GENETIC ALGORITHM Lecture Two: GAs: How Do They Work?

Prof. Ming-Jong Yao
Department of Transportation and Logistics Management
National Chiao Tung University

1. Some General Comments

1.1 Maximization of a Function

- Without any loss of generality, we can assume maximization problems only.
- If the optimization problem is to minimize a function *f*, this is equivalent to maximizing a function *g*, where *g* = -*f*, i.e.,

$$\min f(x) = \max g(x) = \max\{-f(x)\}$$

- Moreover, we may assume that the objective function *f* takes positive values on its domain; otherwise we can add some positive constant *C*, i.e.,

$$\max g(x) = \max\{g(x) + C\}$$

- Suppose we wish to maximize a function of k variables, $f(x_1, ..., x_k) : R^k \to R$.

1.2 Representation

- Suppose further that each variable x_i can take values from a domain $D_i = [a_i, b_i] \subseteq \Re$ and $f(x_1, \ldots, x_k) > 0$ for all $x_i \in D_i$.
- We wish to optimize the function f with some required precision: suppose six decimal places for the variables' values is desirable.

- It is clear that to achieve such precision each domain D_i should be cut into $(b_i - a_i)\cdot 10^6$ equal size ranges.
- Let us denote by m_i the smallest integer such that $(b_i - a_i)\cdot 10^6 \leq 2^{m_i} - 1.$
- Then, a representation having each variable x_i coded as a binary string of length m_i clearly satisfies the precision requirement.

- Additionally, the following formula interprets each such string:

$$x_i = a_i + decimal(1001...001_2) \cdot \frac{b_i - a_i}{2^{m_i} - 1}$$

where decimal(string$_2$) represents the decimal value of that binary string.

- Now, each chromosome (as a potential solution) is represented by a binary string of length $m = \sum_{i=1}^{k} m_i$ the first m_1 bits map into a value from the range $[a_1,b_1]$, the next group of m_2 bits map into a value from the range $[a_2,b_2]$, and so on; the last group of m_k bits map into a value from the range $[a_k,b_k]$.

1.3 Algorithm Overview

- To initialize a population, we can simply set some pop-size number of chromosomes randomly in a bitwise fashion.
  - However, if we do have some knowledge about the distribution of potential optima, we may use such information in arranging the set of initial (potential) solutions.

- The rest of the algorithm is straightforward: in each generation we

1. evaluate each chromosome (using the function *f* on the decoded sequences of variables),
2. select new population with respect to the probability distribution based on fitness values, and
3. alter the chromosomes in the new population by *mutation* and *crossover* operators.

- After some number of generations, when no further improvement is observed, the best chromosome represents an (possibly the global) optimal solution.
  - Often we stop the algorithm after a fixed number of iterations depending on speed and resource criteria.

1.4 Selection Process

- The selection process (selection of a new population with respect to the probability distribution based on fitness values): a roulette wheel with slots sized according used.
- We construct such a roulette wheel as follows.
  - We assume here that the fitness values are positive, otherwise, we can use some scaling mechanism.

f'(x) = f(x) + C. C= 10

f(v1) = 5 — f'(v1) = 8
f(v2) = -1 — f'(v2) = 2
f(v3) = 7 — f'(v1) = 10

1. Calculate the fitness value $eval(v_i)$ for each chromosome v_i ($i = 1, \ldots, pop\_size$)
2. Find the total fitness of the population

$$F = \sum_{i=1}^{pop\_size} eval(v_i)$$

3. Calculate the probability of a selection p_i for each chromosome v_i ($i = 1, \ldots, pop\_size$)

$$p_i = \frac{eval(v_i)}{F}$$

4. Calculate a cumulative probability q_i for each chromosome v_i ($i = 1, \ldots, pop\_size$)

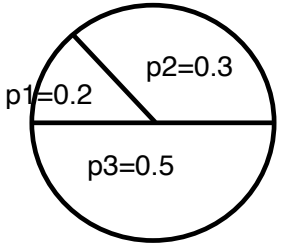
$$q_i = \sum_{j=1}^{i} p_j$$

- The selection process is based on spinning the roulette wheel *pop_size* times: each time we select a single chromosome for a new population in the following way.
  1. Generate a random (float) number r from the range [0, 1].
  2. If $r < q_1$, then select the first chromosome v_1 ; otherwise select the ith chromosome v_i , such that $q_{i-1} < r \leq q_i$ $(2 \leq i \leq pop\_size)$
- Obviously, some chromosomes would be selected more than once.

➢ This is in accordance with the Schema Theorem (see next chapter): the best chromosomes get more copies, the average stay even, and the worst die off.

