An Introduction to Genetic Algorithms

Beatrice Lazzerini

Department of Information Engineering

University of Pisa

ITALY

- *Genetic algorithms* (GAs) are search and optimization methods that mimic natural evolution.
- Progress in natural evolution is based on three fundamental processes:
  - *selection* picks the individuals that will produce offspring,
  - *recombination* (or *crossover*) combines two different individuals to produce offspring,
  - *mutation* is the random variation of the existing genetic material.
- GAs perform a systematic random search in order to improve the likelihood of finding globally optimal solutions.

- A GA considers a population of *chromosomes* (or *individuals*), each of which encodes a potential solution to the optimization problem.
- A chromosome is a set of genes, each of which represents a specific feature of an individual.
- Each chromosome corresponds to a point in the search space.
- The quality of a chromosome with respect to the optimization task is defined by a scalar objective function (*fitness* function).
- Highly fit individuals are more likely to be selected to produce offspring.

Selection, Crossover, Mutation

- The search progress is achieved by changing the chromosome population, generation after generation.
- A *selection* operator is applied to take advantage of the best characteristics of good candidate solutions in order to improve these solutions over the generations.
- Genetic operators (*crossover* and *mutation*) are applied to generate new candidate solutions:
  - *crossover* combines (mates) two chromosomes (parents) to produce new chromosomes (offspring),
  - *mutation* of newly generated offspring restores the lost or unexplored genetic material, thus inducing variability in the population and preventing premature convergence towards suboptimal solutions.

- Iterative genetic search of a solution implies a balance between *exploration* of the search space and *exploitation* of the best available solution.
- Exploration and exploitation correspond to *global search* and *local search*, respectively.
- If the solutions are exploited too much, premature convergence of the search process can occur. In this case, the search ceases to progress and the procedure may end with an unacceptable solution.
- On the other hand, if a particular emphasis is placed on exploration, the information already available may not be properly exploited. In this case, the search process may become very slow.
- GAs make a reasonable compromise between exploitation and exploration of the search space: they explore the problem space through crossover and mutation and exploit the "good" genetic material in the current solution set through selection.

- A GA is an iterative global search procedure whose goal is the optimization of the *fitness function*, the definition of which depends on the problem:
  - for a numerical or combinatorial optimization problem, the fitness function usually coincides with the objective function,
  - for other problems the fitness function can be a cost function, a loss function, etc.,
  - sometimes the quality of a solution can be assessed by comparing it with a set of examples (test cases).
- There must be enough fitness difference among individuals in the population to guide a true evolutionary search process. An algorithm in which fitness is not discriminatory enough can degenerate into a multi-member blind search.

Search progress

- The dynamics of the search process of a genetic algorithm is obtained by chromosome recombination and modification.
- At each iteration *t*, the algorithm creates a population which is interpreted as the *generation* at time *t*.
- In the standard GA approach, all generations have the same size, i.e., the same number of individuals.
- Usually, the new generation contains better individuals, i.e., individuals that have better fitness value.
- With the increase of generations, we can observe a trend of evolution towards the global optimum of the fitness function.

Basic elements of GAs

- The new generation $P(t+1)$ of *independent* individuals representing solutions of the problem is obtained from the population $P(t)$ by means of the following steps.
- **1. Evaluation**

  The GA evaluates the fitness of each individual of the current population.
- **2. Selection for recombination**

  The individuals of population $P(t)$ are selected for recombination according to their fitness.

  The selected individuals represent an intermediate population $P1$.

  Individuals of $P1$ will enter the mating pool with a given probability (*crossover probability*).

- **3. Recombination and modification**

  Individuals in the mating pool are mated using the crossover. A new <u>intermediate</u> <u>population</u> *P*2 is obtained.

  The resulting offspring (population *P*2) are modified by mutation with a *mutation probability*, thus producing population *P*3.

- **4. Selection for replacement and survival**

  The new generation *P*(*t*+1) contains the individuals of population *P*3 and may include other selected individuals.

  Some replacement methods are strictly *generational* procedures; other methods allow generations to overlap.

Canonical GA

- In the canonical GA, the number of chromosomes in each generation is constant.
- The chromosomes are binary strings of constant length.
- The value of each gene is either 0 or 1.

