位位即可表示一个四状态自动机，例如：

00 0110 0011 10010011 1101 0011 0010 0011

序号	当前状态	输入	下一状态	动作
1	00	0	01	10
2	00	1	00	11
3	01	0	10	01
4	01	1	00	11
5	10	0	11	01
6	10	1	00	11
7	11	0	00	10
8	11	1	00	11

二进制编码-人工蚁问题

- 四状态自动机不能解决有间隔的“圣菲轨道”问题，需要更多的状态。
- Jefferson将状态增加到 32个，状态转换表需要64行，动作仍用2位表示，因此表示一个自动机需要的二进制位数为：5+(5+2)*64=453。
- 适应值函数是自动机x引导人工蚁吃到的食物数，最大值为89。
- 成功求解的参数设置：N=65535，M=200

整数编码-旅行商问题

用遗传算法求解旅行商问题，应如何进行编码？

- 二进制编码

  可以用一个矩阵来表示一个可能解。如四城市问题，可用一个4*4矩阵来表示一个可能解，将该矩阵展开得到一个4*4的二进制向量。对n城市问题，则需要一个n*n位二进制向量。可能解在整个状态空间中是非常稀疏的，导致求解效率低下。

- 整数编码

  对城市进行编号，每个城市分别用1到n之间的不同整数表示，n个整数的一个排列就代表了TSP问题的一个可能解。

适应度函数

- 为了评价染色体的适应能力，引入了对问题中的每一个染色体都能进行评价的函数，叫**适应度函数**（fitness function）。
- 一般情况下，可以直接选取问题的指标函数作为适应度函数。
- 例1：求$f(x)$的最大值问题，可以直接采用$f(x)$作为适应度函数，即$F(x)=f(x)$。
- 例2：TSP问题，目标是路径总长度最短，因此可以将路径总长度作为TSP问题的适应度函数。
- 适应度函数要有效反映每一个染色体与问题的最优解染色体之间的差距。

适应度函数

- 某些情况下，$f(x)$在最大值附近的变化可能非常小，以至于很难区分哪个染色体更优，这时应如何定义适应度函数，才能有效反映染色体与最优解染色体的差异?

✓ 非线性加速适应函数

$$F(x)=\begin{cases}\dfrac{1}{f_{\max}-f(x)} & f(x)<f_{\max}\\ M & \text{其它}\end{cases}$$

✓ 线性加速适应函数

$$F(x)=\alpha\cdot f(x)+\beta$$

✓ 利用染色体指标函数值从小到大的排列号作为适应函数值。

按定义，选择概率计算式为：$P(x_i)=\dfrac{i}{\sum_{j=1}^{m} j}=\dfrac{2i}{m(m+1)}$

二进制编码的交配规则

- 双亲双子法

  参与交配的两个双亲染色体确定后，随机地产生一个交配位，双亲染色体交换各自的交配位后的基因给对方，得到两个子染色体。

- 变化交配法

  随机产生交配位时，排除与双亲一样的交配位。

- 多交配位法

  产生多个交配位进行交配，在交配时采用交配区间交替进行的方法。

- 双亲单子法

  两个染色体交配后只得到一个子染色体。一般选择适应值大的。

二进制交配操作的例子

- 如下表所示，第0代种群为：01101,11000,11000,10011。假定交配概率的100%，即种群中所有染色体均参与交配，并按顺序两两交配。
- 交配后得到的子群为：01100,11001,11011,10000

序号	种群	交配对象	交配位	子代	适应值
1	01101	2	4	01100	144
2	11000	1	4	11001	625
3	11000	4	2	11011	729
4	10011	3	2	10000	256

整数编码的交配规则-1

整数编码的交配规则必须保持编码的有效性。

下面以TSP问题的整数编码为例进行说明(P318)。

- 常规交配法

  与二进制编码的双亲双子法类似。子代1交配位之前的基因采用父代1交配位之前的基因，交配位之后的基因从父代2中按顺序选取那些没有出现过的基因。

父代1：12345678
父代2：52373846

⇨

子代1：12345786
子代2：52371468

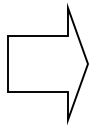
整数编码的交配规则-2

基于次序的交配法

对于两个选定的父代染色体父代1和父代2，首先随机地选择一组位置，设父代1中所选位置对应的数字从左到右依次为$x_1,x_2,\ldots,x_k$，然后从父代2中也找到这k个数字，并从父代2删除，空位置用$x_1,x_2,\ldots,x_k$依次填入，就得到子代1。子代2同理。