A simple example

Chromosome	Evaluation	Probability (p_i) = Eval(v_i) /F	Cumm. Prob.(q_i)	Random number(r)
			q_0=0.0	
v_1	Eval(v_1)=2	p_1=2/10=0.2	q_1=0.2	[0,0.2)
v_2	Eval(v_2)=3	p_2=3/10=0.3	q_2=0.5	[0.2,0.5)
v_3	Eval(v_3)=5	p_3=5/10=0.5	q_3=1.0	[0.5,1.0)
	F=10			



1.5 Crossover Operation

- One of the parameters of a genetic system is probability of crossover p_c .
- This probability gives us the expected number $p_c \cdot$pop_size of chromosomes which undergo the crossover operation.
- We proceed in the following way:
  1. Select candidates for crossover: For each chromosome in the (new) population:
     a) Generate a random (float) number r from the range [0, 1];
     b) If $r < p_c$, select given chromosome for crossover.

1.5 Crossover Operation

2. Mate selected chromosomes randomly: for each pair of coupled chromosomes we generate a random integer number pos from the range [1, *m*-1] (*m* is the total length, i.e., number of bits, in a chromosome).

=cutpoint

- Question: How to get pos?

round_up(r * (m-1)), r e [0, 1)

- The number pos indicates the position of the crossing point.
- Two chromosomes

$$(b_1 b_2 \ldots b_{pos} b_{pos+1} \ldots b_m)$$

and

$$(c_1 c_2 \ldots c_{pos} c_{pos+1} \ldots c_m)$$

are replaced by a pair of their offspring:

$$(b_1 b_2 \ldots b_{pos} c_{pos+1} \ldots c_m)$$

and

$$(c_1 c_2 \ldots c_{pos} b_{pos+1} \ldots b_m)$$

1.6 Mutation Operation

- The next operator, mutation, is performed on a bit-by-bit basis. Another parameter of the genetic system, probability of mutation p_m, gives us the expected number of mutated bits $p_m \cdot m \cdot pop\_size$.

m=total bits in a chromosome

- Every bit (in all chromosomes in the whole population) has an equal chance to undergo mutation, i.e., change from 0 to 1 or vice versa. So we proceed in the following way.

- For each chromosome in the current (i.e., after crossover) population and for each bit within the chromosome:

1. Generate a random (float) number r from the range [0, 1];
2. If $r < p_m$, mutate the bit.

- Following selection, crossover, and mutation, the new population is ready for its next evaluation.
- This evaluation is used to build the probability distribution (for the next selection process), i.e., for a construction of a roulette wheel with slots sized according to current fitness values.
- The rest of the evolution is just cyclic repetition of the above steps.

2 Numerical Example

- We assume that the population size *pop_size* = 20, and the probabilities of genetic operators are p_c = 0.25 and p_m = 0.01. Let us assume also that we maximize the following function:

$$f(x_1, x_2) = 21.5 + x_1 \cdot \sin(4\pi x_1) + x_2 \cdot \sin(20\pi x_2),$$

where $-3.0 < x_1 < 12.1$ and $4.11 < x_2 < 5.8$.

- The graph of the function *f* is given in Figure 1.

