Podejście agentowe w projektowaniu sieci RBF
Agent-based approach to design of the RBFNs

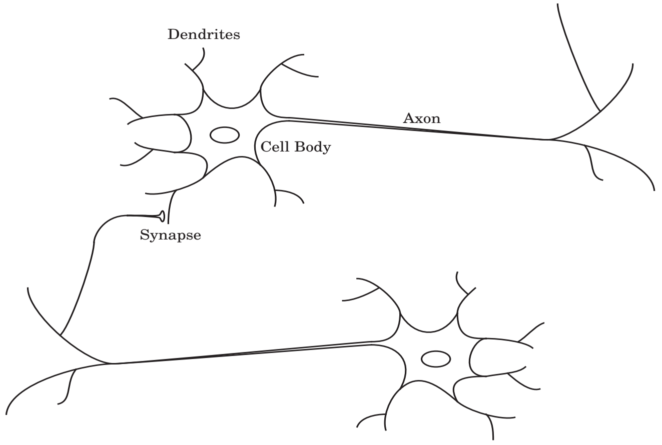
Irek Czarnowski

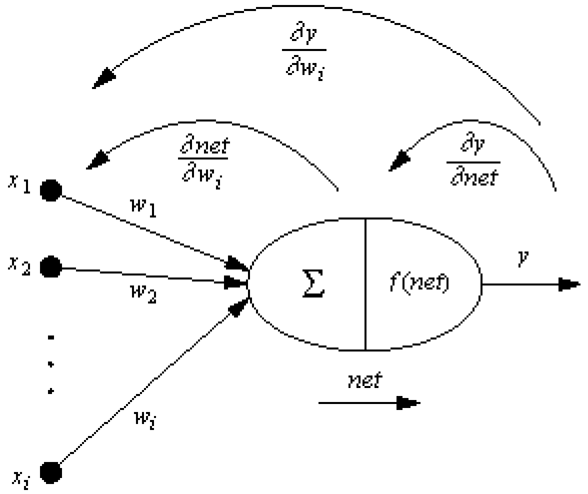
e-mail: i.czarnowski@wpit.am.gdynia.pl

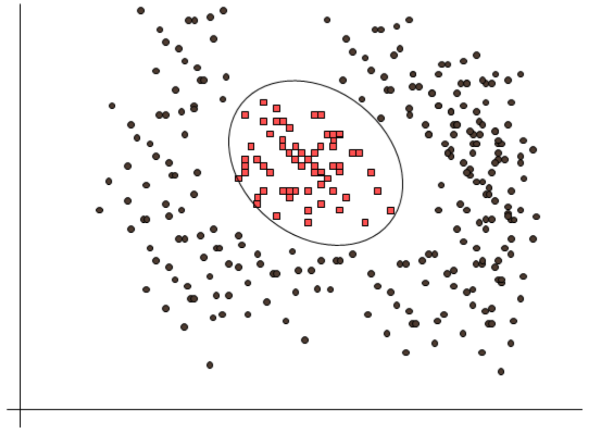
III spotkanie polskiej grupy badawczej systemów uczących się
Wrocław, 17-18 marca 2014r.

- Radial basis function networks (RBFNs) are a popular type of feedforward networks
- The RBF networks can be considered as universal approximation tools similarly to multilayer perceptrons (MLPs)
- RBFNs usually achieve faster convergence than MLPs

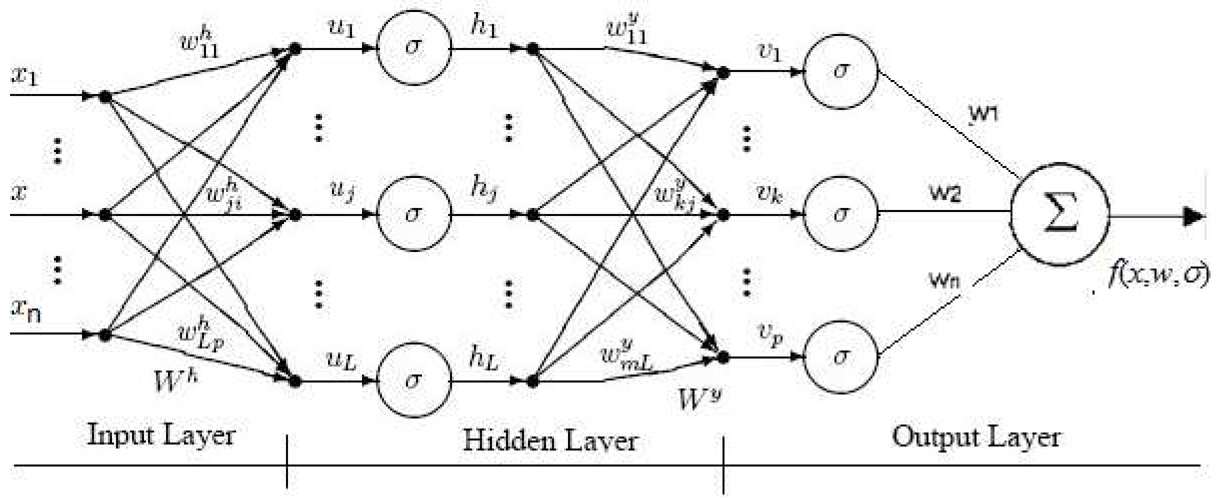
• Perceptron


Dendrites
Axon
Cell Body
Synapse


$\frac{\partial y}{\partial w_i}$
$\frac{\partial net}{\partial w_i}$
$\frac{\partial y}{\partial net}$
x_1
x_2
x_i
w_1
w_2
w_i
Σ
$f(net)$
y
net