S1. Set $t = 0$
S2. Initialize a chromosome population $P(t)$
(Random initialization is the basic choice)
S3. Evaluate $P(t)$ using a fitness measure
S4. **while** (termination condition not satisfied) **do**
begin
S4.1. Select for recombination chromosomes from $P(t)$
Let $P1$ be the set of selected chromosomes
Choose individuals from $P1$ to enter the mating pool (MP)
S4.2. Recombine the chromosomes in MP to form population $P2$
Mutate chromosomes in $P2$ to form population $P3$
S4.3. Select for replacement from $P3$ and $P(t)$ to form $P(t+1)$
S4.4. Set $t = t+1$
end

Termination condition

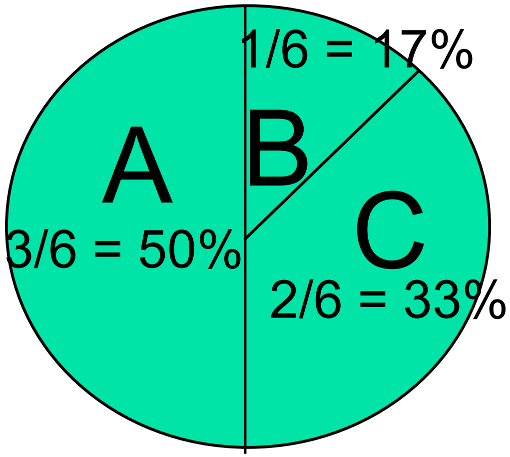
- maximum number of generations,
- desired fitness achieved by the best individual in the population,
- number of successive generations for which there is no change in the population,
- average fitness of all individuals greater than a given percentage (e.g., 97-98%) of the fitness of the best individual.

Solution

- The solution to the problem is generally considered to be the best individual of the last generation.
- But there is no guarantee that a better individual has not previously been obtained. Therefore, it is useful to keep the best individual obtained up to each instant *t*. This can be achieved by means of *elitism*, which means that a percentage of the fittest individuals of the current generation are copied to the next generation (of course these individuals can also be selected as parents).

Selection

- *Proportional selection* (or *roulette wheel selection*): each individual *i* of the current population has a probability *p*(*i*) of being selected proportional to its fitness *f*(*i*):



fitness(A) = 3
fitness(B) = 1
fitness(C) = 2

$$p(i) = \frac{f(i)}{\sum_{j=1}^{n} f(j)}$$

n = population size

- Assign each individual a sector of the roulette wheel.
- Spin the wheel *n* times to select *n* individuals.

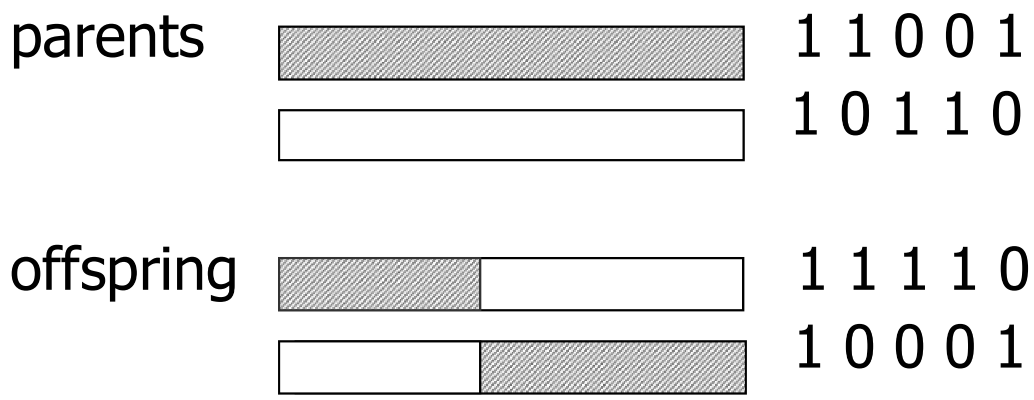
- *Tournament selection*: there are a number of variants, e.g., *k-tournament selection*: *k* individuals are randomly selected from the population and the fittest of them (the winner of the tournament) is considered for reproduction.

Selection pressure

- Selection pressure is the degree to which the best individuals are favoured by the selection operator. The higher the selection pressure, the more the best individuals are favoured.
- Selection pressure influences the convergence rate of the GA and, therefore, has an important effect on the <u>balance between exploitation and exploration</u>: a strong selection pressure can cause a premature convergence to a local optimum, while a too low selection pressure causes the GA to take longer to find the optimal solution.

