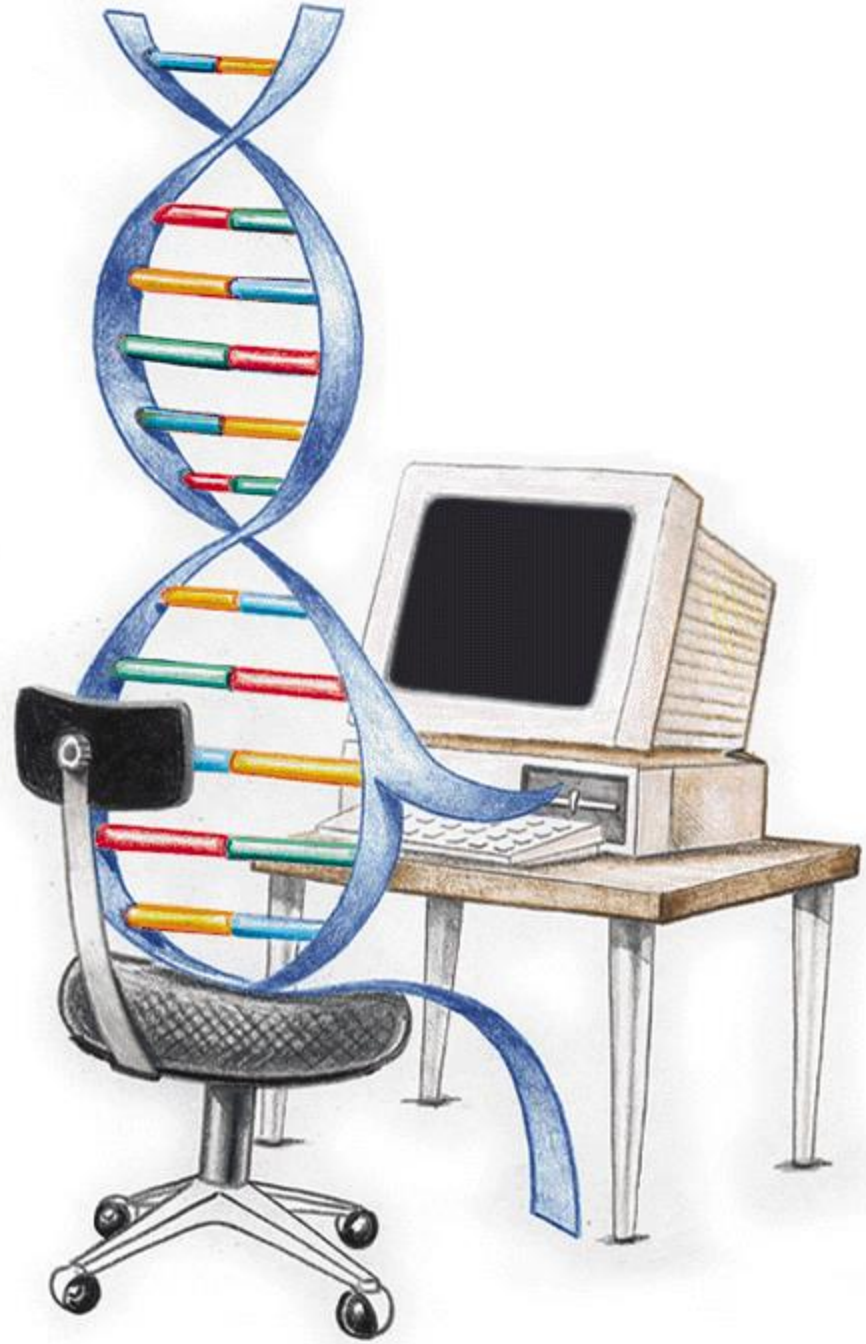
Nature inspired computing



Prof Marko Robnik Šikonja, PhD
November 2014

Evolutionary and natural computation

- Many engineering and computational ideas from nature work fantastically!
- Evolution as an algorithm
- Abstraction of the idea:
- progress, adaptation - learning, optimization
- Survival of the fittest - competition of agents, programs, solutions
- Populations – parallelization
- (Over)specialization – local extremes

