Математички институт САНУ, 70 година: ДАН ОДЕЉЕЊА ЗА РАЧУНАРСТВО И ПРИМЕЊЕНУ МАТЕМАТИКУ

Variable Neighborhood Programming- a new automatic programming method in artificial intelligence

Souhir ELLEUCH

University of Sfax MODILS Laboratory Sfax, Tunisia

Bassem JARBOUI

University of Sfax MODILS Laboratory Sfax, Tunisia

Nenad MLADENOVIC

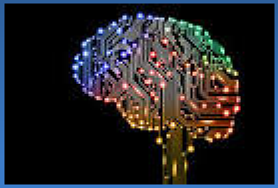
Mathematical Institute, SANU, Belgrade, Serbia

May 27, 2016

Overview

- Introduction
  - Artificial intelligence problem
  - Genetic programming
  - Variable neighborhood search
- Variable Neighborhood Programming algorithm (VNP)
  - Neighborhood structures
- VNP application
  - Forecasting
  - Classification
- Conclusions

Artificial Intelligence



- Artificial intelligence (AI) is the intelligence exhibited by machines or software.
- It is also the name of the academic field of study which studies how to create computers and computer software that are capable of intelligent behavior.
- Today it has become an essential part of the technology industry, providing the heavy lifting for many of the most challenging problems in computer science.
- One of the central challenges of computer science is to get a computer to do what needs to be done, without telling it how to do it.

Automatic Programming

- Automatic programming is growing in artificial intelligence field.
- Genetic programming (Koza, 1992).
  - Based on Genetic Algorithm operators;
  - Each individual Presented as a computer program;
  - Increasingly used in artificial Intelligence problems.
  - Genetic programming achieves goal of *automatic programming* (sometimes called *program synthesis* or *program induction*) by genetically breeding a population of computer programs using the principles of Darwinian natural selection and biologically inspired operations.

Some Genetic Programming applications

- Symbolic regression (Koza,1992; Cai et al., 2006; Quang et al., 2011);
- Data mining (Xing, 2014; Jabeen and Baig, 2009; Pereira et al., 2014);
- Time series forecasting (Eklund, S.E., 2003; Rivero et al., 2005; Yi-Shian Lee et Lee-Ing Tong,2011; Bouaziz et al., 2013);
- Classifications (Jabeen and Baig, 2013; Escalante et al., 2014 ; Shao et al., 2014).

Some Genetic Programming applications

- Symbolic regression searches the space of mathematical expressions to find the model that best fits a given dataset.
- Data mining is the computational process of discovering patterns in large data sets ("big data")
- Time series forecasting is the use of a model to predict future values based on previously observed values.
- Classifications consists in predicting the value of a user-specified goal attribute (the class) based on the values of other attributes, called predicting attributes

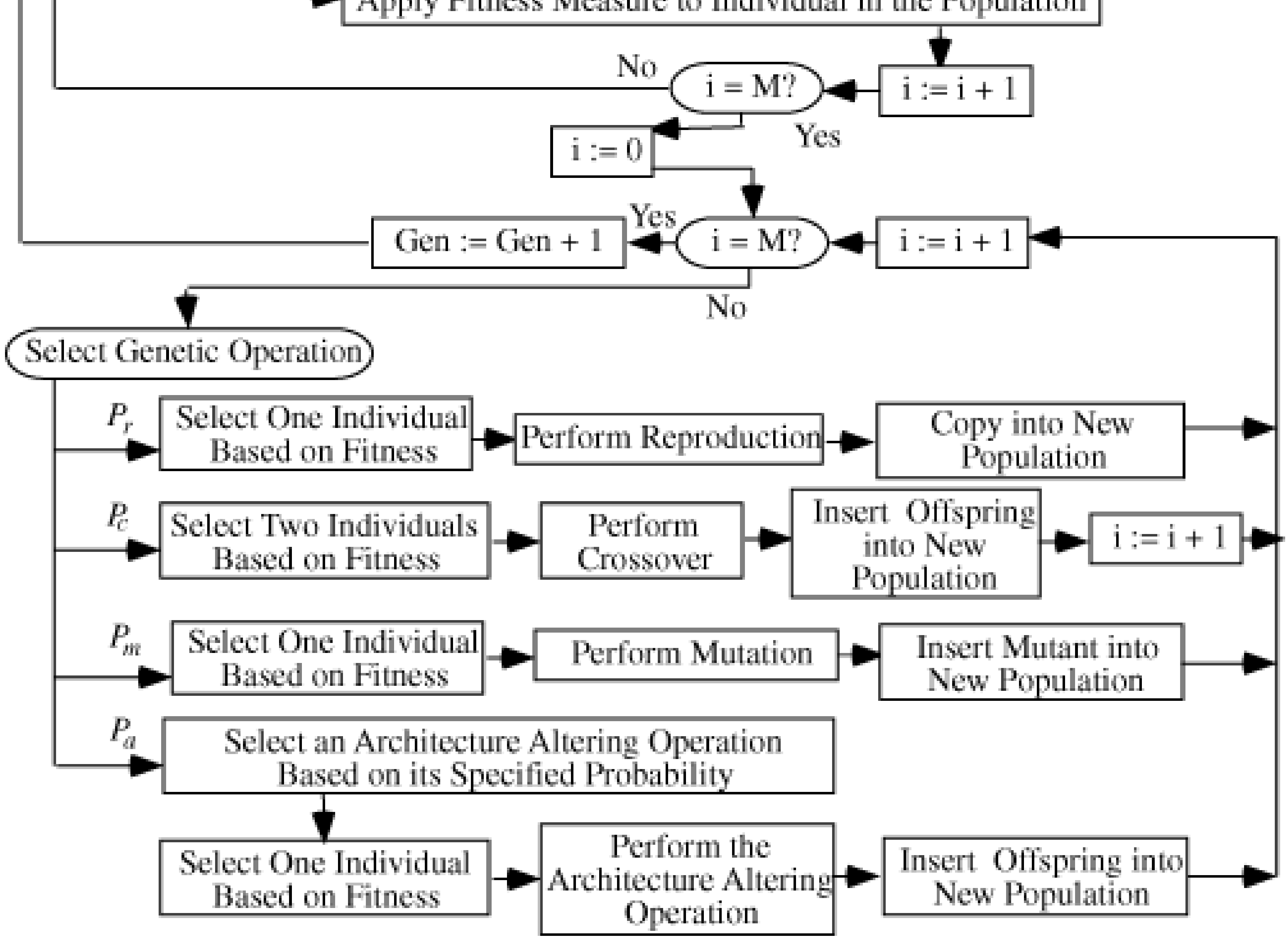
Genetic Programming

- Preparatory Steps of Genetic Programming

. (1) the set of terminals (e.g., the independent variables of the problem, zero-argument functions, and random constants) for each branch of the to-be-evolved program,

- (2) the set of primitive functions for each branch of the to-be-evolved program,
- (3) the fitness measure (for explicitly or implicitly measuring the fitness of individuals in the population),
- (4) certain parameters for controlling the run, and
- (5) the termination criterion and method for designating the result of the run.

- Executional steps of GP


Apply Fitness Measure to Individual in the Population
No
i = M?
i := i + 1
Yes
i := 0
Yes
Gen := Gen + 1
i = M?
i := i + 1
No
Select Genetic Operation
P_r
Select One Individual Based on Fitness
Perform Reproduction
Copy into New Population
P_c
Select Two Individuals Based on Fitness
Perform Crossover
Insert Offspring into New Population
i := i + 1
P_m
Select One Individual Based on Fitness
Perform Mutation
Insert Mutant into New Population
P_a
Select an Architecture Altering Operation Based on its Specified Probability
Select One Individual Based on Fitness
Perform the Architecture Altering Operation
Insert Offspring into New Population

(1) Randomly create an initial population (generation 0) of individual computer programs composed of the available functions and terminals.