• MLP Networks

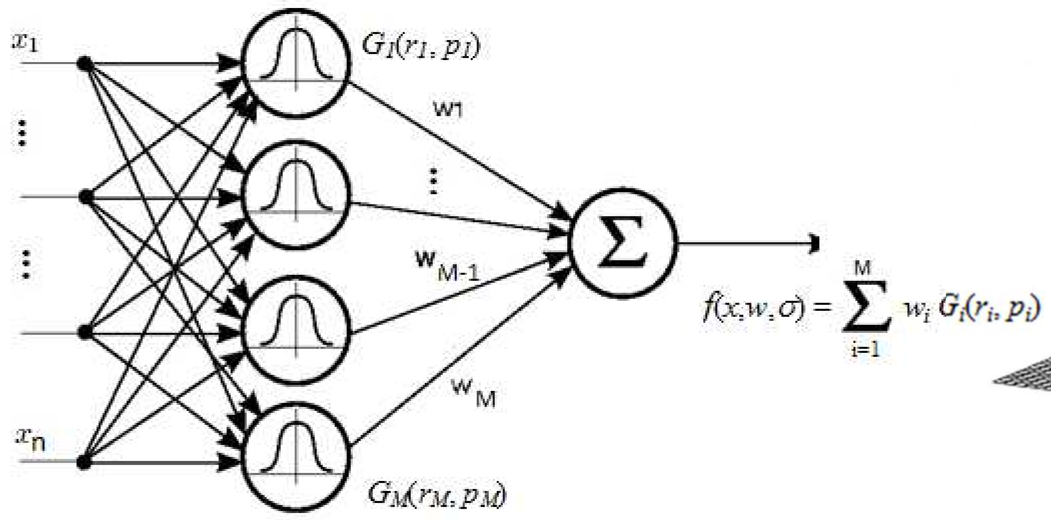


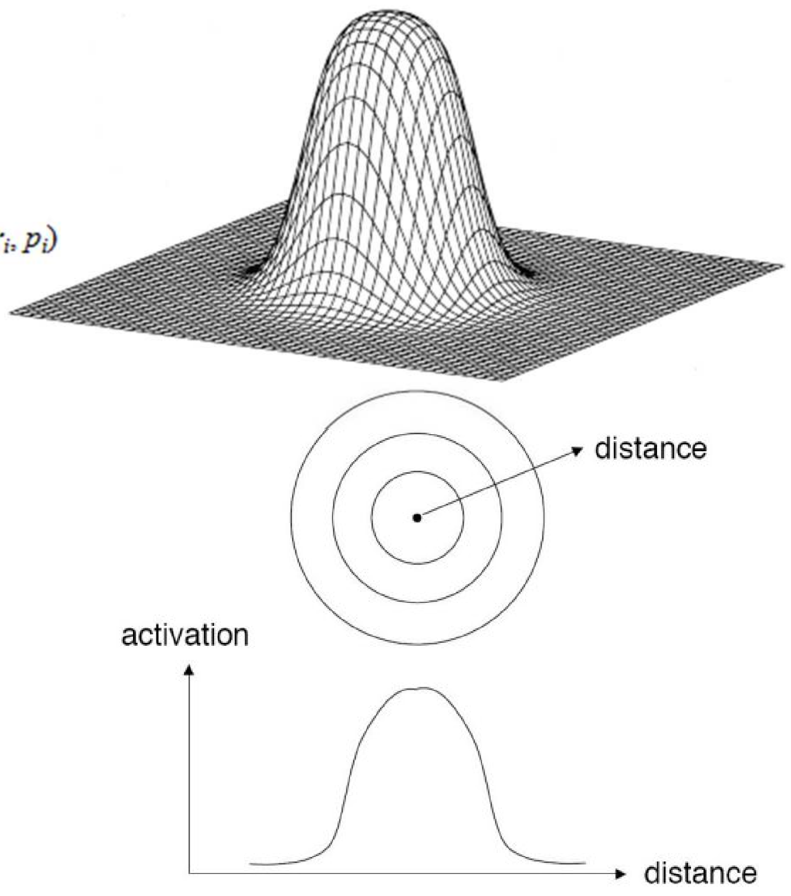
www.dtreg.com/mlfn.htm

There are several issues involved in designing and training a multilayer perceptron network:

- **Selecting how many hidden layers to use in the network**
- **Deciding how many neurons to use in each hidden layer**
- **Finding a globally optimal solution that avoids local minima**
- **Converging to an optimal solution in a reasonable period of time**

• RBF Networks





- Weights of the input layer => kernels of the basis functions
- Hidden layer consists of hidden neurons with radial basis function

• The performance of the RBF network

The performance depends on numerous factors:

- Number of hidden units
- Number of centroids and their location
- Types of transfer functions
- Shape parameters of transfer functions

