ITMO UNIVERSITY

Saint Petersburg, Russia

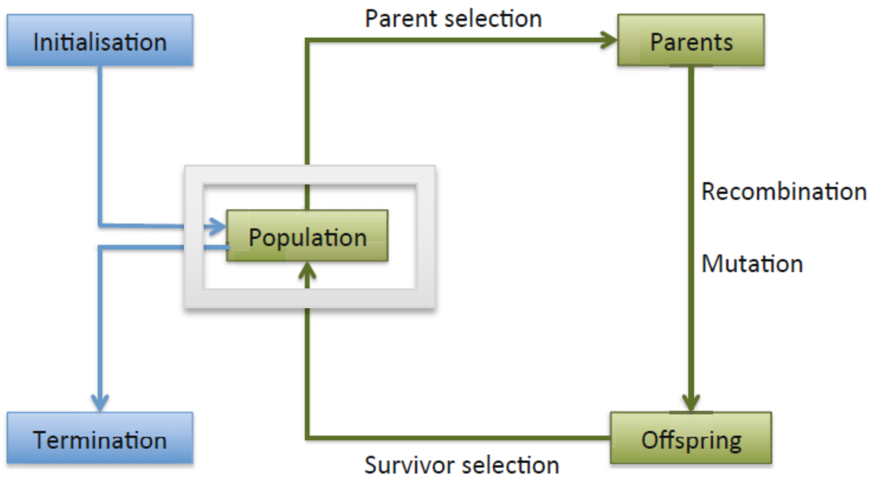
Lecture 2: Representation of individuals

Michael Melnik

mihail.melnik.ifmo@gmail.com

Components of EAs

- **Representation of individuals**
- Population of individuals
- Evaluation function (fitness function)
- Parent selection mechanism
- Variation operators (recombination, mutation)
- Survivor selection mechanism
- Terminate conditions



Representations

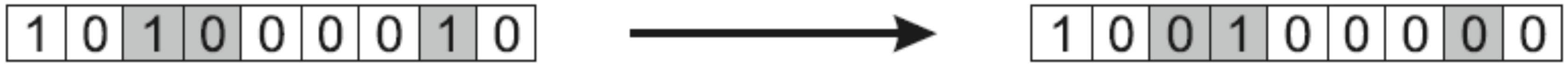
- The first stage of building any evolutionary algorithm is to decide on a genetic representation of a candidate solution to a problem
- Common types of representation:
  - Binary
  - Integer values
  - Real values
  - Permutation
  - Tree representation

Binary representation

1	0	1	0	0	0	0	1	0

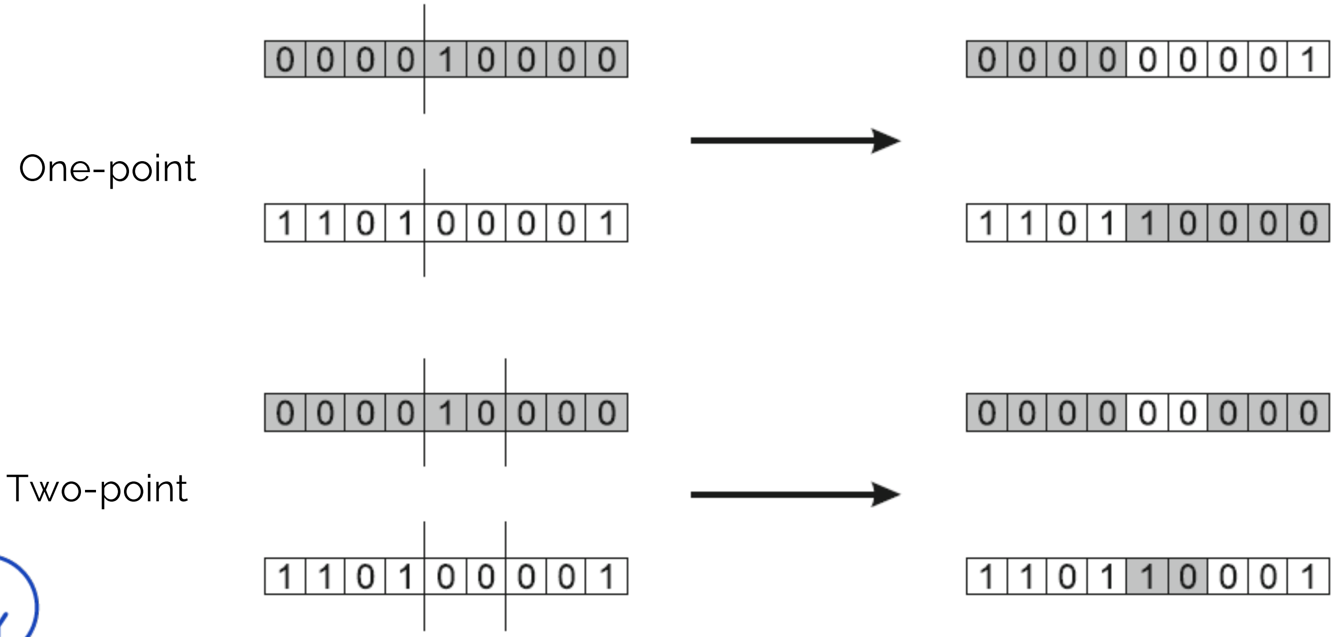
Integer 332 is represented as a binary genotype

Mutation may flip some bits in genotype



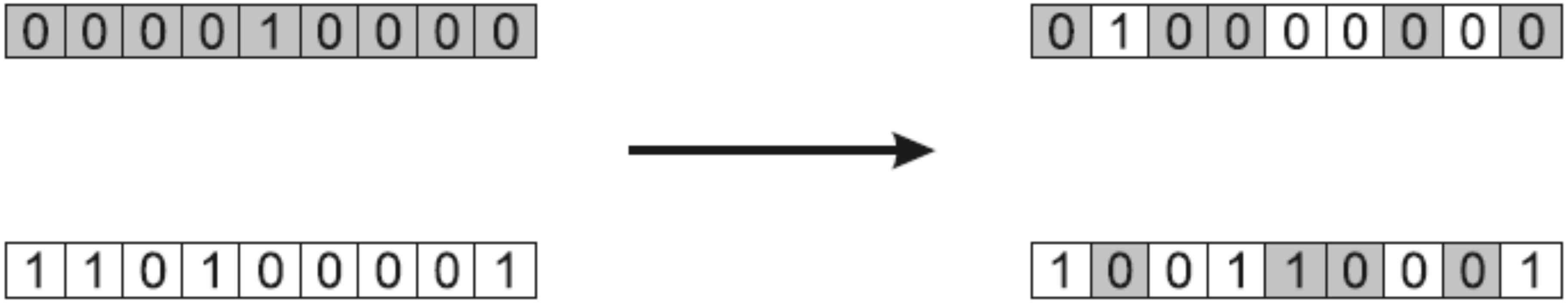
Binary crossover

- Original crossover operator is one-point crossover
- This can be easily extended to n-point crossover



Uniform crossover

- Child may receive genes uniformly



Crossover or mutation

- ✓ Decade long debate: which one is better / necessary / main-background

- ✓ Answer:
  - it depends on the problem, but
  - in general, it is good to have both
  - both have another role
  - mutation-only-EA is possible, xover-only-EA would not work

Crossover OR mutation

- ✓ Only crossover can combine information from two parents
- ✓ Only mutation can introduce new information (genes)

There is co-operation AND competition between them

✓ Crossover is explorative, it makes a *big* jump to an area somewhere “in between” two (parent) areas

✓ Mutation is exploitative, it creates random *small* diversions, thereby staying near (in the area of) the parent

Integer representation

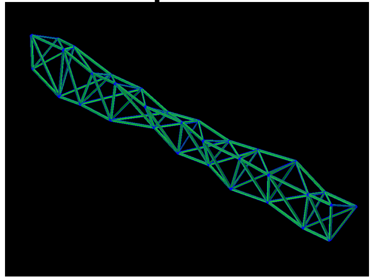
- Nowadays it is generally accepted that it is better to encode numerical variables directly (integers, floating point variables)
- Some problems naturally have integer variables, e.g. image processing parameters
- Others take categorical values from a fixed set e.g. {blue, green, yellow, pink}
- N-point / uniform crossover operators work
- Extend bit-flipping mutation to make
  - "creep" i.e. more likely to move to similar value
    - Adding a small (positive or negative) value to each gene with probability p.
  - Random resetting (esp. categorical variables)
    - With probability p_m a new value is chosen at random
- Same recombination as for binary representation