(2) Iteratively perform the following sub-steps (called a *generation*) on the population until the termination criterion is satisfied:

   (a) Execute each program in the population and ascertain its fitness (explicitly or implicitly) using the problem's fitness measure.

   (b) Select one or two individual program(s) from the population with a probability based on fitness (with reselection allowed) to participate in the genetic operations in (c).

GP steps

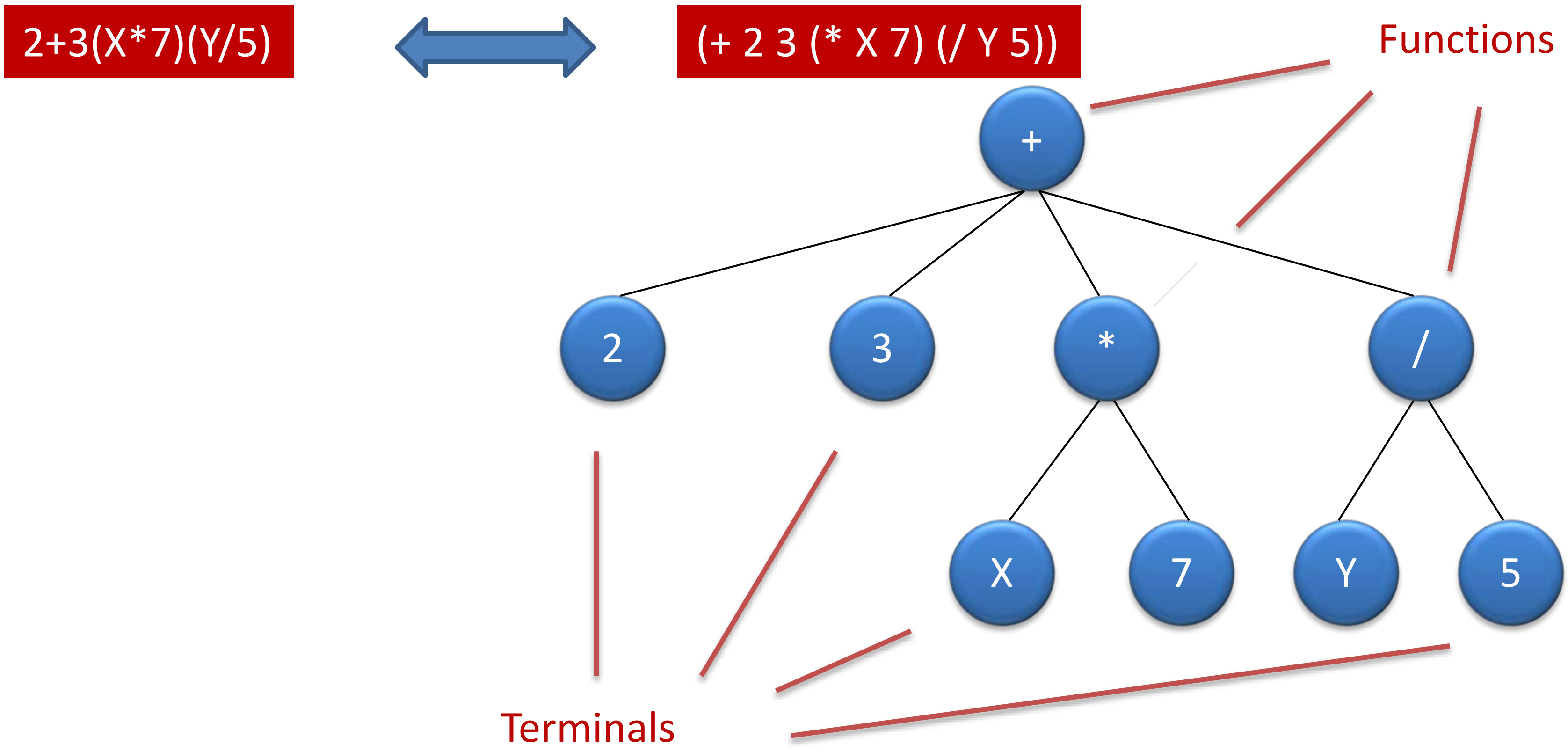
- (c) Create new individual program(s) for the population by applying the following genetic operations with specified probabilities:
  - (i) *Reproduction:* Copy the selected individual program to the new population.
  - (ii) *Crossover:* Create new offspring program(s) for the new population by recombining randomly chosen parts from two selected programs.
  - (iii) *Mutation:* Create one new offspring program for the new population by randomly mutating a randomly chosen part of one selected program.
  - (iv) *Architecture-altering operations:* Choose an architecture-altering operation from the available repertoire of such operations and create one new offspring program for the new population by applying the chosen architecture-altering operation to one selected program.

- (3) After the termination criterion is satisfied, the single best program in the population produced during the run (the best-so-far individual) is harvested and designated as the result of the run. If the run is successful, the result may be a solution (or approximate solution) to the problem.

Variable Neighborhood Programming algorithm

- Inspiring the power of Genetic programming solution representation and Variable Neighborhood Search movements.
- Based on systematic change of neighborhood within a local search.
- Start with a single solution presented by a program
- Apply neighborhood structure movements to reach the global optimum

