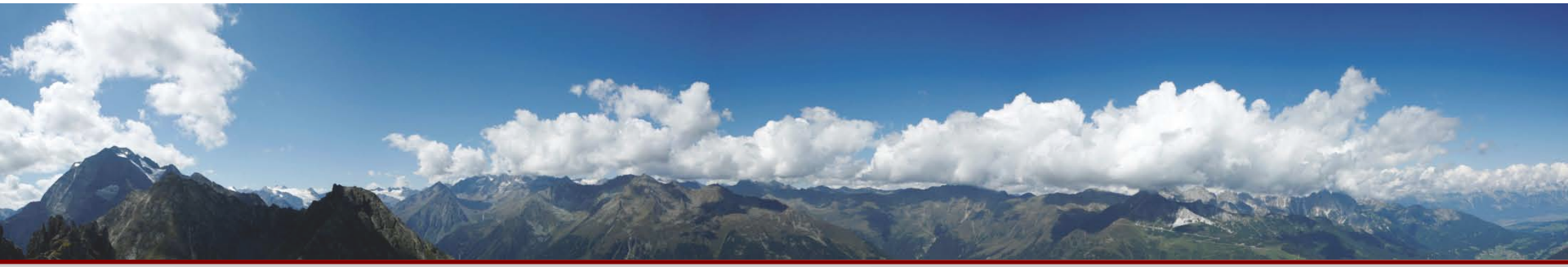
Intelligent Optimization Algorithm

Genetic Algorithms



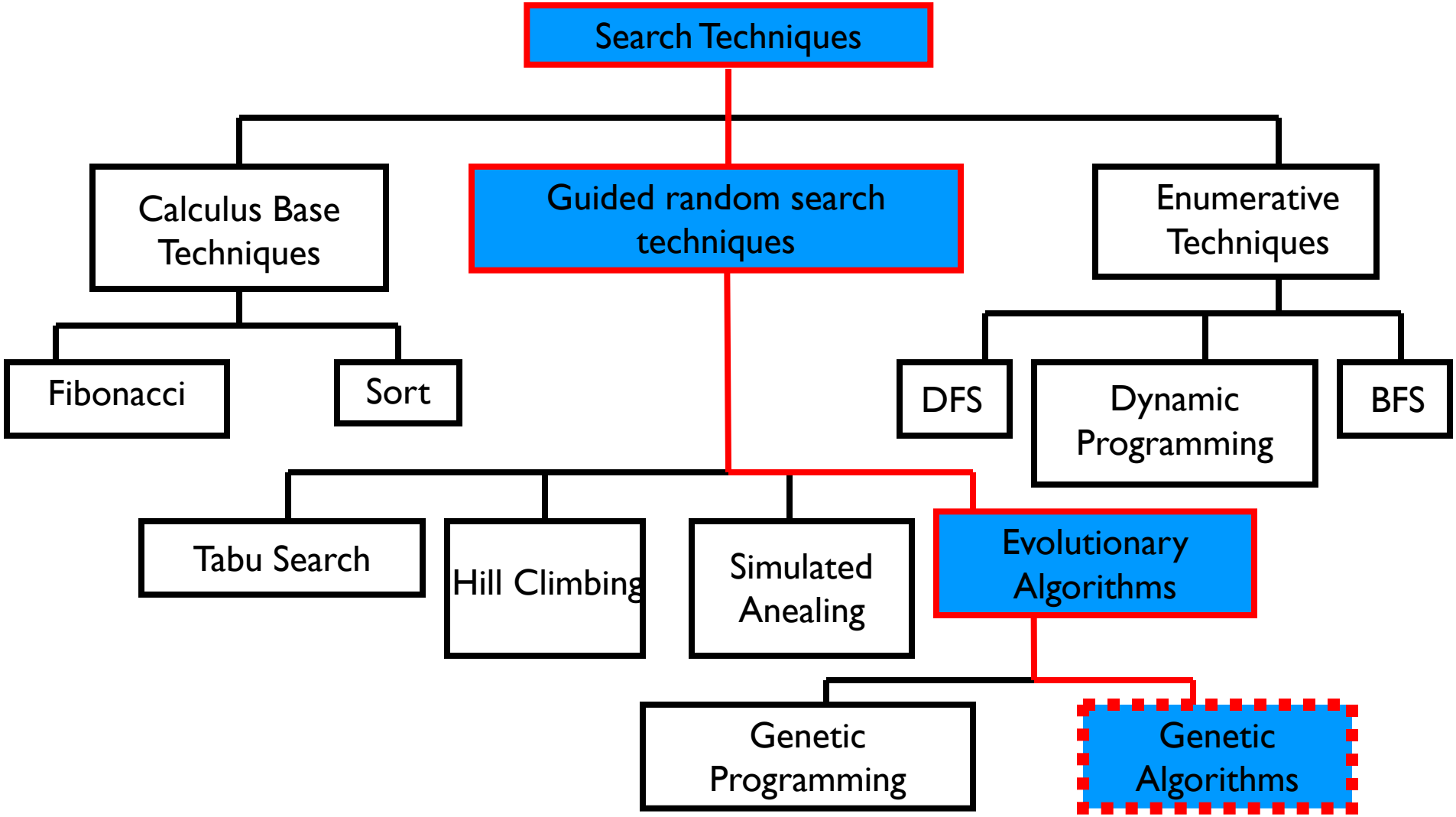
Genetic Algorithms (GA) OVERVIEW

- A class of probabilistic optimization algorithms
- Inspired by the biological evolution process
- Uses concepts of "Natural Selection" and "Genetic Inheritance" (Darwin 1859)
- Originally developed by John Holland (1975)

GA overview (cont)

