Evolutionary Computations, Genetic Algorithms

Pawel Herman

Department of Computational Biology

School of Computer Science and Communication

KTH Royal Institute of Technology

BEST summer school

Stockholm, June 9-13 2014

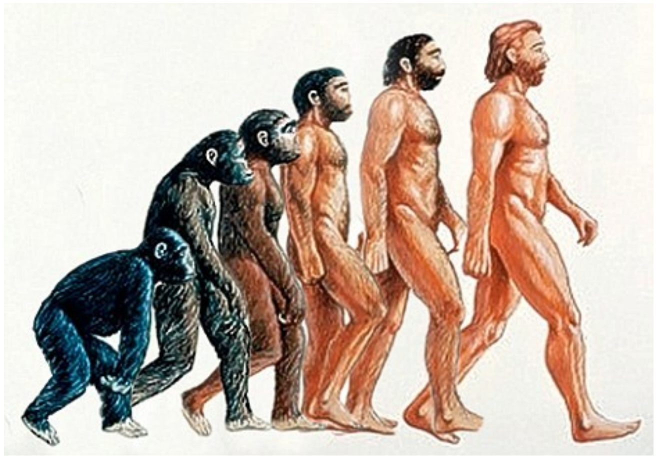
Introduction to EAs – origins and inspiration

- Inspired by imitation of nature
  - organisms (population) living in some environment
  - they have genetic material (chromosomes) that contains information about them
  - through genetic processes they can transfer their features to next generations
  - mechanisms of natural selection will promote good features (adaptation to the environment) - ”survival of the fittest”
  - Concepts of ”natural selection” and ”genetic inheritance” proposed by Darwin (1859)
- Family of methods
  - Genetic Algorithms (GAs)
  - Genetic Programming
  - Particle Swarm Optimisation
  - Ant Colony Algorithms

Biology, nature

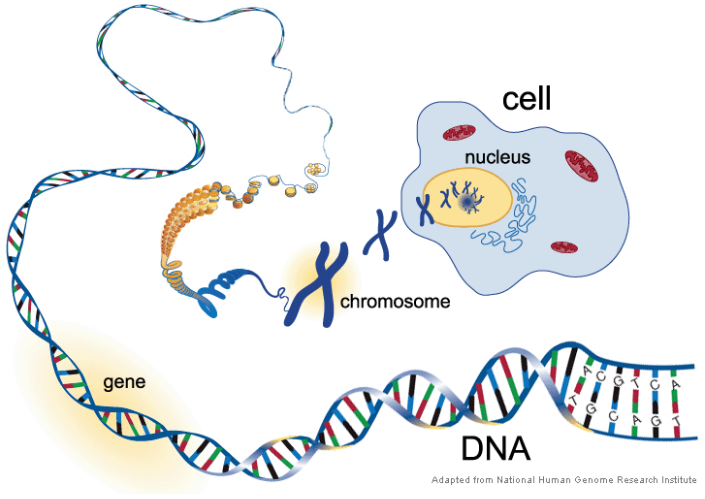
- Natural selection

According to Darwin, *”only the organism best adapted to their environment tend to survive and transmit their genetic characteristics in increasing numbers to succeeding generations while those less adapated tend to be eliminated”*.



Fundamental aspects of genetics

- Each cell contains chromosomes (in nuclei)
- Each chromosome is composed of strands of DNA with genes as segments that determine specifc traits (features)
- We can have more than 20 thousand genes
- Gene are subject to mutation
- Reproduction of individuals is associated with exchange of genetic material between chromosomes



Introduction to EAs – origins and inspiration

- Evolutionary algorithms tackle optimisation problems
- Rather than search from general to specific or another way round, EAs generate successor of hypotheses by recombining parts of best currently known hypotheses
- Why are they attractive?
  - evolution is know to be robust and successful
  - potential to search spaces of hypotheses containing complex interacting parts without having to model explicitly their effect on overall hypothesis
  - Global search heuristic effective at finding approximate solutions (though there is no convergence guarantee)
  - scope for massive parallelism

Optimisation approaches

1. Analytical methods
   - local scope – search for solutions in the vicinity of selected samples
   - relies on the existence of gradients
2. Enumerative methods
   - searching through all samples
3. Random methods
   - blind random search is rather ineffective
   - in EAs random selection is only a tool for supporting search in the coded space of solutions (stochastic nature of search).

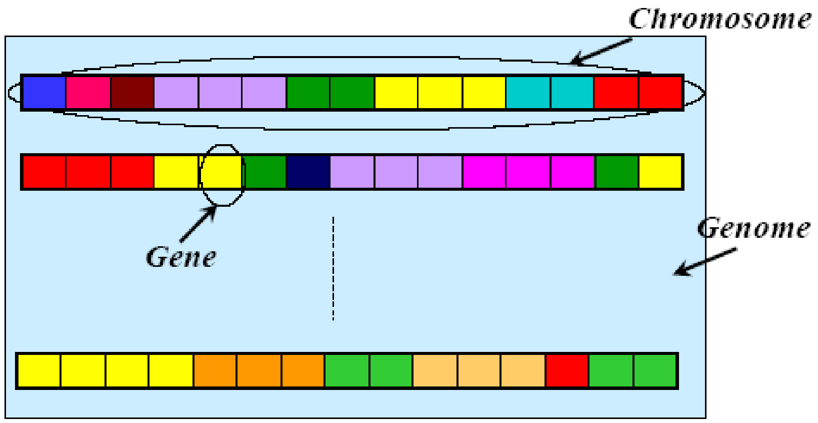
EAs optimisation - key principles

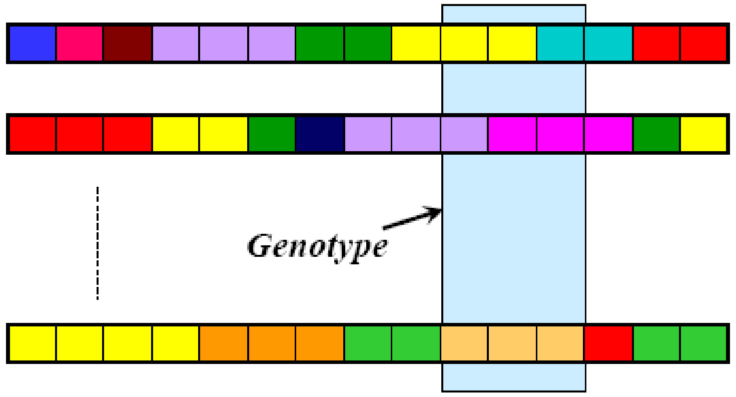
1. EAs do not directly process the task parameters but their codes.
2. They start from a population , not from a single point.
3. EAs use only the objective function, not its derivatives or other information.
4. Probabilistic selection rules are used instead of deterministic.

Genetic nomenclature

- **Population** – a set of individuals coded in the form of chromosomes
- **Chromosomes** – ordered sequences of genes
- **Gene** – a feature that constitutes a single element of the genotype (structure in the chromosome)
- **Phenotype** – a set of values corresponding to a genotype (decoded structure)
- **Allel** – the value of a give gene, feature value
- **Locus** – position indicating the place of the location of a given gene in the chromosme

Chromosome, genes, genotype

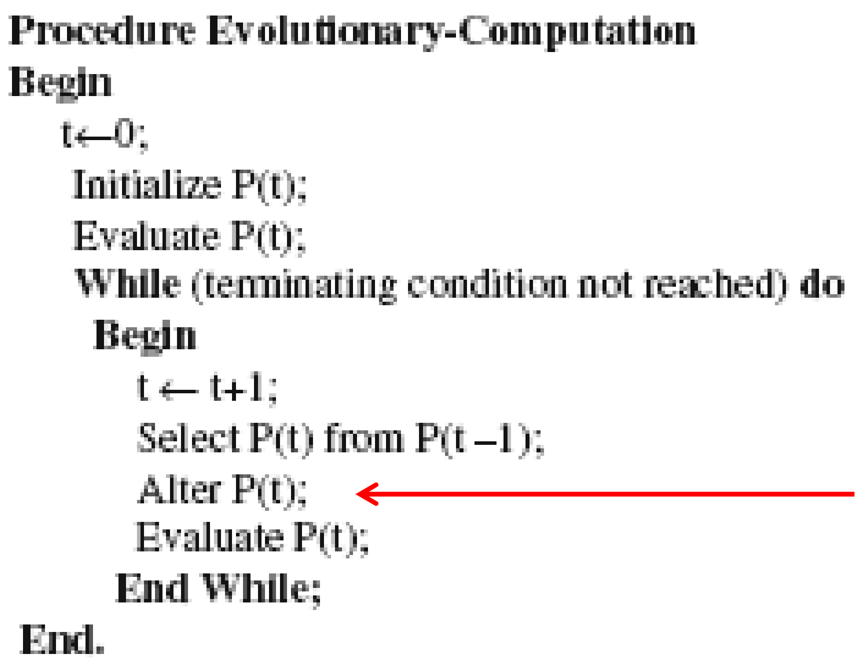




Ingredients needed for using EAs

- Define the problem
- Code the parameters for optimisation – define your genotype and identify the phenotype
- Define genetic operators
- Define the cost (fitness) function
- Determine selection criteria
- Set population parameters