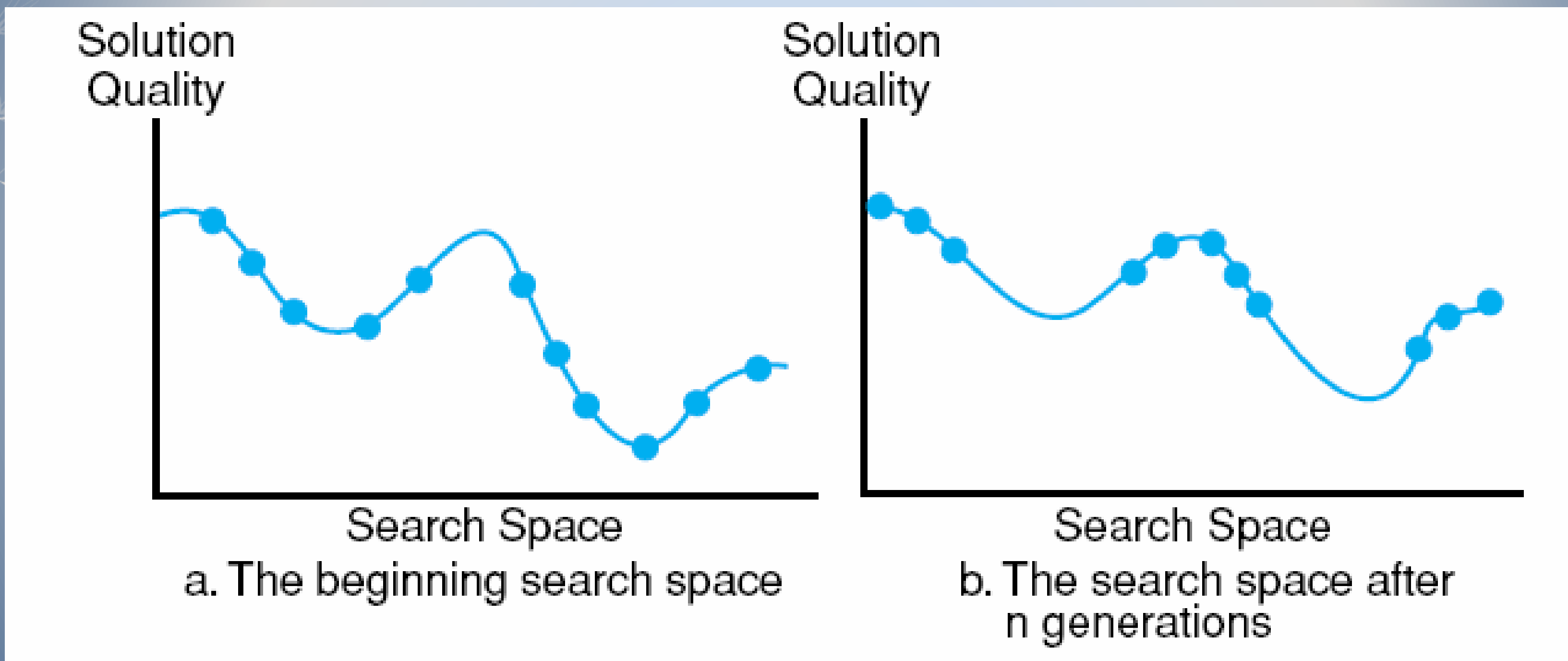
Template of evolutionary program

generate a population of agents (objects, data structures)

```
do {
    compute fitness (quality) of the agents
    select candidates for the reproduction using fitness
    create new agents by combining the candidates
    replace old agents with new ones
} while (not satisfied)
```

- immensely general -> many variants

A result of successful evolutionary program



Strengths and weaknesses

- …
- no-free-lunch theorem

Main approaches

- Genetic algorithms
- Genetic programming
- Swarm methods (particles, ants, bees, …)
- Self organized fields,
- ….