- Particularly well suited for hard problems where little is known about the underlying search space
- Widely-used in business, science and engineering

Classes of Search Techniques



A genetic algorithm maintains a population of candidate solutions for the problem at hand, and makes it evolve by iteratively applying a set of stochastic operators

Stochastic operators

- ***Selection*** replicates the most successful solutions found in a population at a rate proportional to their relative **quality**
- ***Recombination*** decomposes two distinct solutions and then randomly mixes their parts to form novel solutions
- ***Mutation*** randomly perturbs a candidate solution

The Metaphor

Genetic Algorithm	Nature
Optimization problem	Environment
Feasible solutions	Individuals living in that environment
Solutions quality (fitness function)	Individual's degree of adaptation to its surrounding environment

The Metaphor (cont)

Genetic Algorithm	Nature
A set of feasible solutions	A population of organisms (species)
Stochastic operators	Selection, recombination and mutation in nature's evolutionary process
Iteratively applying a set of stochastic operators on a set of feasible solutions	Evolution of populations to suit their environment

The Metaphor (cont)

The computer model introduces simplifications (relative to the real biological mechanisms),

BUT

surprisingly complex and interesting structures have emerged out of evolutionary algorithms

Simple Genetic Algorithm

produce an initial population of individuals

evaluate the fitness of all individuals

while termination condition not met **do**

    select fitter individuals for reproduction

    recombine between individuals

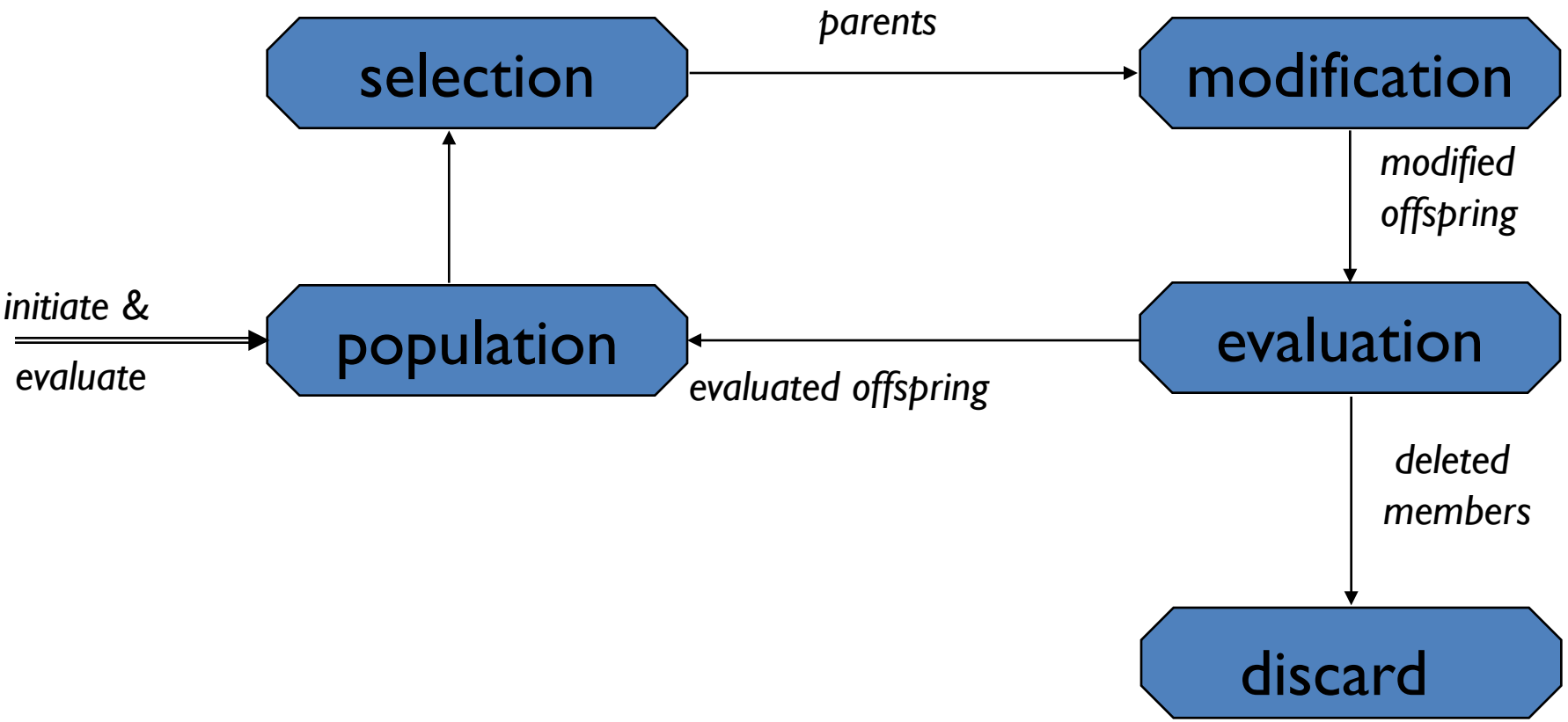
    mutate individuals

    evaluate the fitness of the modified individuals

    generate a new population

End while

The Evolutionary Cycle



基本概念 (Basic Concept)

1. 个体(individual)与种群(population)

- 个体(individual)就是模拟生物个体而对问题中的对象（一般就是问题的解）的一种称呼，一个个体也就是搜索空间中的一个点。

Individual is just the plant in the under-solved problem ,it is also a point in the searching space.

- 种群(population)就是模拟生物种群而由若干个体组成的群体，它一般是整个搜索空间的一个很小的子集。

While a population consists of several individuals. Generally , it is also a subset of the whole searching space.