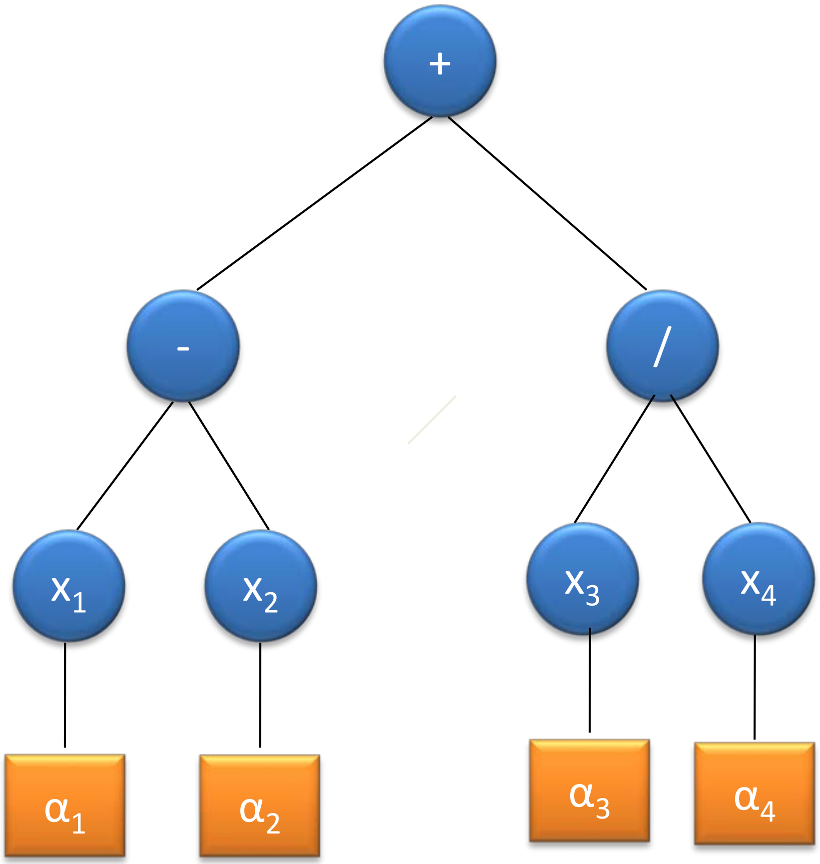
GP solution representation



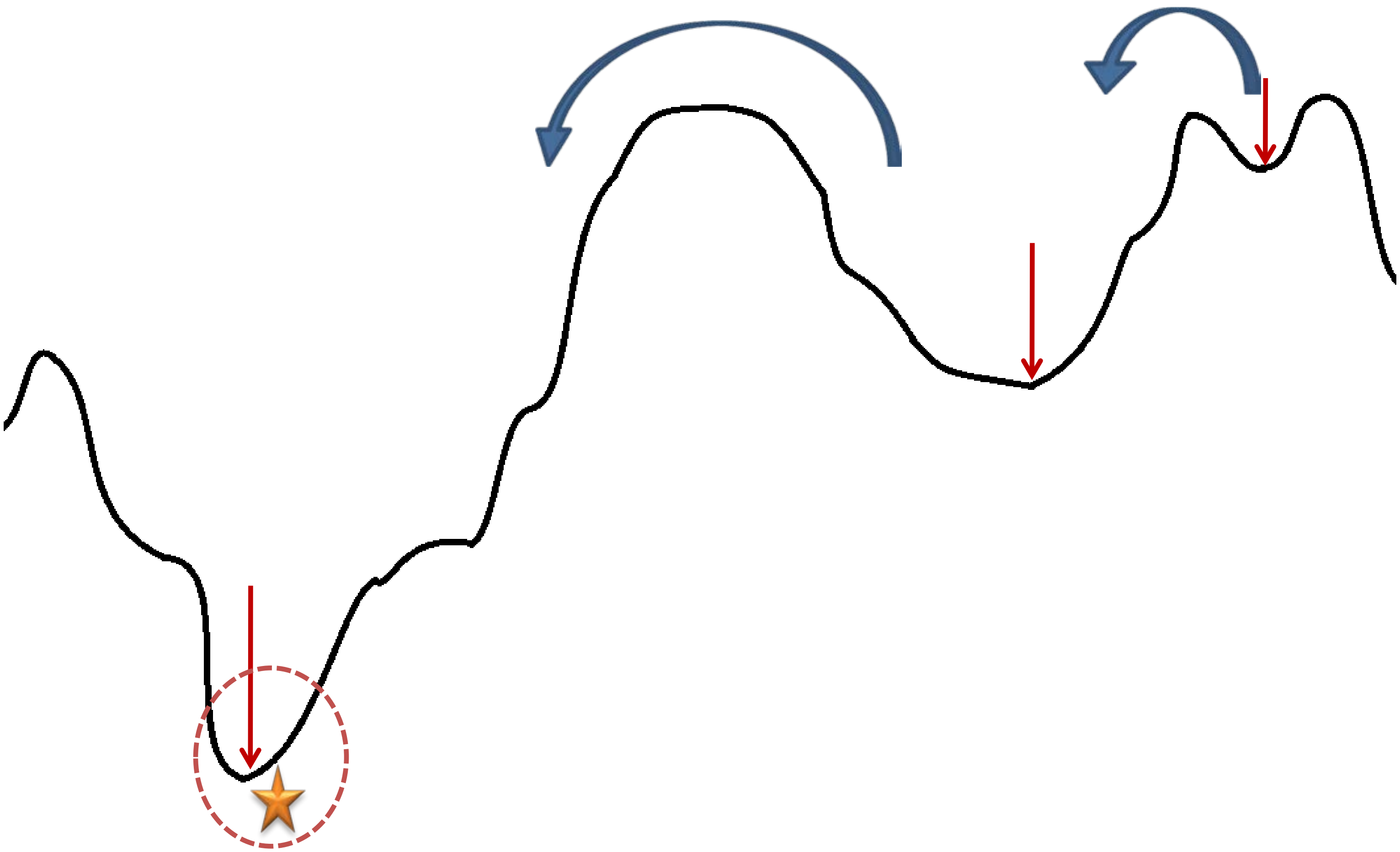
In the majority of previous studies, programs are usually presented as trees rather than as lines of code.

VNP solution representation

We suggest an extended solution illustration adding coefficients. Each terminal node is attached by its own parameter value. These parameters serve to give a weight for each terminal node



VNS algorithm movements



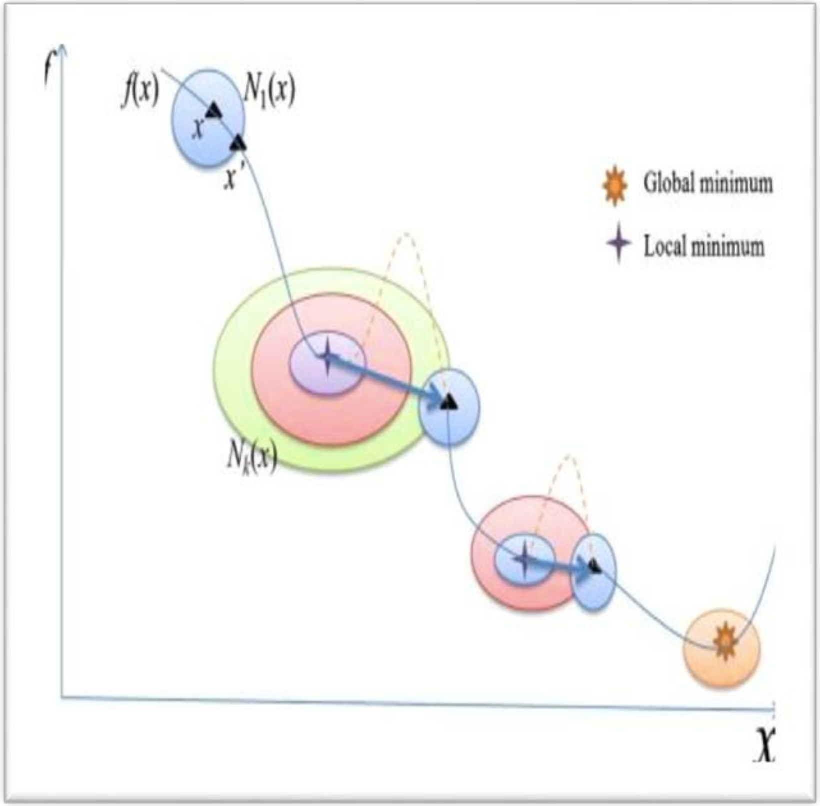
VNS - Overview

- Proposed by Mladenovic and Hansen in 1997
- **Main idea**: Systematically change the neighborhood structures
- **Based on three facts:**
  - A local minimum w.r.t. one neighborhood structure is not necessary so for another
  - A global minimum is local minimum w.r.t. all possible neighborhood structures
  - For many problems local minima w.r.t. one or several neighborhoods are close to each other

Outline of VNS algorithm

Procedure VNS

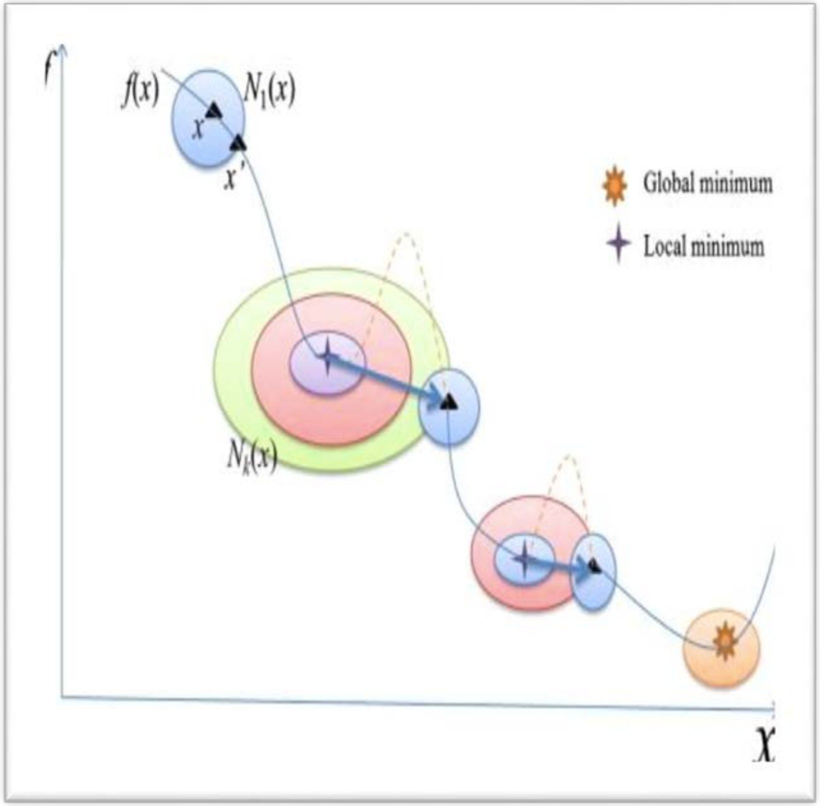
Define neighborhood structures N_k (k=1,...,k_{max})
Generate initial solution $x \in X$
 while *stopping condition is not met* **do**
  $k \leftarrow 1$
  while $k \leq k_{max}$ **do**
   $x' \leftarrow$ Shake(x), $x' \in N_k(x)$;
   $x'' \leftarrow$ Local Search(x');
   if (x'' *is better than* x)
    $x \leftarrow x''$; $k \leftarrow 1$;
   else
    $k \leftarrow k+1$;
  end-while
 end-while