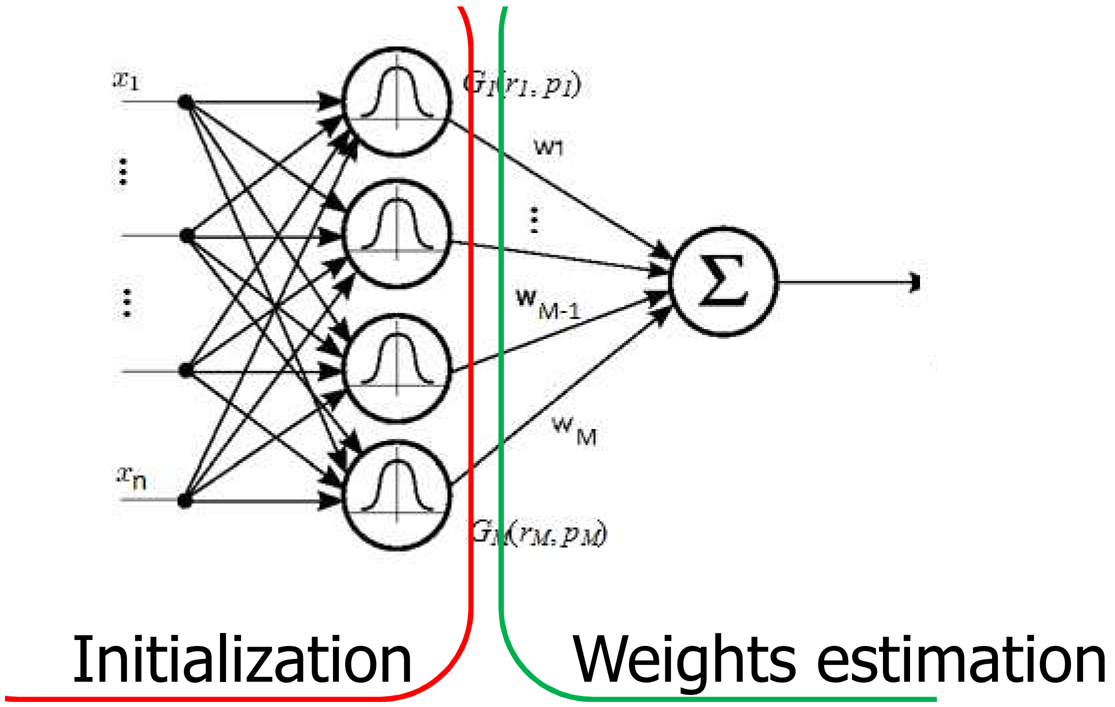
„…the shape of radial basis functions should be changed depending on the data distribution. Such a flexibility should result in assuring better approximation effect in comparison with other approaches, where, for example, radial basis function parameters are set by some ad hoc criterion…”
In: Hanrahan, G.: Artificial Neural Networks in Biological …., 2011

„Fixing centroids decrease chances to find optimal neural network structure.”
In: Wang, L., Yoang, B., Chen, Y., Abraham, A., Sun, H., Chen, Z., Wang, H., Improvemenat of neural network classifiers using floating … , 2012

The feature of the universal transfer function „is an ability to change shape smoothly from one function form to another”
In: Hoffmann G.A.: Adaptive Transfer Function …., 2004

„The choice of transfer functions may strongly influence complexity and performance of neural networks used in classification and approximation tasks”

• RBFN Designing



• RBFN Designing

The RBF design problem boils down to following tasks:

INITIALIZATION

- How to determine centroids and their locations?
- How to determine the parameters associated with the radial-basis functions in the hidden layer?
- Which type of transfer function should be used?

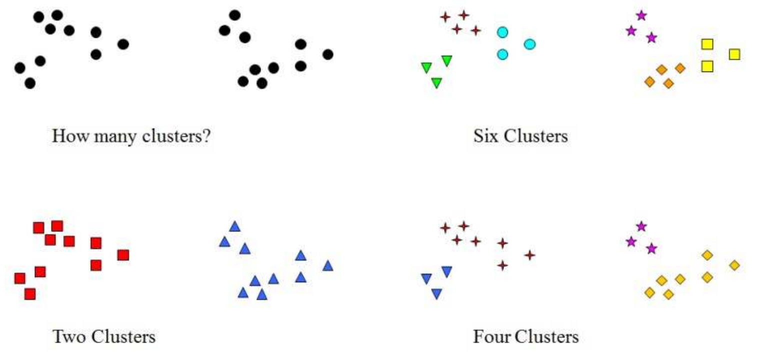
WEIGHTS ESTIMATION

- How to train the hidden-to-output weights?

• RBFN Designing

- Selection of method to calculate cluster centroids together with their parameters
  - Random selection
  - Clustering
  - Sequential growing
  - Systematic seeding
  - Editing techniques

A similarity-based approach is proposed to initialize clusters. It is assumed that from clusters prototypes are selected.



www.practicaldb.com/data-visualization-consulting

• Algorithm for Similarity-Based Clustering

Input: X - the matrix containing values of all instances from the training set, where $X = \{x_{ij}: i = 1,\ldots,N;$
$j = 1,\ldots, n + 1\}$, $n + 1$ element contains the class label with value from the finite set of class labels $C = \{c_l: l = 1,\ldots,k\}$
Output: clusters from which prototypes can be selected

1. Transform data instances: each $\{x_{ij}\}$ for $i=1,\ldots,N$ and $j=1,\ldots,n$ is normalized into interval [0,1] and then rounded to the nearest integer, that is 0 or 1
2. Calculate values:

$$s_j = \sum_{i=1}^{N} x_{ij}, \text{where } j = 1,\ldots,n.$$

3.
   a) For classification problem:
      For l=1 to k do
      For instances from X, belonging to the class c_l, calculate the value of its similarity coefficient I_i:

$$\forall_{x: x_{i,n+1}=c_l} \; I_i = \sum_{j=1}^{n} x_{ij} s_j, \text{where } i = 1,\ldots,N.$$

   b) For regression problem:
      For each instance from X calculate the value of its similarity coefficient I_i:

$$I_i = \sum_{j=1}^{n} x_{ij} s_j, \text{where } i = 1,\ldots,N.$$

4. Map input vectors from X with the same value of similarity coefficient I_i into clusters
5. Let $Y_1,\ldots,Y_t$ denote the obtained clusters such that $D = \bigcup_{i=1}^{t} Y_i$ and $\forall_{i \neq j: i,j=1,\ldots,t} \; Y_i \cap Y_j = \varnothing.$