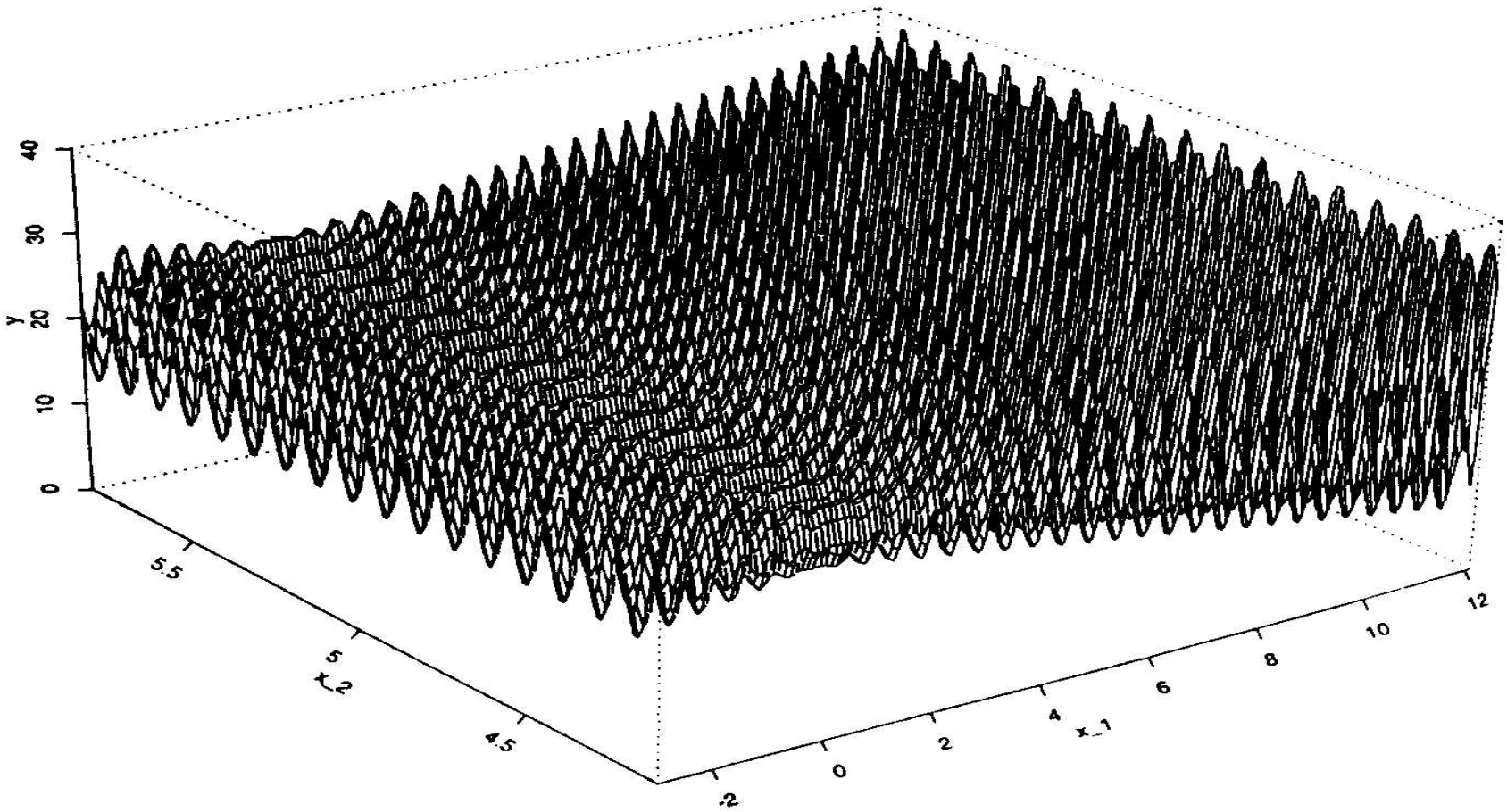


Figure 1: Graph of the function

$$f(x_1, x_2) = 21.5 + x_1 \cdot \sin(4\pi x_1) + x_2 \cdot \sin(20\pi x_2),$$

- Let assume further that the required precision is four decimal places for each variable.
- The domain of variable x_1 has length 15.1; the precision requirement implies that the range [−3.0, 12.1] should be divided into at least $15.1 \cdot \underline{10000}$ equal size ranges.
- This means that 18 bits are required as the first part of the chromosome: $2^{17} < 151000 < 2^{18}$.
- The domain of variable x_2 has length 1.7; the precision requirement implies that the range [4.1, 5.8] should be divided into at least $1.7 \cdot 10000$ equal size ranges.
- This means that 15 bits are required as the second part of the chromosome: $2^{14} < 17000 < 2^{15}$.

- The total length of a chromosome (solution vector) is then m = 18 +15 = 33 bits; the first 18 bits code x_1 and remaining 15 bits (19-33) code x_2.

1. Let us consider an example chromosome: (010001001011010000111110010100010).
   a. The first 18 bits, 010001001011010000, represent

$$x_1 = -3.0 + decimal(010001001011010000_2) \cdot \frac{12.1-(-3.0)}{2^{18}-1}$$

$$= -3.0 + 70352 \cdot \frac{15.1}{26143} = 1.052426$$

b. The next 15 bits, 111110010100010, represent

$$x_2 = 4.1 + decimal(111110010100010_2) \cdot \frac{5.8 - 4.1}{2^{15} - 1}$$

$$= 4.1 + 1.655330 = 5.755330$$

c. So the chromosome (010001001011010000111110010100010) corresponds to

$$(x_1, x_2) = (1.052426, 5.755330).$$

d. The fitness value for this chromosome is

$$f(1.052426, 5.755330) = 20.252640$$

2. To optimize the function *f* using a genetic algorithm, we create a population of *pop_size* = 20 chromosomes.

- All 33 bits in all chromosomes are initialized *randomly*.

