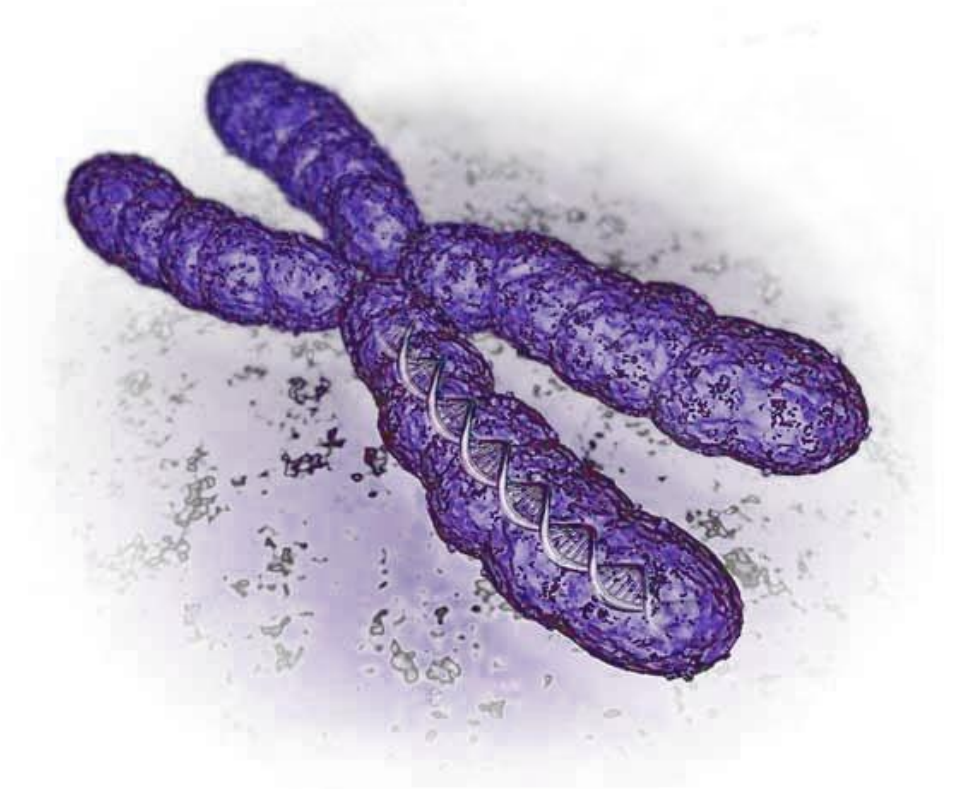
UNIVERSITÉ DU LUXEMBOURG

GA Introduction

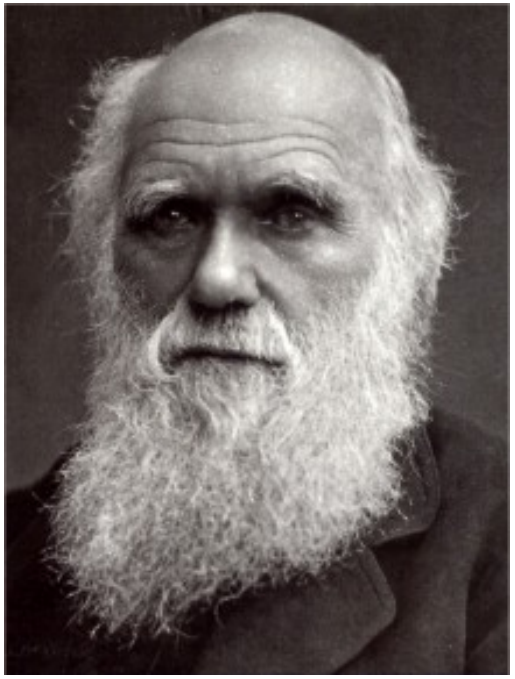


Grégoire Danoy
gregoire.danoy@uni.lu
Office: E006

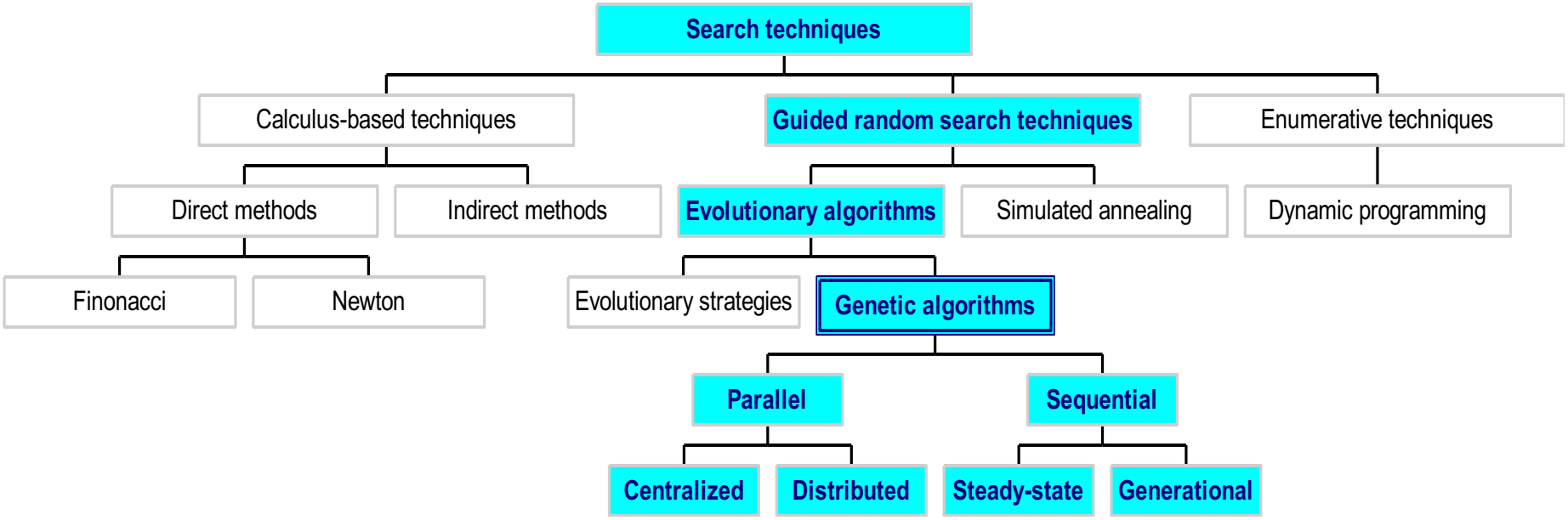
The Genetic Algorithm

- Directed search algorithms based on the mechanics of biological evolution
- Developed by John Holland, University of Michigan (1970's)

i.e. the theories of this man



Classes of Search Techniques



Components of a GA

A problem to solve, and ...