2. 适应度(fitness)与适应度函数(fitness function)

- 适应度(fitness)就是借鉴生物个体对环境的适应程度,而对问题中的个体对象所设计的表征其优劣的一种测度。

Fitness is the standard to judge the individual. We can use it to evaluate the individuals in order to estimate them.

2. 适应度(fitness)与适应度函数(fitness function)

- 适应度函数(fitness function)就是问题中的全体个体与其适应度之间的一个对应关系。

它一般是一个实值函数。该函数就是遗传算法中指导搜索的评价函数。

Fitness function is the relationship between an individual and its fitness. It is an evaluation function in GA.

3. 染色体(chromosome)与基因(gene)

染色体（chromosome）就是问题中个体的某种字符串形式的编码表示。字符串中的字符也就称为基因（gene）。

Chromosome is the expression of the individual which is coded. The character in the string is called gene.

3. 染色体(chromosome)与基因(gene)

Example:

individual		chromosome
9	----	1001
(2, 5, 6)	----	010 101 110

4. 遗传操作(genetic operation)

亦称遗传算子(genetic operator)，就是关于染色体的运算。遗传算法中有三种遗传操作:

- 选择-复制(selection-reproduction)
- 交叉(crossover，亦称交换、交配或杂交)
- 变异(mutation，亦称突变)

There are three kinds of operation in GA. Selection-reproduction, crossover, as well as mutation.

选择-复制(selection-reproduction)

通常做法是：对于一个规模为N的种群S, 按每个染色体$x_i \in S$ 的选择概率$P(x_i)$所决定的选中机会，分N 次从S 中随机选定N个染色体，并进行复制。

We usually do as follows: Choose N chromosomes from population S in N separate times. The probability of one individual being chosen is $P(xi)$.

The computational formula of $P(x_i)$:

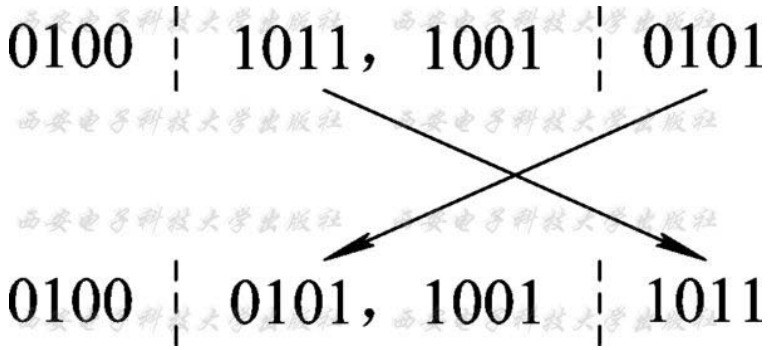
$$P(x_i) = \frac{f(x_i)}{\sum_{j=1}^{N} f(x_j)}$$

交叉(crossover)

就是互换两个染色体某些位上的基因。

There is a chance that the chromosomes of the two parents are copied unmodified as offspring ,or randomly recombined (crossover) to form offspring.

For example, there are two chromosomes s_1=01001011, s_2=10010101, we exchange the last four genes:



s_1'=01000101, s_2'=10011011

They can be considered as the offspring of s_1 and s_2.

变异(mutation)

就是改变染色体某个(些)位上的基因。

There is a chance that a gene of a child is changed randomly. Generally the chance of mutation is low.

For example, the chromosome s=11001101,

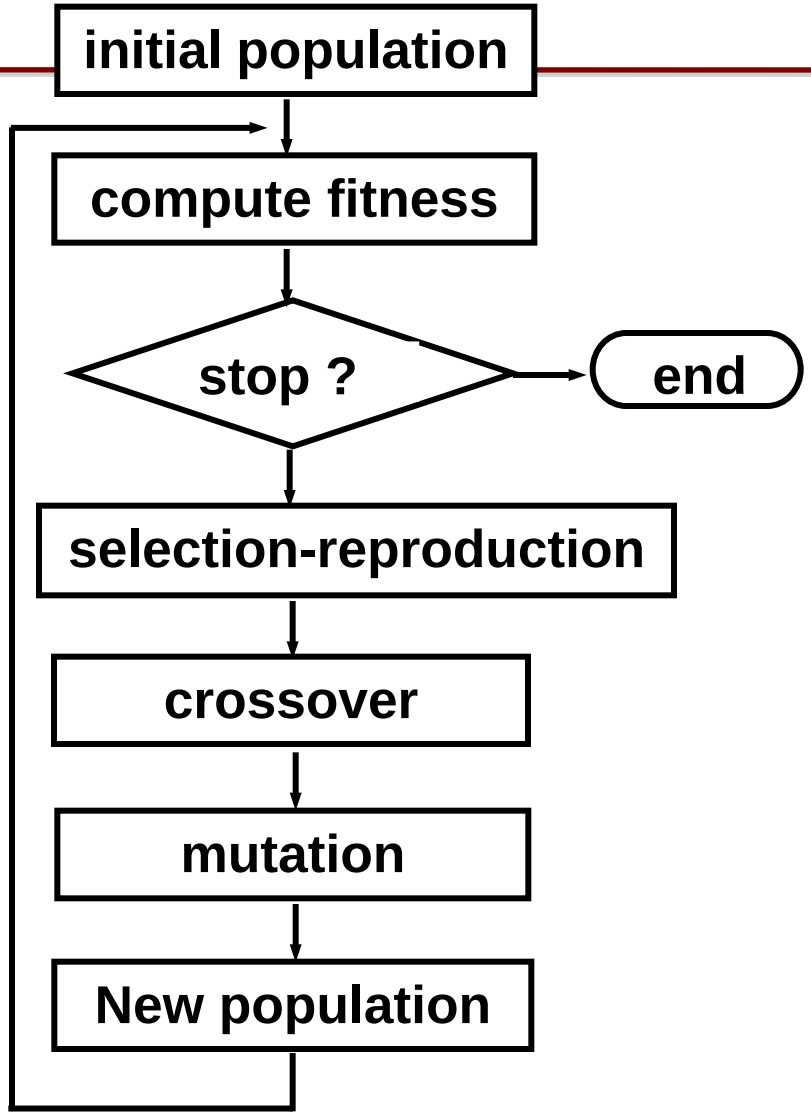
The third bit change into 1 from 0, we get:

$$s=11\underline{0}01101 \rightarrow 11\underline{1}01101= s'。$$

s'can be seen as the offspring of s.