Biological equivalents

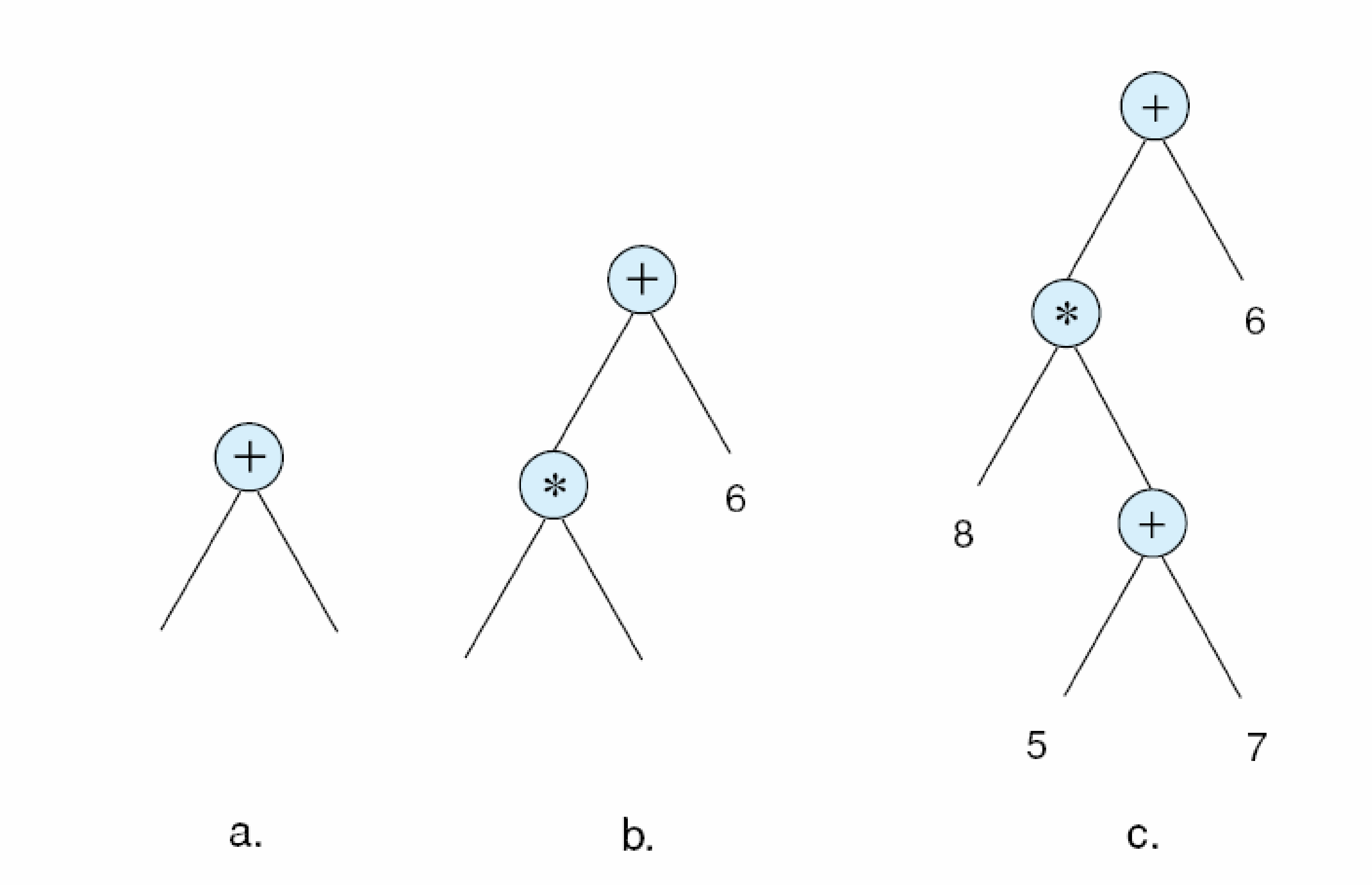
- Gen, allele
- Evolution is a variation of alleles frequencies through time.
- Reproduction, variation (mutation, crossover), selection

Evolutionary computation keywords

- Representation: data structures, operations
- Fitness, heuristics
- Population variability
- Local and global extremes
- Coevolution
- Variability of fitness function

Gen representation

- Bit vector
- Numeric vectors
- Strings
- Permutations
- Trees: functions, expressions, programs
- …


+
+
*
6
+
*
6
8
+
5
7
a.
b.
c.

Crossover

- Single point/multipoint
- Shall preserve individual objects
  - Bit representation