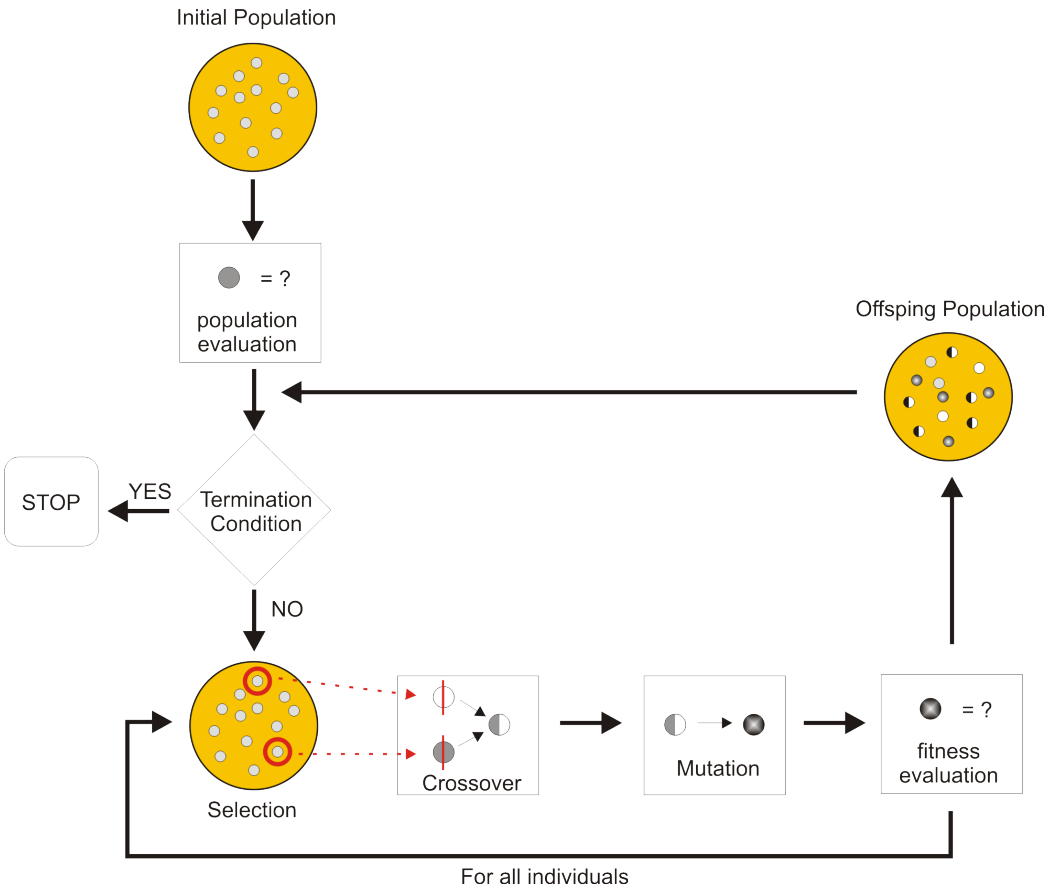
- Encoding technique (*gene, chromosome*)
- Initialization procedure *(creation)*
- Evaluation function *(environment)*
- Selection of parents *(reproduction)*
- Genetic operators *(mutation, recombination)*
- Parameter settings *(practice and art)*

Simple Genetic Algorithm

```
{
    initialize population;
    evaluate population;
    while TerminationCriteriaNotSatisfied
    {
        select parents for reproduction;
        perform recombination and mutation;
        evaluate population;
    }
}
```

GA Functioning

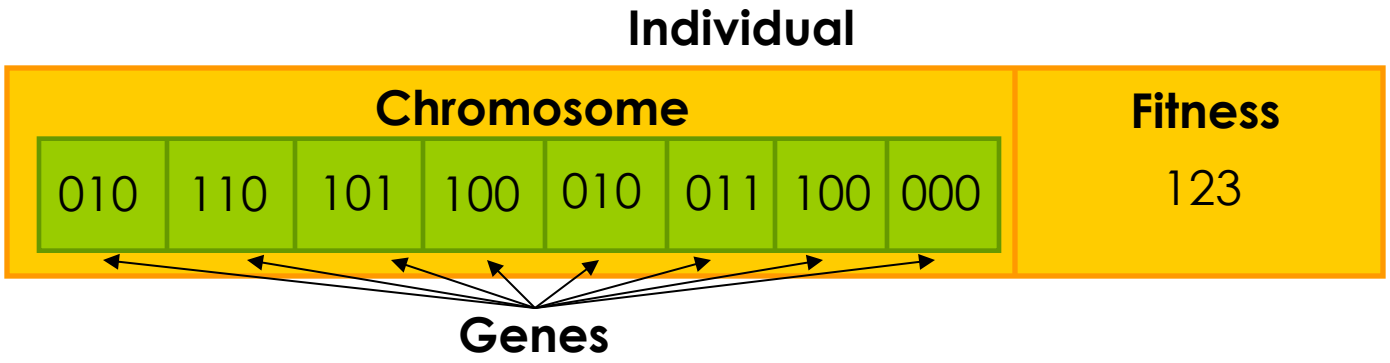
- It is based on a set (population) of potential solutions (individuals) on which it applies some stochastic operators in order to search for an optimum.
- It uses a single population (panmixia) of individuals and apply operators to them as a whole.