General procedure



```
Procedure Evolutionary-Computation
Begin
    t ← 0;
    Initialize P(t);
    Evaluate P(t);
    While (terminating condition not reached) do
      Begin
        t ← t+1;
        Select P(t) from P(t – 1);
        Alter P(t);
        Evaluate P(t);
      End While;
End.
```

$\leftarrow$ decides about the type of EA

Focus on Genetic Algorithms

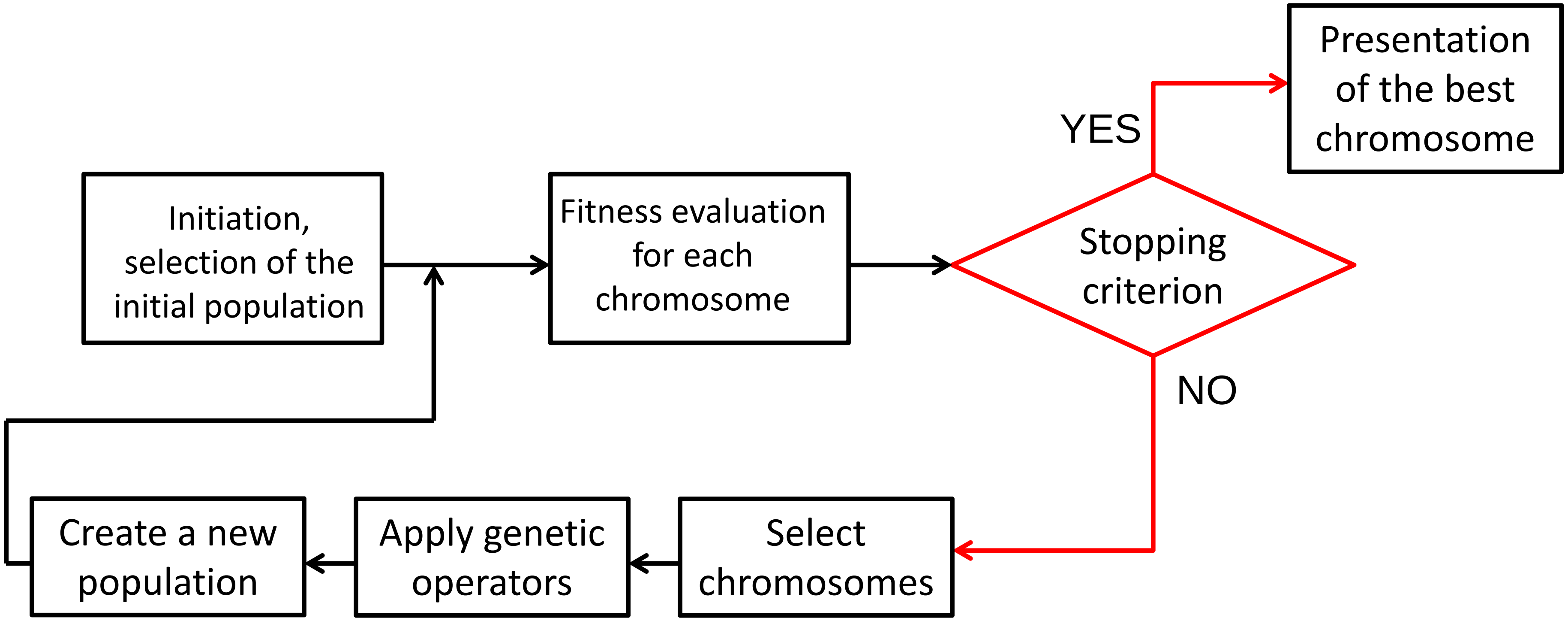
- Originally proposed by Holland in 1975

  ”Adaptation in natural and artificial systems”

- Uniqueness of GAs
  - universal genetic codes
  - multi-parent reproduction
  - fitness proportional selection
  - problem independence – mechanism for evolution defined in a general manner

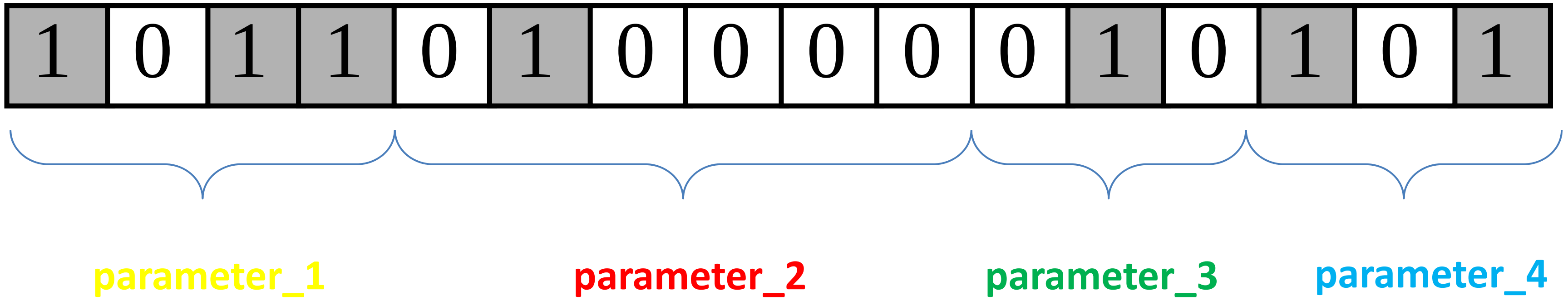
”GAs are good at taking large, potentially huge search spaces and navigating them, looking for optimal combinations of things, solutions you might not otherwise find in a lifetime”

Classic GA cycyle


Initiation, selection of the initial population
Fitness evaluation for each chromosome
Stopping criterion
YES
Presentation of the best chromosome
NO
Select chromosomes
Apply genetic operators
Create a new population

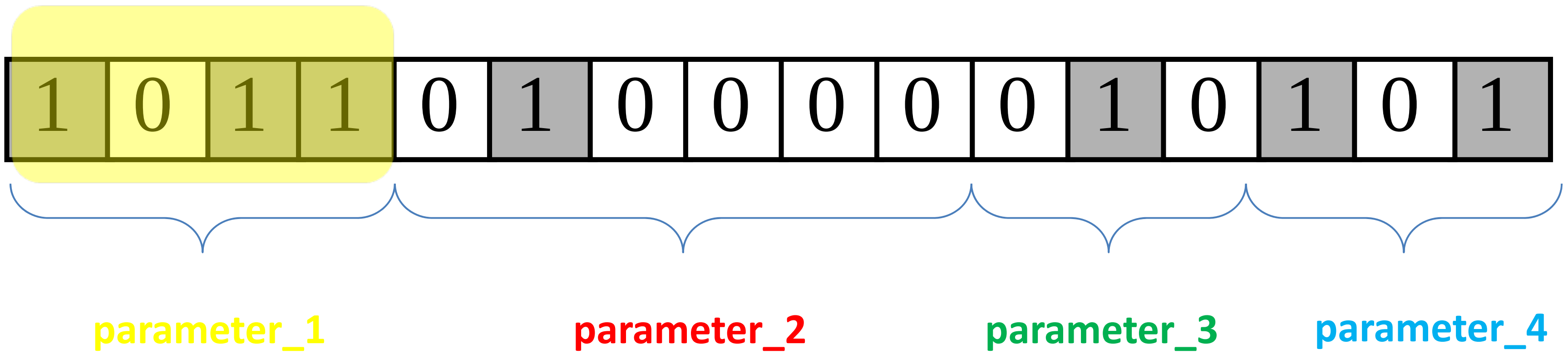
Coding, genetic representations

- Representation of an individual uses discrete values (binary, integer, etc.), the most common are binary strings