a. Assume that after the initialization process we get the following population:

$$
\begin{aligned}
v_1 &= (100110100000001111111010011011111)\\
v_2 &= (111000100100110111001010100011010)\\
v_3 &= (000010000011001000001010111011101)\\
v_4 &= (100011000101101001111000001110010)\\
v_5 &= (000111011001010011010111111000101)\\
v_6 &= (000101000010010101001010111111011)\\
v_7 &= (001000100000110101111011011111011)\\
v_8 &= (100001100001110100010110101100111)\\
v_9 &= (010000000101100010110000001111100)\\
v_{10} &= (000001111000110000011010000111011)\\
v_{11} &= (011001111110110101100001101111000)\\
v_{12} &= (110100010111101101000101010000000)\\
v_{13} &= (111011111010001000110000001000110)\\
v_{14} &= (010010011000001010100111100101001)\\
v_{15} &= (111011101101110000100011111011110)\\
v_{16} &= (110011110000011111100001101001011)\\
v_{17} &= (011010111111001111010001101111101)\\
v_{18} &= (011101000000001110100111110101101)\\
v_{19} &= (000101010011111111110000110001100)\\
v_{20} &= (101110010110011110011000101111110)
\end{aligned}
$$

b. During the evaluation phase we decode each chromosome and calculate the fitness function values from (x_1,x_2) values just decoded. We get:

$$
\begin{aligned}
eval(\boldsymbol{v}_1) &= f(6.084492, 5.652242) = 26.019600 \\
eval(\boldsymbol{v}_2) &= f(10.348434, 4.380264) = 7.580015 \\
eval(\boldsymbol{v}_3) &= f(-2.516603, 4.390381) = 19.526329 \\
eval(\boldsymbol{v}_4) &= f(5.278638, 5.593460) = 17.406725 \\
eval(\boldsymbol{v}_5) &= f(-1.255173, 4.734458) = 25.341160 \\
eval(\boldsymbol{v}_6) &= f(-1.811725, 4.391937) = 18.100417 \\
eval(\boldsymbol{v}_7) &= f(-0.991471, 5.680258) = 16.020812 \\
eval(\boldsymbol{v}_8) &= f(4.910618, 4.703018) = 17.959701 \\
eval(\boldsymbol{v}_9) &= f(0.795406, 5.381472) = 16.127799 \\
eval(\boldsymbol{v}_{10}) &= f(-2.554851, 4.793707) = 21.278435 \\
eval(\boldsymbol{v}_{11}) &= f(3.130078, 4.996097) = 23.410669 \\
eval(\boldsymbol{v}_{12}) &= f(9.356179, 4.239457) = 15.011619 \\
eval(\boldsymbol{v}_{13}) &= f(11.134646, 5.378671) = 27.316702 \\
eval(\boldsymbol{v}_{14}) &= f(1.335944, 5.151378) = 19.876294 \\
eval(\boldsymbol{v}_{15}) &= f(11.089025, 5.054515) = 30.060205 \\
eval(\boldsymbol{v}_{16}) &= f(9.211598, 4.993762) = 23.867227 \\
eval(\boldsymbol{v}_{17}) &= f(3.367514, 4.571343) = 13.696165 \\
eval(\boldsymbol{v}_{18}) &= f(3.843020, 5.158226) = 15.414128 \\
eval(\boldsymbol{v}_{19}) &= f(-1.746635, 5.395584) = 20.095903 \\
eval(\boldsymbol{v}_{20}) &= f(7.935998, 4.757338) = 13.666916
\end{aligned}
$$

c. It is clear, that the chromosome v_{15} is the strongest one, and the chromosome v_2 the weakest.

$$
\begin{aligned}
eval(\boldsymbol{v}_1) &= f(6.084492, 5.652242) = 26.019600 \\
eval(\boldsymbol{v}_2) &= f(10.348434, 4.380264) = 7.580015 \\
eval(\boldsymbol{v}_3) &= f(-2.516603, 4.390381) = 19.526329 \\
eval(\boldsymbol{v}_4) &= f(5.278638, 5.593460) = 17.406725 \\
eval(\boldsymbol{v}_5) &= f(-1.255173, 4.734458) = 25.341160 \\
eval(\boldsymbol{v}_6) &= f(-1.811725, 4.391937) = 18.100417 \\
eval(\boldsymbol{v}_7) &= f(-0.991471, 5.680258) = 16.020812 \\
eval(\boldsymbol{v}_8) &= f(4.910618, 4.703018) = 17.959701 \\
eval(\boldsymbol{v}_9) &= f(0.795406, 5.381472) = 16.127799 \\
eval(\boldsymbol{v}_{10}) &= f(-2.554851, 4.793707) = 21.278435 \\
eval(\boldsymbol{v}_{11}) &= f(3.130078, 4.996097) = 23.410669 \\
eval(\boldsymbol{v}_{12}) &= f(9.356179, 4.239457) = 15.011619 \\
eval(\boldsymbol{v}_{13}) &= f(11.134646, 5.378671) = 27.316702 \\
eval(\boldsymbol{v}_{14}) &= f(1.335944, 5.151378) = 19.876294 \\
eval(\boldsymbol{v}_{15}) &= f(11.089025, 5.054515) = 30.060205 \\
eval(\boldsymbol{v}_{16}) &= f(9.211598, 4.993762) = 23.867227 \\
eval(\boldsymbol{v}_{17}) &= f(3.367514, 4.571343) = 13.696165 \\
eval(\boldsymbol{v}_{18}) &= f(3.843020, 5.158226) = 15.414128 \\
eval(\boldsymbol{v}_{19}) &= f(-1.746635, 5.395584) = 20.095903 \\
eval(\boldsymbol{v}_{20}) &= f(7.935998, 4.757338) = 13.666916
\end{aligned}
$$