Population

Chromosomes could be:

- 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 ...



Evaluation

- The evaluator decodes a chromosome and assigns it a fitness measure
- The evaluator is the only link between a classical GA and the problem it is solving

GA Operators

There are 3 main operators for a serial GA:

Selection

Crossover

Mutation

Selection

- The method by which population members (candidate solutions) are chosen.

- The chosen individuals will be combined with each other to form offspring.

Selection methods

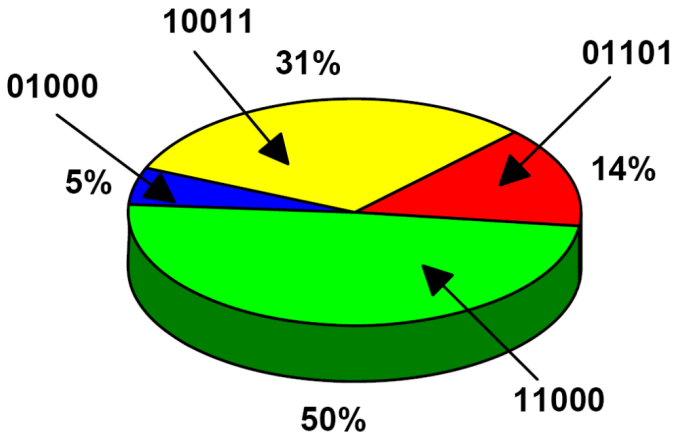
Common selection methods used in GAs are:

- Fitness Proportionate Selection
- Rank Selection
- Tournament Selection

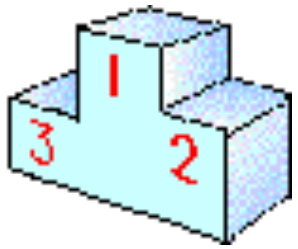
Fitness proportionate selection

- Can be achieved using the roulette wheel algorithm
  - Construct a roulette wheel with a marker proportional to the fitness of each individual as shown.
  - When the arrow is spun the probability of selecting an individual is thus proportional to the fitness of that individual.

$$P_{Si} = \frac{f_i}{\sum_{j=1}^{n} f_j}$$



Rank Selection



- All individuals are sorted according to their fitness.
- Each individual is then assigned a probability of being selected from some prior probability density.

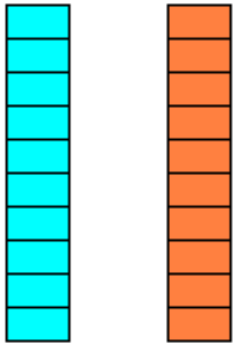
Tournament Selection

- Select a group of *N* *(N>1)* members (usually N=2 → binary tournament)

- Select the fittest member of this group and discard the rest.

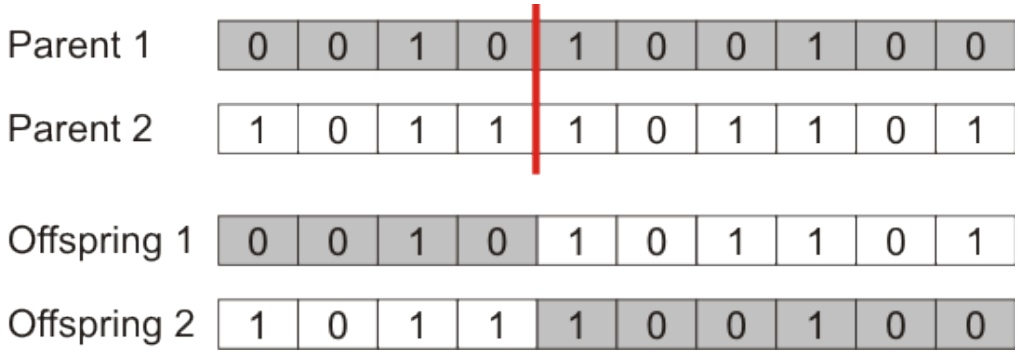
Crossover

- The means by which individuals are combined to form offspring.
- A method of mixing solutions to produce better ones
- Applied with a probability p_c ($0.6 < p_c < 1$)