Real-valued (floating-point) representation

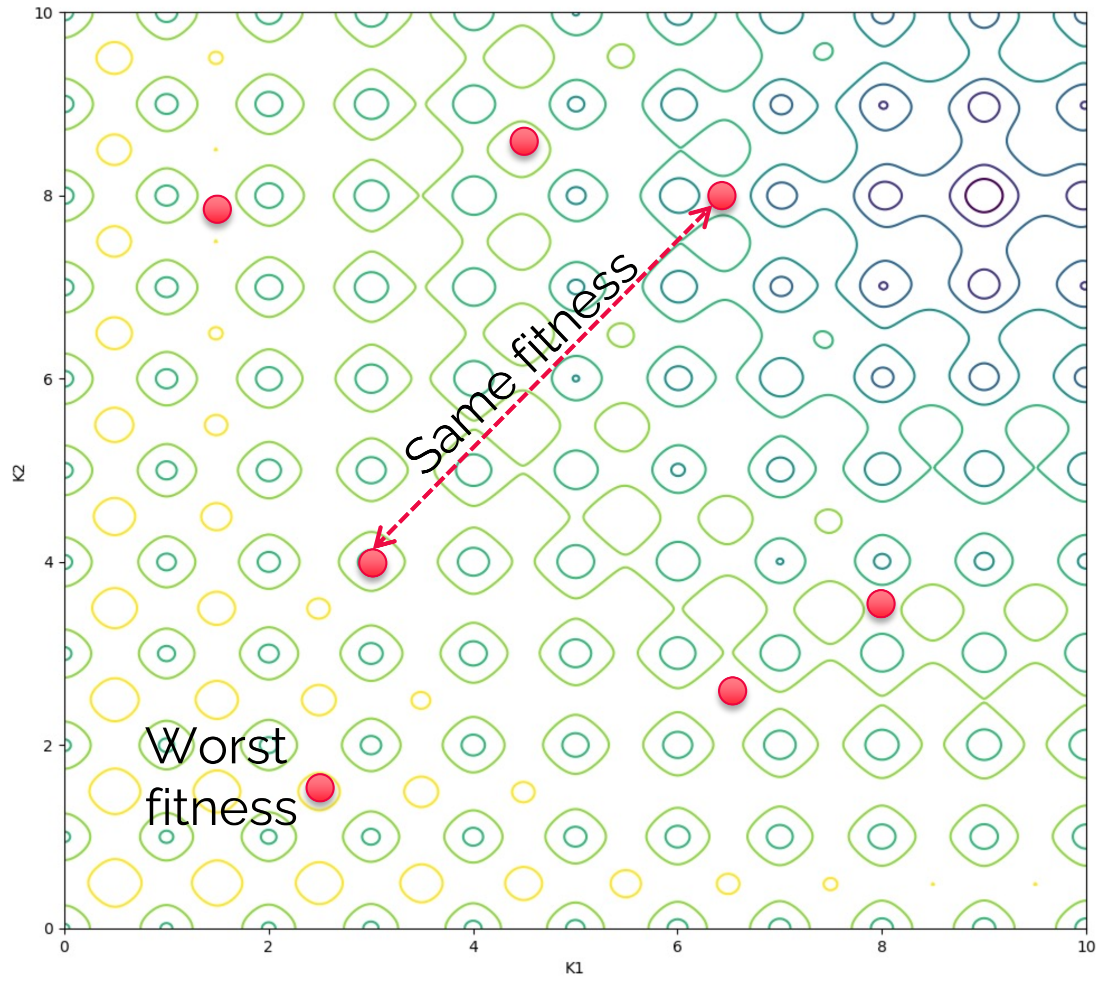
- Genotype for a solution with is now a vector of real values

0.5	0.9	0.1	1.5	0.6

- Floating-point ignores discretization and considers genes as continuous values
- Real valued representation is widely used
- For example: satellite holder can be encoded as a set of float values, which responsible for different angles and spar lengths



Population



- **Randomly generated points: [(x1, x2),....]**

Mutation

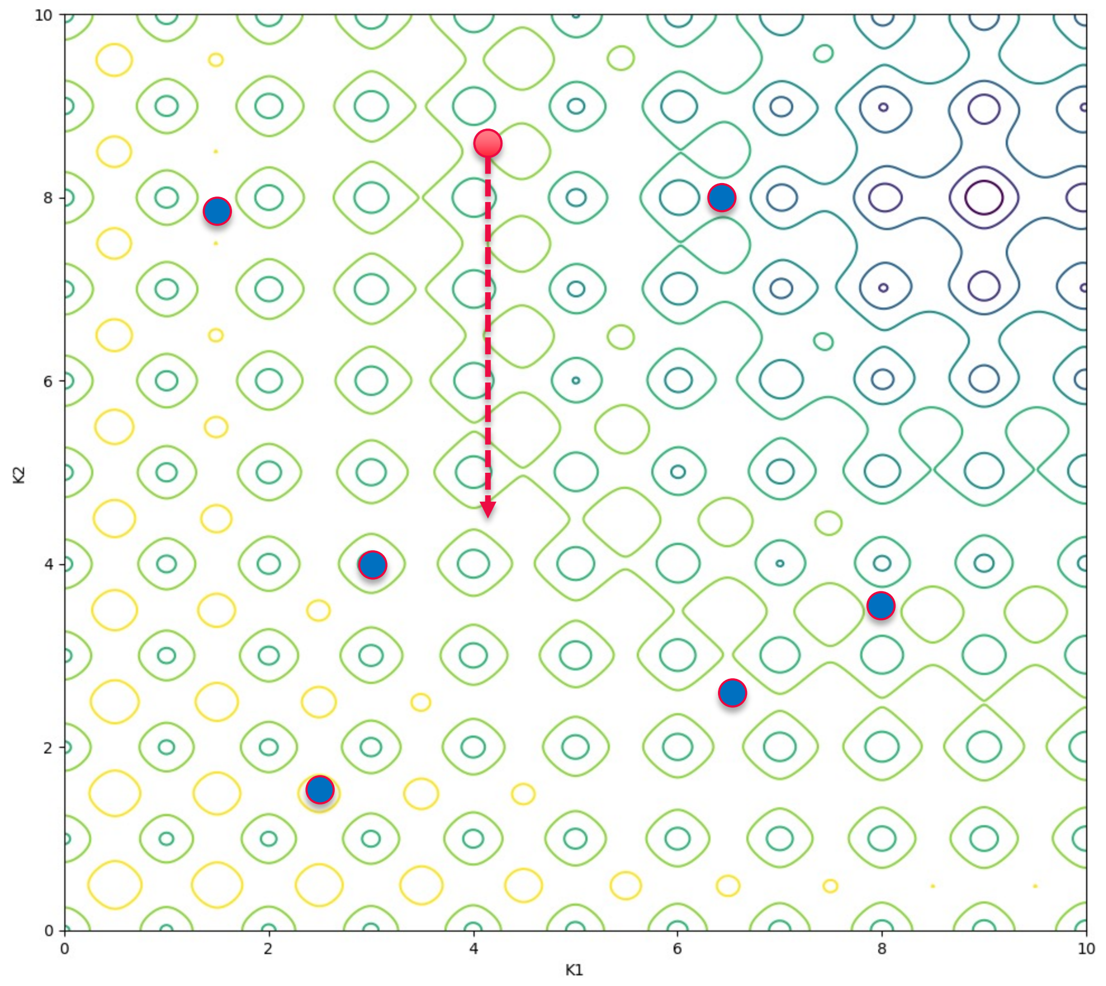
- For real-valued representation it is common to change value of each gene randomly within its domain given by lower and upper bounds

$$< x_1, \dots, x_n > \to < x_1', \dots, x_n' >, where\ x_i, x_i' \in [L_i, U_i]$$

Two types of mutations can be defined:

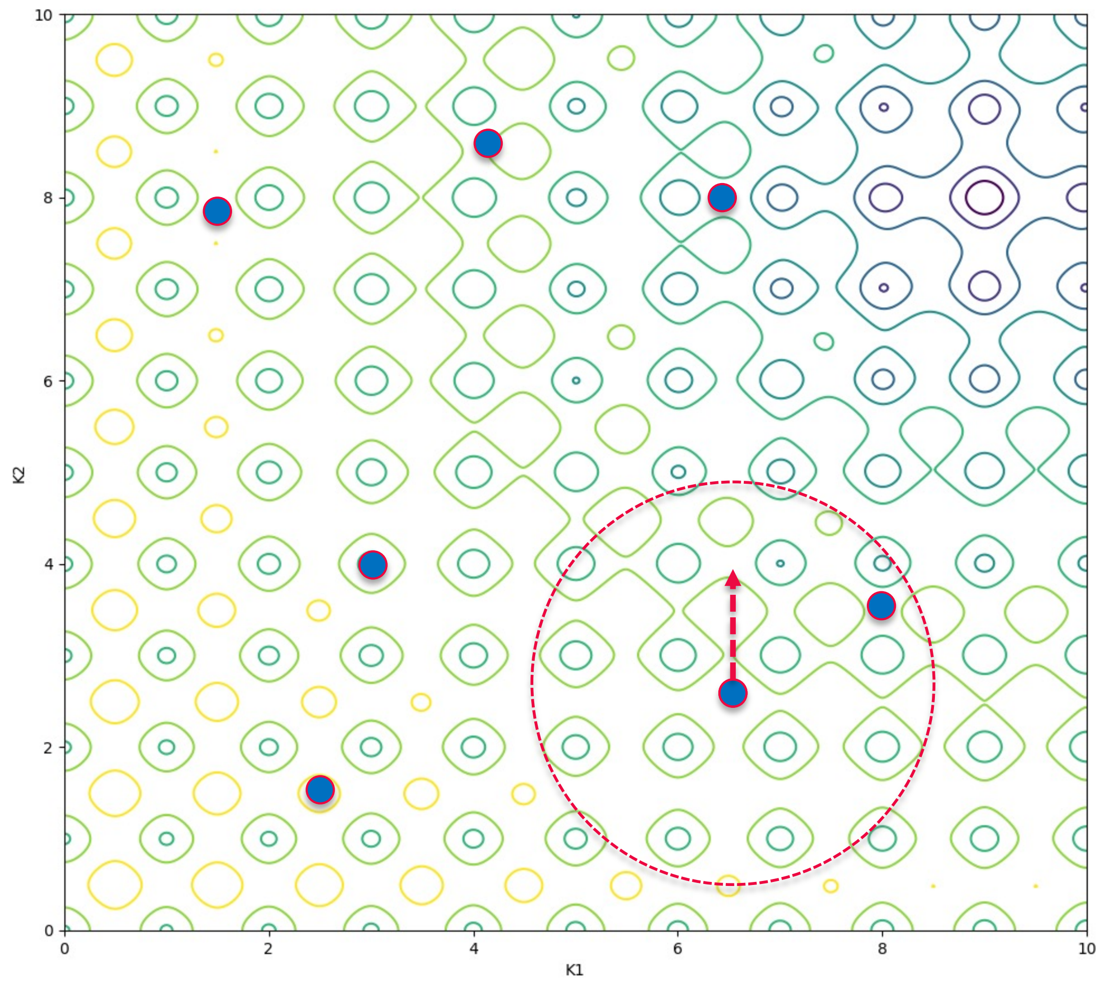
- uniform
- nonuniform

Uniform mutation



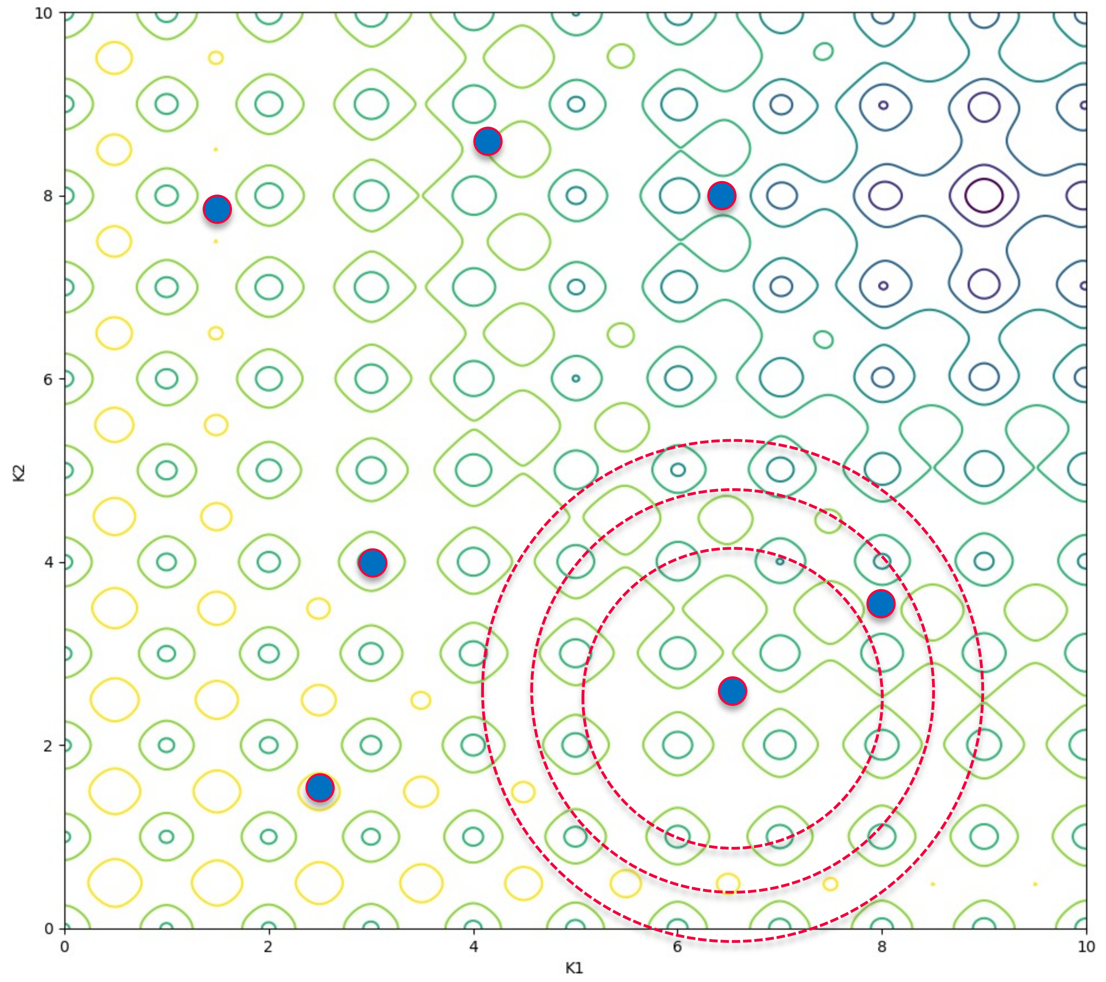
- Uniform random value from range $[L_i, U_i]$.

Nonuniform mutation



- Nonuniform creep mutation.
- $x'_i = x_i + N(0,\sigma)$

Uniform mutation N(0,σ)



How to choose σ?

- ✓ Hand-tuned by problem range
- ✓ Hyper-parameter

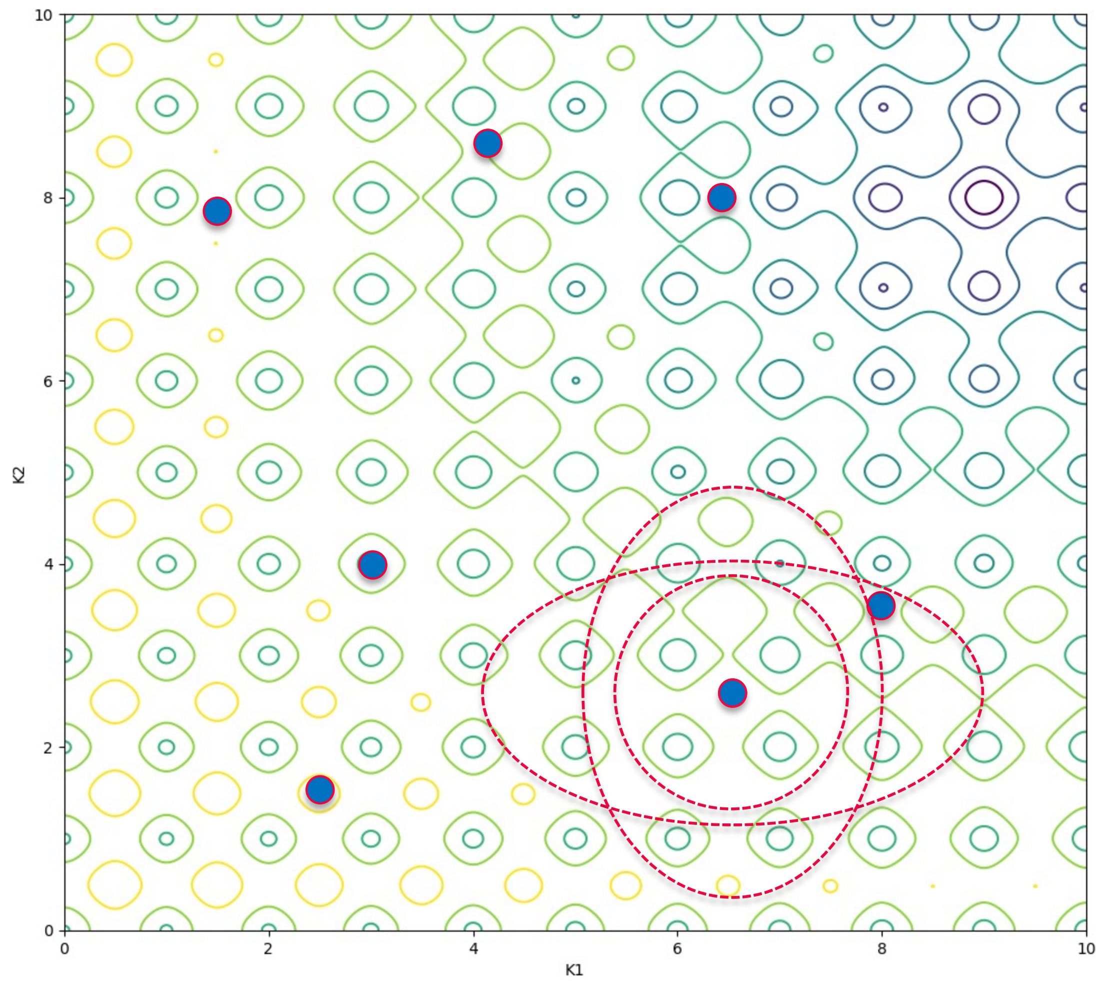
Self-adaptive mutation

- Nonuniform mutation is done by adding some random variables from a Gaussian distribution with specified std.
- Concept of self-adaptive mutation is that std evolves with solutions
- $\langle x_1,\dots,x_n, \sigma \rangle$

Self adaptive mutation

- Mutate σ first
- Mutation effect: ⟨ x, σ ⟩ → ⟨ x', σ' ⟩
- Order is important:
  - first σ → σ'
  - then x → x' = x + N(0,σ')
- Reversing mutation order this would not work
  - Primary: x' is good if f(x') is good
  - Secondary: σ' is good if the x' it created is good

Nonuniform mutation N(0,σ)



How to choose **σ**?