Binary encoding crossover

- **One-point crossover**

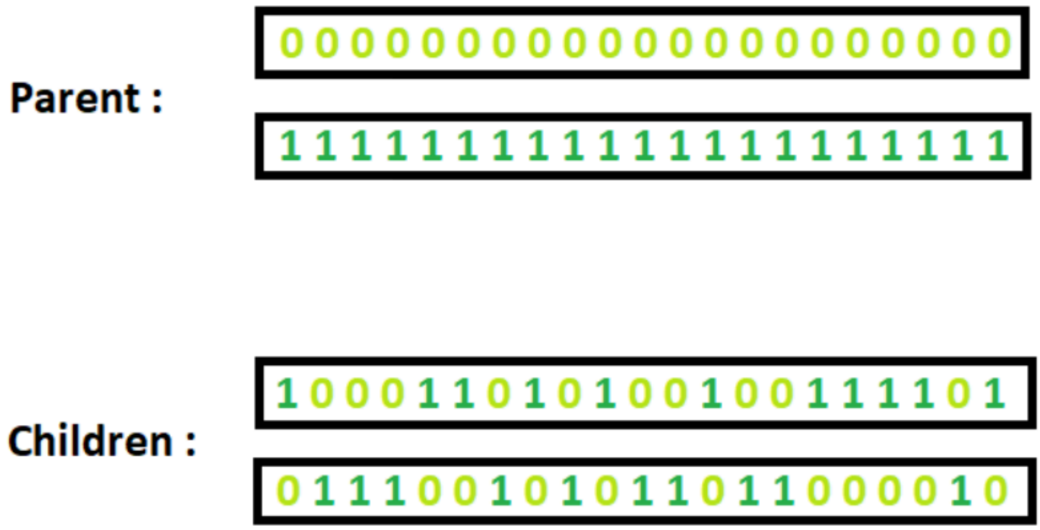


- **Two-point crossover**

- **Uniform crossover**

Uniform crossover does not use predefined crossover points:

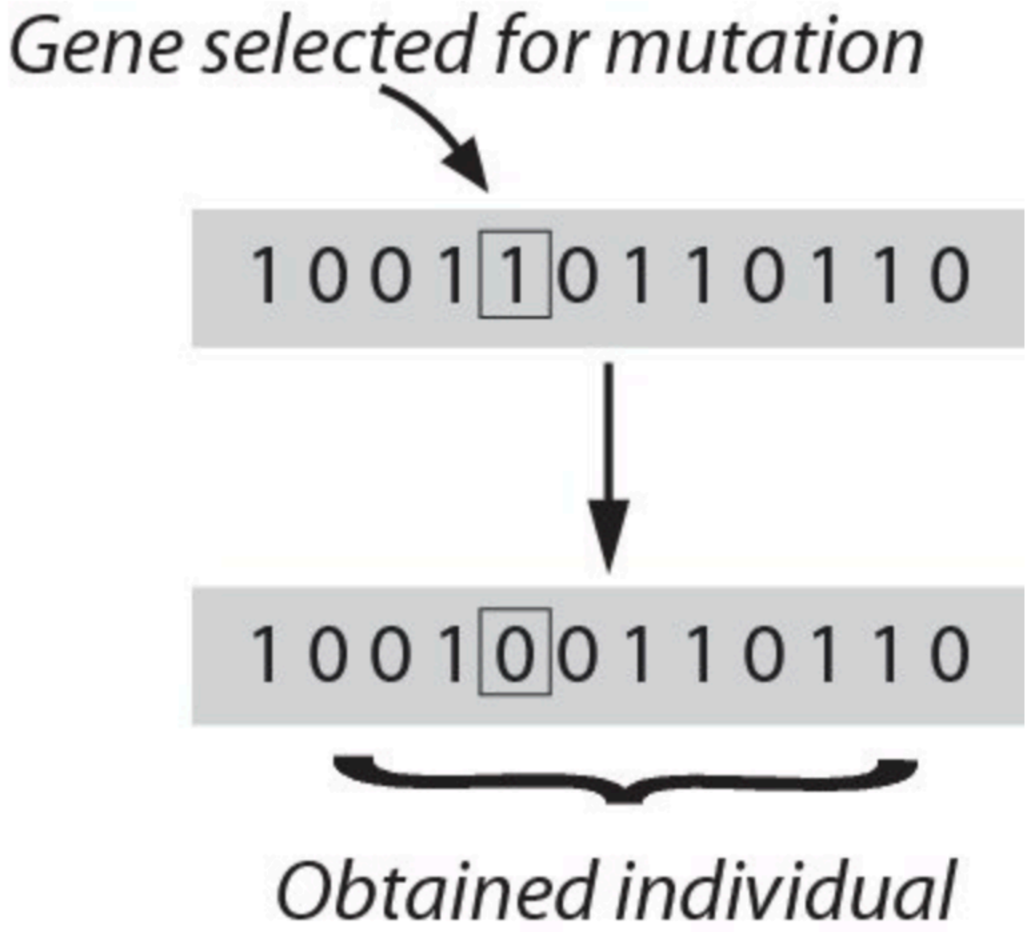
- for each gene of the first descendant, the parent that will give the value of that gene is chosen (with a given probability). For the second offspring, we take the value of the corresponding gene from the other parent;
- the genes of each offspring can also be calculated independently. Thus, a parent can give the value of a gene to both descendants.