• RBFN Designing

Selection of output functions of the RBF hidden units together with their parameters

- **The Gaussian function**

$$G(r,b) = e^{-(\frac{r}{b})^2}$$

r=||x-c|| is a norm function, c is a centroid, b is a value of dispersion of the radial basis function

- **The bicentral functions**

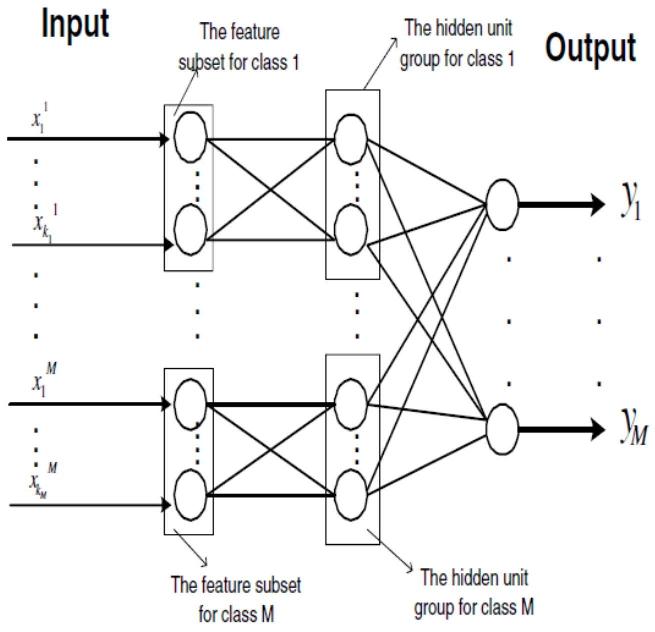
$$G(x,c,b,s) = \prod_{i=1}^{N} \sigma(e^{s_i} \cdot (x_i - c_i + e^{b_i}))(1 - \sigma(e^{s_i'} \cdot (x_i - c_i - e^{b_i})))$$

c is a centroid, b is a corresponding width of the radial basis function, N is a dimension of the instacnes, s represents a slope of the function, σ is a sigmoid function

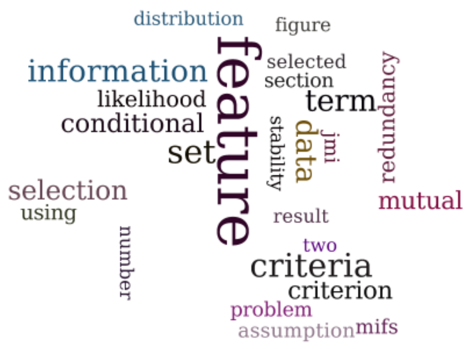
• RBFN Designing

Data Dimensionality Reduction

- Class-independent feature selection
- Class-dependent feature selection
- Cluster-dependent feature selection


Input
The feature subset for class 1
The hidden unit group for class 1
Output
x_1^1
$x_{k_1}^1$
x_1^M
$x_{k_M}^M$
y_1
y_M
The feature subset for class M
The hidden unit group for class M

Source: Lipo Wang, Xiuju Fu, A Rule Extraction System with Class-Dependent Features Evolutionary Computation in Data Mining. Studies in Fuzziness and Soft Computing Volume 163, 2005, pp 79-99



www.evolved-analytics.com

• Agent-Based Approach to RBFN

- The RBF network designing is a process, where the set of different parameters needs to be calculated or drawn.
- It is performed during the RBFN tuning
- The RBF neural network initialization (and training) belong to the class of computationally difficult combinatorial optimization problems, thus it is reasonable to apply to solve this task one of the known metaheuristics
- An agent-based population learning algorithm is proposed to
  - Producing clusters and determining their centroids
  - Determining the kind of transfer function for each hedden units and other parameters of the transfer function
  - Determining the hidden-to-output weights

• Agent-Based Population Learning Algorithm

Main features of the agent-based population learning algorithm:

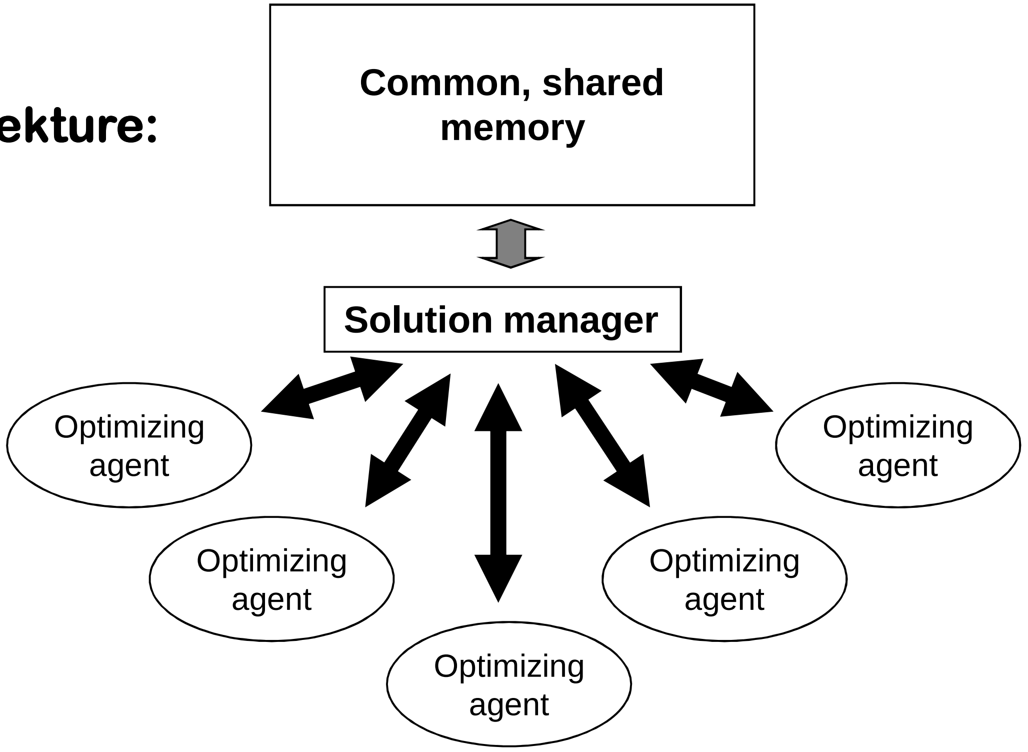
- The A-Team architecture, as a problem-solving paradigm, has been used to design the population learning algorithm
- The A-Team architecture was originally proposed by S. Talukdar as a set of objects including multiple agents and memories which through interactions produce solutions of optimisation problems
- A-Team – Asynchronous Team, where all agents can work asynchronously and in parallel
- Within an A-Team agents achieve an implicit cooperation by sharing a population of solutions
- Each agent possesses some problem-solving skills and each memory contains a population of temporary solutions to the problem at hand
- During their work agents cooperate to construct, find and improve solutions which are read from the shared, common memory

• Agent-Based Population Learning Algorithm

The functionality of the algorithm is implemented by two types of agents:

- **Optimizing agent (agent optymalizacyjny)**
- **Solution manager (agent nadzorujący)**

A-Team Architekture:

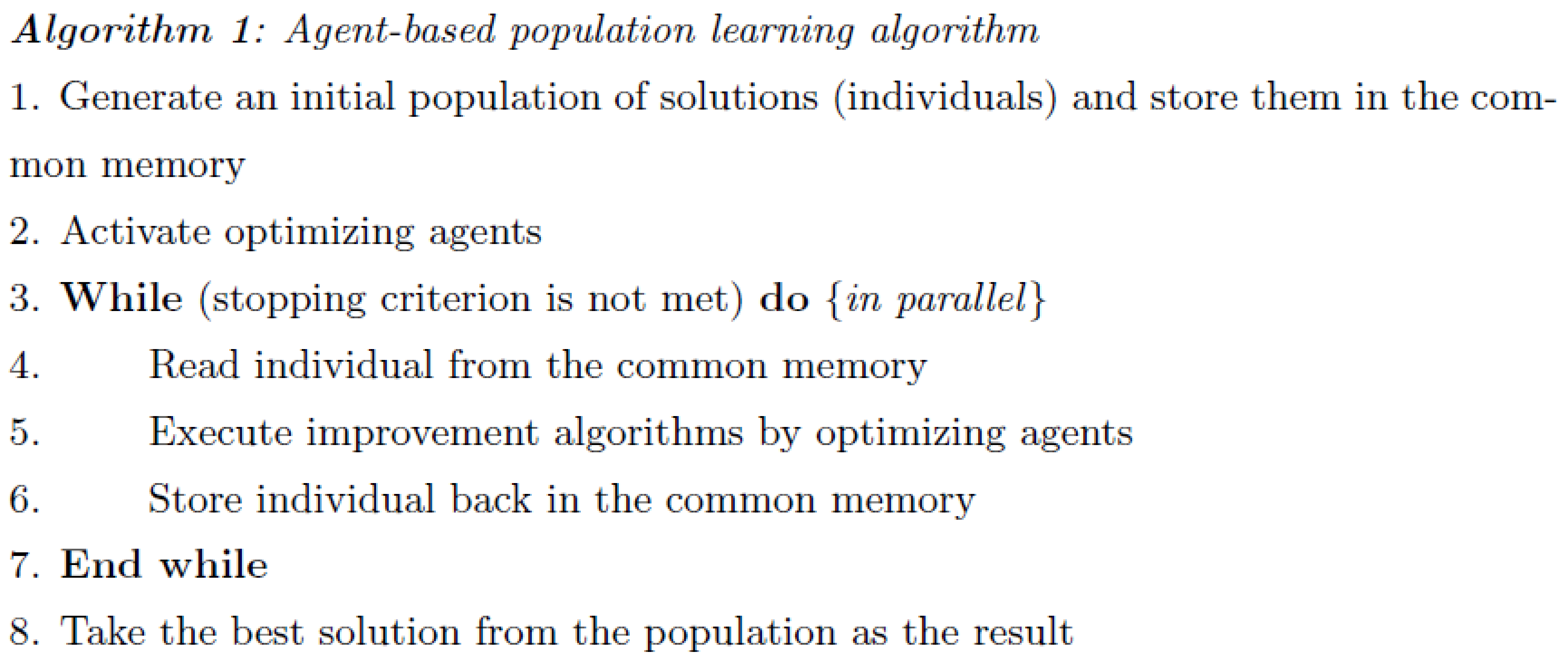


• Agent-Based Population Learning Algorithm

Process of searching for the best solution involves the following steps:

- Generation of the initial population of solutions to be stored in the common memory
- Activation of optimizing agents which apply solution improvement procedures to solutions drawn from the common memory and store them back after the attempted improvement applying some user defined replacement strategy
- Continuation of the reading-improving-replacing cycle until a stopping criterion is met. After computation has been stopped the best solution achieved so far is accepted as final

• Agent-Based Population Learning Algorithm



Algorithm 1: Agent-based population learning algorithm

```
1. Generate an initial population of solutions (individuals) and store them in the com-
mon memory
2. Activate optimizing agents
3. While (stopping criterion is not met) do {in parallel}
4.         Read individual from the common memory
5.         Execute improvement algorithms by optimizing agents
6.         Store individual back in the common memory
7. End while
8. Take the best solution from the population as the result
```

• Agent-Based Population Learning Algorithm

To implement the agent-based population learning algorithm one has to set and dene the following:

- Solution representation format
- Initial population of individuals
- Fitness function
- Improvement procedures
- Replacement strategy implemented for managing the population of individuals

• Agent-Based Population Learning Algorithm for RBFN

The main goal is to find the optimal set of RBF network parameters with respect to:

- Producing clusters and determining their centroids (including cluster-dependent feature selection)
- Determining the kind of transfer function for each hidden units and other parameters of the transfer function
- Determining output weights of the RBF network.

• Agent-Based Population Learning Algorithm for RBFN

The basic assumptions are following:

- Shared memory of the A-Team is used to store a population of solutions to the RBFN considered problem

- A solution is represented by a string consisting of two parts:
  - The first contains integers representing numbers of instances selected as centroids
  - The second part consists integers representing numbers of feature selected for each centroid
  - The third part consists of real numbers for representing left and right slope of the transfer functions
  - The fourth part consists of real numbers for representing weights of connections between neurons of the network

- The initial population is generated randomly

- Initially, the numbers included in the solutions are generated randomly

- Each solution from the population is evaluated and the value of its fitness is calculated
  - The evaluation is carried out by estimating classification accuracy or error approximation of the RBFN, assuming it is initialized using centroids, set of transfer function parameters and set of weights as indicated by the solution produced by the proposed approach.

• Agent-Based Population Learning Algorithm for RBFN

Determining the kind of transfer function

- The number of cluster centroids determines the kind of transfer function associated with a given hidden unit
- When only one centroid is selected, the output of the RBF hidden unit is calculated using the Gaussian function
- When the number of selected centroids is greater than one the output of the RBF hidden unit is calculated using the bicentral function

• Agent-Based Population Learning Algorithm for RBFN

To solve the RBFN tuning problem four groups of optimizing agents have been considered

- The first group includes agents executing procedures for centroid selection
- The second group includes agents executing procedures for cluster-dependent feature selection
    - Local search with the tabu list for prototype selection
    - Simple local search
- The third group of optimizing agents includes procedures for estimation of the slope parameters
    - Standard mutation
    - Non-uniform mutation
- The fourth group of optimizing agents includes procedures for estimation of the output weights
    - Gradient mutation
    - Grudient adjustment