Coding, genetic representations

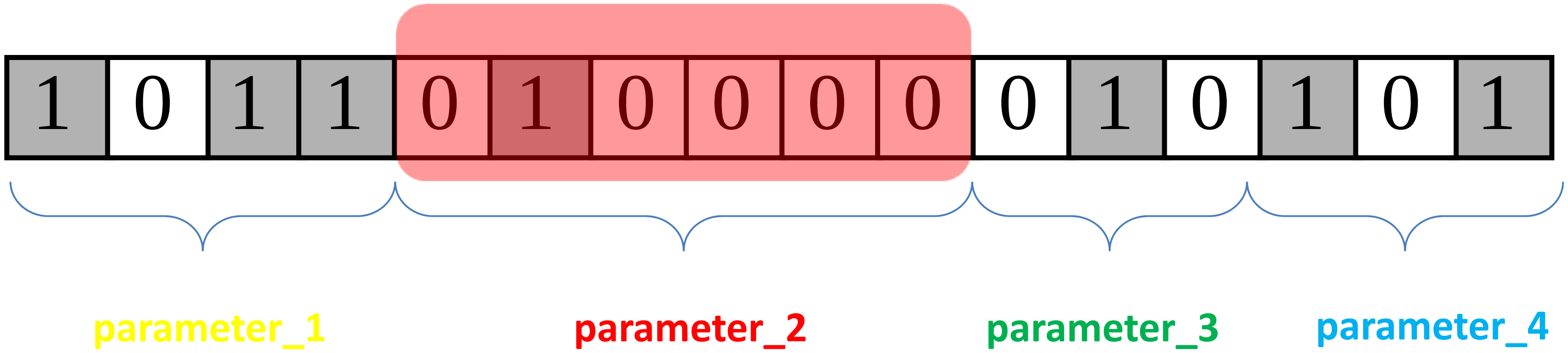
- Representation of an individual uses discrete values (binary, integer, etc.), the most common are binary strings



1011 -> 11

Coding, genetic representations

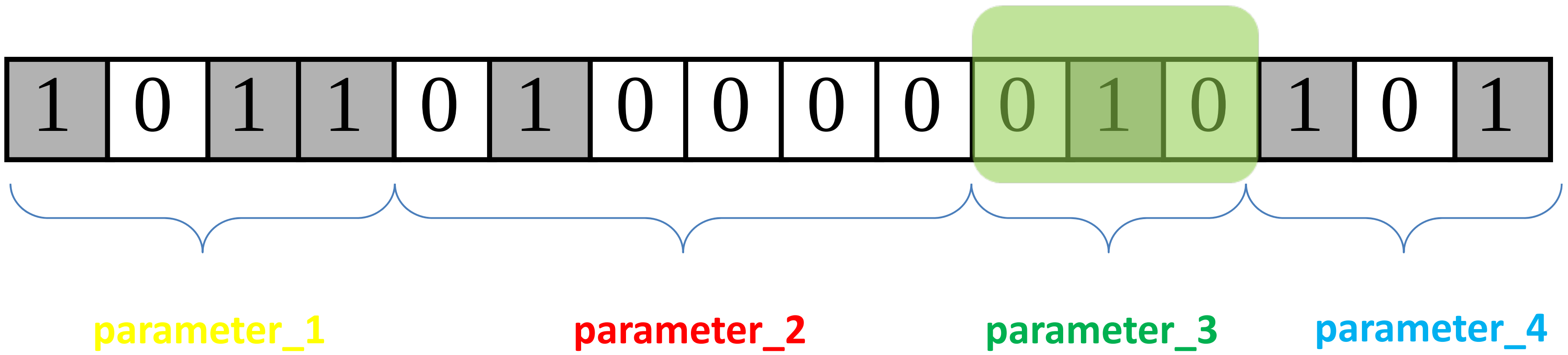
- Representation of an individual uses discrete values (binary, integer, etc.), the most common are binary strings



010000 -> 16

Coding, genetic representations

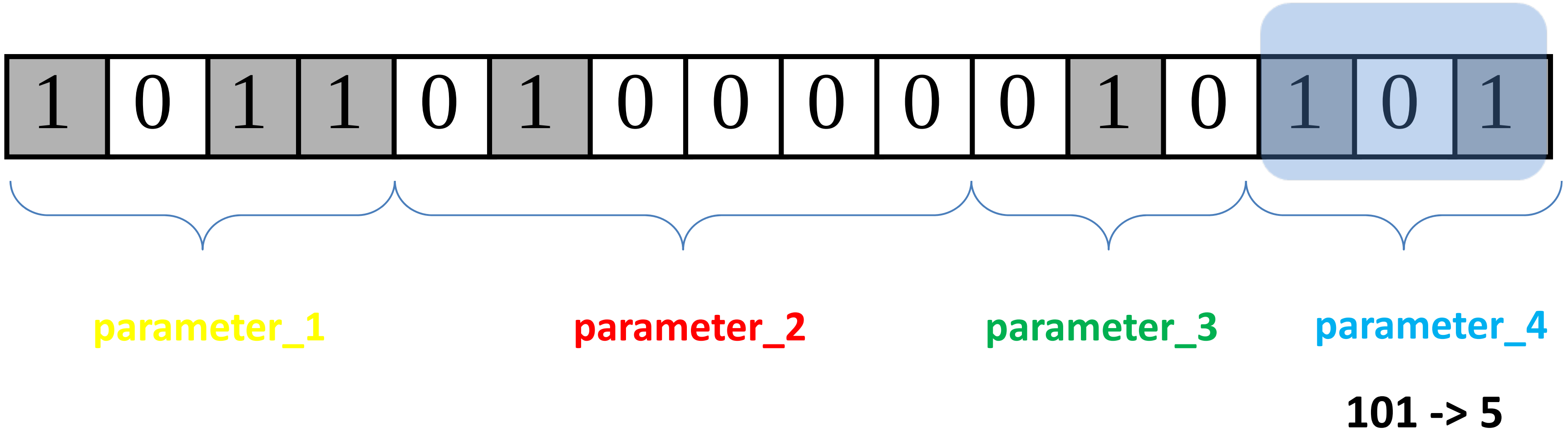
- Representation of an individual uses discrete values (binary, integer, etc.), the most common are binary strings



010 -> 2

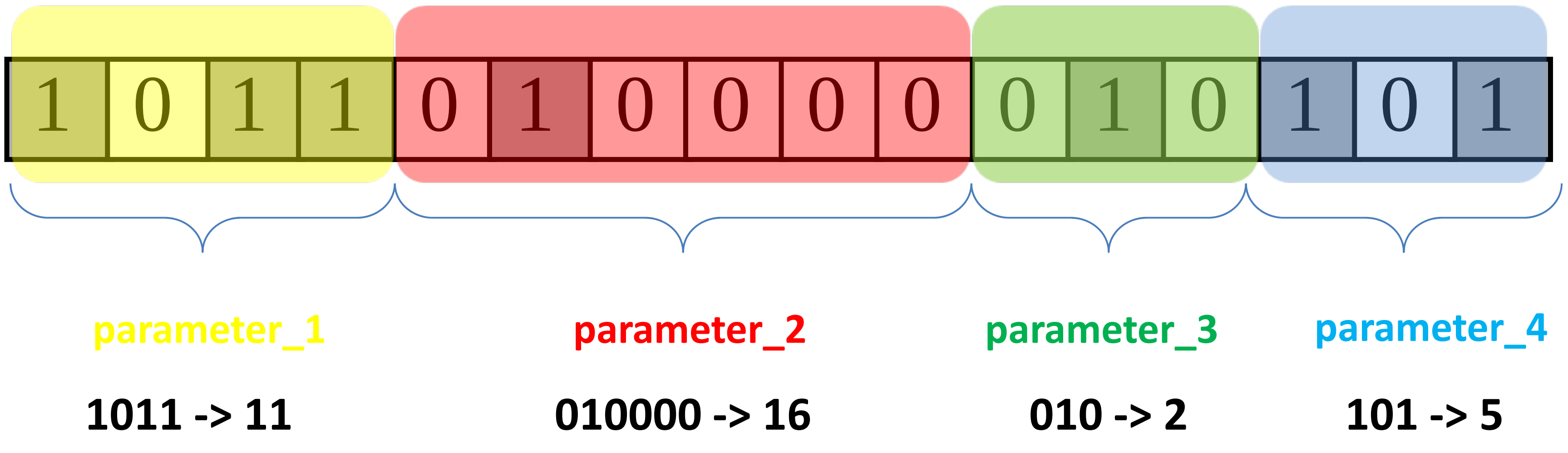
Coding, genetic representations

- Representation of an individual uses discrete values (binary, integer, etc.), the most common are binary strings

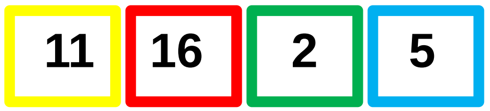


Coding, genetic representations

- Representation of an individual uses discrete values (binary, integer, etc.), the most common are binary strings