Uniform Crossover

Binary encoding mutation

- The effect of mutation is to change a single gene (or, possibly, more genes) within a chromosome. Mutation ensures that the full range of gene values is available for the search.
- In *strong* mutation, the position selected for mutation automatically changes its value. In *weak* mutation, the selected position changes value with a certain probability.


Gene selected for mutation
1 0 0 1 1 0 1 1 0 1 1 0
1 0 0 1 0 0 1 1 0 1 1 0
Obtained individual

Real-valued encoding crossover

- In real-valued encoding, each chromosome is a vector of real numbers. Each gene is a real number.

Discrete crossover

- Discrete crossover is analogous to uniform crossover in binary encoding.
- For each position *i* of the first offspring, we choose (with a fixed probability) the parent whose *i*-th gene will be transmitted to this descendant. The corresponding position of the second offspring will be given the value of the corresponding gene of the other parent.

Average crossover

- Some genes are chosen at random and the corresponding genes in the descendants represent the arithmetic mean of the corresponding genes in the parents. E.g., suppose genes 3 and 5 are selected

parents $x = (x1, x2, x3, x4, x5)$

$y = (y1, y2, y3, y4, y5)$

offspring $x' = (x1, x2, (x3 + y3)/2, x4, (x5 + y5)/2)$

$y' = (y1, y2, (x3 + y3)/2, y4, (x5 + y5)/2)$

Convex crossover

- The descendant(s) is (are) expressed as a convex combination of the parents. E.g.:
  - the *i*-th gene of the only descendant of chromosomes *x* and *y* is:

$$z_i = \alpha\ x_i + (1-\alpha)\ y_i,\ \ \alpha \in [0,1]$$

  - the *i*-th gene of the descendants *u* and *v* are:

$$u_i = \alpha\ x_i +\ (1-\alpha)\ y_i$$
$$v_i = \alpha\ y_i +\ (1-\alpha)\ x_i$$

Real-valued encoding mutation

- We can have *uniform* and *non-uniform* mutation. The action of a non-uniform operator depends on the generation.

Uniform mutation

- *One-position mutation*: a single (randomly chosen) gene is replaced with a randomly generated real number within the domain of the corresponding parameter.
- *All-positions mutation*: all genes are similarly perturbed. There are several methods, e.g., an additive normal mutation:

$$x_i' = x_i + \alpha_i N(0, \sigma_i)$$

where the real parameters α_i and the standard deviations σ_i can be the same or different for the genes considered.

Non-uniform mutation

- Genes undergo significant changes in the first generations. Then the changes gradually decrease. Therefore, in the initial stage of the search process, important progress is made and in the last phases there is a refinement, or fine control, of the search process.

Crossover and mutation probability

- The crossover probability p_c generally assumes values of the order of magnitude 10^{-1}.
- Small values of mutation probability p_m are traditionally used ($p_m \in [0.001, 0.01]$).
- We have already observed that the balance between exploitation and exploration is essential: it can be adjusted by the selection pressure of the selection operator (as stated before), and by the probability of crossover and mutation.

Solving an optimization problem using a GA

- The following five issues need to be addressed:
  - a genetic representation of the candidate solutions,
  - creating an initial population of solutions,
  - definition of the fitness function,
  - choice of selection operator and genetic operators,
  - choice of the parameters of the GA, e.g., population size, number of generations, probabilities of genetic operators.