- ✓ Hand-tuned by problem range
- ✓ Hyper-parameter
- ✓ Hyper-parameters for each x value:

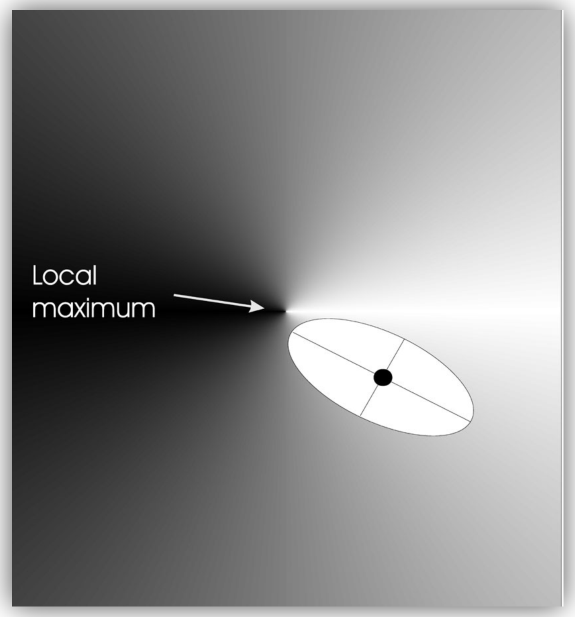
  N(0, σ1, σ2)

Uncorrelated mutation with n stds

- Chromosomes: $\langle x_1,\dots,x_n, \sigma_1,\dots, \sigma_n \rangle$
  - $\sigma'_i = \sigma_i \cdot \exp(\tau' \cdot N(0,1) + \tau \cdot N_i(0,1))$
  - $x'_i = x_i + \sigma'_i \cdot N_i(0,1)$
- Two learning rate parameters:
  - τ' overall learning rate
  - τ coordinate wise learning rate

Correlated mutation

- Mutants with equal likelihood



- Ellipse: mutants having the same chance to be created

Correlated mutation

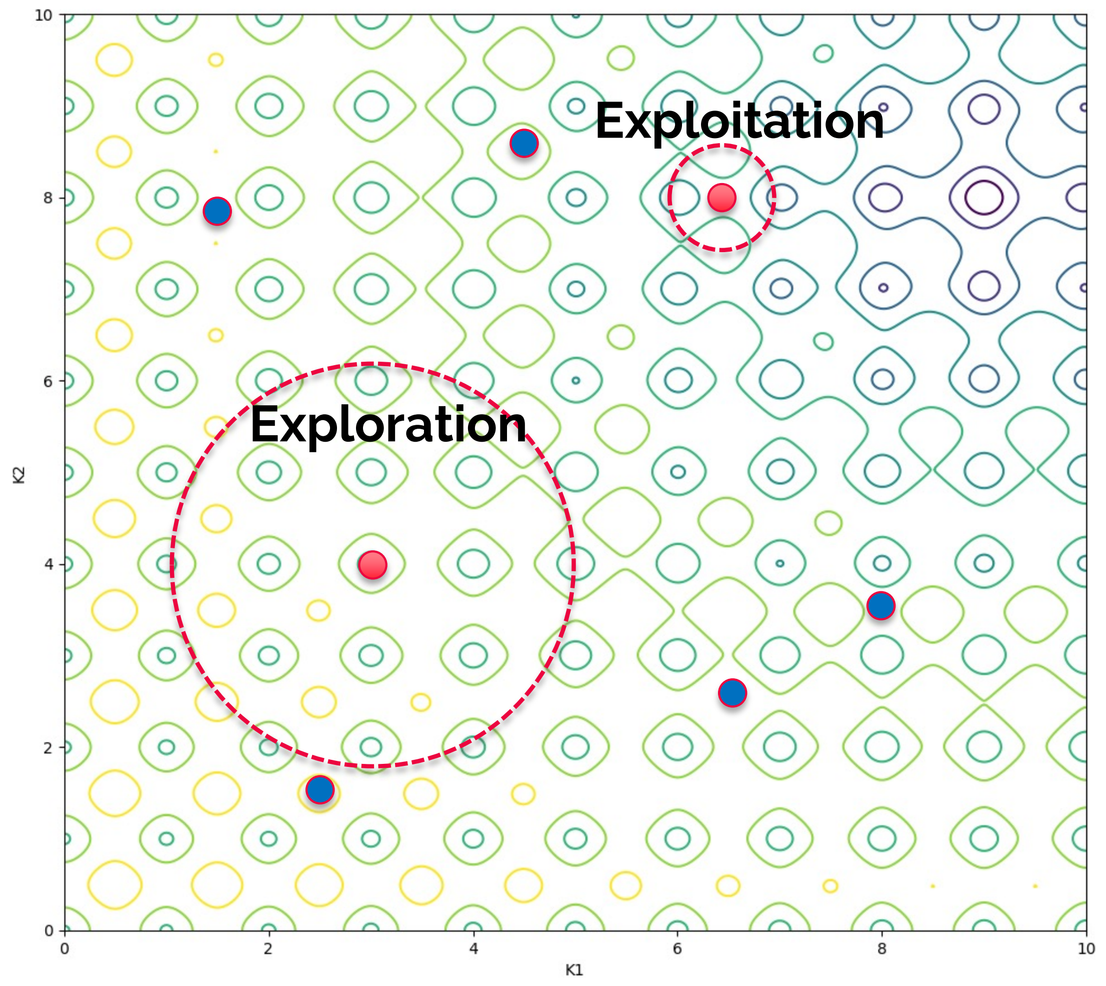
- Chromosomes: $\langle x_1,\dots,x_n, \sigma_1,\dots, \sigma_n, \alpha_1,\dots, \alpha_k \rangle$
  where $k = n \cdot (n-1)/2$
- Covariance matrix C is defined as:
  - $c_{ii} = \sigma_i^2$
  - $c_{ij} = 0$ if i and j are not correlated
  - $c_{ij} = ½ \cdot (\sigma_i^2 - \sigma_j^2) \cdot \tan(2\alpha_{ij})$ if i and j are correlated
- Note the numbering / indices of the α's

Correlated mutation

The mutation mechanism is then:

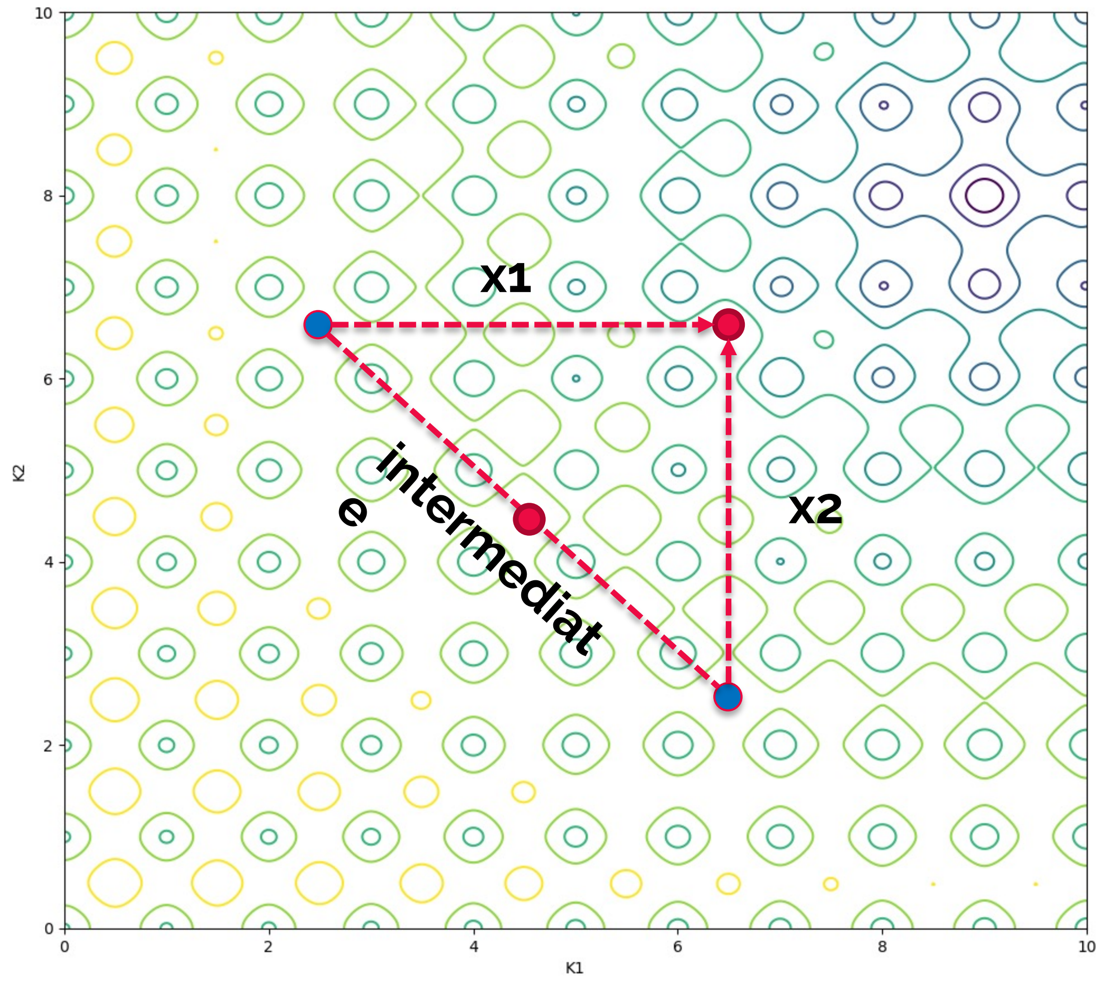
- ✓ $\sigma'_i = \sigma_i \cdot \exp(\tau' \cdot N(0,1) + \tau \cdot N_i(0,1))$
- ✓ $\alpha'_j = \alpha_j + \beta \cdot N(0,1)$
- ✓ $\boldsymbol{x}' = \boldsymbol{x} + N(0, C')$
  - $\boldsymbol{x}$ stands for the vector $\langle x_1, \ldots, x_n \rangle$
  - C' is the covariance matrix C after mutation of the α values