VNS outline of algorithm

Procedure VNS

Define neighborhood structures N_k (k=1,...,k_{max})
Generate initial solution $x \in X$
while *stopping condition is not met* **do**
$k \leftarrow 1$
while $k \leq k_{max}$ **do**
$x' \leftarrow$ Shake(x), $x' \in N_k(x)$;
$x'' \leftarrow$ Local Search(x');
if (x'' *is better than* x)
$x \leftarrow x''$; $k \leftarrow 1$;
else
$k \leftarrow k+1$;
end-while
end-while



Variants of VNS algorithms

Variable Neighborhood Search (VNS) Variants

- Reduced VNS (RVNS)
- Skewed VNS (SVNS)
- General VNS (GVNS)
- VN Decomposition Search (VNDS)
- Two-level GVNS
- Nested VNS
- Parallel VNS (PVNS)
- Primal Dual VNS (P-D VNS)
- Reactive VNS
- Formulation Space Search (FSS)
- VN Branching . . .

Variable Neighborhood Descent (VND) Variants

- In VND, shaking phase is removed from VNS
- VND can be used as a part of VNS in the local search phase

- Sequential VND
- Cyclic VND
- Pipe VND
- Union VND
- Nested VND
- Mixed-nested VND
- Etc.

Variants of VNS algorithms

- 3 level VNS
- Backward VNS
- 2-phase VNS
- Gaussian VNS for continuous opt.
- Best improvement VNS
- VN Pump
- VNS Hybrids
- etc

Varaiable Neighborhood Descent (VND)

Procedure VNS

Define neighborhood structures N_k ($k=1,\ldots,k_{max}$)

Generate initial solution $s \in S$

while stopping condition is not met **do**

$k \leftarrow 1$

~~**while** $k \leq k_{max}$ **do**~~

$s' \leftarrow$ Shake(s), $s' \in N_k(s)$;

$s'' \leftarrow$ LocalSearch(s'), $s'' \in S$;

if (s'' is better than s)

$s \leftarrow s''$; $k \leftarrow 1$;

else

$k \leftarrow k+1$;

endif

end-while

end-while

End-Procedure

Variable Neighborhood Descent (VND)

In VND, shaking phase is removed from VNS so that the algorithm explores local optima by using neighborhood structures only. VND can be used as a part of VNS in the local search phase

Variants of VND

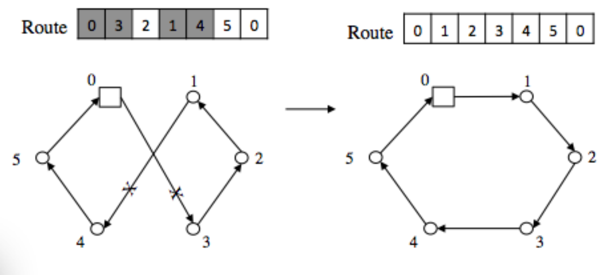
- Basic VND (BVND):
-

```
Procedure BVND
  Define neighborhood structures N_k (k=1,...,k_max)
  Generate initial solution s ∈ S
  k=1;
      while k ≤ k_max do
         s'← LocalSearch(s), s'∈ N_k;
         if (s' is better than s)
              s ← s'; k ← 1;
         else
            k ← k+1;
        end-if
      end-while
End-Procedure
```

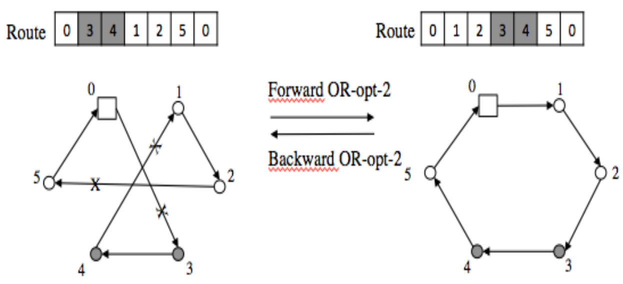
If there is an improvement w.r.t. some neighborhood N_k, exploration is continued in the first neighborhood