Genotype and its corresponding phenotype



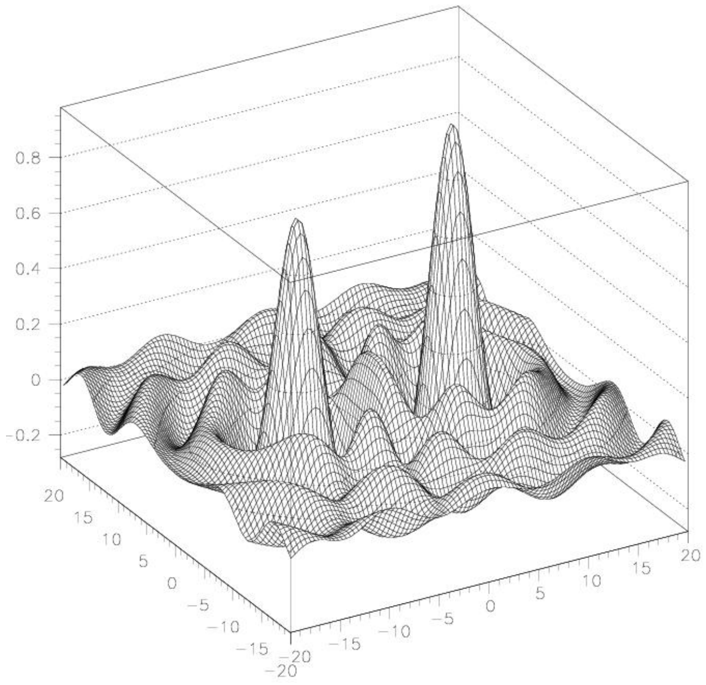
Initiation

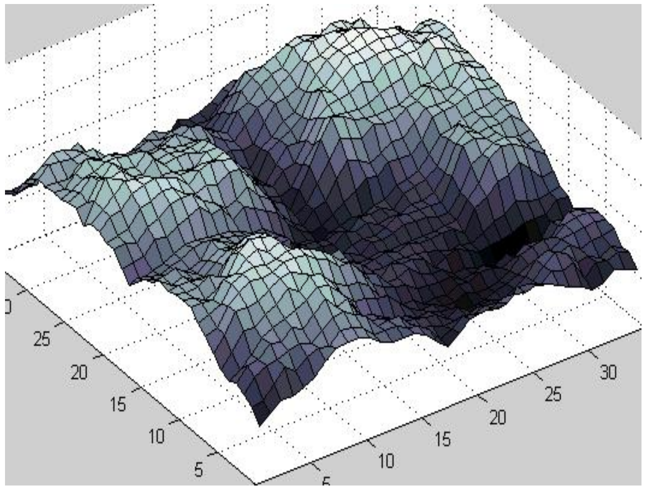
- Creation of an initial population
  - initially many individual solutions are randomly generated
  - the population size depends on the nature of the problem (typically contains several hundreds or thousands of possible solutions)
- Traditionally, the population is generated randomly, but occasionally the solutions may be "seeded" in areas where optimal solutions are likely to be found.

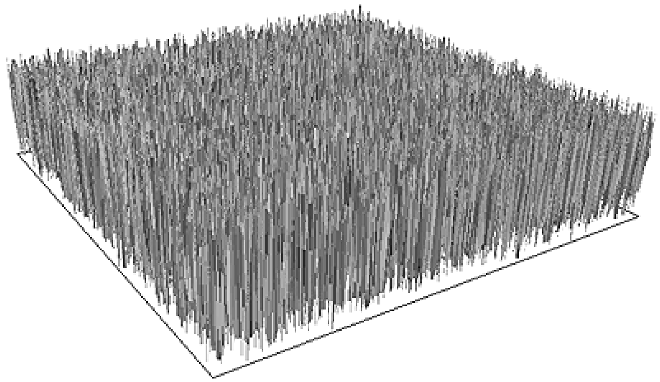
Fitness function

- This serves as the only obvious link to a specific optimisation problem being addressed.
- Most often it is the most computationally demanding aspect of GAs since the cost function (fitness) must be evaluated for each indidual.
- Scaling of the fitness function is sometimes performed to control the convergence especially in the early (risk of harmful premature convergence) and late stages.

Optimisation landscapes

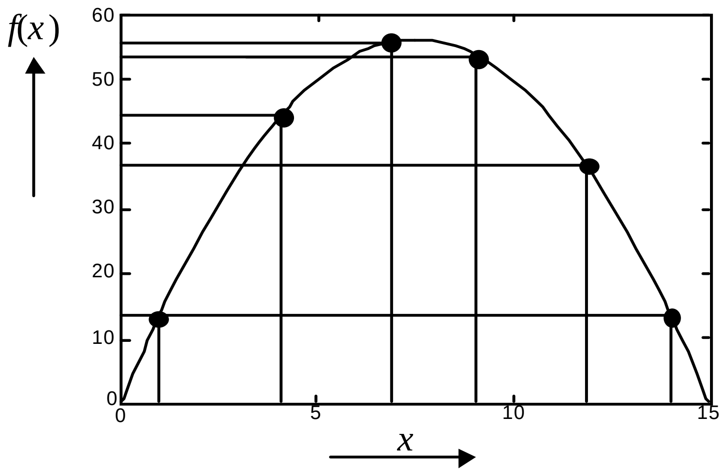




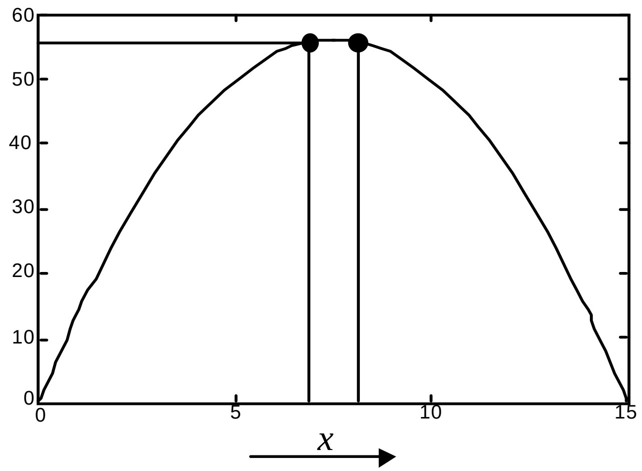


Search for the max *f*, $f(x) = 15x - x^2$

Chromosome label	Chromosome string	Decoded integer	Chromosome fitness	Fitness ratio, %
X1	1 1 0 0	12	36	16.5
X2	0 1 0 0	4	44	20.2
X3	0 0 0 1	1	14	6.4
X4	1 1 1 0	14	14	6.4
X5	0 1 1 1	7	56	25.7
X6	1 0 0 1	9	54	24.8



(*a*) Chromosome initial locations.



(*b*) Chromosome final locations.

Negnevitsky, 2005

Selection

- During each successive generation, a proportion of the existing population is selected to breed a new generation – mating pool
- Individual solutions are selected through a fitness-based process.
- Basic idea – the fitter are more likely to be selected
- Most functions are stochastic and designed so that a small proportion of less fit solutions are selected
  - keeping the diversity of the population large
  - preventing premature convergence on poor solutions
  - elitist strategy as an option

Roulette wheel selection

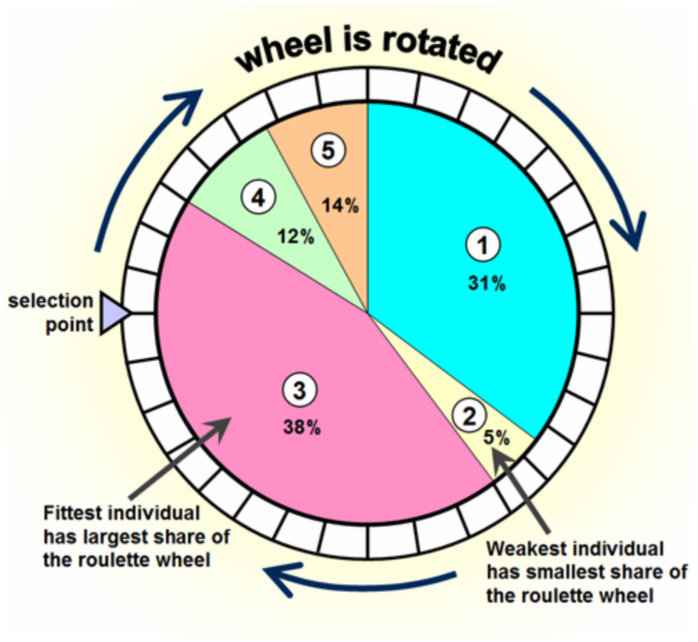
Selection based on fitness.