父代1：1 2 3 4 5 6 7 8 9 10
父代2：5 9 2 4 6 1 10 7 3 8
选 定： * * * *

父代2‘：b 9 b 4 6 1 10 7 b b
父代1‘：1 b 3 4 5 b b 8 b 10



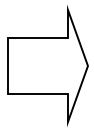
子代1： 2 9 3 4 6 1 10 7 5 8
子代2： 1 9 3 4 5 2 6 8 7 10

整数编码的交配规则

基于位置的交配法

对于两个选定的父代染色体父代1和父代2，首先随机地选择一组位置，对于这些位置上的基因，子代1从父代2直接得到，子代1其他位置的基因，按顺序从父代1中选取那些不相重的基因。子代2同理。

父代1：1 2 3 4 5 6 7 8 9
父代2：5 9 2 4 6 1 7 3 8
选 定： * * * *



子代1：1 9 2 4 6 5 7 3 8

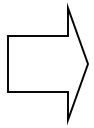
子代2：9 2 3 4 5 6 1 8 7

整数编码的交配规则

基于部分映射的交配法

对于两个选定的父代染色体父代1和父代2，随机产生两个位置，两个父代在这两个位置之间的基因产生对应对，然后用这种对应对分别去替换两个父代的基因，从而产生两个子代。

父代1：2 6 4 3 8 1 5 7 9
父代2：8 5 1 7 6 2 4 3 9
选 定：　　*　*



子代1：1 8 4 7 6 2 5 3 9

子代2：6 5 2 3 8 1 4 7 9

变异操作

- 变异操作发生在某个染色体的某个基因上，它将可变性引入群体中，增强了群体的多样性，从而提供了从局部最优中跳出来的一种手段。
- 一般通过一个很小的变异概率来控制变异
- 对二进制编码，随机产生一个变异位，被选中的基因由0变为1，或者由1变为0。

序号	种群	是否变异	变异位	新群体	适应值
1	11011	N		11011	729
2	11001	Y	3	11101	841
3	10000	N		10000	256

整数编码的变异规则

对整数编码，必须考虑变异后染色体的合理性。

常用的变异规则有如下几种（祥见P320）：

- 基于位置的变异

  随机地产生两个变异位，然后将第二个变异位上的基因移动到第一个变异位之前。

- 基于次序的变异

  随机地产生两个变异位，然后交换这两个变异位上的基因。

- 打乱变异

  随机选取染色体上的一段，然后打乱在该段内的基因次序。逆序交换方式是打乱变异的一个特例。

控制参数

遗传算法的控制参数主要包括：

- 群体规模

  每一代中群体的规模一般是固定的。扩大群体的规模可以防止早熟现象的发生。

- 停止准则

  一般可以通过规定进化的最大代数来定义；或者定义为经过连续的几代进化后，得到的最优解没有任何变化。

- 进化代数

  根据实际需要以及时间约束设置。

- 交配概率

- 变异概率

遗传算法的特点

- 遗传算法是对参数集合的编码而非针对参数本身进行进化；
- 遗传算法是从问题解的编码组开始而非从单个解开始搜索；
- 遗传算法利用目标函数的适应度这一信息而非利用导数或其它辅助信息来指导搜索；
- 遗传算法利用选择、交叉、变异等算子而不是利用确定性规则进行随机操作。
- 随机性：遗传算法是一个随机搜索算法，每一次运行得到的结果可能是不一样的。
- 通用性：遗传算法经过编码表示后除适应值计算外几乎不需要任何与问题有关的知识，而且对待求解问题的指标函数没有什么特殊的要求；
- 并行性：遗传算法具有天然的并行性，适用于并行求解；

遗传算法的收敛性定理

- 关于遗传算法数学基础更全面的知识，请参考[Goldberg,1987]的有限马可夫链数学理论分析理论或[张文修 2000]一书。
- 如果在代的进化过程中，遗传算法每次保留到目前为止的最好解，并且算法以交配和变异为其随机化操作，则对于一个全局最优化问题，当进化代数趋于无穷时，遗传算法找到最优解的概率为1。
- 该定理从理论上保证了只要进化代数足够多，则遗传算法找到最优解的可能性非常大。
- 实际使用中，要考虑在可接受的有限时间内终止算法，因此解的质量与算法的控制参数，如群体的规模、进化代数等有很大的关系。