- Now the system constructs a roulette wheel for the selection process. The total fitness of the population is

$$F = \sum_{i=1}^{20} eval(v_i) = 387.776822$$

a) The probability of a selection p_i for each chromosome v_i (i=1, …,20) is:

$$p_1 = eval(\boldsymbol{v}_1)/F = 0.067099 \qquad p_2 = eval(\boldsymbol{v}_2)/F = 0.019547$$
$$p_3 = eval(\boldsymbol{v}_3)/F = 0.050355 \qquad p_4 = eval(\boldsymbol{v}_4)/F = 0.044889$$
$$p_5 = eval(\boldsymbol{v}_5)/F = 0.065350 \qquad p_6 = eval(\boldsymbol{v}_6)/F = 0.046677$$
$$p_7 = eval(\boldsymbol{v}_7)/F = 0.041315 \qquad p_8 = eval(\boldsymbol{v}_8)/F = 0.046315$$
$$p_9 = eval(\boldsymbol{v}_9)/F = 0.041590 \qquad p_{10} = eval(\boldsymbol{v}_{10})/F = 0.054873$$
$$p_{11} = eval(\boldsymbol{v}_{11})/F = 0.060372 \qquad p_{12} = eval(\boldsymbol{v}_{12})/F = 0.038712$$
$$p_{13} = eval(\boldsymbol{v}_{13})/F = 0.070444 \qquad p_{14} = eval(\boldsymbol{v}_{14})/F = 0.051257$$
$$p_{15} = eval(\boldsymbol{v}_{15})/F = 0.077519 \qquad p_{16} = eval(\boldsymbol{v}_{16})/F = 0.061549$$
$$p_{17} = eval(\boldsymbol{v}_{17})/F = 0.035320 \qquad p_{18} = eval(\boldsymbol{v}_{18})/F = 0.039750$$
$$p_{19} = eval(\boldsymbol{v}_{19})/F = 0.051823 \qquad p_{20} = eval(\boldsymbol{v}_{20})/F = 0.035244$$

b) The cumulative probabilities q_i for each chromosome v_i (i=1, …,20) is:

q1+p2

$$q_1 = 0.067099 \quad q_2 = 0.086647 \quad q_3 = 0.137001 \quad q_4 = 0.181890$$
$$q_5 = 0.247240 \quad q_6 = 0.293917 \quad q_7 = 0.335232 \quad q_8 = 0.381546$$
$$q_9 = 0.423137 \quad q_{10} = 0.478009 \quad q_{11} = 0.538381 \quad q_{12} = 0.577093$$
$$q_{13} = 0.647537 \quad q_{14} = 0.698794 \quad q_{15} = 0.776314 \quad q_{16} = 0.837863$$
$$q_{17} = 0.873182 \quad q_{18} = 0.912932 \quad q_{19} = 0.964756 \quad q_{20} = 1.000000$$

4. Now we are ready to spin the roulette wheel 20 times; each time we select a single chromosome for a new population.
   a) Let us assume that a (random) sequence of 20 numbers from the range [0, 1] is:

0.513870	0.175741	0.308652	0.534534	0.947628
0.171736	0.702231	0.226431	0.494773	0.424720
0.703899	0.389647	0.277226	0.368071	0.983437
0.005398	0.765682	0.646473	0.767139	0.780237

b) The first number $r = 0.513870$ is greater than q_{10} and smaller than q_{11}, meaning the chromosome is selected for the new population; the second number $r = 0.175741$ is greater than q_3 and smaller than q_4 meaning the chromosome v_4 is selected for the new population, etc.