But, why not choose the best chromosomes?

Balance between ***exploration*** and ***exploitation***

$$\nu(\mathrm{ch}_i) = p_s(\mathrm{ch}_i) \cdot 100\%,$$

$$p_s(\mathrm{ch}_i) = \frac{F(\mathrm{ch}_i)}{\sum_{j=1}^{K} F(\mathrm{ch}_j)},$$


wheel is rotated
5
14%
4
12%
1
31%
selection point
3
38%
2
5%
Fittest individual has largest share of the roulette wheel
Weakest individual has smallest share of the roulette wheel

Rank-based selection

1. Organise population
   - Sort (rank) individuals according to fitness
   - Ascending or descending order (minimization or maximization)
2. Selection
   - Select individuals with probability proportional to their rank only (ignoring the fitness value)
   - The better the rank, the higher the probability of being selected

It still requires global sorting of individuals, reducing potential for parallel processing.

Tournament selection

1. A number of “tournaments” are run
2. Several chromosomes chosen at random
3. The chromosome with the highest fitness is selected each time

Larger tournament size means that weak chromosomes are less likely to be selected.

Advantages:

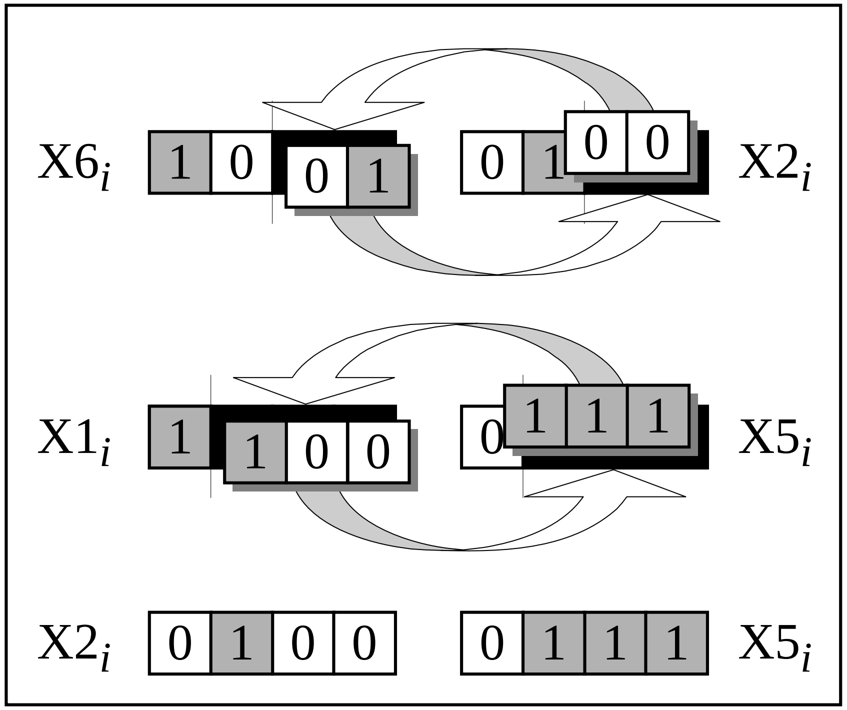
- It is efficient to code
- It works on parallel architectures

Reproduction

- Another generation population of solutions (offspring) is produced from those selected for breeding (mating pool)
- Genetic operators are applied
  - **crossover** (also called recombination)
  - **mutation**.
- A new offspring solution typically shares many of the characteristics of its parents.
- After reproduction a new population can
  - replace the entire parent generation population (generational GA)
  - replace a certain part of the older generation (steady-state GA)

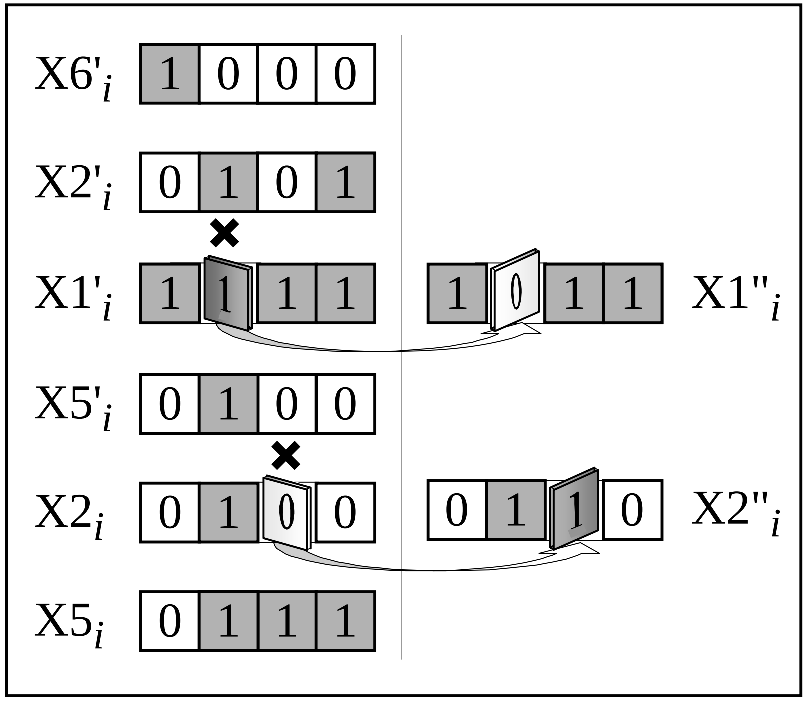
Crossover

- The first stage is selecting a pair of parents from the mating pool with the probability, p_c
- For each pair, a crossover locus is randomly drawn
- The exchange of chromosome substrings is performed between the parent individual to produce a pair of offspring.



Mutation

- Change in gene with low probability, p_m
- Gene is flipped in elected individuals
- It prevents from getting stuck in local minima (noise factor)
- The probability, p_m, usually does not exceed 0.01.



Termination

- GA cycles are repeated until a termination condition is reached.
- Common terminating conditions are:
  - minimum criteria are satisfied by a solution
  - fixed number of generations reached
  - the highest ranking solution's fitness is reaching or has reached a plateau such that successive iterations no longer produce better results
  - possible combinations.

Hypothetical explanation (Holland’s Schema theorem)

”A schema with an above average fitness tends to increase at an exponential rate until it becomes a significant portion of the population.”

A schema is a template that identifies a subset of strings with similarities at certain string positions, much like a regular expression.

- deterministic argument
- stochastic argument

Schema theorem

- survival of schemas with above average fitness will grow exponentiallys
- schema with low order has better chances to survive under mutation
- schema with low defining length has better chances to survive under crossover
- short, low-order, above average schemata are represented by exponentially growing number of trials in GA generations

Schema theorem

- urvival of schemas with above average fitness will grow exponentiallys
- schema with low order has better chances to survive under mutation
- schema with low defining length has better chances to survive under crossover
- short, low-order, above average schemata are represented by exponentially growing number of trials in GA generations

*So, it leads to the so-called **building block hypothesis.***

GA seems to seek (near) optimal solutions through the combination of short, low-order, high performance schemata – **building blocks**

Building block hypothesis

“Just as a child creates magnificent fortresses through the arrangement of simple blocks of wood [building blocks], so does a genetic algorithm seek near optimal performance through the juxtaposition of short, low-order, high performance schemata, or building blocks.”