Parents: 1101011100 0111000101

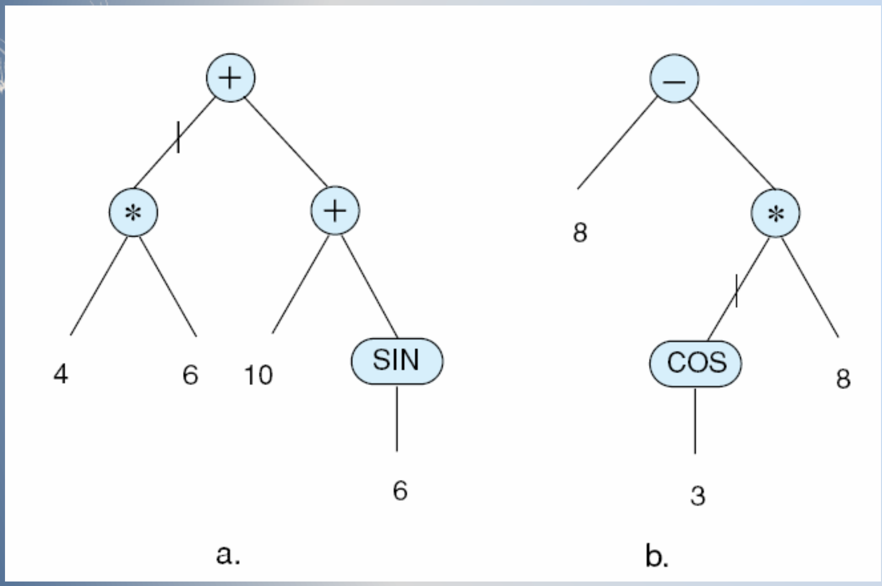
Children: 1101010101 0111001100

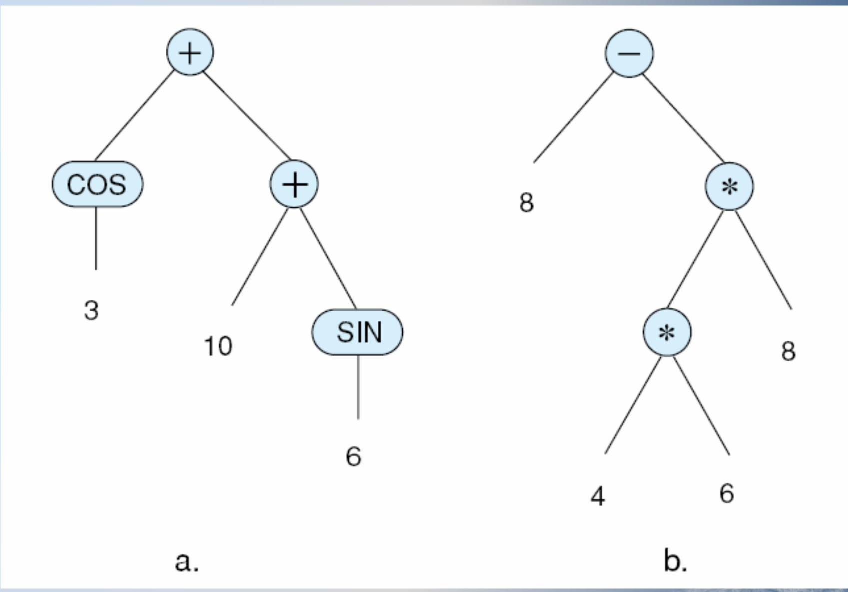
- Vector representation

Parents: (6.13, 4.89, 17.6, 8.2) (5.3, 22.9, 28.0, 3.9)

Children: (6.13 , 22.9, 28.0, 3.9) (5.3, 4.89, 17.6, 8.2)

Crossover on trees





Gray coding of binary numbers

- Keeping similarity

Binary	Gray
0000	0000
0001	0001
0010	0011
0011	0010
0100	0110
0101	0111
0110	0101
0111	0100
1000	1100
1001	1101
1010	1111
1011	1110
1100	1010
1101	1011
1110	1001
1111	1000

Adaptive crossover

- Different evolution phases
- Crossover templates
- 0 – first parent, 1 second parent
- Different dynamics of template crossover

	Gene	Template
Parent 1	1.2 3.4 5.6 4.5 7.9 6.8	010101
Parent 2	4.7 2.3 1.6 3.2 6.4 7.7	011100
Child 1	1.2 2.3 5.6 3.2 7.9 7.7	010100
Child 2	4.7 3.4 1.6 4.5 6.4 6.8	011101

Mutation

- Adding new information
- Random search?
- Binary representation:

  0111001100 --> 0011001100

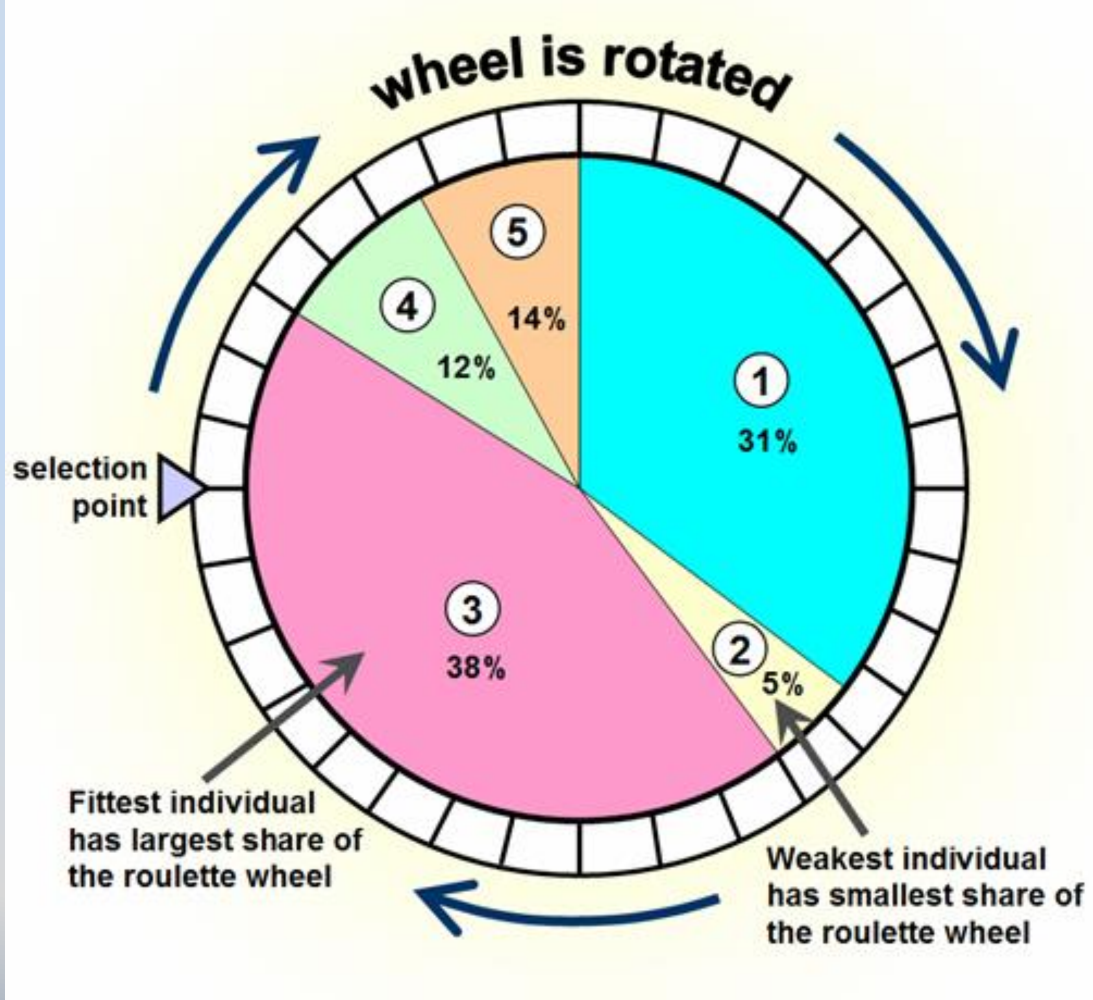
- Single point/multipoint
- Lamarckian

Evolutional model

- Keeping the good
- Prevent premature convergence
- heterogeneity of population

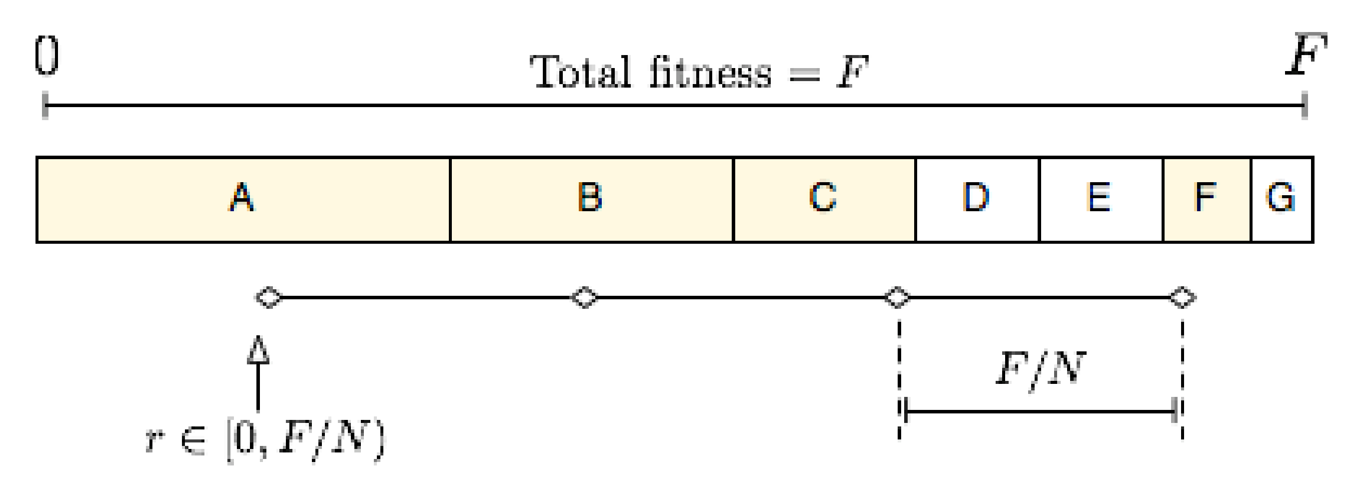
Selection

- Proportional
- Rank proportional
- Tournament
- Single tournament
- Stochastic universal sampling



Stochastic universal sampling (SUS)

- unbiased
- r ∈ [0, F/N]
- r + i*N, i ∈ 0, 1,..., N-1]



Replacement

- All
- According to fitness
- Elitism
- Local elitism

Population size

Niche specialization

$f'_i = f_i / q(r,i)$

$$q(r,i) = \begin{cases} 1 & ;\ sim(i) <= 4, \\ sim(i)/4 & ;\ otherwise \end{cases}$$

Stopping criteria

Why genetic algorithms work?