遗传算法的收敛性定理

- 模式定理和隐性并行性是Holland为解释基于二进制编码的标准遗传算法的功效而建立的，是最早对遗传算法全局收敛性作出定性分析的理论基础，它说明了遗传算法比一般的随机搜索能更有地解决优化问题。
- 设个体链长为l，种群规划为N，用表示个体Xi的适应值，用$F(t)$第t代中个体的总适应值，即

$$F(t)=\sum_{i=1}^{N} f(X_i)$$ ，记 $$\overline{F}(t)=\frac{F(t)}{N}$$ 。

遗传算法的收敛性定理

- **定义1(模式)** 设 $a_{i_k}\in\{0,1\}$，$1\le i_k<i_{k+1}\le l$，一个模式是个体空间中的“超平面”，记作

$$s=[(i_k,a_{i_k}),0\le k\le K]$$

$$=\{X=(x_1,\ldots,x_n)\in S;x_{i_k}=a_{i_k}(k\le K)\}$$

- 其中K称作为模式的阶，记作$O(s)=K$。$\{i_1,\ldots,i_k\}$称为ψ的定义分量(或称为定义基因位置)，称a_{i_k}作定义分量值，模式ψ 的定义长度为 $\delta(s)=i_k-i_1$ 。
- 在不至于引起混淆的情况下，将模式$s=[(i_k,a_{i_k}),0\le k\le K]$简记为$s$。

遗传算法的收敛性定理

- 直观地说，如果染色体采用二进制编码，则模式是一些字符串的特征描述，可表示为一个有些位为*(可为0或1)的字符串，代表一些串的共同特征，如*11*0代表第2，3位是1且第5位是0的二进制串，即表示了模式s[(2,1), (3,1), (5,0)]={01100，01110，11100，11110}。
- 定义长度为*n*的模式有3n个，但同样长度的二进制串只有2n个，因此一个二进制串可能属于不同的模式。模式的阶是格模式中确定字符的个数，模式 的阶记为O(*s*)=3。
- 模式的定义长度是模式中第一个确定字符与最后一个确定字符间的距离， 模式的定义长度=3。

遗传算法的收敛性定理

- **定义2(**模式的适应值) 对一个模式s，在第t代的适应值$f(s,t)$定义为在种群$\vec{X}(t)$中属于该模式s的个体的平均适应值。设种群中有P个个体$X_1,\ldots,X_p$属于模式s，则

$$f(s,t)=\frac{\sum_{i=1}^{P} f(X_i)}{P}$$

- **定义3(**模式的存活概率) 对一个模式s，在第t代的适应值$f(s,t)$，定义模式s的存活概率$P(s)$为 。

$$\frac{f(s,t)}{\overline{F}(t)}$$

- 记第t代种群$\vec{X}(t)$含有s中元素个数的期望为$\xi(s,t)$，则第$t+1$代种群$(t+1)$含有s中元素个数的期望为$\xi(s,t+1)$。

遗传算法的收敛性定理

- **定理1**(模式定理) 设标准遗传算法采用适应值选择算子，交叉概率和变异异概率分别为P_c和P_m，且P_m取值较小，模式s的定义长度为　　，δ阶为　　，则以下不等式成立

$$\xi(s,t+1) \geq \xi(s,t) \bullet \frac{f(s,t)}{\overline{F}(t)} \bullet (1 - P_c \frac{\delta(s)}{l-1} - P_m \bullet O(s))$$

遗传算法的收敛性定理-证明

- 首先确定适应值选择算子后，属于s的个体的期望个数的变化。

  因为第t代种群中属于s的个体的选择概率平均为

$$\frac{f(s,t)}{F(t)}$$

  独立选择N次，一个个体被选择的概率为$N\cdot\frac{f(s,t)}{F(t)}$

  第t代种群中属于s的个体的期望个数有 $\xi(s,t)$

  于是选择算子后属于s的个体个数有

$$\xi(s,t)\bullet N\bullet\frac{f(s,t)}{F(t)} \qquad (5.1.2)$$

遗传算法的收敛性定理-证明

- 记第t代种群中个体平均适应值为 $\overline{F}(t)=\frac{F(t)}{N}$

于是式(5.1.2)变为

$$\xi(s,t)\bullet N\bullet\frac{f(s,t)}{F(t)}$$

$$=\xi(s,t)\bullet\frac{f(s,t)}{\frac{F(t)}{N}}$$

$$=\xi(s,t)\bullet\frac{f(s,t)}{\overline{F}(t)}$$

遗传算法的收敛性定理-证明

- 设属于模式s的个体的适应值$f(s,t)$比平均高 ε，

即 $f(s,t)=\overline{F}(t)+\varepsilon\overline{F}(t)$

则有

$$\xi(s,t)\bullet\frac{f(s,t)}{\overline{F}(t)}$$

$$=\xi(s,t)\bullet\frac{(\overline{F}(t)+\varepsilon\overline{F}(t))}{\overline{F}(t)}$$