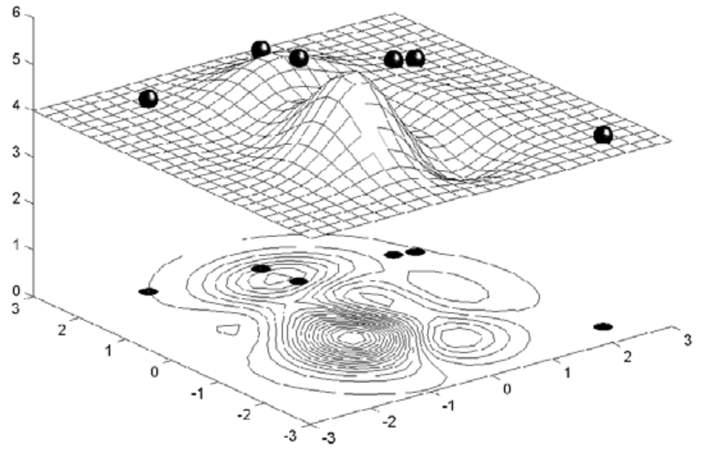
Goldberg, 1989

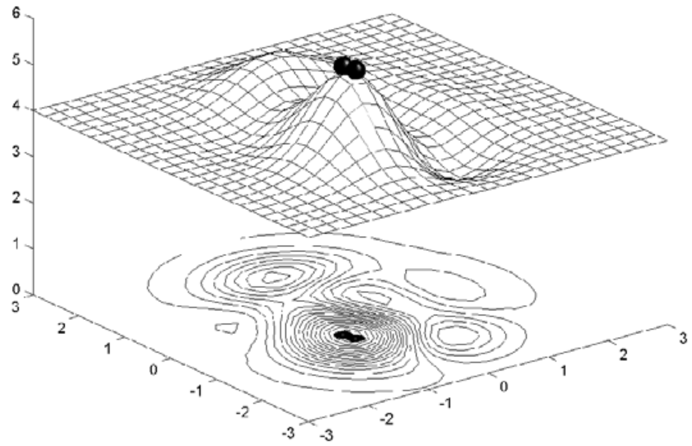
Benefits of GA

- The concept is rather intuitive and easy to comprehend
- Supports multi-objective optimization
- Good for noisy environments
- Focus on finding sufficiently good solutions within reasonable time
- Robust for a wide range of problems and suitable to optimise functions that cannot be analytically treated
- It always produces an answer and answer gets better with time
- Inherently parallel; easily distributed

Example 1

$$f(x,y) = (1-x)^2 e^{-x^2-(y+1)^2} - (x - x^3 - y^3)\, e^{-x^2-y^2}$$

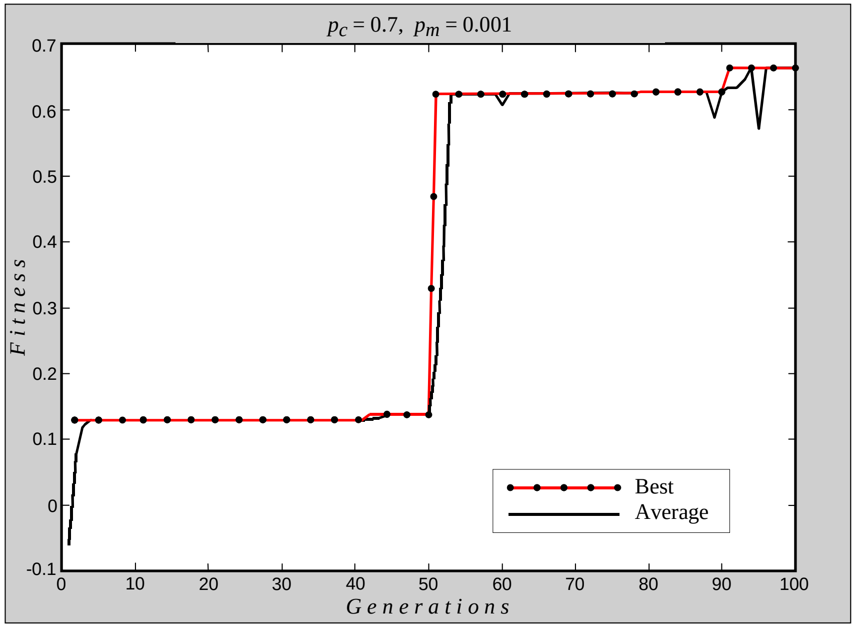




Negnevitsky, 2005

Example 1

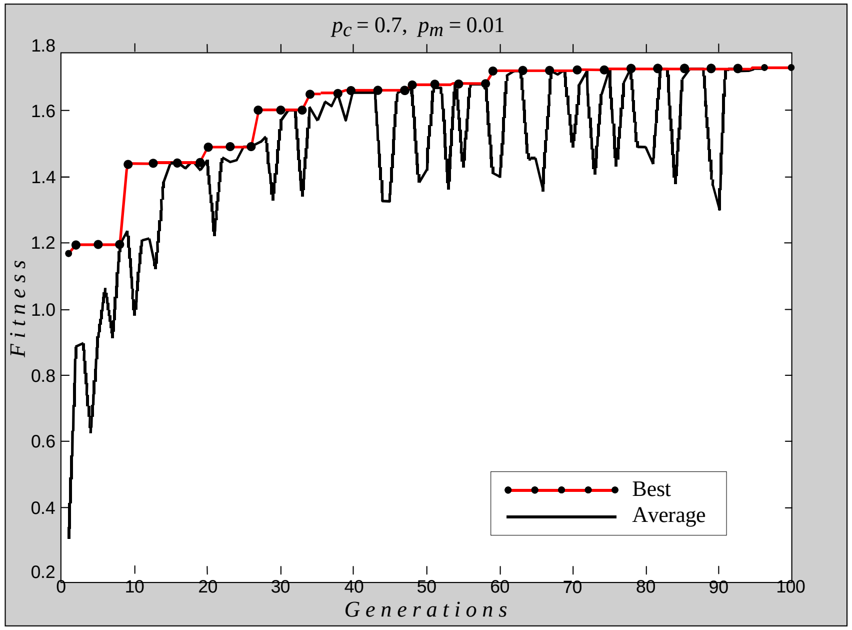
$$f(x,y) = (1-x)^2 e^{-x^2-(y+1)^2} - (x - x^3 - y^3)\, e^{-x^2-y^2}$$



Negnevitsky, 2005

Example 1

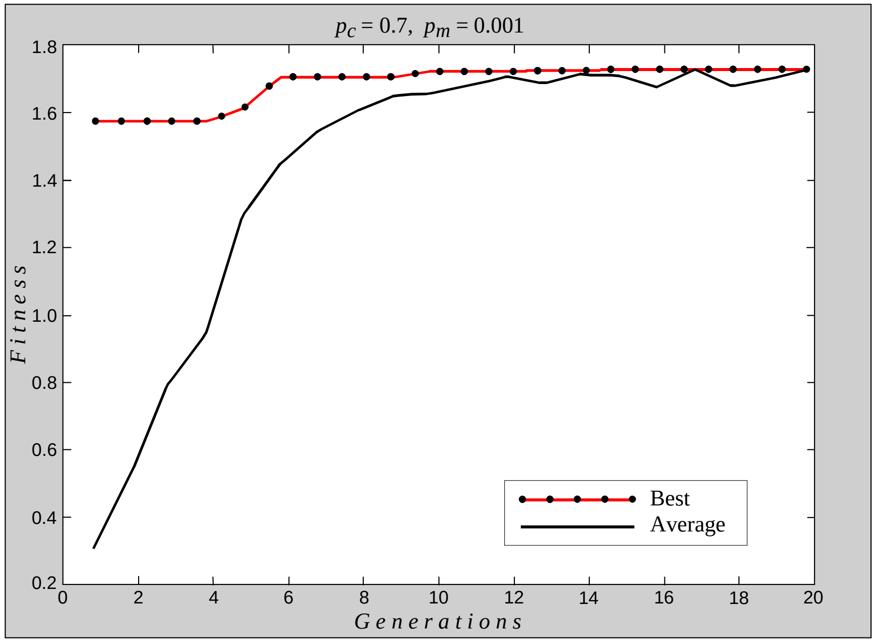
$$f(x,y) = (1-x)^2 e^{-x^2-(y+1)^2} - (x - x^3 - y^3)\, e^{-x^2-y^2}$$