$q_1 = 0.067099$	$q_2 = 0.086647$	$q_3 = 0.137001$	$q_4 = 0.181890$
$q_5 = 0.247240$	$q_6 = 0.293917$	$q_7 = 0.335232$	$q_8 = 0.381546$
$q_9 = 0.423137$	$q_{10} = 0.478009$	$q_{11} = 0.538381$	$q_{12} = 0.577093$
$q_{13} = 0.647537$	$q_{14} = 0.698794$	$q_{15} = 0.776314$	$q_{16} = 0.837863$
$q_{17} = 0.873182$	$q_{18} = 0.912932$	$q_{19} = 0.964756$	$q_{20} = 1.000000$

5. Finally, the new population consists of the following chromosomes:

$$
\begin{aligned}
v'_1 &= (011001111110110101100001101111000)\ (v_{11})\\
v'_2 &= (100011000101101001111000001110010)\ (v_4)\\
v'_3 &= (001000100000110101111011011111011)\ (v_7)\\
v'_4 &= (011001111110110101100001101111000)\ (v_{11})\\
v'_5 &= (000101010011111111110000110001100)\ (v_{19})\\
v'_6 &= (100011000101101001111000001110010)\ (v_4)\\
v'_7 &= (111011101101110000100011111011110)\ (v_{15})\\
v'_8 &= (000111011001010011010111111000101)\ (v_5)\\
v'_9 &= (011001111110110101100001101111000)\ (v_{11})\\
v'_{10} &= (000010000011001000001010111011101)\ (v_3)\\
v'_{11} &= (111011101101110000100011111011110)\ (v_{15})\\
v'_{12} &= (010000000101100010110000001111100)\ (v_9)\\
v'_{13} &= (000101000010010101001010111111011)\ (v_6)\\
v'_{14} &= (100001100001110100010110101100111)\ (v_8)\\
v'_{15} &= (101110010110011110011000101111110)\ (v_{20})\\
v'_{16} &= (100110100000001111111010011011111)\ (v_1)\\
v'_{17} &= (000001111000110000011010000111011)\ (v_{10})\\
v'_{18} &= (111011111010001000110000001000110)\ (v_{13})\\
v'_{19} &= (111011101101110000100011111011110)\ (v_{15})\\
v'_{20} &= (110011110000011111100001101001011)\ (v_{16})
\end{aligned}
$$

6. Now we are ready to apply the recombination operator crossover, to the individuals in the new population.
   a) The probability of crossover $p_c = 0.25$, so we expect that (on average) 25% of chromosomes (i.e., 5 out of 20) undergo crossover.
   b) We proceed in the following way: for each chromosome in the (new) population we generate a random number r from the range [0, 1]; if $r < 0.25$, we select a given chromosome for crossover.

c) Let us assume that the sequence of random numbers is:

0.822951	0.151932	0.625477	0.314685	0.346901
0.917204	0.519760	0.401154	0.606758	0.785402
0.031523	0.869921	0.166525	0.674520	0.758400
0.581893	0.389248	0.200232	0.355635	0.826927

d) This means that the chromosomes v_2', v_{11}', v_{13}' and v_{18}' were selected for crossover. (We were lucky: the number of selected chromosomes is even, so we can pair them easily.

- If the number of selected chromosomes were odd, we would either add one extra chromosome or remove one selected chromosome - this choice is made randomly as well.)

e) Now we mate selected chromosomes randomly: say, the first two (i.e., v_2' and v_{11}') and the next two (i.e., v_{13}' and v_{18}') are coupled together.

f) For each of these two pairs, we generate a random integer number *pos* from the range [1, 32] (33 is the total length - number of bits - in a chromosome). The number pos indicates the position of the crossing point.

i. The first pair of chromosomes is

$$v_2' = (100011000\,|\,101101001111000001110010)$$
$$v_{11}' = (111011101\,|\,101110000100011111011110)$$

and the generated number *pos* = 9. These chromosomes are cut after the 9th bit and replaced by a pair of their offspring:

$$v_2'' = (100011000\,|\,101110000100011111011110)$$
$$v_{11}'' = (111011101\,|\,101101001111000001110010).$$

ii. The second pair of chromosomes is

$$v'_{13} = (000101000010010101001010111111011)$$
$$v'_{18} = (111011111010001000110000001000110)$$

and the generated number *pos* = 20. These chromosomes are replaced by a pair of their offspring:

$$v''_{13} = (000101000010010101000000001000110)$$
$$v''_{18} = (111011111010001000110101011111011).$$

The current version of (the first 7 chromosomes of) the population is:

$$
\begin{aligned}
\boldsymbol{v}'_1 &= (011001111110110101100001101111000) \\
\boldsymbol{v}''_2 &= \boxed{(100011000101110000100011111011110)} \\
\boldsymbol{v}'_3 &= (001000100000110101111011011111011) \\
\boldsymbol{v}'_4 &= (011001111110110101100001101111000) \\
\boldsymbol{v}'_5 &= (000101010011111111110000110001100) \\
\boldsymbol{v}'_6 &= (100011000101101001111000001110010) \\
\boldsymbol{v}'_7 &= (111011101101110000100011111011110)
\end{aligned}
$$