$$=\xi(s,t)\bullet(1+\varepsilon)$$

因此，若不考虑其它算子的破坏作用，则适应值高于平均值的属于模式s的个体的期望数目，选择算子将使其随进化迭代次数而指数级增长。

遗传算法的收敛性定理-证明

- 下面考虑其它算子对模式s的破坏作用。
- 我们分析单点交叉算子的情况。由于交叉是在l-1个可能位置中以等概率进行的，因此模式s被交叉破坏的概率为

$$P_d(s) = P_c \bullet \frac{\delta(s)}{l-1}$$

- 或者说模式s 被保留下来的概率为

$$P_{s,c}(s) = 1 - P_c \bullet \frac{\delta(s)}{l-1}$$

遗传算法的收敛性定理-证明

- 最后我们分析变异算子的情况，由于变异是以变异概率*Pm*独立地对每一位进行，于是每一位保持不变的概率为1-*Pm*，因为各位变异是独立进行的，因此每个属于模式s的个体在变异中被保留的概率为。

$$P_m = (1-P_m)^{O(s)} \approx 1-O(s)\bullet P_m$$

- 综合而得，第*t*+1代种群中属于s的个体个数满足

$$\xi(s,t+1) \geq \xi(s,t)\bullet\frac{f(s,t)}{\overline{F(t)}}\bullet(1-P_c\frac{\delta(s)}{l-1}-P_m\bullet O(s))$$

它说明定义长度短、阶低且适应值高于平均水平的模式的数量在遗传过程中将指数级增长。

遗传算法的隐性并行性定理

- **定理2(**隐性并行性定理) 设ε 为一小正 数，

$$l_s < \varepsilon(l-1)+1, N = 2^{l_s/2} \quad ,$$

则SGA一次处理的存活概率不小于1- ε 且定义长度不大于l_s的模式数为O(N^3)。

遗传算法同时处理多个模式，说明了它具有隐性并行性，这就是其效率高的根本原因。

在有限个体空间中的遗传算法的收敛性，主要有下面几个定理。

- **定理3** 在有限个体空间中，SGA不能以概率1收敛于全局最优解。 □
- **定理4**在有限个体空间中，EGA以概率1收敛于全局最优解。□
- 这两个定理都说明，遗传算法的收敛性与适应值函数没有直接关系。

求解SAT的遗传算法

- SAT问题
- SAT-GA设计
- 测试结果分析

SAT问题

- 可满足性(SAT)问题是命题逻辑中的一个经典问题，也是计算机科学理论与应用的一个核心问题。研究解决SAT问题的有效算法不仅具有重大的理论意义和实际的应用价值。自从SAT问题提出后，一直有不少学者在研究，有的学者致力寻找可行的算法，也有不少学者致力于研究SAT问题的内在规律，获得了不少成果和认识。
- 在算法研究方面，分为完全算法和不完全算法研究两大类。前者的主要成果有：早期的在上世纪60、70年代的DP算法[1]、归结法等，近期的贺思敏[00]等的吴方法求解方法等。完全算法理论上能保证找到解，但是SAT问题是一个NP完全问题，不太可能找到效率可行的完全算法。上世纪90年代以来，有关SAT问题的算法研究热点转向了不完全算法研究。不完全算法虽然不能保证一定能在有限时间内找到解，但多数情况下其求解速度比完全算法快，实用性更高。

SAT问题描述

- 近年来不完全算法的研究成果非常丰富，如Selman和Kautz的GSAT算法和WALKSAT算法、梁东敏等的WALKSAT改进算法、李未等的数学物理方法和拟人拟物算法、张德富等的拟人退火算法、顾均的SAT1.3算法等等。虽然近期有用遗传算法求解NP问题的文章中，以NP的一个例子的形式提到了SAT问题，如张铃、张钹的佳点集遗传算法等等。

SAT问题描述

定义1 用符号U表示一个命题变元的集合。对n个命题变元$u_1, u_2, \ldots, u_n$组成，那么，$U=\{u_1, u_2, \ldots, u_n\}$，用$|U|$表示集合U的元素数目。