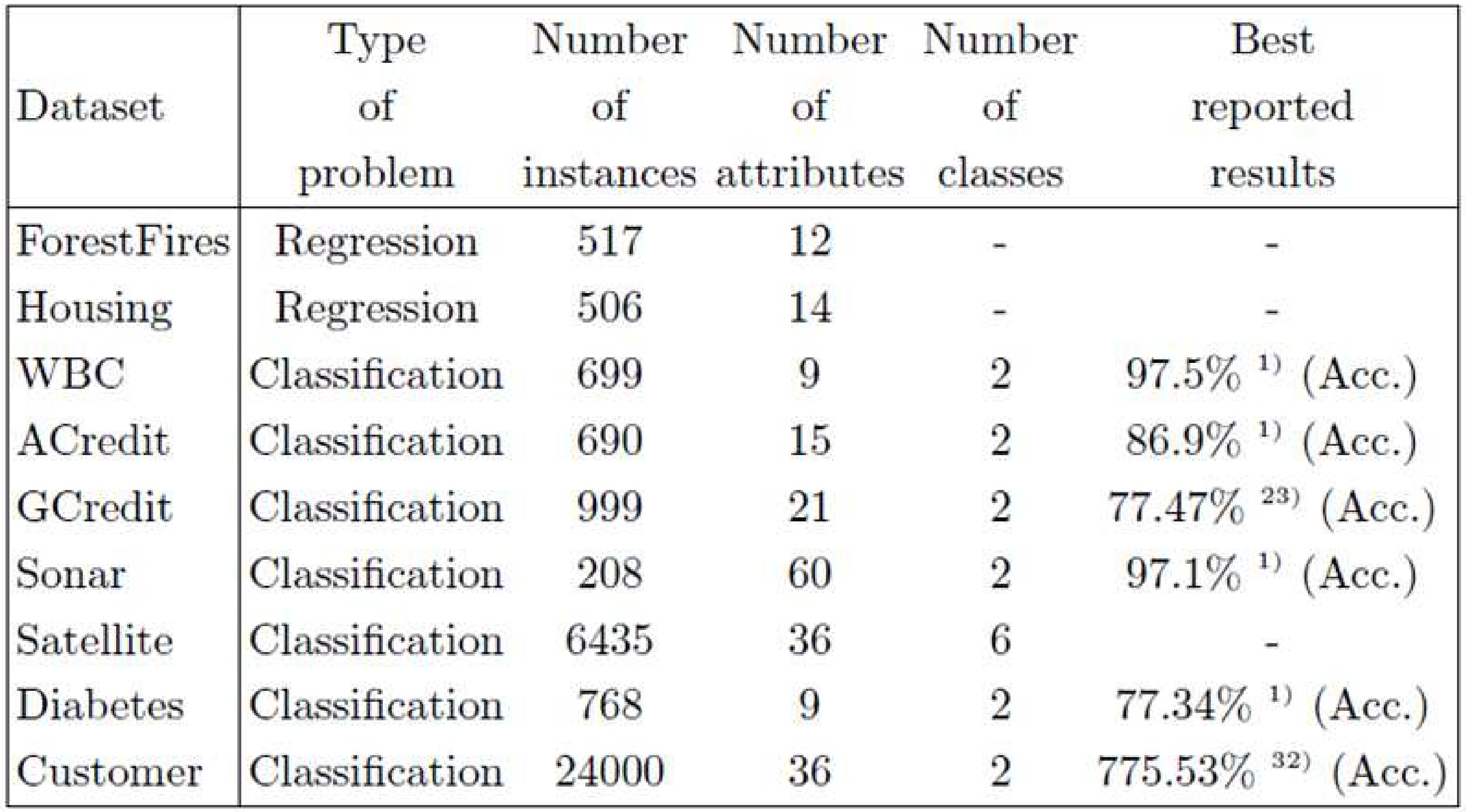
• Computational Experiments

The experiment aimed at answering the following basic questions:

- Does the proposed RBFN designing approach perform better than classical methods to RBFN initialization, training?
- Does the agent-based RBF network (ABRBFN) is competitive with other algorithms?

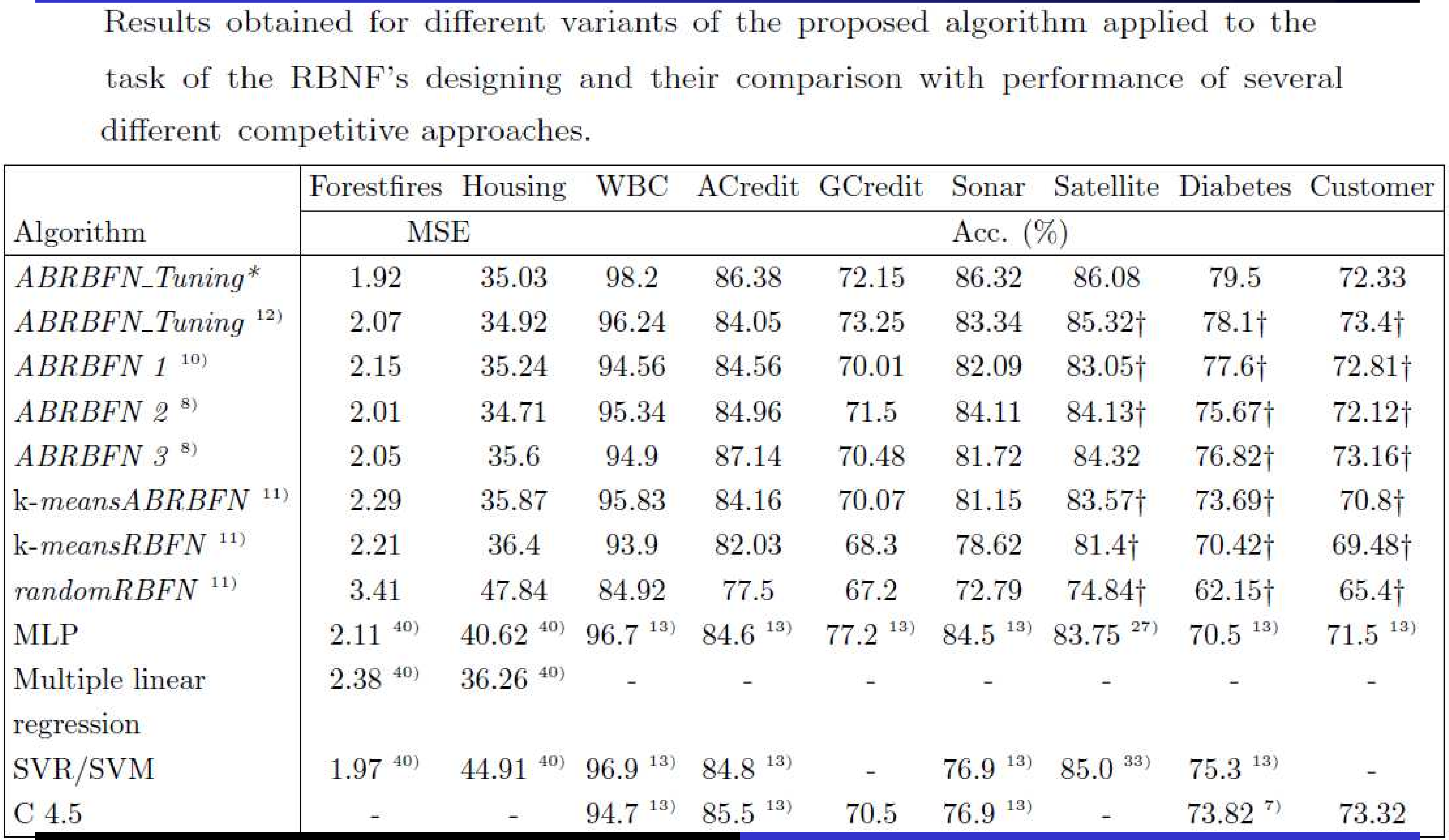
• Computational Experiments

Datasets used in the reported experiment.