基本遗传算法

Basic flow chart



算法中的一些控制参数：

- **种群规模** (population scale)
- **最大换代数** (Max generation number)
- **交叉率**(crossover rate)就是参加交叉运算的染色体个数占全体染色体总数的比例，记为P_c，取值范围一般为0.4～0.99。
- **变异率**(mutation rate)是指发生变异的基因位数所占全体染色体的基因总位数的比例，记为P_m，取值范围一般为0.0001～0.1。

steps

step1 在搜索空间U上定义一个适应度函数$f(x)$，给定种群规模N，交叉率P_c和变异率P_m，代数T；

step2 随机产生U中的N个个体s_1，s_2，…，s_N，组成初始种群$S=\{s_1, s_2, \cdots, s_N\}$，置代数计数器$t=1$；

step3 计算S中每个个体的适应度f；

step4 若终止条件满足，则取S中适应度最大的个体作为所求结果，算法结束。

step5　按选择概率$P(x_i)$所决定的选中机会，每次从S中随机选定1个个体并将其染色体复制，共做N次，然后将复制所得的N个染色体组成群体S_1；

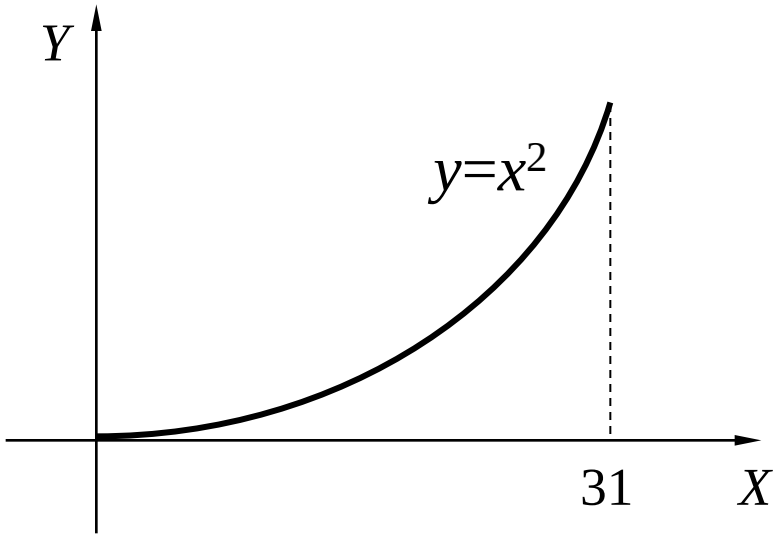
step6　按交叉率P_c所决定的参加交叉的染色体数c，从S_1中随机确定c个染色体，配对进行交叉操作，并用产生的新染色体代替原染色体，得群体S_2；

step7 按变异率P_m所决定的变异次数m，从S_2中随机确定m个染色体，分别进行变异操作，并用产生的新染色体代替原染色体，得群体S_3；

step8 将群体S_3作为新一代种群，即用S_3代替S，$t = t+1$，转步3；

遗传算法应用举例 (Application)

Use GA to compute the maximum of the function $y=x^2$ in the interval [0,31].



分析

原问题可转化为在区间［0, 31］中搜索能使y取最大值的点a的问题。那么，［0, 31］中的点x就是个体, 函数值$f(x)$恰好就可以作为x的适应度，区间［0, 31］就是一个(解)空间 。这样, 只要能给出个体x的适当染色体编码, 该问题就可以用遗传算法来解决。

Solve

(1) Set the size of the population ,code the chromosomes ,initial population

We choose the size of the population as 4; the coding number has 5 bits ; the initial population S_1:

$$s_1 = 13\ (01101),\quad s_2 = 24\ (11000)$$

$$s_3 = 8\ (01000),\quad s_4 = 19\ (10011)$$

(2) define the fitness function:

$$f(x) = x^2$$

(3) Compute the fitness of each individual, apply the genetic operation to the chromosomes ,until the individual who has the largest fitness appears.

Calculate the fitness $f(s_i)$ of each individual in S_1:

$s_1 = 13(01101)$, $s_2 = 24(11000)$

$s_3 = 8(01000)$, $s_4 = 19(10011)$

We have:

$$f(s_1) = f(13) = 13^2 = 169$$
$$f(s_2) = f(24) = 24^2 = 576$$
$$f(s_3) = f(8) = 8^2 = 64$$
$$f(s_4) = f(19) = 19^2 = 361$$