Exploration vs Exploitation

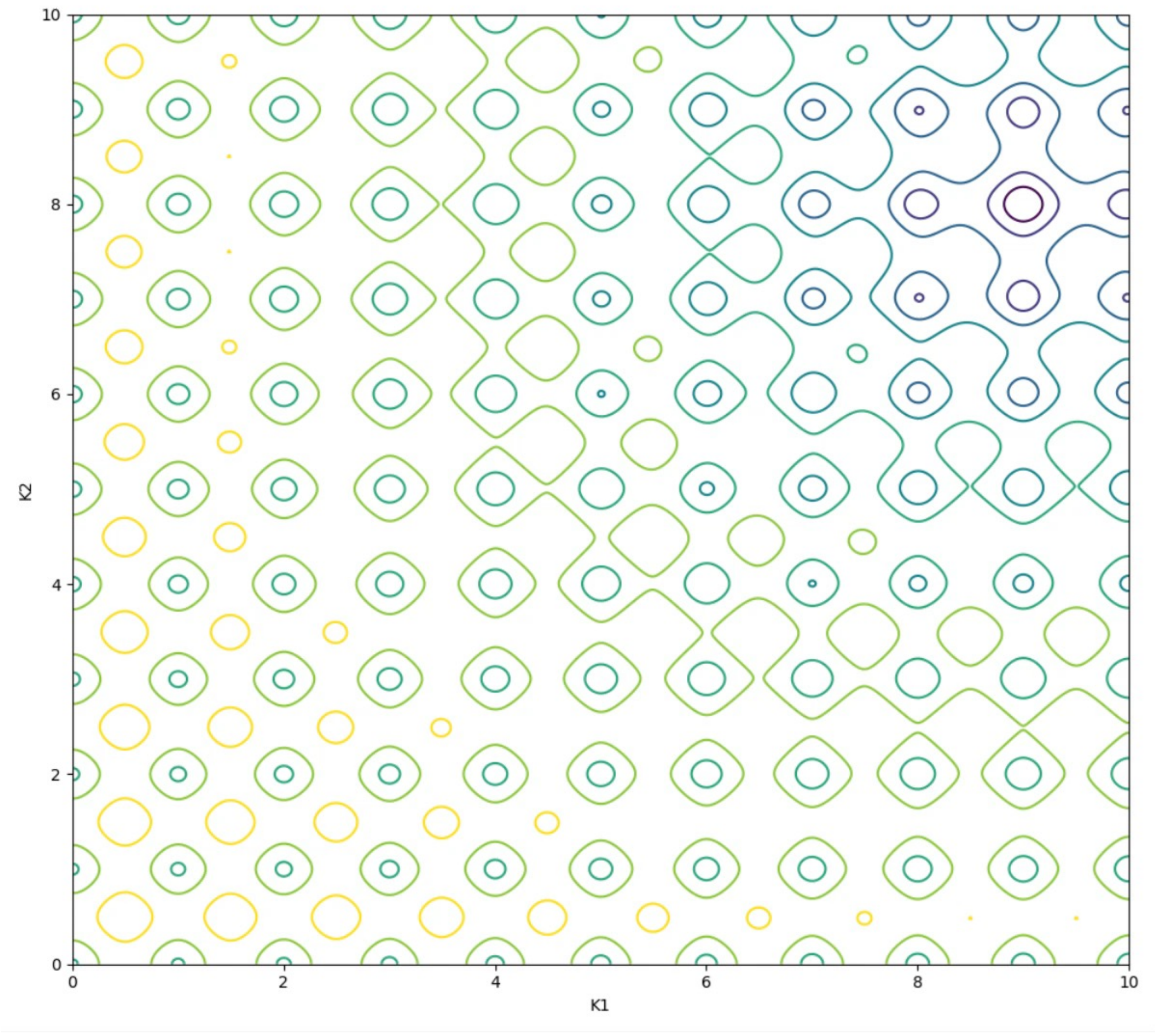


- ✓ Exploration
  - Global search
- ✓ Exploitation
  - Local search

Crossover



- Discrete:
  - each gene value in offspring *z* comes from one of its parents *(x,y)* with equal probability: $z_i = x_i$ or y_i
- Intermediate
  - $z_i = \alpha x_i + (1 - \alpha) y_i$ where $\alpha: 0 \leq \alpha \leq 1$.


K2
K1
0
2
4
6
8
10

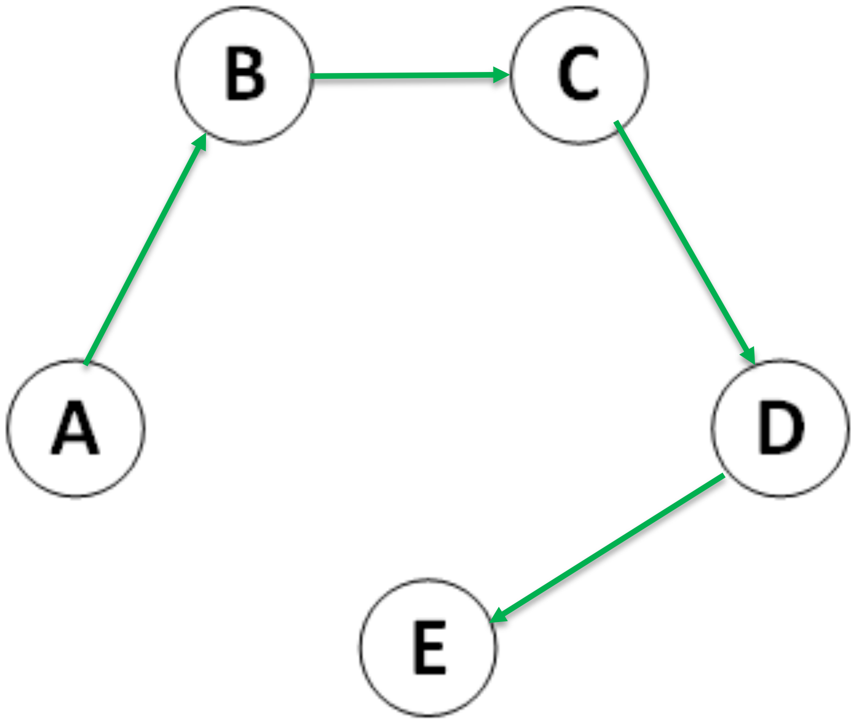
Representations

- The first stage of building any evolutionary algorithm is to decide on a genetic representation of a candidate solution to a problem
- Common types of representation:
  - Binary
  - Integer values
  - Real values
  - **Permutation**
  - Tree representation

Permutation representations

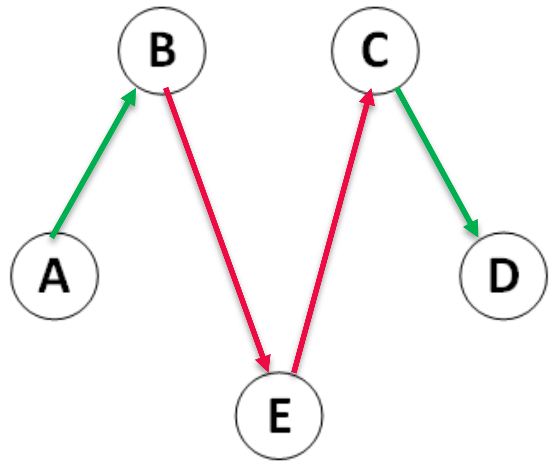
- Ordering/sequencing problems form a special type
- Task is (or can be solved by) arranging some objects in a certain order
- Example: Travelling Salesman Problem (TSP) : important thing is which elements occur next to each other

Traveling salesman problem

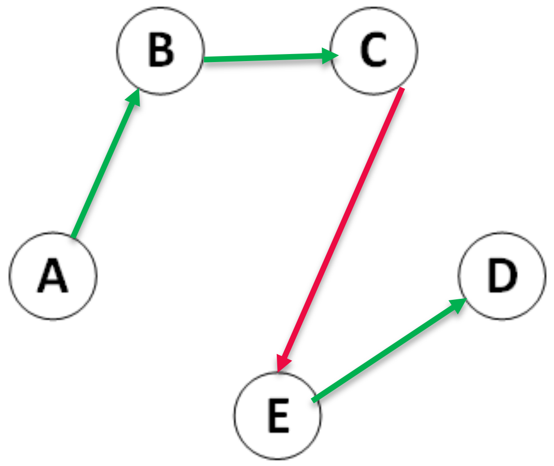


- 5 cities
- Need to find the shortest path

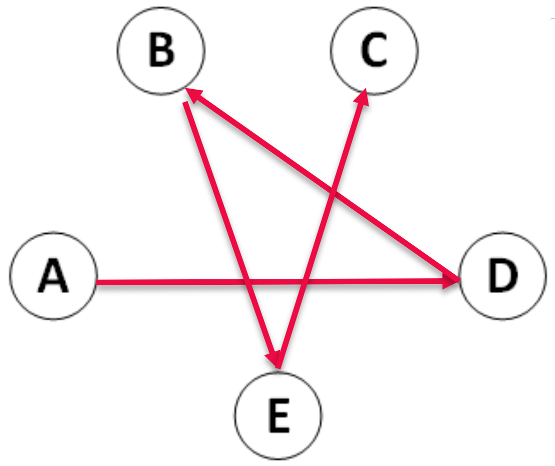
Population



[A, B, E, C, D]