TSP neighborhoods

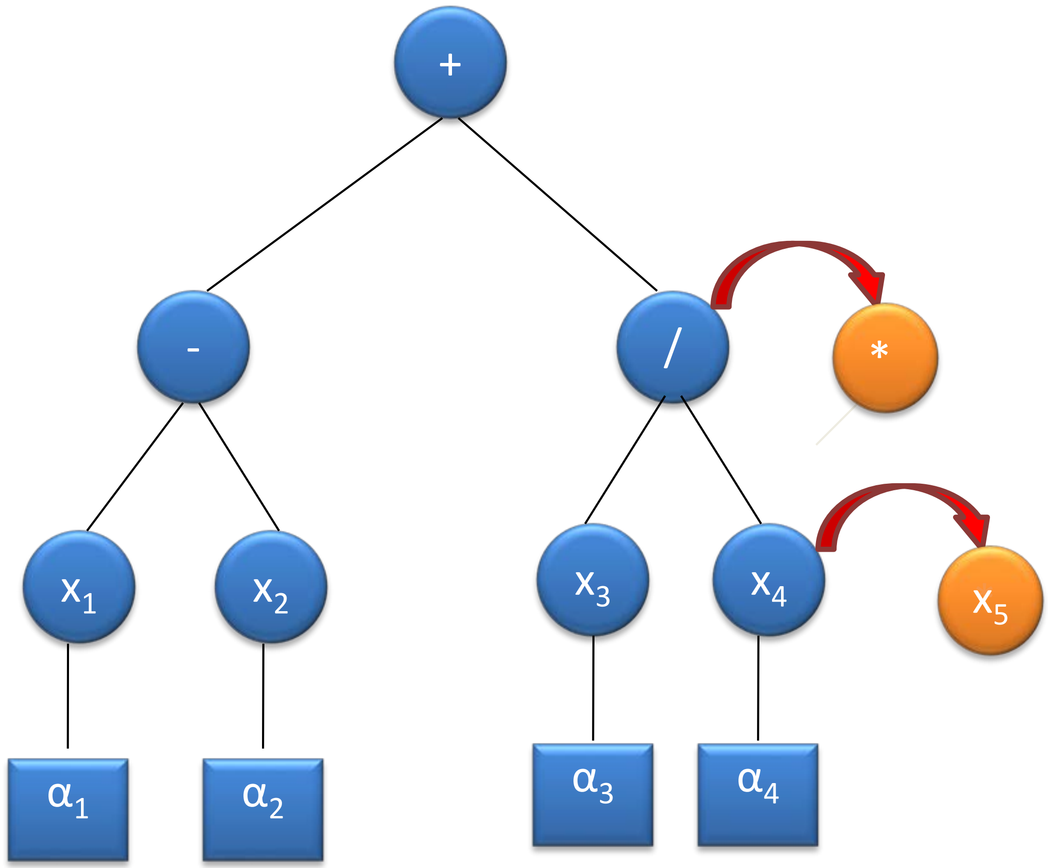
- 2-opt



- OR-opt_1
- OR-opt_2

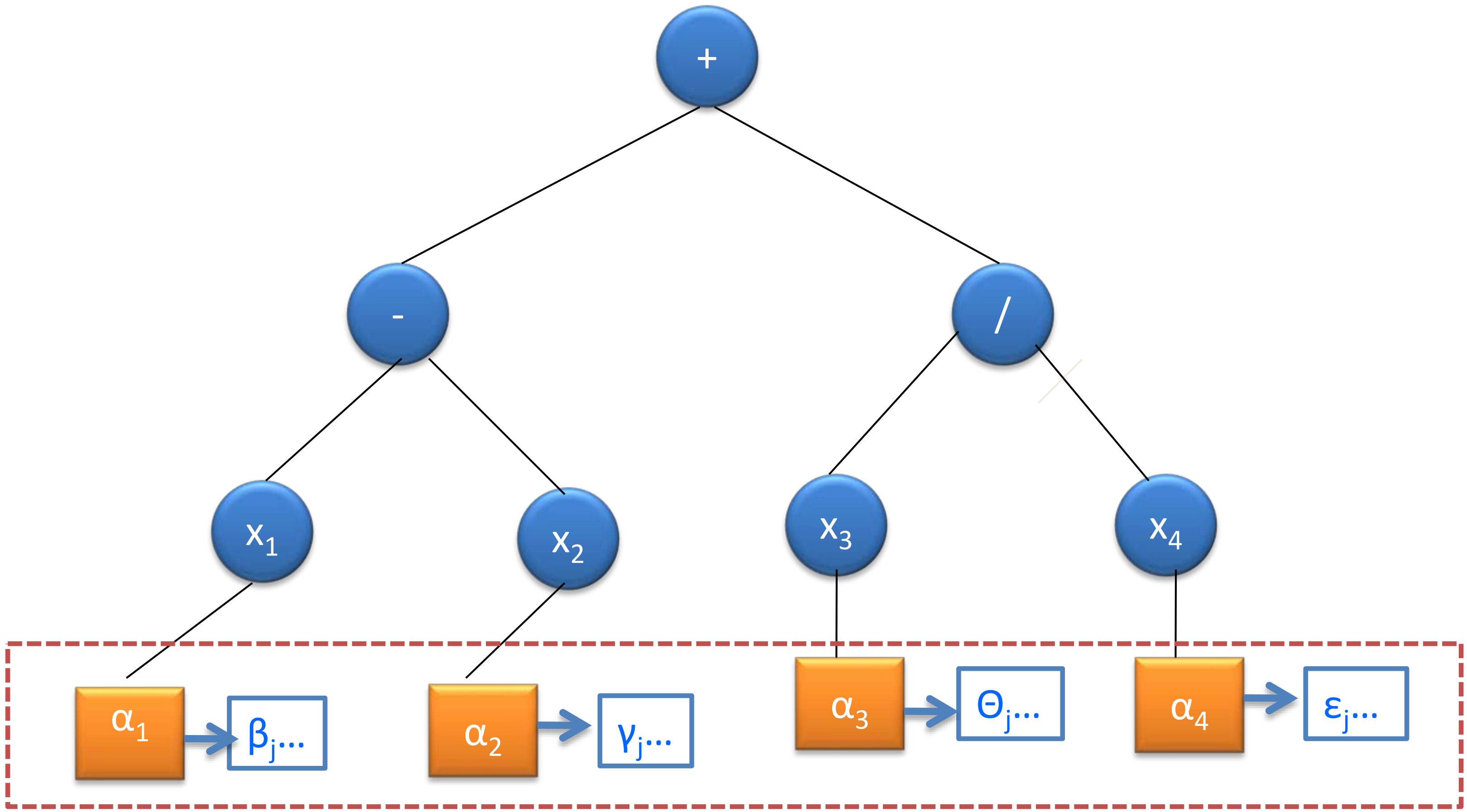


VNP neighborhood structures(1/7)



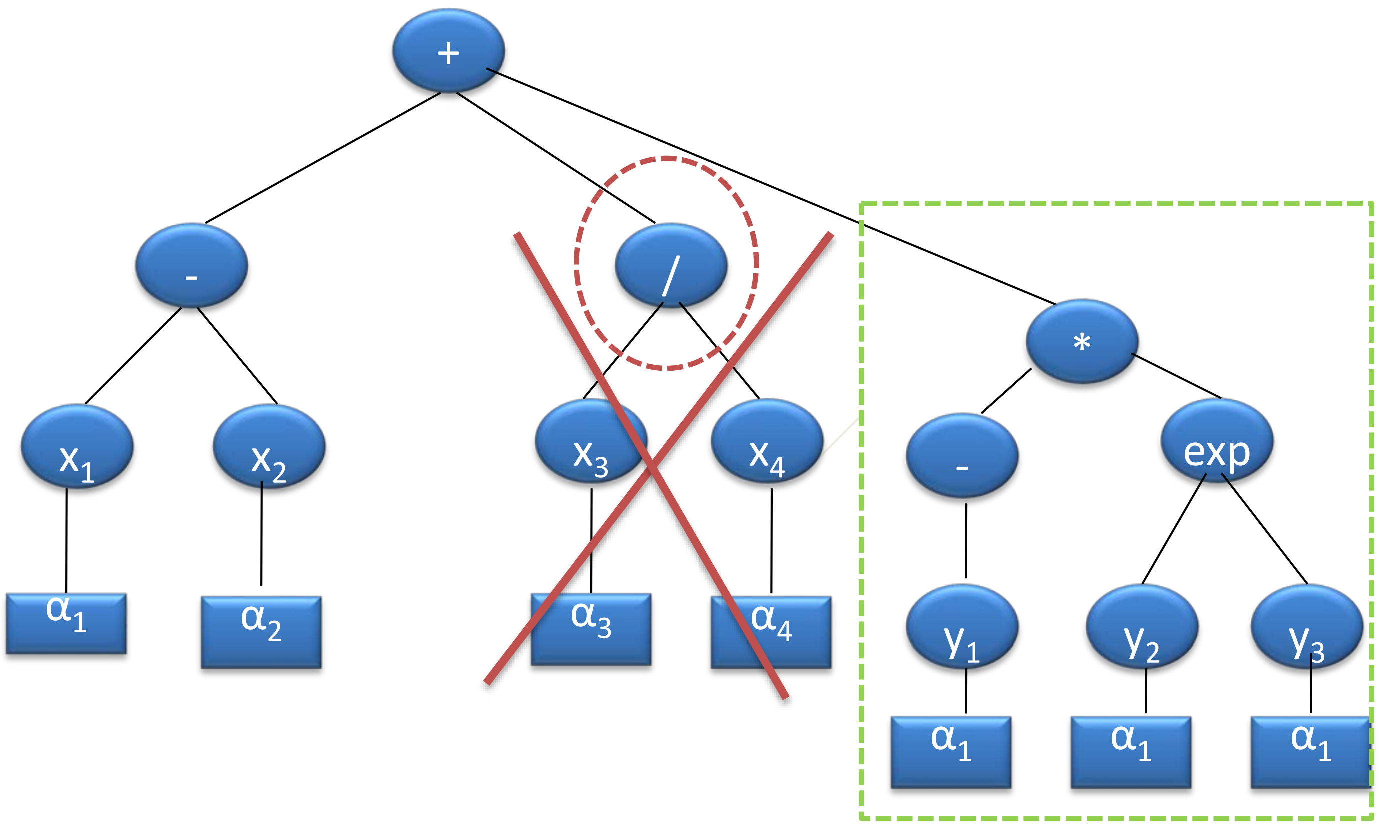
Changing node value operator

VNP neighborhood structures(2/7)