定义2 U的一个真值指派t是一个映射: $U \rightarrow \{0, 1\}$，它可用一个n维向量$<t_1, t_2, \ldots, t_n>$表，其中$t_i \in \{0, 1\}$，0和1分别代表命题常元False和True。

对任意一个命题变元u_i，如果其在真值指派t下取真，则记为$t(u_i)=1$，否则，$t(u_i)=0$。

在U上，存在2^n个不同的真值指派，所有真值指派构成的n维向量空间，记为U_a。

定义3 对任意一个变元u，称符号u和¬u是其文字，且称u是正文字，¬u是负文字。

用符号L表示一个文字。

SAT问题描述

- 定义4　子句是U上的若干个文字的析取，用C表示。用|C|表示C中的文字数，称为子句长度。
- 定义5　子句集A是由U的子句组成的集合。它在真值指派t下是可满足的当且仅当其中每个子句在真值指派t下都是真。
- 定义6　SAT问题可以定义为：给定命题变元集U的子句集A，问是否存在一个关于U的真值指派t，使得A是可满足的，记为P(U, A)。
- 对任一P(U, A)，设|U|=n，|A|=m，用Z_P表示整数集{0,1,2,…,2^n-1}和用Z_q表示整数集{0, 1, …, m-1}。
- 定义7　从整数集Z_p到Z_q的函数s：$Z_P \rightarrow Z_q$，即任一个二进制整数$k=t_1t_2\ldots t_n$，s(k)就是真值指派$\langle t_1, t_2, \ldots, t_n\rangle$所能满足的子句数目，显然$0 \leqslant s(k) \leqslant m$。

SAT问题描述

- 例如：U={x, y, z}，A={x∨y，¬x∨z，z∨¬y}，|U|=3，|A|=3，Z_P={0, 1, 2, 3, 4, 5, 6, 7}，Z_q={0, 1, 2, 3}

  假设有一个真值指派t：<x, y, z>=<1, 0, 1>。

  真值指派<1，0，1>能满足A中的3个子句，所以，$s(101_B)$=3。

  本例的真值指派集Ua={<0,0,0>，<0,0,1>，<0,1,0>，<0,1,1>，<1,0,0>，<1,0,1>，<1,1,0>，<1,1,1>}，各真值指派可满足的子句数目对应为：2、2、2、3、1、3、1和3。

SAT-GA

- 染色体是一个由n个二进制位的位串，实际上就是指真值指派中各变元的取值构成的。设|U|=n，对任一变元u，如果在一个真值指派t中，有t(u)=1，则t(¬u)=0，反之亦然，所以只要n个二进制位就可以表示一个真值指派t。
- 用二进制的位串表示真值指派的染色体是很直观的编码方法，采用这种编码方案，充分利用了SAT本身的特点，便于计算适应值函数和设计各种遗传算子。

SAT-GA

- 适应值函数f_1

前文定义7所定义的函数s。真值指派能满足的子句数量能一定程度上反映其优劣。这个函数必存在最大值，若子句集是可满足的，其最大值为子句数目。

这个目标函数的设计是合理且直观的,计算也简单，实验表明该函数具有一定的导向能力，能快速引导个本向解的进化，缺点是搜索的最后阶段算法易出现平台现象，求解成功率偏低。

- 适应值函数f_2

定义为f_2(t)=真值指派t下为真的子句的静态权重之和。

在结构化SAT问题中求解中，子句长短不一，实验中发现，短子句难满足，长子句易满足，因此子句长度是反映子句满足的难易程度的信息，是SAT问题求解时自身存在的、可用以引导进化搜索的很重要的启发信息。

SAT-GA描述

SAT-GA算法如下：

1). 输入变元集和子句集，初始化参数，随机生成M个真值指派作为初始第一代的群体；

2). 对每一个真值指派t，求其适应值，同时计算出当代群体的总适应值和最优的真值指派的适应值，进而计算每个真值指派的选择概率，构成选择种群的概率滚花轮;

3). 若满足结束条件，算法结束。

4). 选择种群：当代的一个最优的真值指派直接复制到种群中，其余的M-1个体由模拟滚花轮的方式从当代群体中选择。

5). 产生下一代：当代的一个最优个体直接复制到下一代中，其余M-1个真值指派的产生过程是：顺序地从种群中选出一个真值指派作为父亲，以概率Pc参与交叉，若要交叉，则从种群中随机配对，找到另一个真值指派作为母亲，然后进行交叉操作，产生两个下一代的真值指派，对新真值指派实施变异操作，放入下一代中。如此重复，直到产生了M-1个的新一代群体为止。