- building blocks hypothesis
- Controversial (mutations)

Applications

Where to use evolutionary algorithms?

- Local extremes
- Just fitness, without derivations
- No specialized methods
- Multiobjective optimization
- Robustness
- Combined approaches

Multiobjective optimization

- Fitness function with several objectives
- $\min F(x) = \min (f_1(x), f_2(x), \ldots, f_n(x))$
- Pareto optimal solution

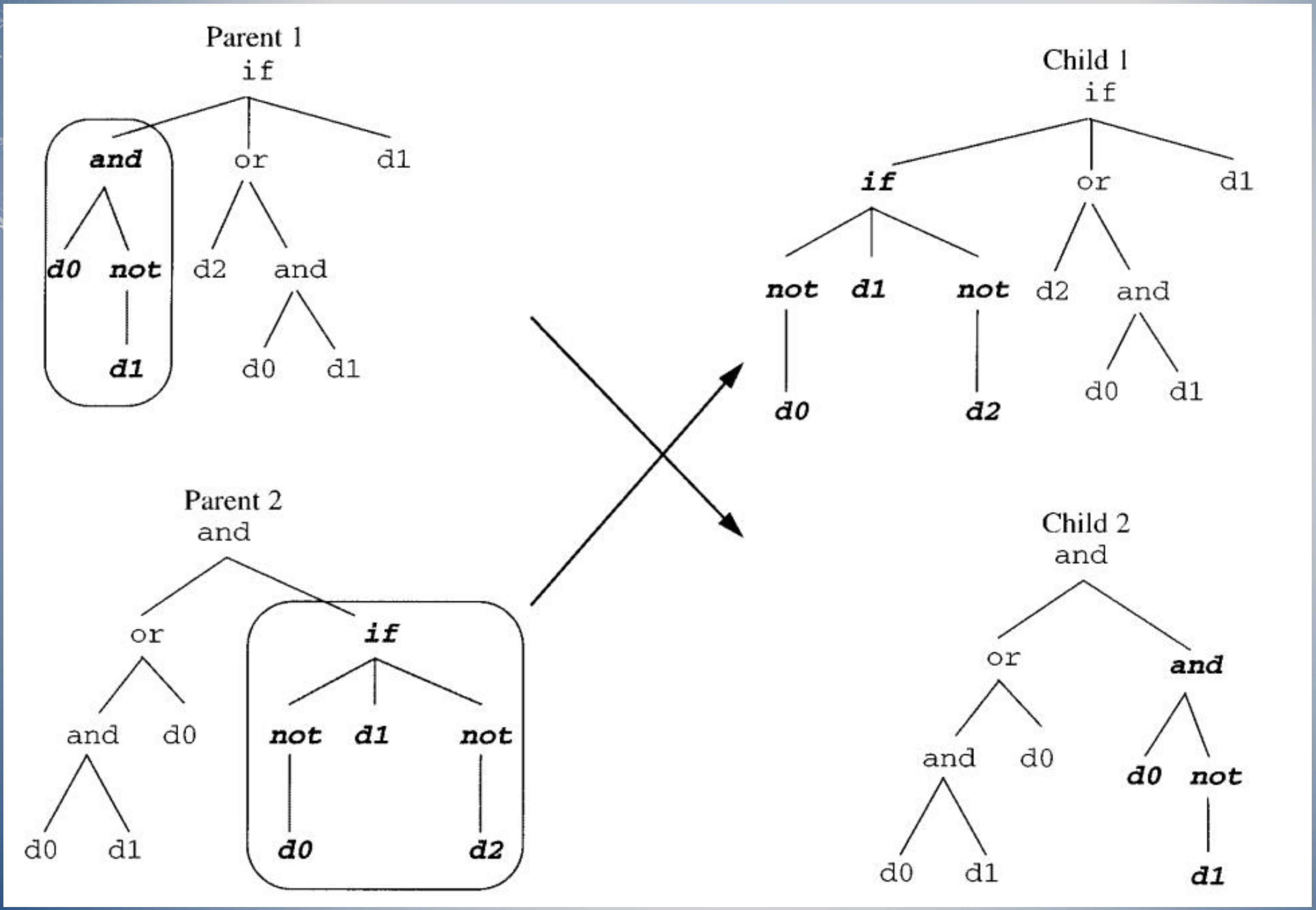
Toolboxes and libraries

- CIlib – computational intelligence library
- EO (C++) - evolutionary computation library
- ECJ (Java)
- EvA2 (Java),
- JAGA (Java)
- ECF- Evolutionary Computation Framework (C++)
- Matlab, …
- R: Rfreak, ppso, numDeriv,…

Genetic programming

- Functions, programs, expression trees
- Keep the structures valid
- Tree crossover, type closure
- applications