Then we compute the probability of each individual in S_1

Calculation formula:

$$P(x_i) = \frac{f(x_i)}{\sum_{j=1}^{N} f(x_j)}$$

then,

$$P(s_1) = P(13) = 0.14$$

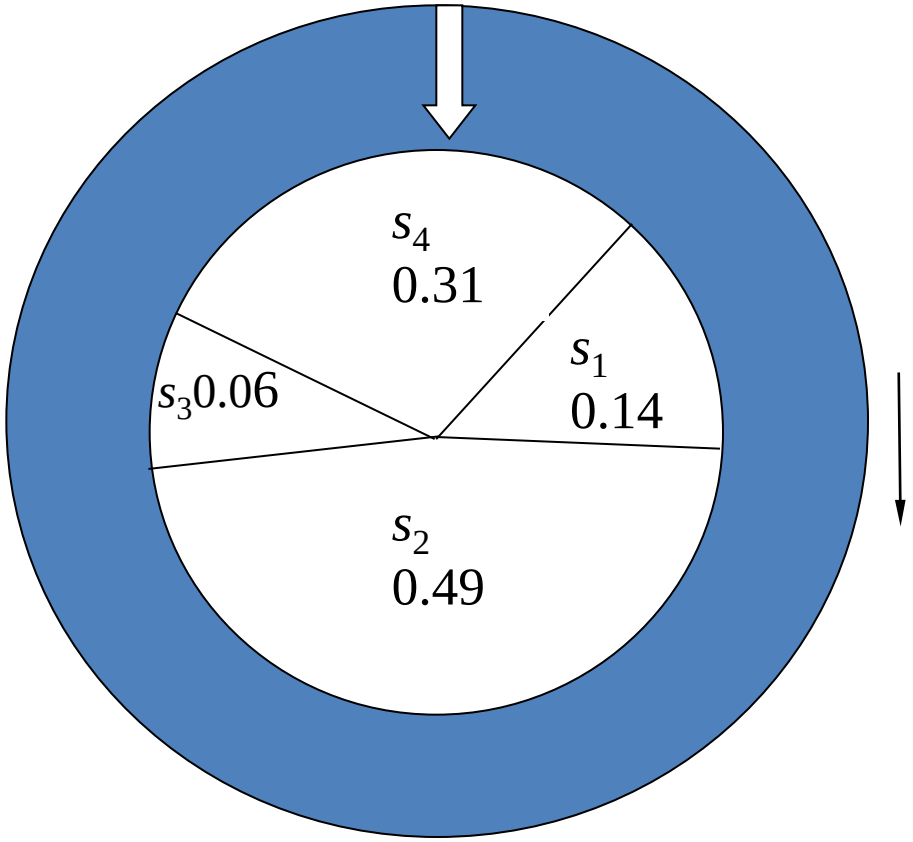
$$P(s_2) = P(24) = 0.49$$

$$P(s_3) = P(8) = 0.06$$

$$P(s_4) = P(19) = 0.31$$

赌轮选择法(roulette wheel)

Next we apply fitness proportionate selection with the roulette wheel method:



赌轮选择**示意**

在算法中赌轮选择法可用下面的子过程来模拟：

① 在［0，1］区间内产生一个均匀分布的随机数r。

② 若$r \leqslant q_1$，则染色体x_1被选中。

③ 若$q_{k-1} < r \leqslant q_k (2 \leqslant k \leqslant N)$，则染色体$x_k$被选中。其中的$q_i$称为染色体$x_i$（$i$=1，2，…，$n$）的**积累概率**，其计算公式为

$$q_i = \sum_{j=1}^{i} P(x_j)$$

Selection-reproduction

Create four random numbers from the interval[0, 1]:

$$r_1 = 0.450126, \quad r_2 = 0.110347$$

$$r_3 = 0.572496, \quad r_4 = 0.98503$$

染色体	适应度	选择概率	积累概率	选中次数
s_1=01101	169	0.14	0.14	1
s_2=11000	576	0.49	0.63	2
s_3=01000	64	0.06	0.69	0
s_4=10011	361	0.31	1.00	1

After reproduction, we get a new population：

$s_1'=11000(24),\ s_2'=01101(13)$

$s_3'=11000(24),\ s_4'=10011(19)$

Crossover

Set p_c=100%, all of the chromosomes in S_1take part in the crossover operation.

Assume that s_1′pairs with s_2′ , s_3′pairs with s_4′.Exchange the last two genes, we get a new chromosome:

$$s_1''=11001(25),\ s_2''=01100(12)$$

$$s_3''=11011(27),\ s_4''=10000(16)$$

Mutation

Set mutation rate p_m=0.001。

Then, in population S_1

$$4 \times 5 \times 0.001 = 0.02$$

The number of genes which is going to be mutate is 0.02.

Obviously, 0.02 is not enough to achieve 1, so no mutation in this operation.

Then, we get the second population S_2:

s_1=11001(25), s_2=01100(12)

s_3=11011(27), s_4=10000(16)

第二代种群S_2中各染色体的情况

染色体	适应度	选择概率	积累概率	选中次数
s_1=11001	625	0.36	0.36	1
s_2=01100	144	0.08	0.44	1
s_3=11011	729	0.41	0.85	1
s_4=10000	256	0.15	1.00	1

Assume that in this operation, all of the four chromosomes in the population S_2 are selected, then we get the population:

$$s_1' = 11001(25),\ s_2' = 01100(12)$$

$$s_3' = 11011(27),\ s_4' = 10000(16)$$

By doing crossover operation, make s_1' and s_2', s_3' and s_4' exchange the last three bits separately, we have

$$s_1'' = 11100(28),\ s_2'' = 01001(9)$$

$$s_3'' = 11000(24),\ s_4'' = 10011(19)$$

There is no mutation operation also.