7. The next operator, mutation, is performed on a bit-by-bit basis. The probability of mutation

   $p_m = 0.01$ so we expect that (on average) 1% of bits would undergo mutation. There are

   $m \cdot pop\_size = 33 \cdot 20 = 660$ bits in the whole population; we expect (on average) 6.6 mutations per generation.

   a) Every bit has an equal chance to be mutated, so, for every bit in the population, we generate a random number r from the range [0,1]; if $r < 0.01$, we mutate the bit.

b) This means that we have to generate 660 random numbers. In a sample run, 5 of these numbers were smaller than 0.01; the bit number and the random number are listed below:

Bit position	Random number
112	0.000213
349	0.009945
418	0.008809
429	0.005425
602	0.002836

c) The following table translates the bit position into chromosome number and the bit number within the chromosome:

Bit position	Chromosome number	Bit number within chromosome
112	4	13
349	11	19
418	13	22
429	13	33
602	19	8

d) This means that four chromosomes are affected by the mutation operator; one of the chromosomes (the 13th) has <u>two</u> bits changed.

8. The final population is listed below; the mutated bits are typed in boldface. We have just completed one iteration (i.e., one generation) of the while loop in the genetic procedure.

$$
\begin{aligned}
\boldsymbol{v}_1 &= (011001111110110101100001101111000)\\
\boldsymbol{v}_2 &= (100011000101110000100011111011110)\\
\boldsymbol{v}_3 &= (001000100000110101111011011111011)\\
\boldsymbol{v}_4 &= (011001111110\boxed{0}1010110000110111 1000)\\
\boldsymbol{v}_5 &= (000101010011111111110000110001100)\\
\boldsymbol{v}_6 &= (100011000101101001111000001110010)\\
\boldsymbol{v}_7 &= (111011101101110000100011111011110)\\
\boldsymbol{v}_8 &= (000111011001010011010111111000101)\\
\boldsymbol{v}_9 &= (011001111110110101100001101111000)\\
\boldsymbol{v}_{10} &= (000010000011001000001010111011101)\\
\boldsymbol{v}_{11} &= (111011101101101001\boxed{0}1100000011100 10)\\
\boldsymbol{v}_{12} &= (010000000101100010110000001111100)\\
\boldsymbol{v}_{13} &= (000101000010010101000\boxed{1}0000100001\boxed{1}1)\\
\boldsymbol{v}_{14} &= (100001100001110100010110101100111)\\
\boldsymbol{v}_{15} &= (101110010110011110011000101111110)\\
\boldsymbol{v}_{16} &= (100110100000001111111010011011111)\\
\boldsymbol{v}_{17} &= (000001111000110000011010000111011)\\
\boldsymbol{v}_{18} &= (111011111010001000111010111111011)\\
\boldsymbol{v}_{19} &= (111011\boxed{0}0101110000100011111011110)\\
\boldsymbol{v}_{20} &= (110011110000011111100001101001011)
\end{aligned}
$$

9. It is interesting to examine the results of the evaluation process of the new population. During the evaluation phase we decode each chromosome and calculate the fitness function values from (x_1,x_2) just decoded. We get:

a) Note that the total fitness of the new population f is 447.049688, <u>much higher</u> than total fitness of the previous population, 387.776822. (best environment)

b) Also, the best chromosome now (v_{11}) has a better evaluation (33.351874) than the best chromosome (v_{15}) from the previous population (30.060205). (best individual)

$$
\begin{aligned}
eval(\boldsymbol{v}_1) &= f(3.130078, 4.996097) = 23.410669 \\
eval(\boldsymbol{v}_2) &= f(5.279042, 5.054515) = 18.201083 \\
eval(\boldsymbol{v}_3) &= f(-0.991471, 5.680258) = 16.020812 \\
eval(\boldsymbol{v}_4) &= f(3.128235, 4.996097) = 23.412613 \\
eval(\boldsymbol{v}_5) &= f(-1.746635, 5.395584) = 20.095903 \\
eval(\boldsymbol{v}_6) &= f(5.278638, 5.593460) = 17.406725 \\
eval(\boldsymbol{v}_7) &= f(11.089025, 5.054515) = 30.060205 \\
eval(\boldsymbol{v}_8) &= f(-1.255173, 4.734458) = 25.341160 \\
eval(\boldsymbol{v}_9) &= f(3.130078, 4.996097) = 23.410669 \\
eval(\boldsymbol{v}_{10}) &= f(-2.516603, 4.390381) = 19.526329 \\
eval(\boldsymbol{v}_{11}) &= f(11.088621, 4.743434) = 33.351874 \\
eval(\boldsymbol{v}_{12}) &= f(0.795406, 5.381472) = 16.127799 \\
eval(\boldsymbol{v}_{13}) &= f(-1.811725, 4.209937) = 22.692462 \\
eval(\boldsymbol{v}_{14}) &= f(4.910618, 4.703018) = 17.959701 \\
eval(\boldsymbol{v}_{15}) &= f(7.935998, 4.757338) = 13.666916 \\
eval(\boldsymbol{v}_{16}) &= f(6.084492, 5.652242) = 26.019600 \\
eval(\boldsymbol{v}_{17}) &= f(-2.554851, 4.793707) = 21.278435 \\
eval(\boldsymbol{v}_{18}) &= f(11.134646, 5.666976) = 27.591064 \\
eval(\boldsymbol{v}_{19}) &= f(11.059532, 5.054515) = 27.608441 \\
eval(\boldsymbol{v}_{20}) &= f(9.211598, 4.993762) = 23.867227
\end{aligned}
$$