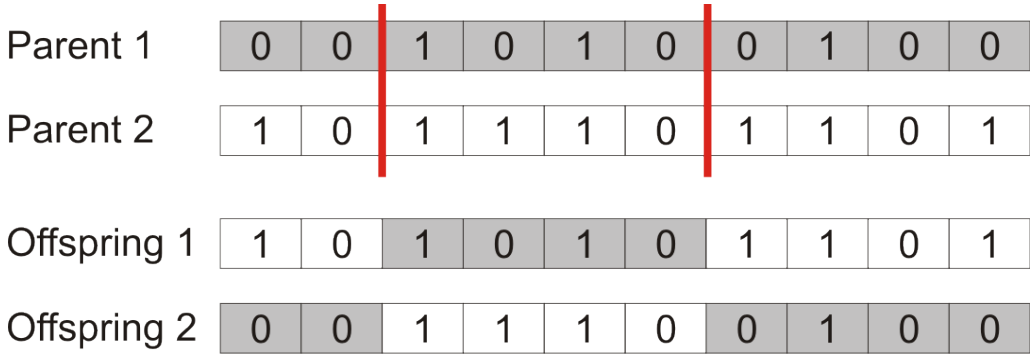
1-Point Crossover

- Randomly selects one position
- Interchanges the two parent chromosomes at this point to produce two new offspring



2-Point Crossover

- Randomly selects two positions



Uniform Crossover

- A crossover mask, the same length as the individual structure, is created at random
- The parity of the bits in the mask indicates which parent will supply the offspring at every position

Parent 1	0	0	1	0	1	0	0	1	0	0
Parent 2	1	0	1	1	1	0	1	1	0	1
Mask 1	0	1	1	0	0	0	1	1	0	1
Mask 2	1	0	0	1	1	1	0	0	1	0
Offspring 1	1	0	1	1	0	1	0	1	1	1
Offspring 2	1	1	0	1	1	0	0	0	0	0

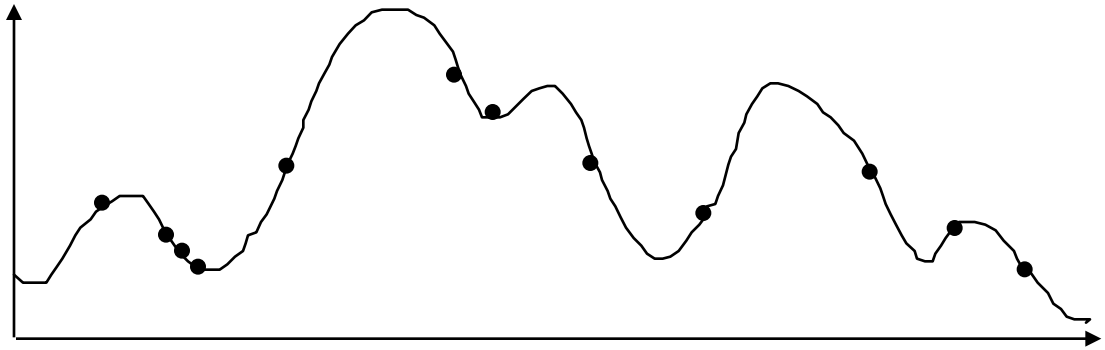
Mutation

- An operator to introduce diversity within the population
- The probability p_m of mutating a particular bit is typically very small ($0.001 < p_m < 0.01$)