[A, B, C, E, D]



[A, D, B, E, C]

Permutation Representations: Mutation

- Normal mutation operators lead to inadmissible solutions
  - e.g. bit-wise mutation: let gene *i* have value *j*
  - changing to some other value *k* would mean that *k* occurred twice and *j* no longer occurred

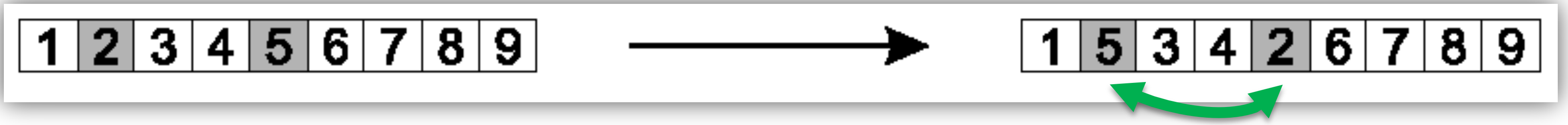
$$[A\ B\ C\ D\ E] \longrightarrow [A\ B\ C\ D\ \underline{\mathbf{A}}]$$

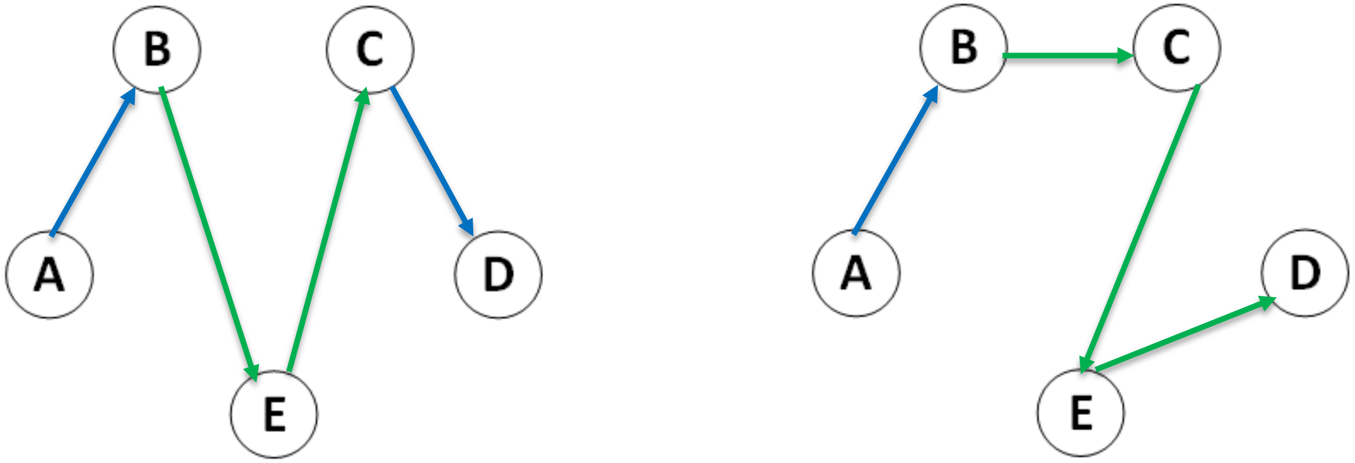
$$[A\ B\ C\ D\ E] \longrightarrow [A\ \underline{\mathbf{C}\ \mathbf{B}}\ D\ E]$$

- Therefore must change at least two values
- Mutation parameter now reflects the probability that some operator is applied once to the whole string, rather than individually in each position

Permutation Representations: Swap mutation

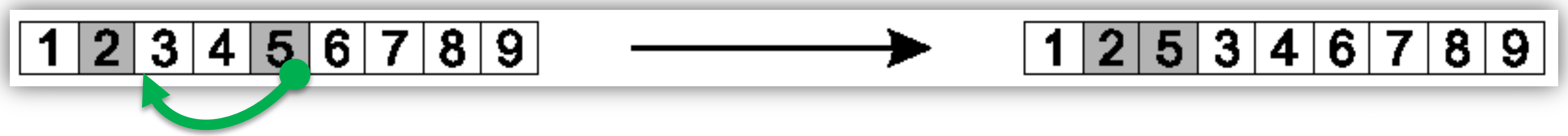
- Pick two alleles at random and swap their positions





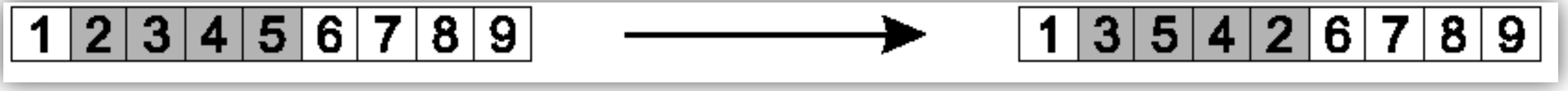
Permutation Representations: Insert Mutation

- Pick two allele values at random
- Move the second to follow the first, shifting the rest along to accommodate
- Note that this preserves most of the order and the adjacency information



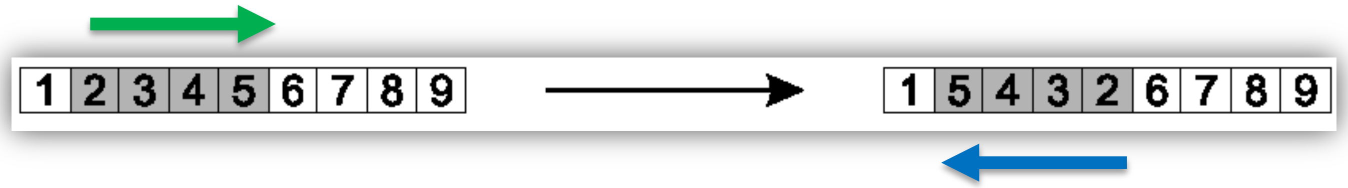
Permutation Representations: Scramble mutation

- Pick a subset of genes at random
- Randomly rearrange the alleles in those positions



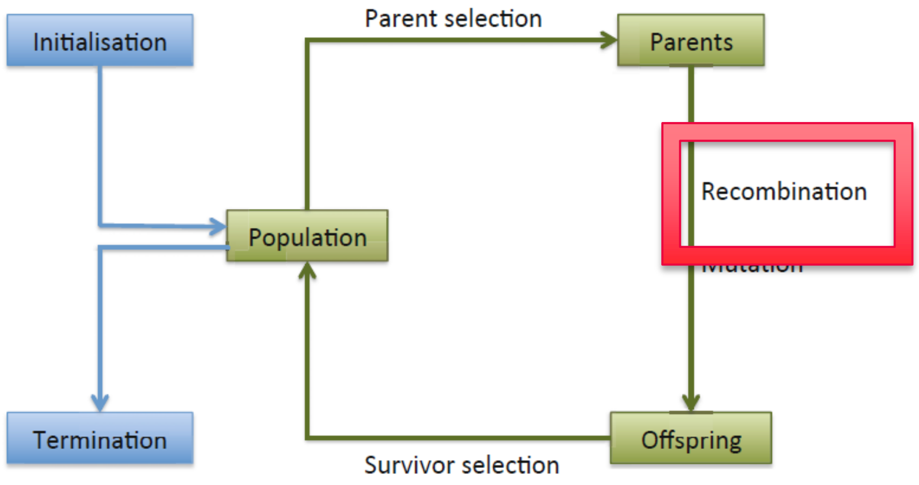
Permutation Representations: Inversion mutation

- Pick two alleles at random and then invert the substring between them.
- Preserves most adjacency information (only breaks two links) but disruptive of order information



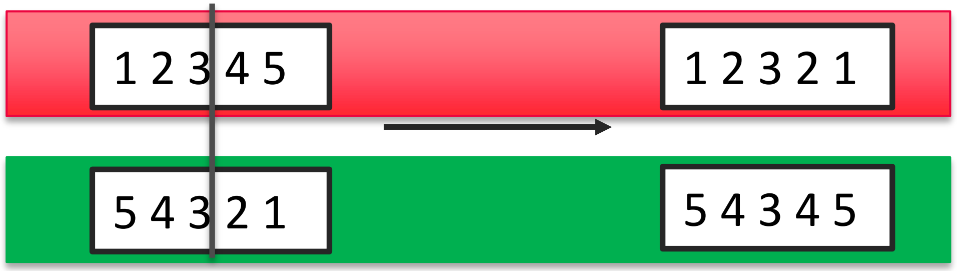
Components of EAs

- **Representation of individuals**
- Population of individuals
- Evaluation function (fitness function)
- Parent selection mechanism
- Variation operators (**recombination**, mutation)
- Survivor selection mechanism
- Terminate conditions