The third population $S3$:

s_1=11100（28）, s_2=01001（9）

s_3=11000（24）, s_4=10011（19）

第三代种群S_3中各染色体的情况

染色体	适应度	选择概率	积累概率	估计的选中次数
s_1=11100	784	0.44	0.44	2
s_2=01001	81	0.04	0.48	0
s_3=11000	576	0.32	0.80	1
s_4=10011	361	0.20	1.00	1

The result of selection-reproduction：

$$s_1'=11100(28),\ s_2'=11100(28)$$

$$s_3'=11000(24),\ s_4'=10011(19)$$

By doing crossover operation，make *s*1′and *s*2′，*s*3′and *s*4′exchange the last two bits separately，we have

$$s_1''=11111(31),\ s_2''=11100(28)$$

$$s_3''=11000(24),\ s_4''=10000(16)$$

There is no mutation operation also.

The fourth population S_4:

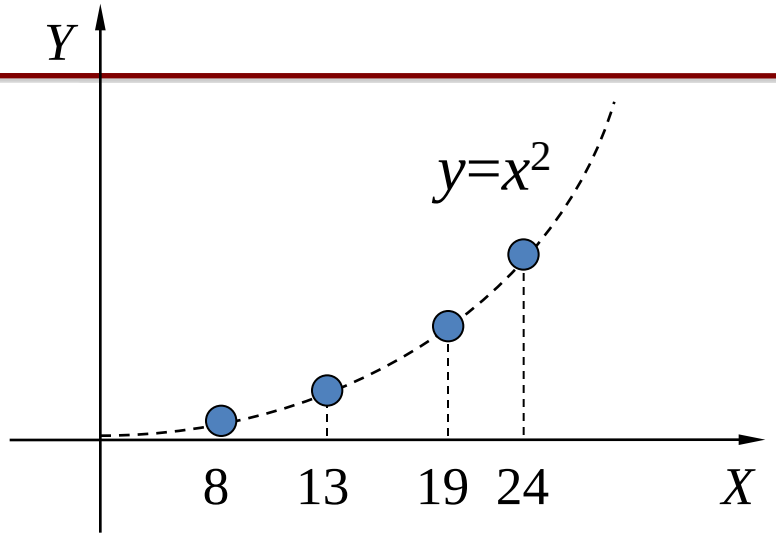
s_1=11111（31）， s_2=11100（28）

s_3=11000（24）， s_4=10000（16）

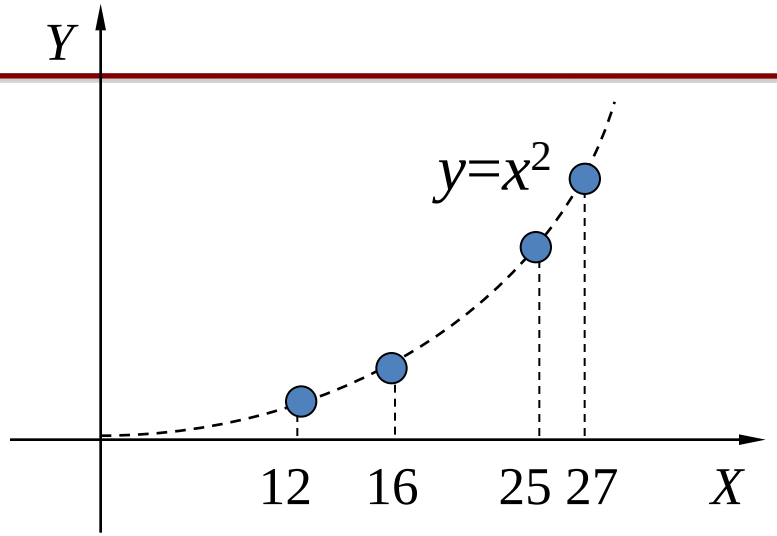
Obviously, we get the fittest chromosome s_1=11111 in this operation. Then comes the end of the genetic operation. Output the string "11111" as the best result .

Then, decode the chromosome "11111" into the phenotype to get the optimal solution: 31。

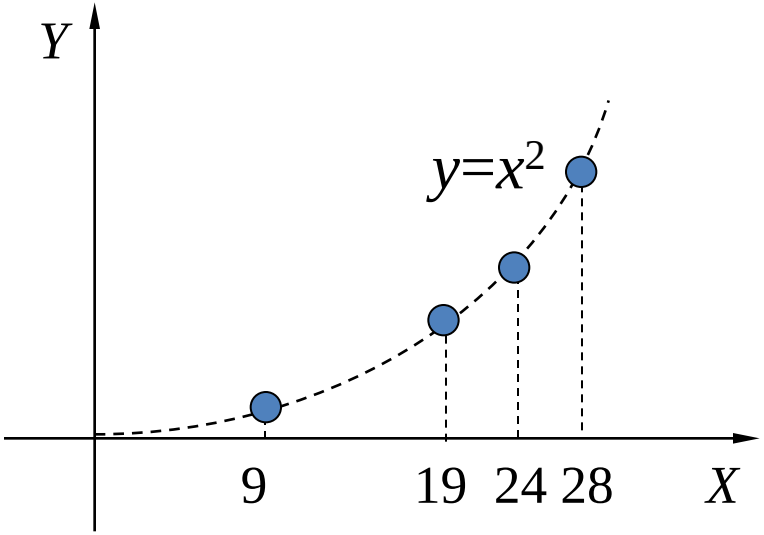
Put 31 into the function $y=x^2$, then we get the solution of the problem, the maximum of function $y=x^2$ is 961。