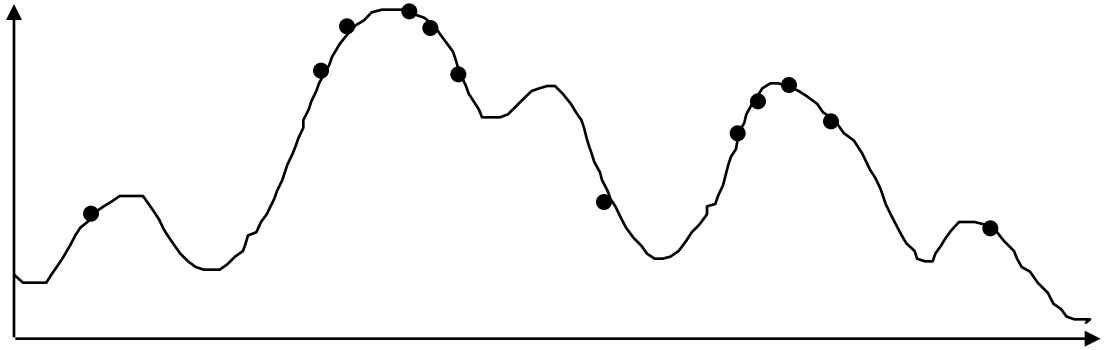
Before:	(0	1	1	1	0	1	1	0)
After:	(0	1	1	0	0	1	1	0)

Before:	(1.38	-69.4	326.44	0.1)
After:	(1.38	-67.5	326.44	0.1)

Abstract Example

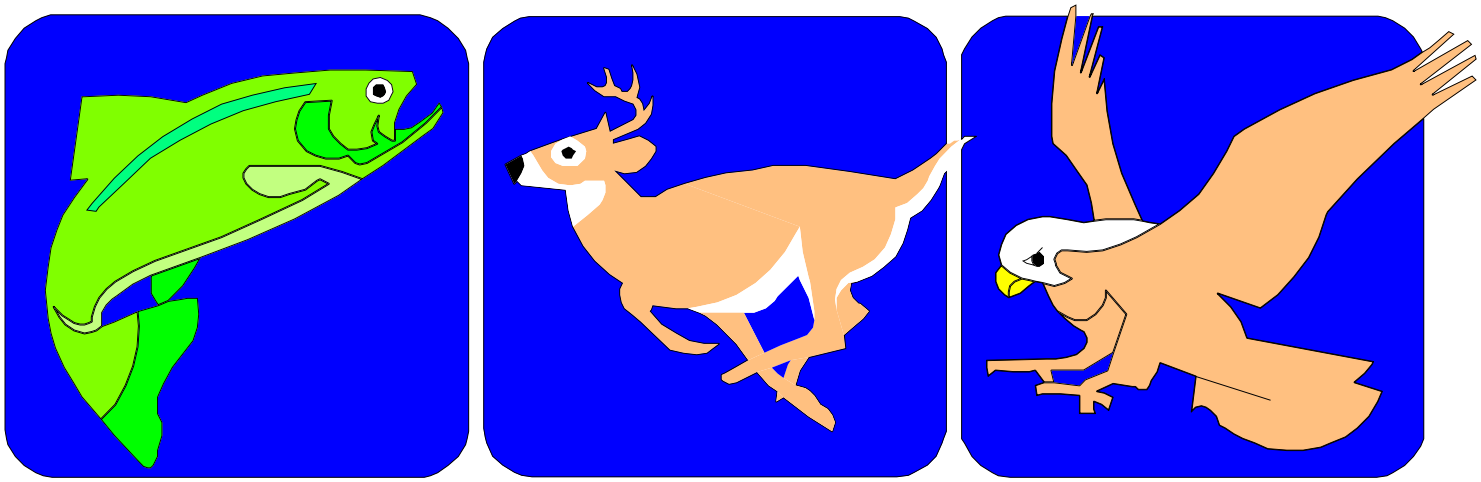


Distribution of Individuals in Generation 0



Distribution of Individuals in Generation N

A Simple Example



"The Gene is by far the most sophisticated program around."