Negnevitsky, 2005

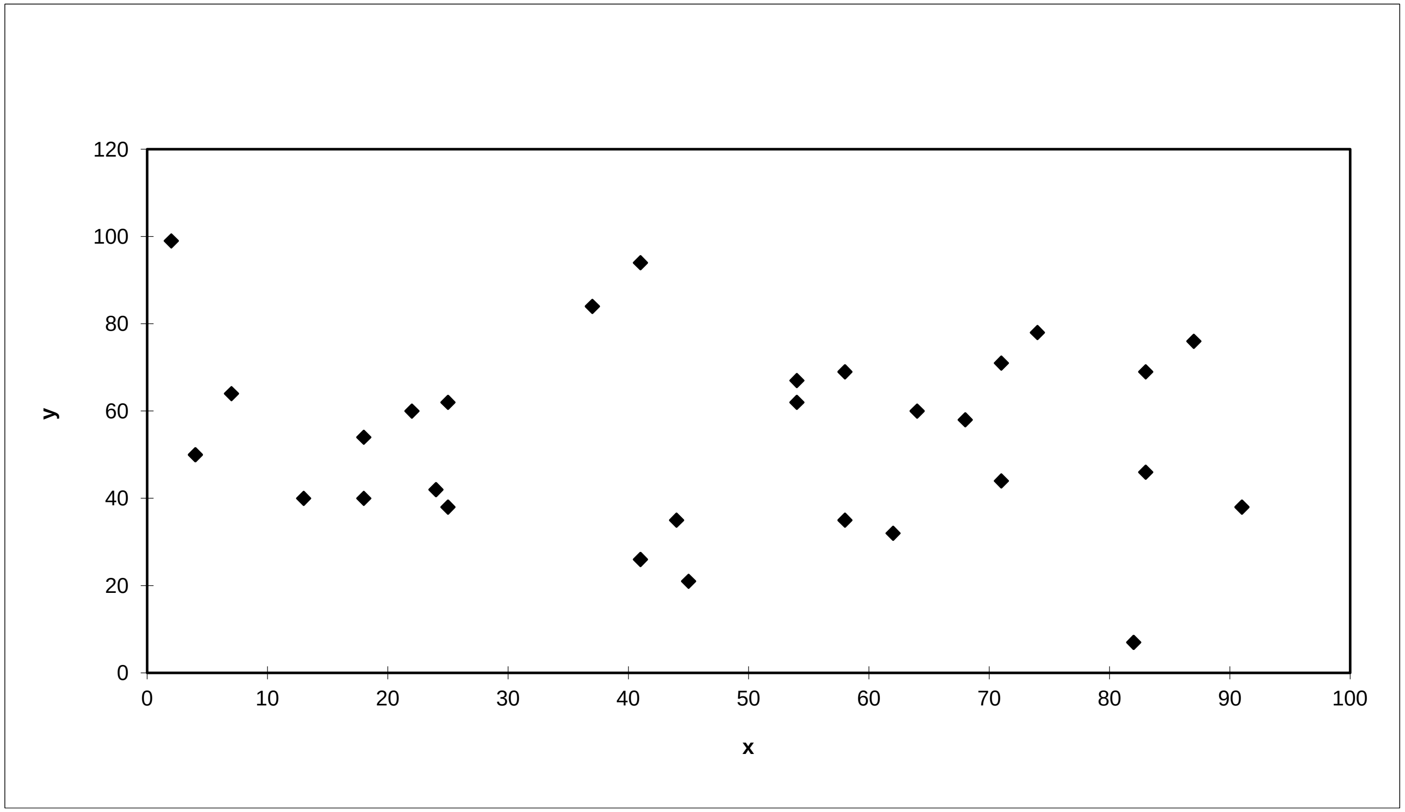
Example 1

$$f(x,y) = (1-x)^2 e^{-x^2-(y+1)^2} - (x - x^3 - y^3)\, e^{-x^2-y^2}$$

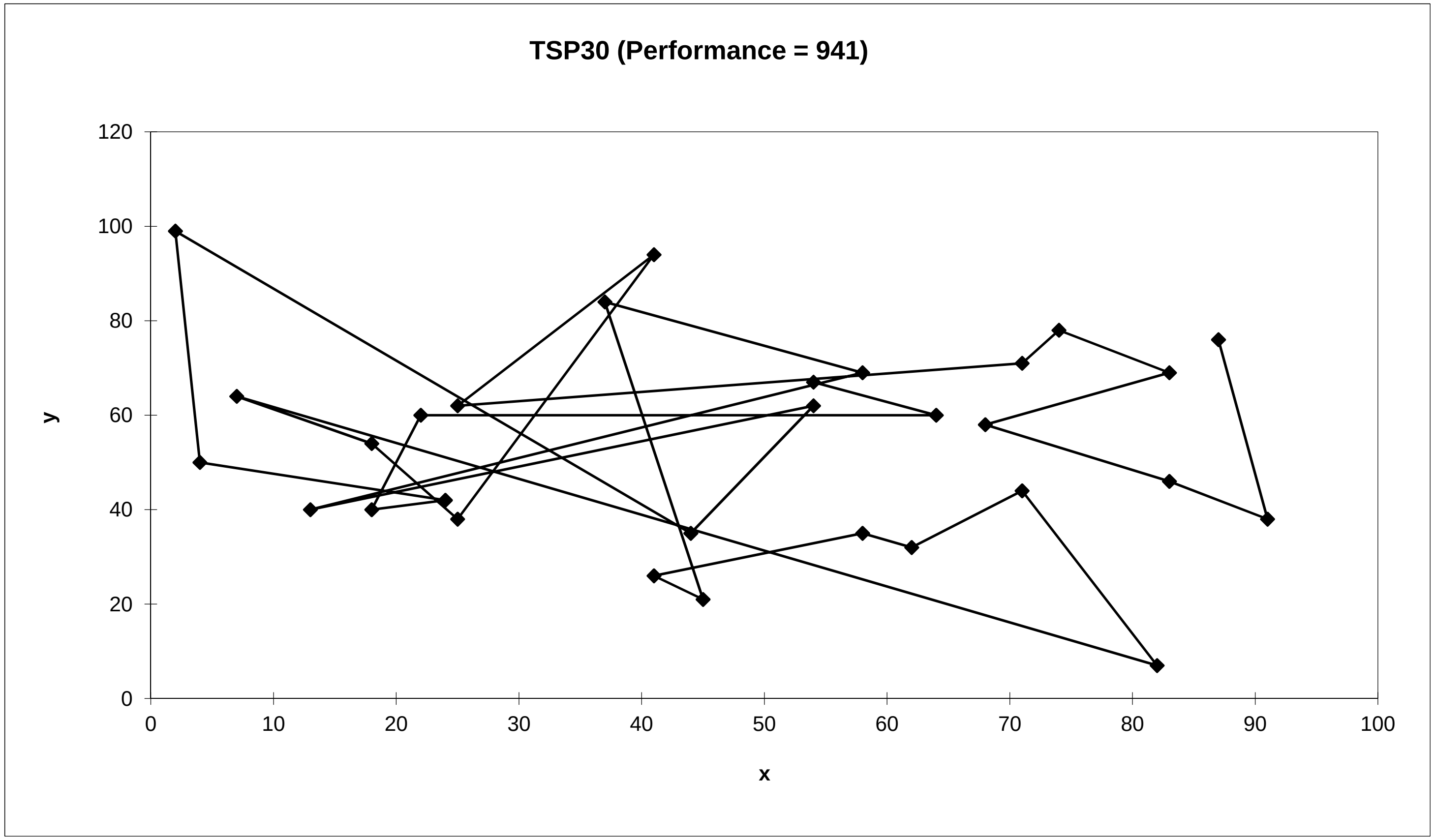


Negnevitsky, 2005

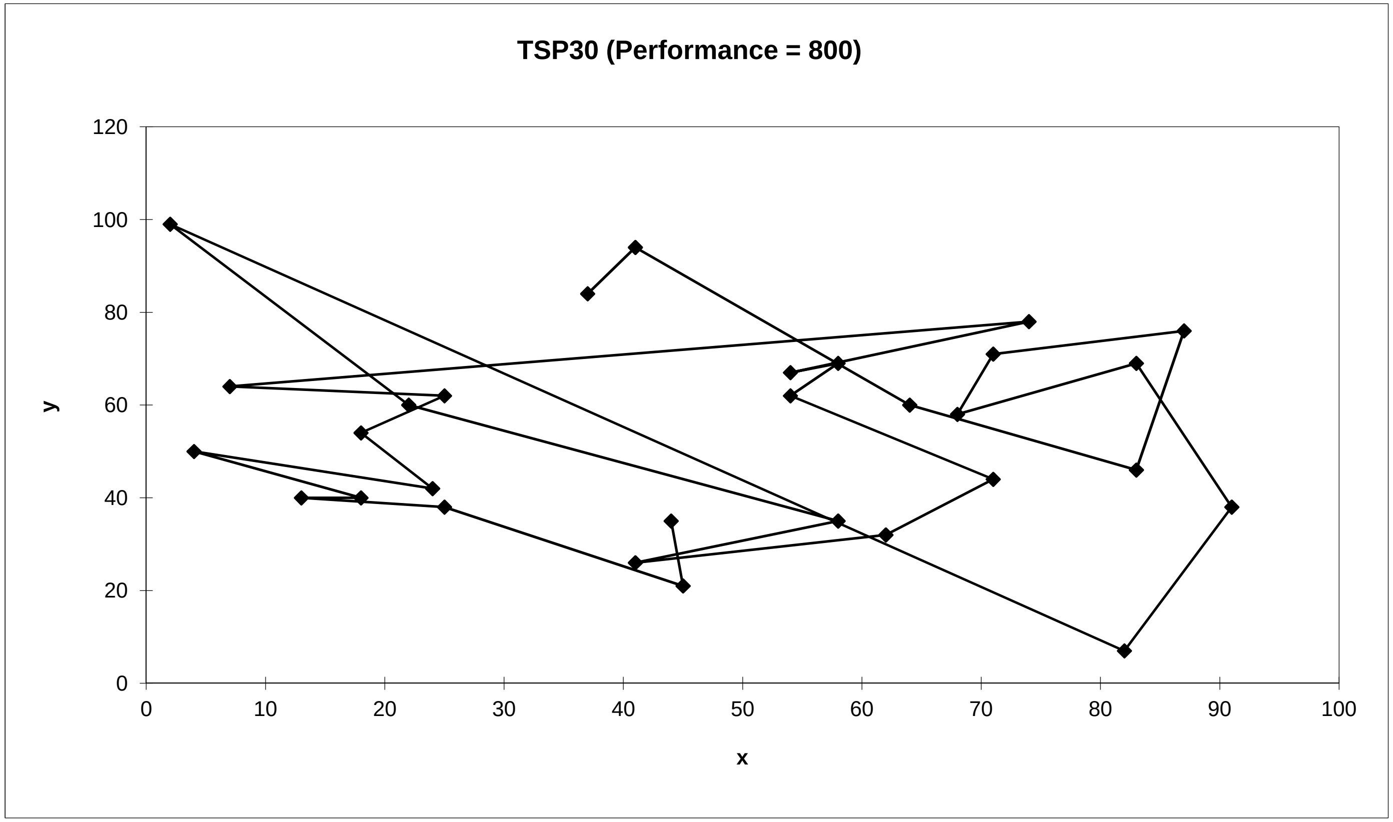
Example 2 – travelling salesman problem