Crossover of expression trees



Other nature inspired methods

- properties
  - Fixed population
  - Communication between agents
- particle swarm optimization
  - Collective behavior: swarms of insects, fish, bird flocks
  - Autonomous individual
  - Communication, sharing food information
- ant colony optimization

Swarm intelligence, PSO

- Particle location and velocity
  - Location *x = (x_1, x_2, …)*
  - *velocity v = (v_1, v_2, …)*

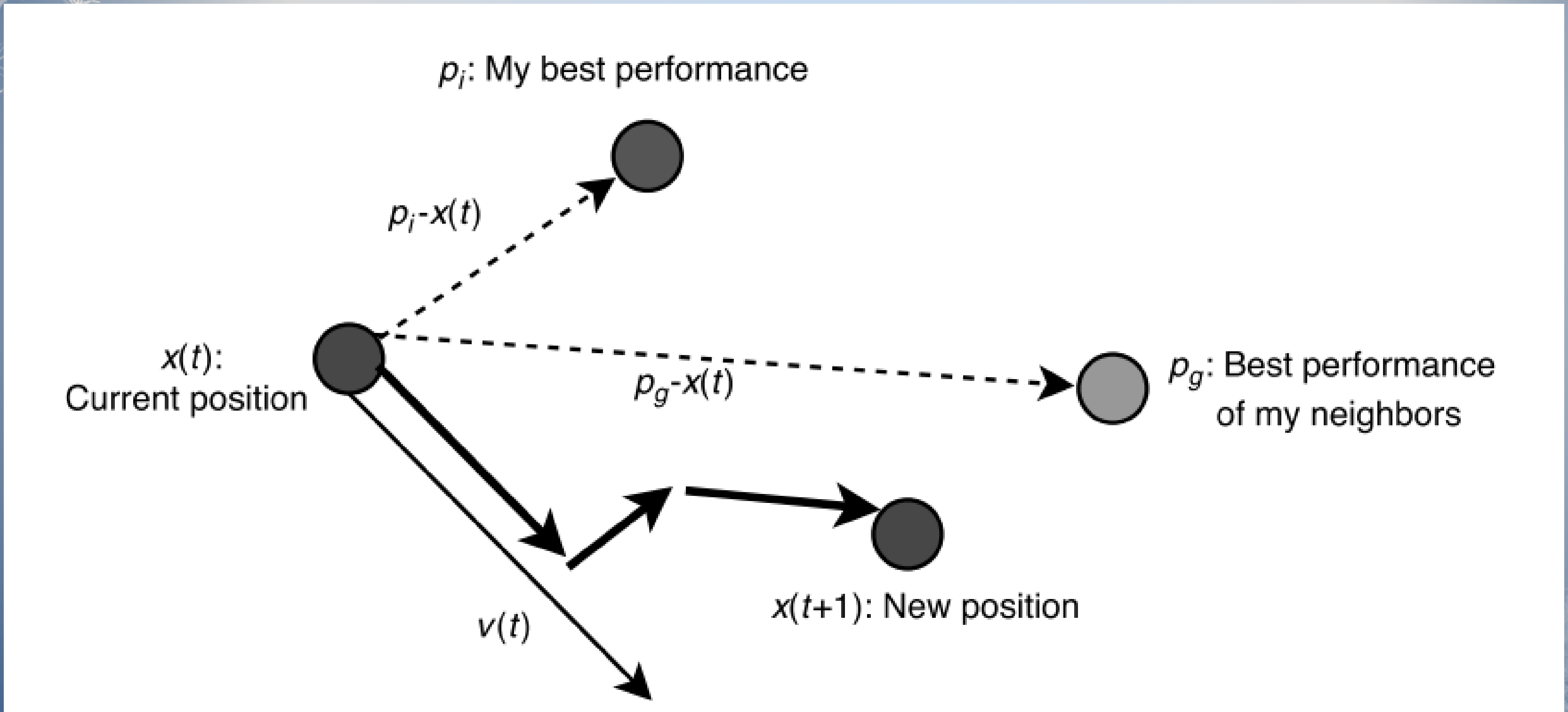
$$\vec{v} = \vec{x}(t) - \vec{x}(t-1)$$

- Initialization of locations and velocities

Information exchange in the swarm

- Historically best location x^*
- Best location of informants x^+
- Globally best location $x^!$