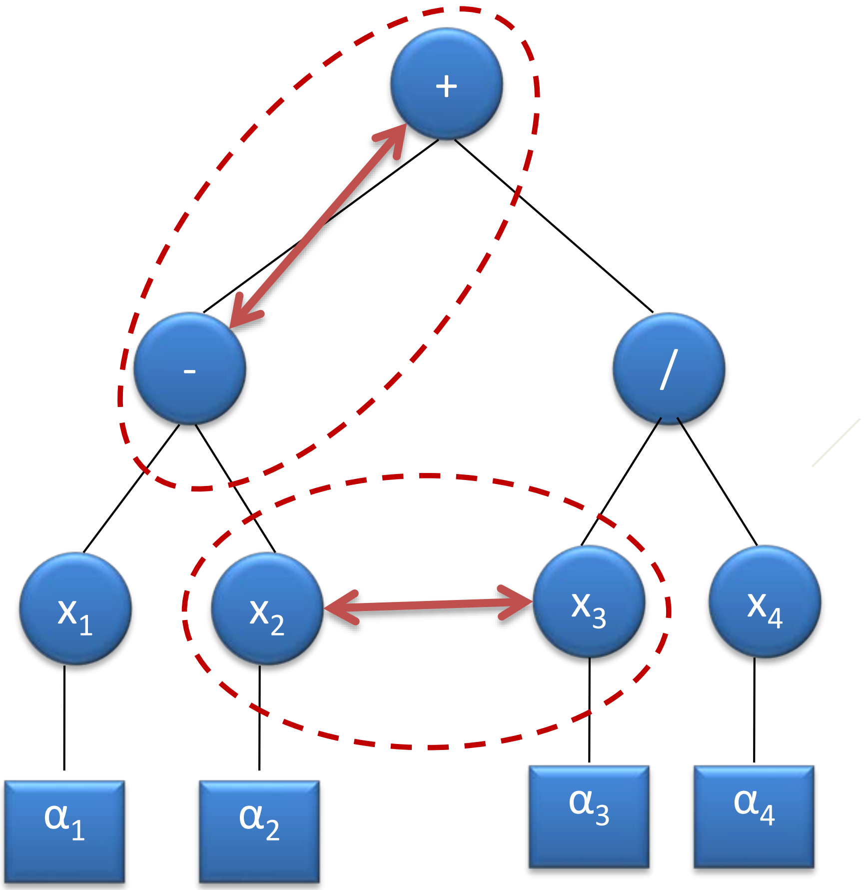
Changing parameter value operator

VNP neighborhood structures(3/7)



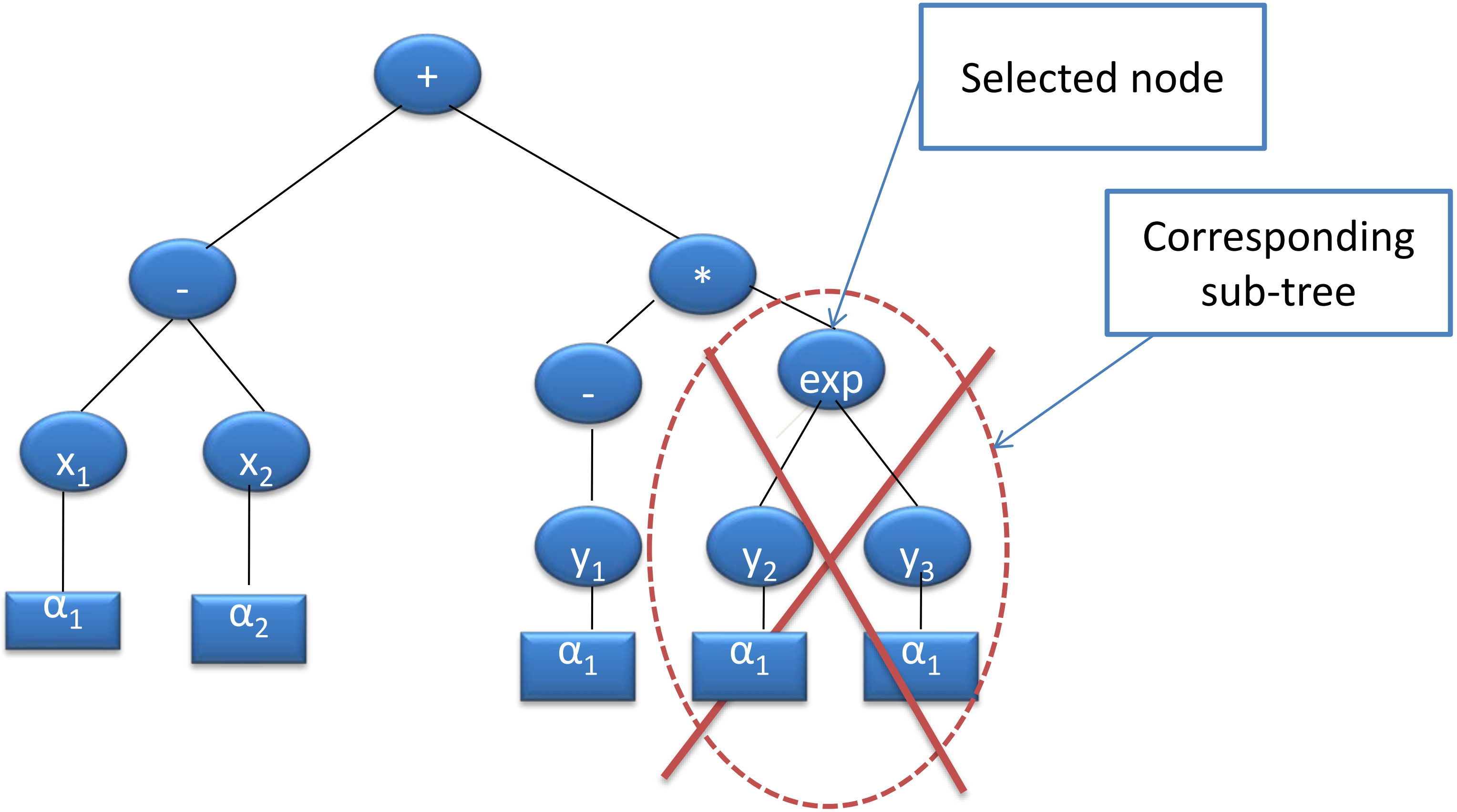
Exchange operator

VNP neighborhood structures(4/7)