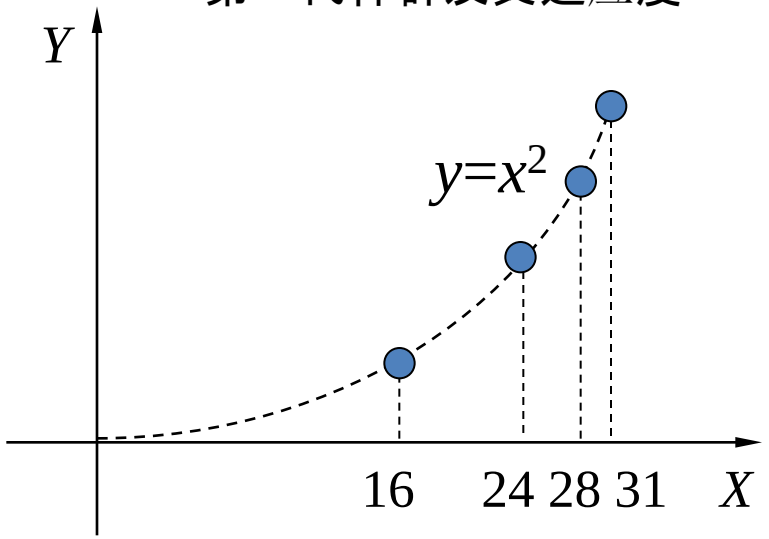
第一代种群及其适应度



第二代种群及其适应度



第三代种群及其适应度



第四代种群及其适应度

Summarize

- The genetic algorithm begins its search with the set of solutions' code set rather than the single solution's code.
- The genetic algorithm can be used to search the global optimum easily.

Practice

Use GA to compute the maximum of the function as follows:

$$\begin{cases} \max & f(x_1,x_2)=x_1^2+x_2^2 \\ \text{s.t.} & x_1 \in \{1,2,3,4,5,6,7\} \\ & x_2 \in \{1,2,3,4,5,6,7\} \end{cases}$$

遗传算法的手工模拟计算示例

为更好地理解遗传算法的运算过程，下面用手工计算来简单地模拟遗传算法的各个主要执行步骤。

例：求下述二元函数的最大值：

$$\begin{cases} \max & f(x_1,x_2)=x_1^2+x_2^2 \\ \text{s.t.} & x_1 \in \{1,2,3,4,5,6,7\} \\ & x_2 \in \{1,2,3,4,5,6,7\} \end{cases}$$

(1) 个体编码

遗传算法的运算对象是表示个体的符号串，所以必须把变量 x_1, x_2 编码为一种符号串。本题中，用无符号二进制整数来表示。

因 x_1, x_2 为 0 ~ 7之间的整数，所以分别用3位无符号二进制整数来表示，将它们连接在一起所组成的6位无符号二进制数就形成了个体的基因型，表示一个可行解。

例如，基因型 X＝101110 所对应的表现型是：x＝[5，6]。

个体的表现型x和基因型X之间可通过编码和解码程序相互转换。

(2) 初始群体的产生

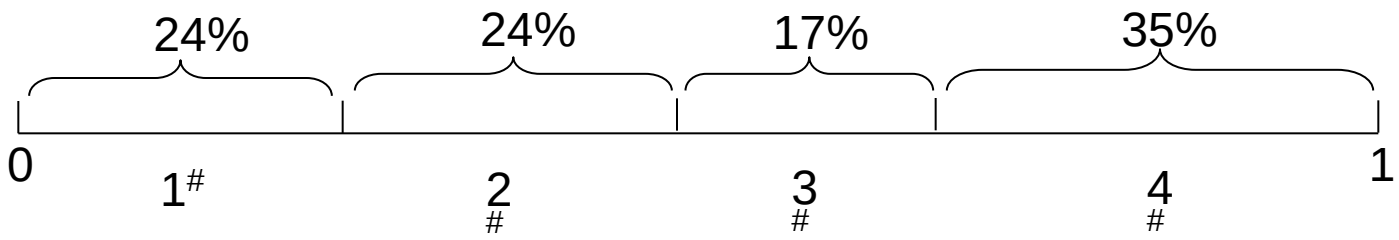
(3) 适应度计算

(4) 选择

本例中，我们采用与适应度成正比的概率来确定各个个体复制到下一代群体中的数量。其具体操作过程是：

- 先计算出群体中所有个体的适应度的总和 Σf_i (i=1,2,...,M);
- 其次计算出每个个体的相对适应度的大小 $f_i / \Sigma f_i$ ，它即为每个个体被遗传到下一代群体中的概率，
- 每个概率值组成一个区域，全部概率值之和为1；
- 最后再产生一个0到1之间的随机数，依据该随机数出现在上述哪一个概率区域内来确定各个个体被选中的次数。

个体编号	初始群体p(0)	x_1	x_2	适值	占总数的百分比	选择次数	选择结果
1	011101	3	5	34	0.24	1	011101
2	101011	5	3	34	0.24	1	111001
3	011100	3	4	25	0.17	0	101011
4	111001	7	1	50	0.35	2	111001
总和				143	1		



(5) 交叉运算

交叉运算是遗传算法中产生新个体的主要操作过程，它以某一概率相互交换某两个个体之间的部分染色体。

本例采用单点交叉的方法，其具体操作过程是：

- 先对群体进行随机配对；
- 其次随机设置交叉点位置；
- 最后再相互交换配对染色体之间的部分基因。

个体编号	选择结果	配对情况	交叉点位置	交叉结果
1 2 3 4	01 1101 11 1001 1010 11 1110 01	1-2 3-4	1-2:2 3-4:4	011001 111101 101001 111011

可以看出，其中新产生的个体“111101”、“111011”的适应度较原来两个个体的适应度都要高。

(6) 变异运算

变异运算是对个体的某一个或某一些基因座上的基因值按某一较小的概率进行改变，它也是产生新个体的一种操作方法。

本例中，我们采用基本位变异的方法来进行变异运算，其具体操作过程是：

- 首先确定出各个个体的基因变异位置，下表所示为随机产生的变异点位置，其中的数字表示变异点设置在该基因座处；
- 然后依照某一概率将变异点的原有基因值取反。

个体编号	交叉结果	变异点	变异结果	子代群体p(1)
1	011001	4	011101	011101
2	111101	5	111111	111111
3	101001	2	111001	111001
4	111011	6	111010	111010