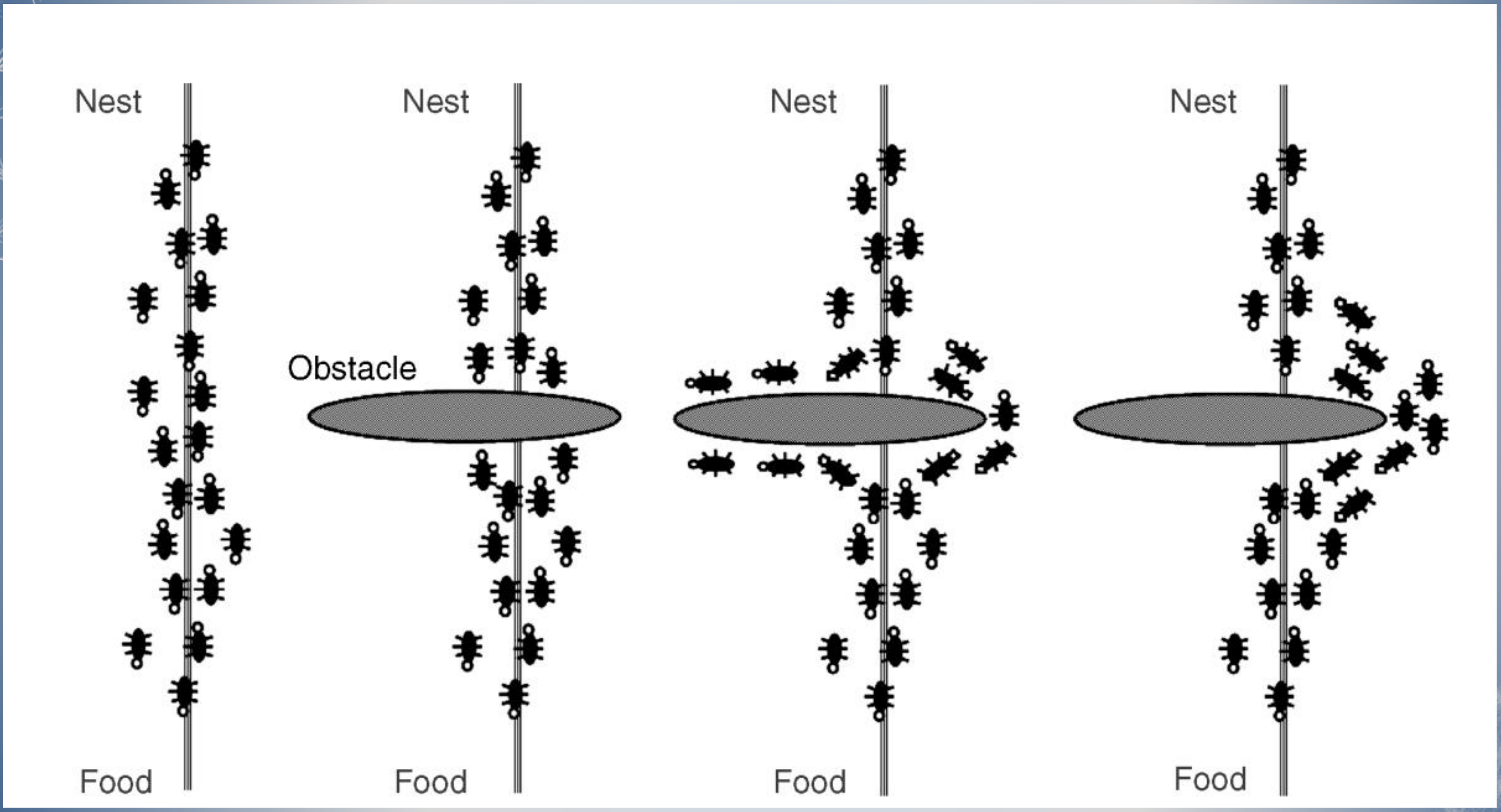
Computing new position



Ant colony

- Pheromones
- Ants lead their sisters to food source
- Evaporation
- Moving targets

Illustration of the idea



ACO pseudo code

```
Initialization of pheromones
do {
   for each  ant
      find solution using pheromones and quality
      update pheromones: enforcement, evaporation
} while (! satisfied)
return best overall solution
```

ACO details

- Pheromones updates
  - ρ speed of evaporation
- Trails updates
- Many variants

$$\tau_{i,j} = (1-\rho)\tau_{i,j} + \Delta\tau_{i,j}$$

$$\Delta\tau_{i,j} = \begin{Bmatrix} 1/C & \text{if ant takes the connection between i, j} \\ 0 & \text{otherwise} \end{Bmatrix},$$

$$\text{where } C \text{ is a cost of edge i, j}$$

ACO for TSP

- cities 1,2,...,n
- cost $c_{i,j}$
- Attractiveness $\eta_{i,j} = 1/ c_{i,j}$
- Probability of ant's transition
- α - impact of pheromones
- β - impact of transition cost

$$p_{i,j} = \frac{\tau_{i,j}^{\alpha}\eta_{i,j}^{\beta}}{\sum \tau_{i,j}^{\alpha}\eta_{i,j}^{\beta}}$$

More ideas

- Bee swarm
- Immune systems
- Simulated annealing
- …