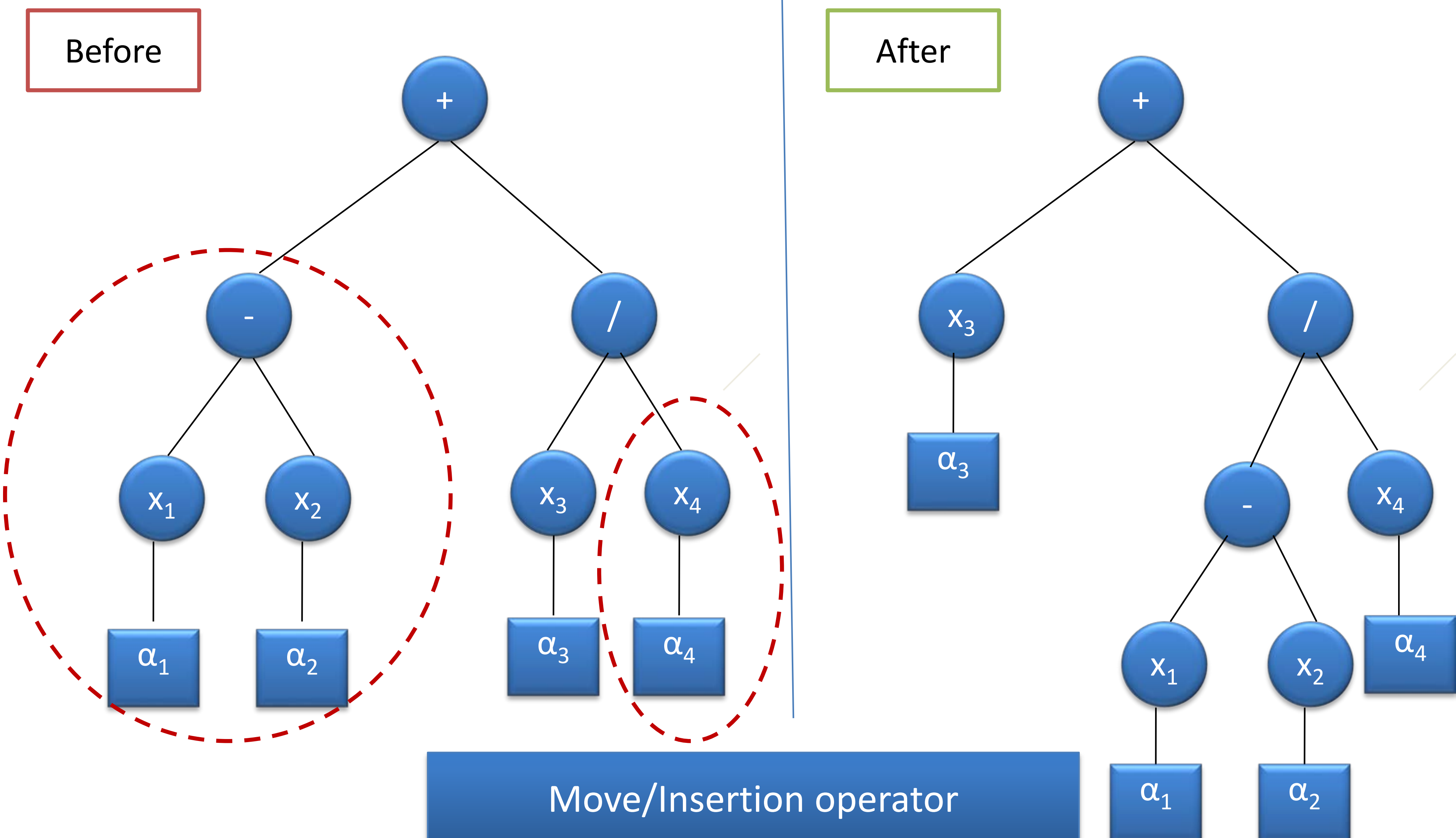
Inversion operator

VNP neighborhood structures(5/7)

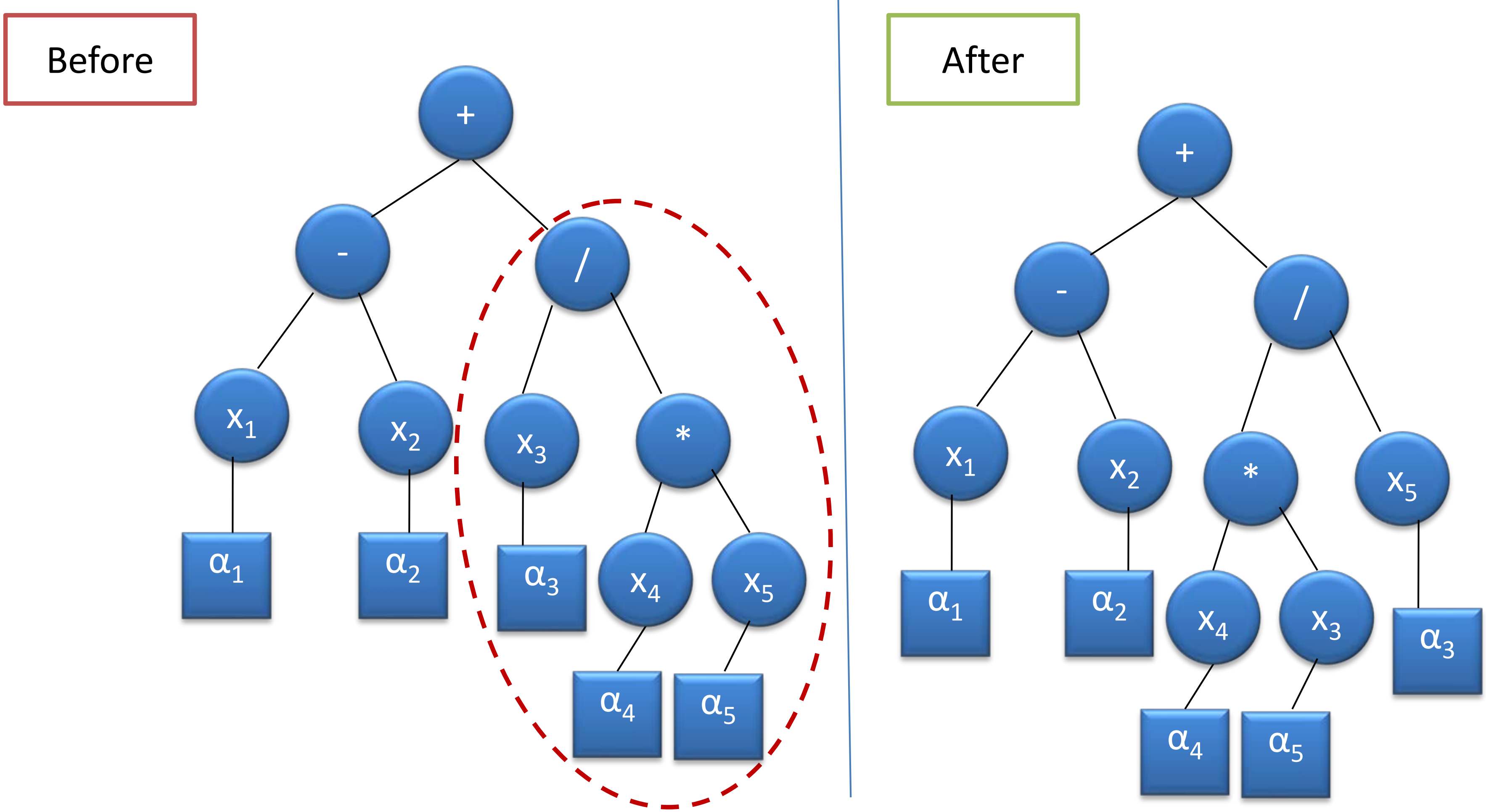


Remove operator

VNP neighborhood structures(6/7)