- Bill Gates, *Business Week*, June 27, 1994

A Simple Example

The Traveling Salesman Problem:

Find a tour of a given set of cities so that

- each city is visited only once
- the total distance traveled is minimized

Representation

Representation is an ordered list of city numbers known as an *order-based* GA.

1) London	3) Dunedin	5) Beijing	7) Tokyo
2) Venice	4) Singapore	6) Phoenix	8) Victoria

CityList1 (3 5 7 2 1 6 4 8)

CityList2 (2 5 7 6 8 1 3 4)

Crossover

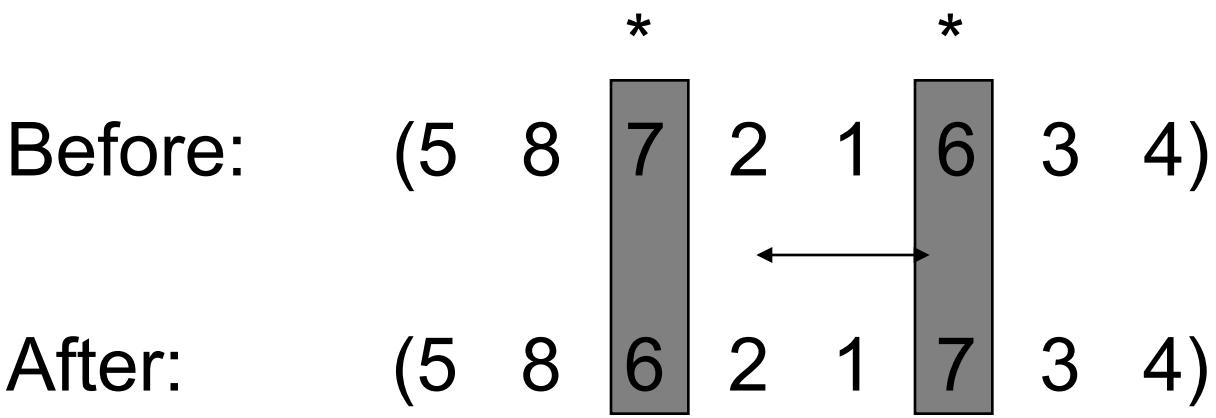
Crossover combines inversion and recombination:

			*			*		
Parent1	(3	5	7	2	1	6	4	8)
Parent2	(2	5	7	6	8	1	3	4)
Child	(5	8	7	2	1	6	3	4)