对群体P(t)进行一轮选择、交叉、变异运算之后可得到新一代的群体p(t+1)。

个体编号	子群体p(1)	x_1	x_2	适值	占总数的百分比
1	011101	3	5	34	0.14
2	111111	7	7	98	0.42
3	111001	7	1	50	0.21
4	111010	7	2	53	0.23
总和				235	1

从上表中可以看出，群体经过一代进化之后，其适应度的最大值、平均值都得到了明显的改进。事实上，这里已经找到了最佳个体“111111”。

[注意]

需要说明的是，表中有些栏的数据是随机产生的。这里为了更好地说明问题，我们特意选择了一些较好的数值以便能够得到较好的结果，而在实际运算过程中有可能需要一定的循环次数才能达到这个最优结果。

个体编号	初始群体p(0)	x_1	x_2	适值 $f_i(x_1,x_2)$		占总数的百分比 $f_i / \Sigma f$
1	011101	3	5	34	$\Sigma f_i=143$	0.24
2	101011	5	3	34	$f_{max}=50$	0.24
3	011100	3	4	25	$\bar{f}=35.75$	0.17
4	111001	7	1	50		0.35

选择次数	选择结果	配对情况	交叉点位置	交叉结果	变异点	变异结果
1	011101			011001	4	011101
1	111001	1-2	1-2:2	111101	5	111111
0	101011	3-4	3-4:4	101001	2	111001
2	111001			111011	6	111010

子代群体p(1)	x_1	x_2	适值 $f_i(x_1,x_2)$		占总数的百分比 $f_i / \Sigma f$
011101	3	5	34	$\Sigma f_i=253$	0.14
111111	7	7	98	$f_{max}=98$	0.42
111001	7	1	50	$\bar{f}=58.75$	0.21
111010	7	2	53		0.23

Components of a GA

A problem definition as input, and

- Encoding principles (gene, chromosome)
- Initialization procedure (creation)
- Selection of parents (reproduction)
- Genetic operators (mutation, recombination)
- Evaluation function (environment)
- Termination condition

Representation (encoding)

Possible individual's encoding

- Bit strings (0101 ... 1100)
- Real numbers (43.2 -33.1 ... 0.0 89.2)
- Permutations of element (E11 E3 E7 ... E1 E15)
- Lists of rules (R1 R2 R3 ... R22 R23)
- Program elements (genetic programming)
- ... any data structure ...

Representation (cont.)

When choosing an encoding method rely on the following key ideas

- Use a data structure as close as possible to the natural representation
- Write appropriate genetic operators as needed
- If possible, ensure that all genotypes correspond to feasible solutions
- If possible, ensure that genetic operators preserve feasibility

HW: The Traveling Salesman Problem (TSP)

The traveling salesman must visit every city in his territory exactly once and then return to the starting point; given the cost of travel between all cities, how should he plan his itinerary for minimum total cost of the entire tour?

TSP $\in$ NP-Complete

Note: we shall discuss a single possible approach to approximate the TSP by GAs

TSP (Representation, Evaluation, Initialization and Selection)

A vector $v = (i_1\ i_2 \ldots\ i_n)$ represents a tour (v is a permutation of {1,2,…,n})

Fitness f of a solution is the inverse cost of the corresponding tour

Initialization: use either some heuristics, or a random sample of permutations of {1,2,…,n}

We shall use the fitness proportionate selection

TSP (Crossover1)

O_x – builds offspring by choosing a sub-sequence of a tour from one parent and preserving the relative order of cities from the other parent and feasibility

Example:

p_1 = (1 2 3 4 5 6 7 8 9) and

p_2 = (4 5 2 1 8 7 6 9 3)

First, the segments between cut points are copied into offspring

o_1 = (x x x 4 5 6 7 x x) and

o_2 = (x x x 1 8 7 6 x x)

TSP (Crossover2)

Next, starting from the second cut point of one parent, the cities from the other parent are copied in the same order

The sequence of the cities in the second parent is

9 – 3 – 4 – 5 – 2 – 1 – 8 – 7 – 6

After removal of cities from the first offspring we get

9 – 3 – 2 – 1 – 8

This sequence is placed in the first offspring

o_1 = (2 1 8 4 5 6 7 9 3), and similarly in the second

o_2 = (3 4 5 1 8 7 6 9 2)

TSP (Inversion)

The sub-string between two randomly selected points in the path is reversed

Example:

(1 2 3 4 5 6 7 8 9) is changed into (1 2 7 6 5 4 3 8 9)

Such simple inversion guarantees that the resulting offspring is a legal tour

References

- *C. Darwin. On the Origin of Species by Means of Natural Selection; or, the Preservation of flavored Races in the Struggle for Life. John Murray, London, 1859.*
- *W. D. Hillis. Co-Evolving Parasites Improve Simulated Evolution as an Optimization Procedure. Artificial Life 2, vol 10, Addison-Wesley, 1991.*
- *J. H. Holland. Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor, Michigan, 1975.*
- *Z. Michalewicz. Genetic Algorithms + Data Structures = Evolution Programs. Springer-Verlag, Berlin, third edition, 1996.*
- *M. Sipper. Machine Nature: The Coming Age of Bio-Inspired Computing. McGraw-Hill, New-York, first edition, 2002.*
- *M. Tomassini. Evolutionary algorithms. In E. Sanchez and M. Tomassini, editors, Towards Evolvable Hardware, volume 1062 of Lecture Notes in Computer Science, pages 19-47. Springer-Verlag, Berlin, 1996.*