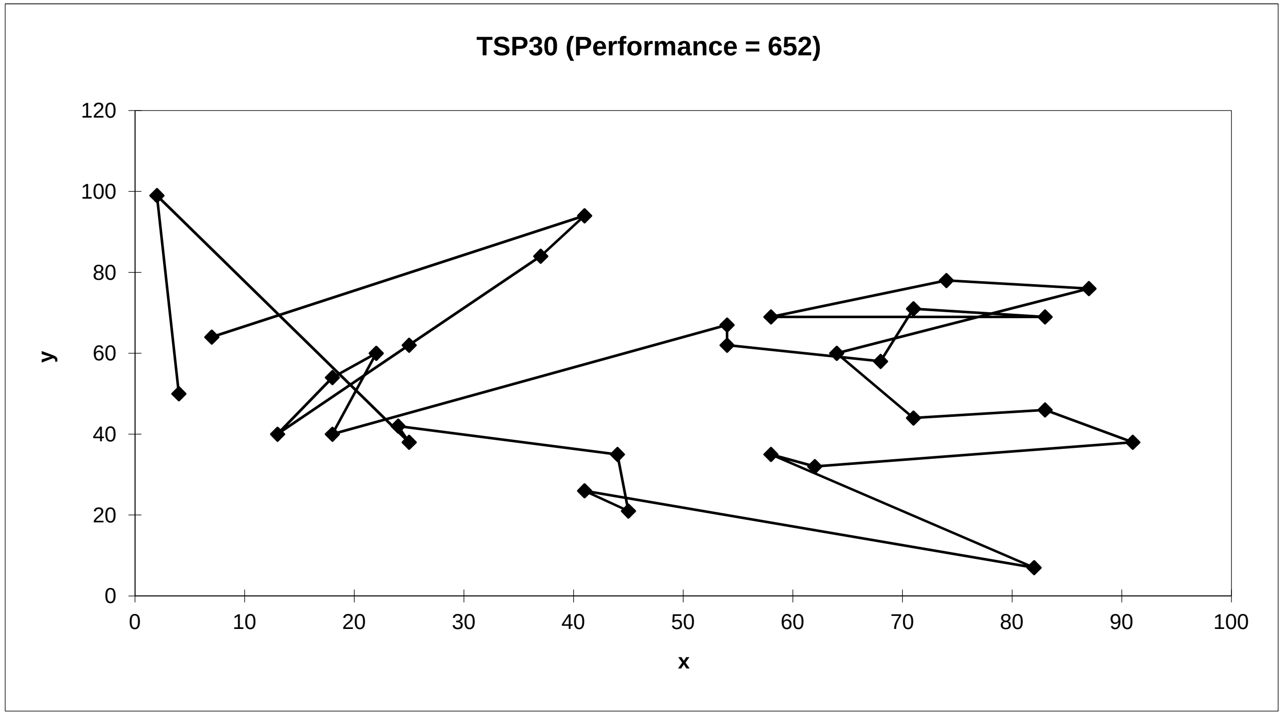
Example 2 – travelling salesman problem



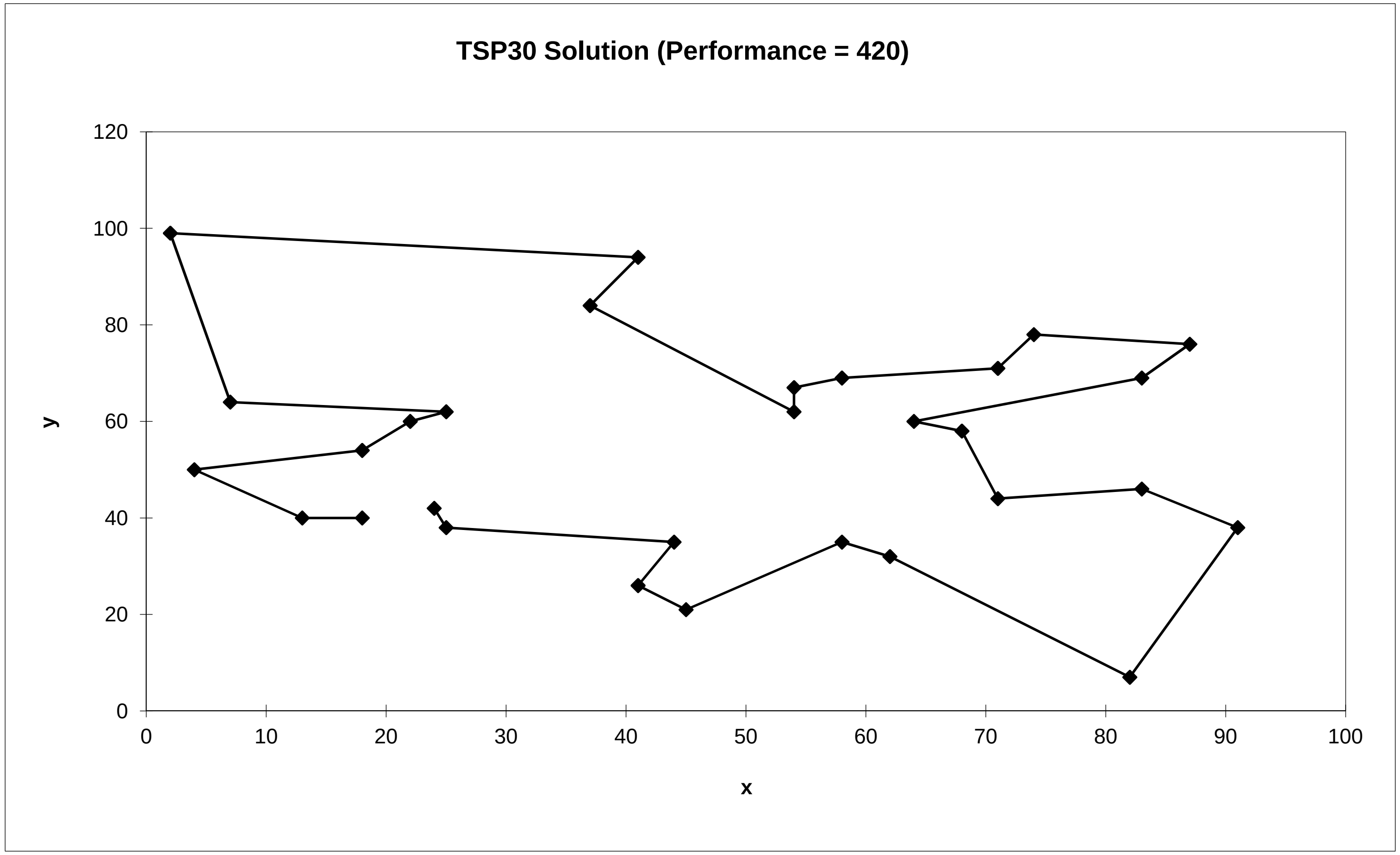
Example 2 – travelling salesman problem



Example 2 – travelling salesman problem



Example 2 – travelling salesman problem


TSP30 Solution (Performance = 420)
y
x