Permutation Representations: Crossover operators

- "Normal" crossover operators will often lead to inadmissible solutions



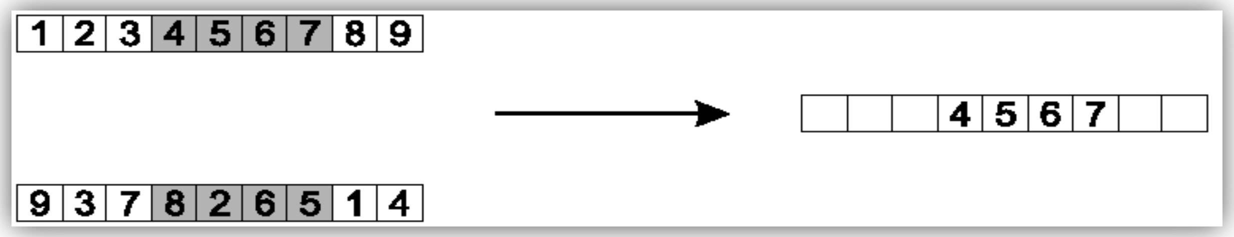
- Many specialised operators have been devised which focus on combining order or adjacency information from the two parents

Permutation Representations: Order 1 crossover (1/2)

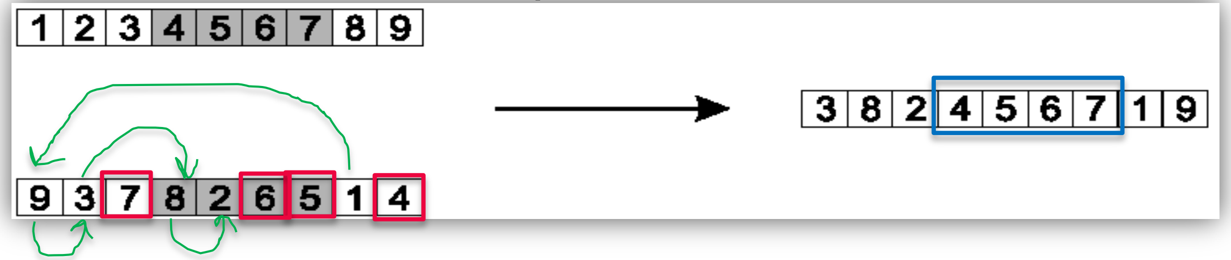
- Idea is to preserve relative order that elements occur
- Informal procedure:
  - 1. Choose an arbitrary part from the first parent
  - 2. Copy this part to the first child
  - 3. Copy the numbers that are not in the first part, to the first child:
    - starting right from cut point of the copied part,
    - using the **order** of the second parent
    - and wrapping around at the end
  - 4. Analogous for the second child, with parent roles reversed

Permutation Representations: Order 1 crossover (2/2)

- Copy randomly selected set from first parent



- Copy rest from second parent in order 1,9,3,8,2



Representations

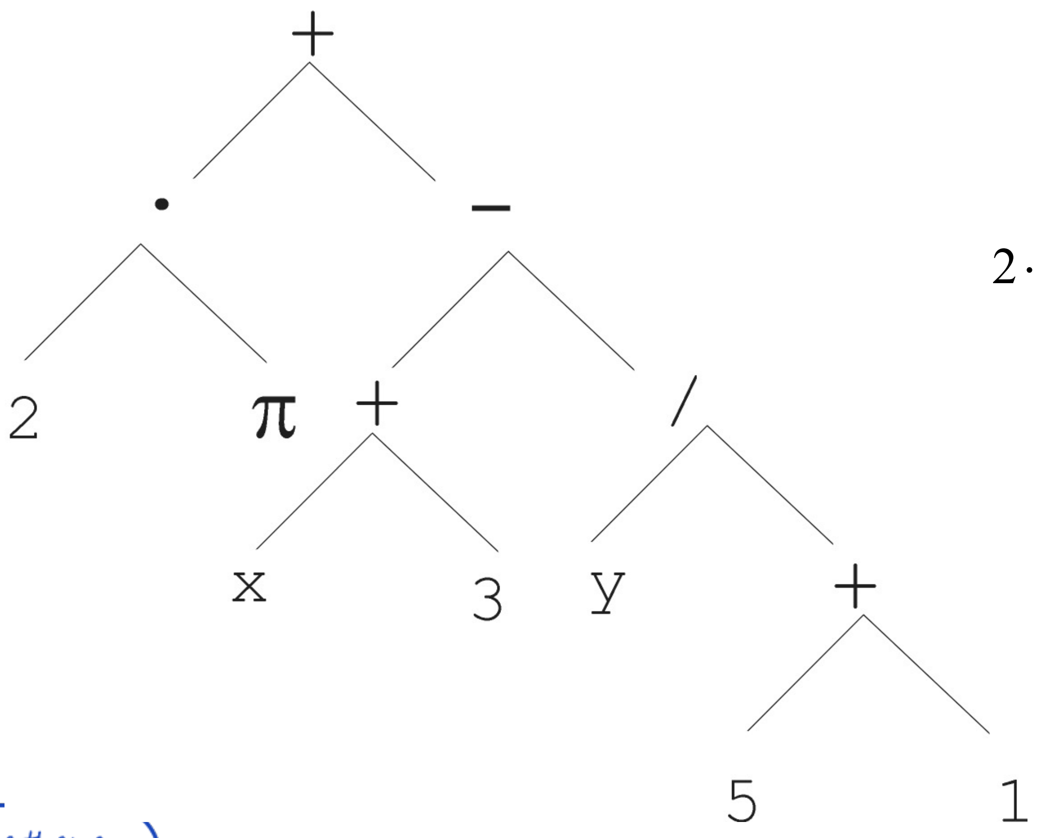
- The first stage of building any evolutionary algorithm is to decide on a genetic representation of a candidate solution to a problem
- Common types of representation:
  - Binary
  - Integer values
  - Real values
  - Permutation
  - **Tree representation**

Tree Representation (1/6)

- Trees are a universal form, e.g. consider
- Arithmetic formula: $2 \cdot \pi + \left((x+3) - \frac{y}{5+1} \right)$
- Logical formula: $(x \wedge \text{true}) \rightarrow ((x \vee y) \vee (z \leftrightarrow (x \wedge y)))$
- Program:

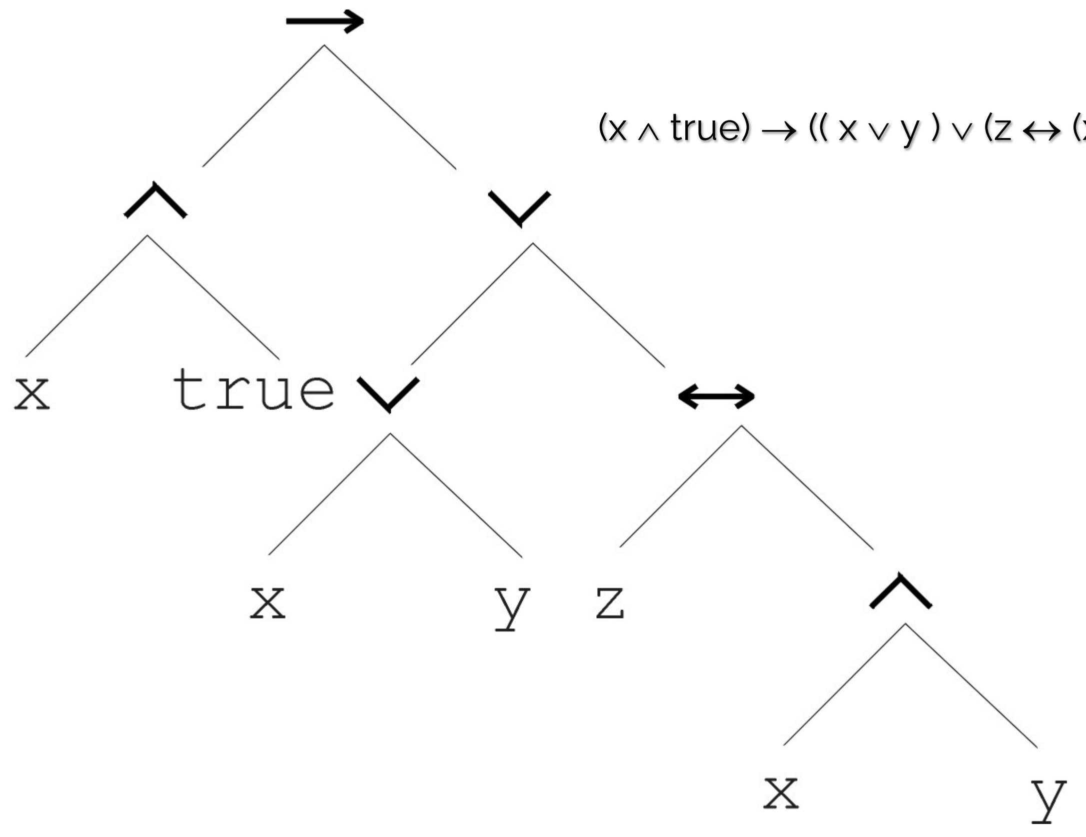
```
i =1;
while (i < 20)
{
        i = i +1
}
```