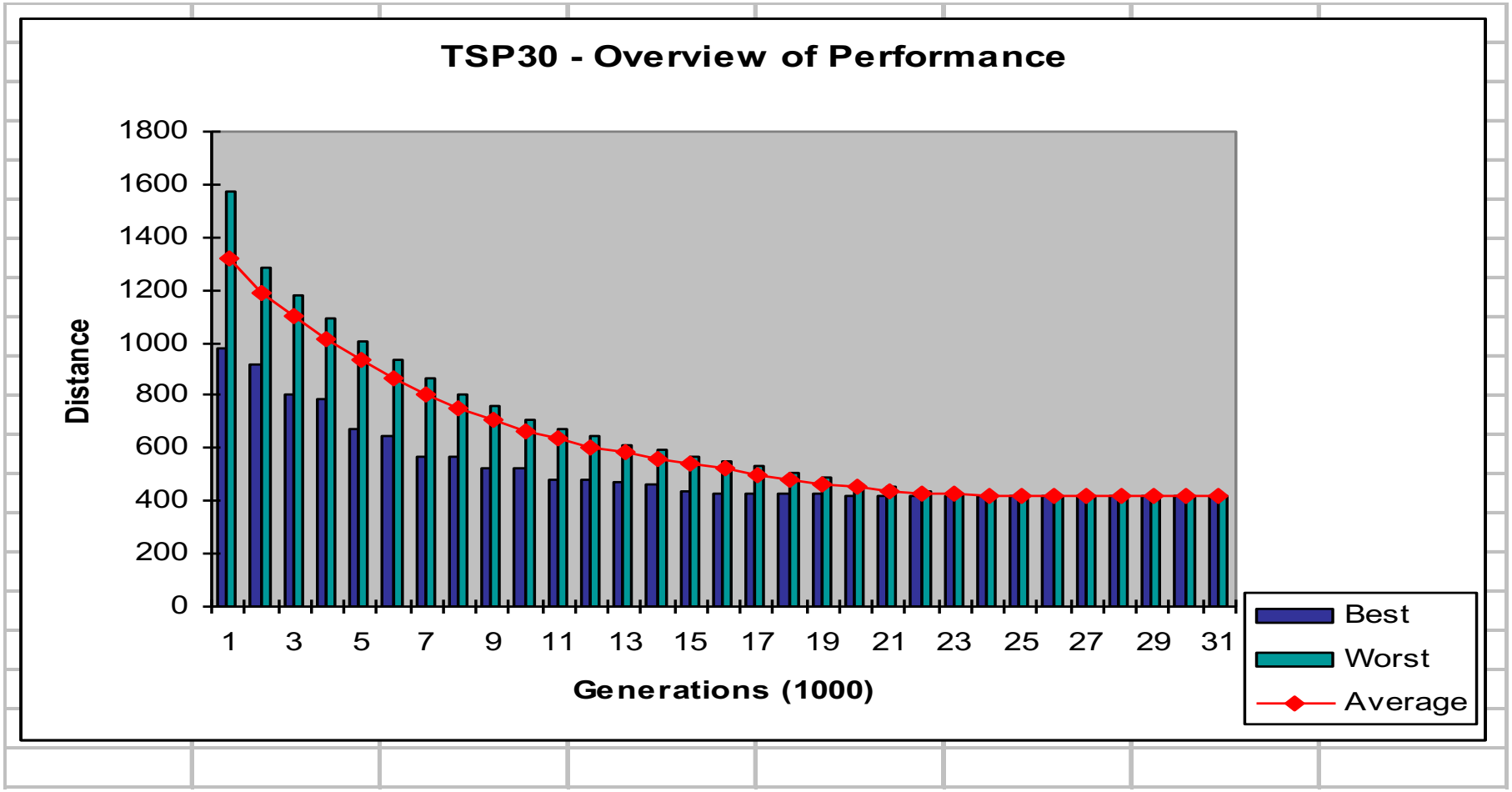
This operator is called the *Order1* crossover.

Mutation

Mutation involves reordering of the list:



Overview of Performance



Issues for GA Practitioners

- Choosing basic implementation issues:
  - representation
  - population size, mutation rate, ...
  - selection, deletion policies
  - crossover, mutation operators
- Termination Criteria
- Performance, scalability
- Solution is only as good as the evaluation function (often hardest part)

Benefits of Genetic Algorithms

- Concept is easy to understand
- Modular, separate from application
- Supports multi-objective optimization
- Good for “noisy” environments
- Always an answer; answer gets better with time
- Inherently parallel; easily distributed

Benefits of Genetic Algorithms

- Many ways to speed up and improve a GA-based application as knowledge about problem domain is gained
- Easy to exploit previous or alternate solutions
- Flexible building blocks for hybrid applications
- Substantial history and range of use

When to Use a GA

- Alternate solutions are too slow or overly complicated
- Need an exploratory tool to examine new approaches
- Problem is similar to one that has already been successfully solved by using a GA
- Want to hybridize with an existing solution
- Benefits of the GA technology meet key problem requirements

Some GA Application Types

Domain	Application Types
Control	gas pipeline, pole balancing, missile evasion, pursuit
Design	semiconductor layout, aircraft design, keyboard configuration, communication networks
Scheduling	manufacturing, facility scheduling, resource allocation
Robotics	trajectory planning
Machine Learning	designing neural networks, improving classification algorithms, classifier systems
Signal Processing	filter design
Game Playing	poker, checkers, prisoner's dilemma
Combinatorial Optimization	set covering, travelling salesman, routing, bin packing, graph colouring and partitioning