6). 累计代数加1，转第二步；

SAT-GA实验结果

SAT、GSAT和WALKSAT求解3-SAT问题

变元数 m=100	子句数= 200		子句数= 300		子句数= 400		子句数= 500		子句数= 600	
算法名称	成功	时间	成功	时间	成功	时间	成功	时间	成功	时间
SAT-GA	100	0.2	100	0.3	100	0.7	100	1.6	100	1.9
GSAT	100	1.2	100	2.4	100	10.4	80	20.8	90	38.3
WALKSAT	100	0.6	100	1.0	100	2.3	100	8.0	100	11.3

SAT-GA实验结果

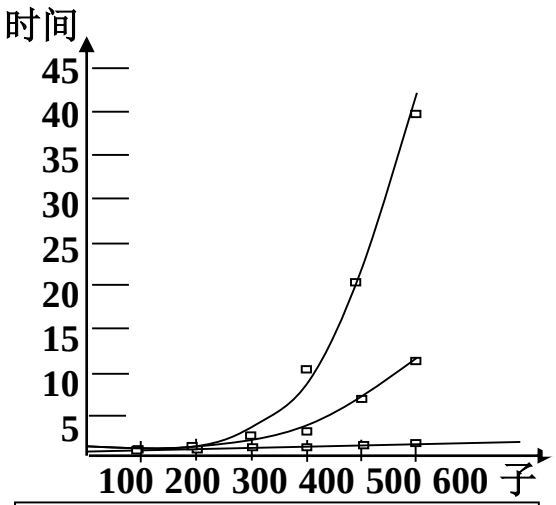


图1 100变元的3-SAT样本求解时间对比图

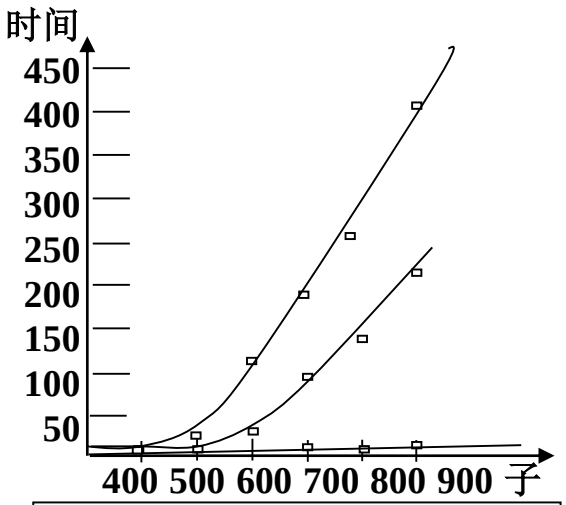


图2　200变元的3-SAT样本求解时间对比图

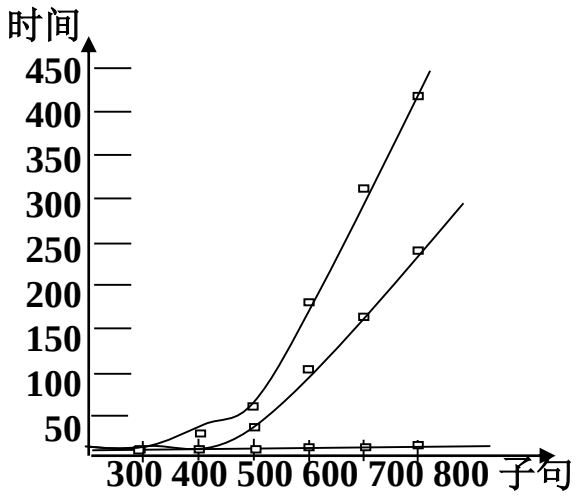


图3　300变元的3-SAT样本求解时间对比图

SAT-GA实验结果

SAT求解结构化SAT问题

问题规模	子句集规模(子句长度不多于变元数的10%)											
	×1		×2		×3		×4		×5		×6	
	成功	时间	成功	时间	成功	时间	成功	时间	成功	时间	成功	时间
1024 变元	100	1.5	100	6.5	100	13.6	100	23.2	100	31.3	100	53.8
2048 变元	100	9.6	100	32.2	100	44.3	100	78.2	-	-	-	-
4096 变元	100	16.4	100	80.4	-	-	-	-	-	-	-	-
8172 变元	100	502.0	-	-	-	-	-	-	-	-	-	-