10. Now we are ready to run the selection process again and apply the genetic operators, evaluate the next generation, etc.
    a) After 1000 generations the population is:

$$
\begin{aligned}
\boldsymbol{v}_1 &= (111011110110011011100101010111011)\\
\boldsymbol{v}_2 &= (111001100110000100010101010111000)\\
\boldsymbol{v}_3 &= (111011110111011011100101010111011)\\
\boldsymbol{v}_4 &= (111001100010000110000101010111001)\\
\boldsymbol{v}_5 &= (111011110111011011100101010111011)\\
\boldsymbol{v}_6 &= (111001100110000100000100010100001)\\
\boldsymbol{v}_7 &= (110101100010010010001100010110000)\\
\boldsymbol{v}_8 &= (111101100010001010001101010010001)\\
\boldsymbol{v}_9 &= (111001100010010010001100010110001)\\
\boldsymbol{v}_{10} &= (111011110111011011100101010111011)\\
\boldsymbol{v}_{11} &= (110101100000010010001100010110000)\\
\boldsymbol{v}_{12} &= (110101100010010010001100010110001)\\
\boldsymbol{v}_{13} &= (111011110111011011100101010111011)\\
\boldsymbol{v}_{14} &= (111001100110000100000101010111011)\\
\boldsymbol{v}_{15} &= (111001101010111001010100110110001)\\
\boldsymbol{v}_{16} &= (111001100110000101000100010100001)\\
\boldsymbol{v}_{17} &= (111001100110000100000101010111011)\\
\boldsymbol{v}_{18} &= (111001100110000100000101010111001)\\
\boldsymbol{v}_{19} &= (111101100010001010001110000010001)\\
\boldsymbol{v}_{20} &= (111001100110000100000101010111001)
\end{aligned}
$$

The fitness values are:

$$
\begin{aligned}
eval(\boldsymbol{v}_1) &= f(11.120940, 5.092514) = 30.298543 \\
eval(\boldsymbol{v}_2) &= f(10.588756, 4.667358) = 26.869724 \\
eval(\boldsymbol{v}_3) &= f(11.124627, 5.092514) = 30.316575 \\
eval(\boldsymbol{v}_4) &= f(10.574125, 4.242410) = 31.933120 \\
eval(\boldsymbol{v}_5) &= f(11.124627, 5.092514) = 30.316575 \\
eval(\boldsymbol{v}_6) &= f(10.588756, 4.214603) = 34.356125 \\
eval(\boldsymbol{v}_7) &= f(9.631066, 4.427881) = 35.458636 \\
eval(\boldsymbol{v}_8) &= f(11.518106, 4.452835) = 23.309078 \\
eval(\boldsymbol{v}_9) &= f(10.574816, 4.427933) = 34.393820 \\
eval(\boldsymbol{v}_{10}) &= f(11.124627, 5.092514) = 30.316575 \\
eval(\boldsymbol{v}_{11}) &= f(9.623693, 4.427881) = 35.477938 \\
eval(\boldsymbol{v}_{12}) &= f(9.631066, 4.427933) = 35.456066 \\
eval(\boldsymbol{v}_{13}) &= f(11.124627, 5.092514) = 30.316575 \\
eval(\boldsymbol{v}_{14}) &= f(10.588756, 4.242514) = 32.932098 \\
eval(\boldsymbol{v}_{15}) &= f(10.606555, 4.653714) = 30.746768 \\
eval(\boldsymbol{v}_{16}) &= f(10.588814, 4.214603) = 34.359545
\end{aligned}
$$

11. It is relatively easy to keep track of the best individual in the evolution process. In a way, the algorithm would report the best value found during the whole process (as opposed to *the best value in the final population*).