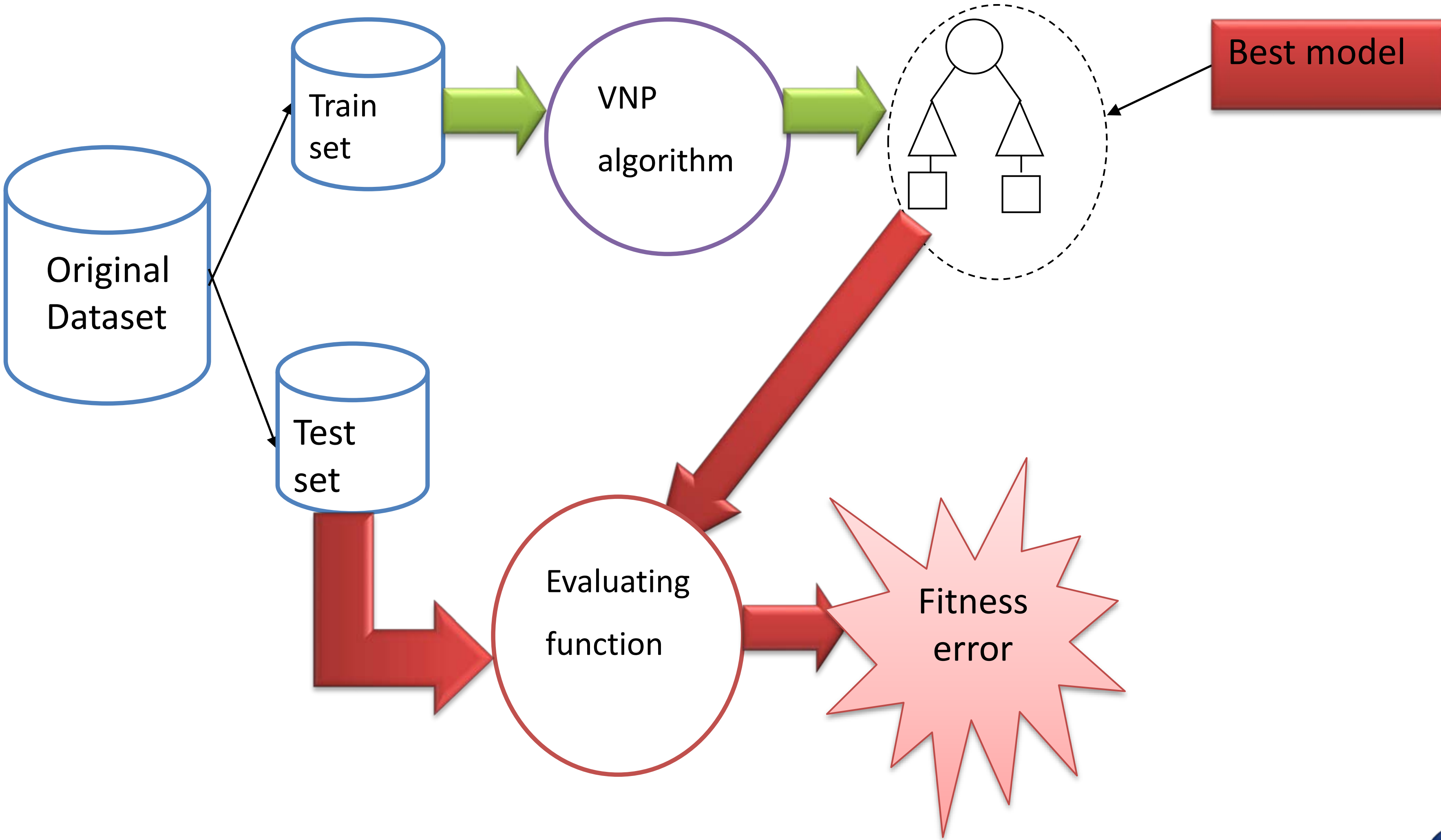


VNP neighborhood structures(7/7)



Shuffle operator

Learning Process



VNPD algorithm

Algorithm 1: VNPD (T, l_{max})

Input: the set of neighborhood structures: $\mathcal{N}_l, l = 1,..,l_{max}$ and an initial solution T

$l \leftarrow 1;$

while $l < l_{max}$

Find the best neighbor T' of $\mathcal{N}_l(T)$

Move or not: $\begin{cases} if\ fitness(T')\ \text{is better than}\ fitness(T)\ then \\ move\ T \leftarrow T';\ l \leftarrow 1; \\ else \\ l \leftarrow l + 1; \end{cases}$

End while

Return T

VNP algorithm

Algorithm 2: GeneralVNP (k_{max}, l_{max})

Initialization:

(1) Fix the set of neighborhood structures for the tree structure optimization and the parameter vector optimization, applied to the local search phase: $N_k, k = 1..k_{max}$ and the set of neighborhood structures for the shaking phase:$N_k, k = 1,..,k_{max}$

(2) Select the set of functions and terminals adequate for the studied problem.

(3) Generate randomly an initial tree T aspresented in Figure 1b).

(4) Choose the stopping condition.

Repeat

$k \leftarrow 1;$

while $k < k_{max}$

(a) $T' \leftarrow$Shake(T)// Find the first neighbor T' in $N_k(T)$

(b) $T'' \leftarrow$VNPD(T', l_{max}) //Local Search

(c) Move or not: $\left\{ \begin{array}{c} \textit{if } fitness(T'') \textit{ is better than } fitness(T) \textit{ then} \\ move\ T \leftarrow T'';\ k \leftarrow 1; \\ else \\ k \leftarrow k+1; \end{array} \right.$

End while

until termination condition is met

Return$T$;

Time forecasting problem

- Time series forecasting is the use of a model to predict future values based on previously observed values.
- The Mackey-Glass series is based on the Mackey-Glass differential equation (Mackey, 2002).
- The gas furnace data of Box and Jenkins was collected from a combustion process of a methane–air mixture (Box and Jenkins, 1976).
- The fitness function is the Root Mean Square Error.

Time forecasting problem

Method	Training error RMSE	Testing error RMSE
PSO BBFN	____	0.027
HMDDE–BBFNN	0.0094	0.0170
Classical RBF	0.0096	0.0114
CPSO	0.0199	0.0322
HCMSPSO	0.0095	0.0208
FBBFNT	0.0061	0.0068
VNP	**0.0021**	**0.0042**

Mackey-Glass dataset results

Time forecasting problem

Methods	Prediction error RMSE
ODE	0.5132
HHMDDE	0.3745
FBBFNT	0.0047
VNP	**0.0038**

Box and Jenkins dataset results

Classification problem

- Classification consists on predicting the appropriate class of an input vector based on a set of attributes.
- We choose five datasets of radically different nature which are the Iris, Wine, Statlog, Glass identification and Yeast datasets
- The performance measure is the Accuracy

Classification problem

Datasets	Classes	Attributes	Type	Instances
Iris	3	4	Real	150
Statlog	4	18	Integer	946
Yeast	10	8	Real	1484
Wine	3	13	Integer, Real	178
Glass identification	6	10	Real	214

Datasets characteristics

Classification problem

Dataset	KNN (%)	DT (%)	SVM(%)	S2GP (%)	VNP (%)
IRIS	95	91	94	96	96.7
VEHICLE	54	51	51	56	55.3
YEAST	50	55	58	61	58.2
WINE	84	84	83	85	89.1
GLASS	60	62	63	64	66

Classification results

Preventive maintenance planning in railway transportation

Overview

- Railway transportation is highly regulated by the state.
- The maintenance of the railway is important for keeping freight and passenger trains moving safely.
- Railroad companies make an inspection run for each time period and record the characteristic of found defects.
- If a defect does not satisfy Federal Railroad Administration (FRA) standards, then it is classified as a red tag and must be repaired immediately. Otherwise the defect belongs to yellow class and its fixation is not urgent.
- The Railway Application Section (RAS) provides the historic of the data describing the status of a several numbers of points in the railway.

Preventive maintenance planning in railway transportation

Problematic

2015 RAS Problem Solving Competition is to predict the color of a selected defect in a predefined milepost value after a given period.

Preventive maintenance planning in railway transportation

Solution

we can extract two different problems:

- Prevision problem: The prediction of the attribute values responsible for the determination of the defect severity after a selected number of days.
- Classification problem: we use the updated attribute values to classify a given defect (VNP indicates if the defect color is red or yellow).
- VNP algorithm is flexible to be applied in the classification and the prediction fields.

Honor Mention

Conclusions

- New algorithm introduction called VNP and based on local search and manipulating programs;
- New solution representation ameliorating the property of generalization;
- The optimization combining simultaneously the structure of the tree and its corresponding parameters;
- VNP algorithm application on two types of time series problems and five datasets of classification;
- The results indicating the good generalization and the effectiveness of the algorithm.