Tree Representation (2/6)



$$2\cdot\pi+\left((x+3)-\frac{y}{5+1}\right)$$

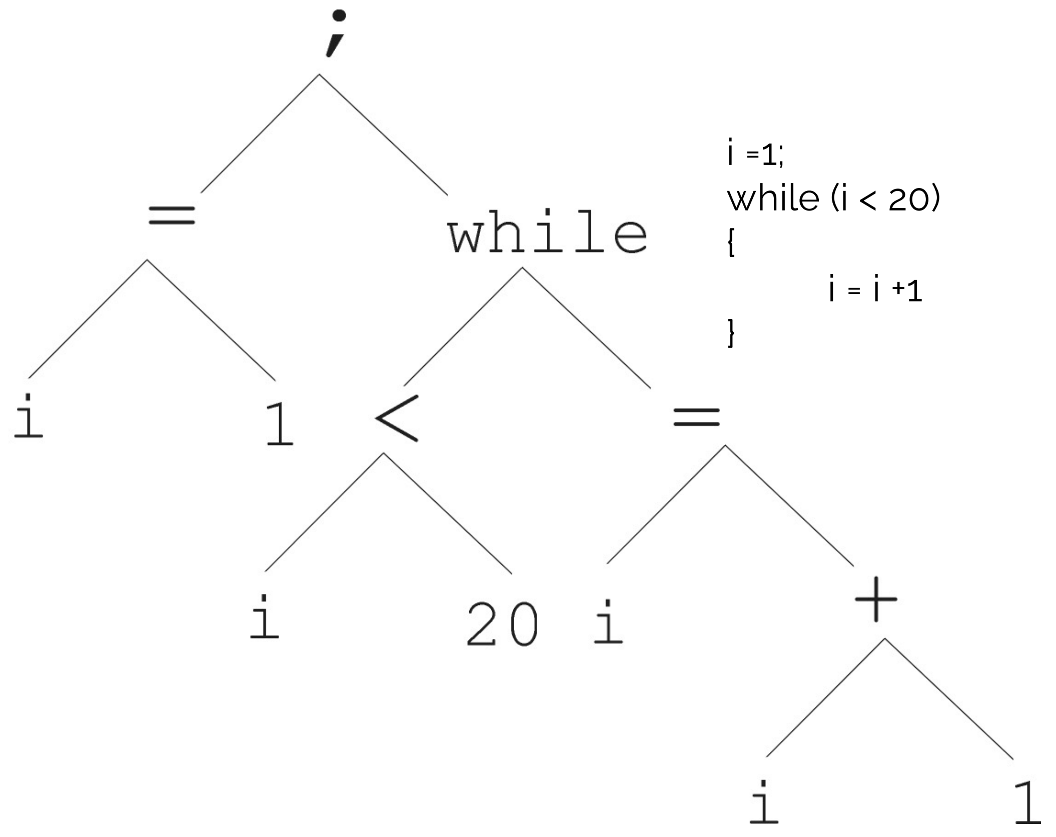
Tree Representation (3/6)



$(x \wedge \text{true}) \to ((x \vee y) \vee (z \leftrightarrow (x \wedge y)))$

Tree Representation (4/6)

Genetic programming



```
i =1;
while (i < 20)
{
        i = i +1
}
```

Tree Representation (5/6)

- In GA, ES, EP chromosomes are linear structures (bit strings, integer string, real-valued vectors, permutations)
- Tree shaped chromosomes are non-linear structures
- In GA, ES, EP the size of the chromosomes is fixed
- Trees in GP may vary in depth and width

Tree Representation (6/6)

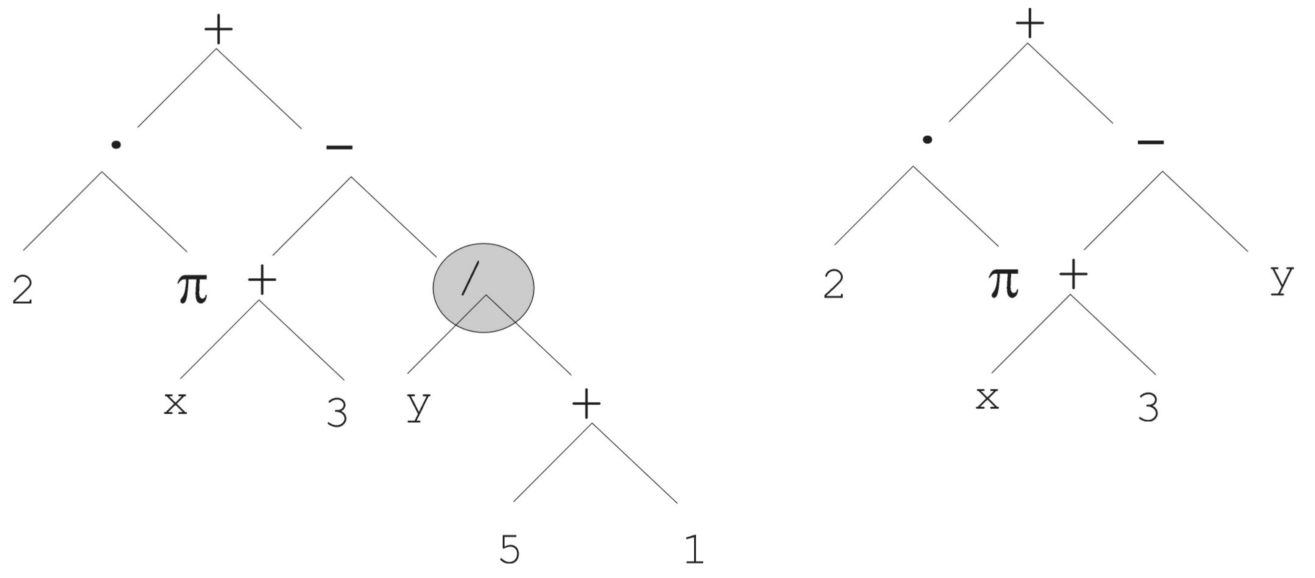
- Symbolic expressions can be defined by
  - Terminal set T
  - Function set F (with the arities of function symbols)

Function set	$\{+, -, \cdot, /\}$
Terminal set	$\mathbb{R} \cup \{x, y\}$

- Adopting the following general recursive definition:
  - Every t ∊ T is a correct expression
  - $f(e_1, \ldots, e_n)$ is a correct expression if f ∊ F, arity(f)=n and $e_1, \ldots, e_n$ are correct expressions
  - There are no other forms of correct expressions
- In general, expressions in GP are not typed (closure property: any f ∊ F can take any g ∊ F as argument)

Tree Representation: Mutation (1/2)

- Most common mutation: replace randomly chosen subtree by randomly generated tree



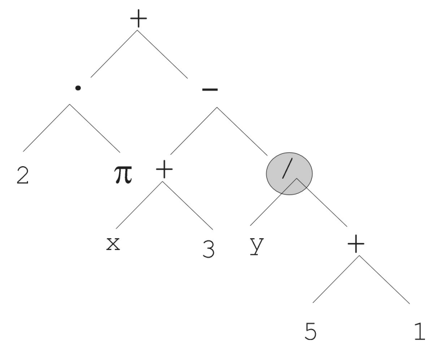
Tree Representation: Mutation (2/2)

- Mutation has two parameters:
  - Probability p_m to choose mutation
  - Probability to chose an internal point as the root of the subtree to be replaced
- Remarkably p_m is advised to be 0 (Koza'92) or very small, like 0.05 (Banzhaf et al. '98)
- The size of the child can exceed the size of the parent

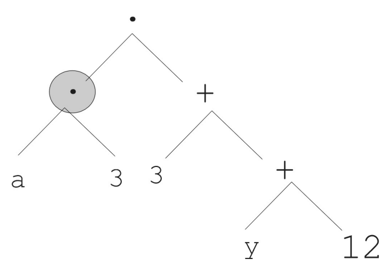
Tree Representation: Recombination (1/2)

- Most common recombination: exchange two randomly chosen subtrees among the parents
- Recombination has two parameters:
  - Probability p_c to choose recombination
  - Probability to chose an internal point within each parent as crossover point
- The size of offspring can exceed that of the parents

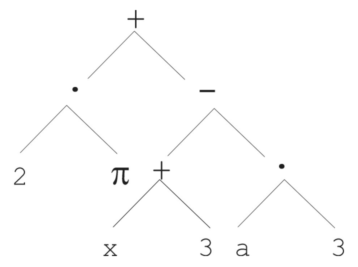
Tree Representation: Recombination (2/2)



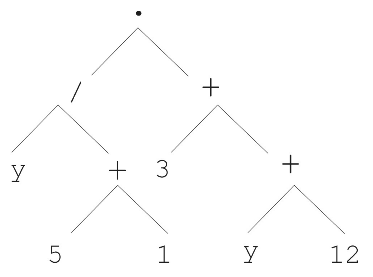
Parent 1



Parent 2



Child 1



Child 2

Lab2

- Function optimization
- Need to implement Factory, Mutation and Crossover
- Fitness function estimates n dimensional double array with values from -5.0 to 5.0
- Max possible fitness is 10.0.

Thank you for your attention!

www.ifmo.ru

IT'sMOre than a UNIVERSITY