Dataset	Type of problem	Number of instances	Number of attributes	Number of classes	Best reported results
ForestFires	Regression	517	12	-	-
Housing	Regression	506	14	-	-
WBC	Classification	699	9	2	97.5% [1] (Acc.)
ACredit	Classification	690	15	2	86.9% [1] (Acc.)
GCredit	Classification	999	21	2	77.47% [23] (Acc.)
Sonar	Classification	208	60	2	97.1% [1] (Acc.)
Satellite	Classification	6435	36	6	-
Diabetes	Classification	768	9	2	77.34% [1] (Acc.)
Customer	Classification	24000	36	2	775.53% [32] (Acc.)

• Computational Experiments

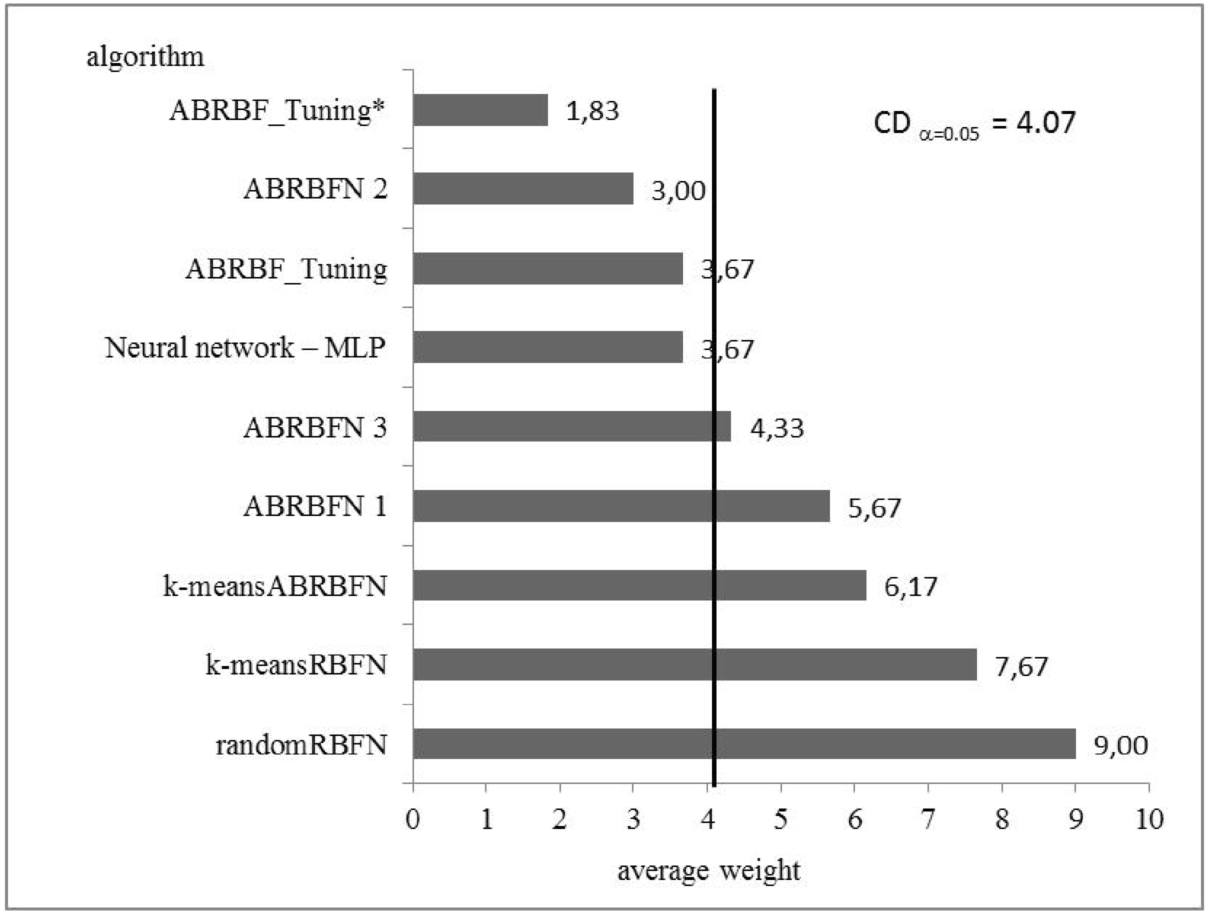


Results obtained for different variants of the proposed algorithm applied to the task of the RBNF's designing and their comparison with performance of several different competitive approaches.

Algorithm	Forestfires	Housing	WBC	ACredit	GCredit	Sonar	Satellite	Diabetes	Customer
	MSE		Acc. (%)						
*ABRBFN_Tuning**	1.92	35.03	98.2	86.38	72.15	86.32	86.08	79.5	72.33
ABRBFN_Tuning [12]	2.07	34.92	96.24	84.05	73.25	83.34	85.32†	78.1†	73.4†
ABRBFN 1 [10]	2.15	35.24	94.56	84.56	70.01	82.09	83.05†	77.6†	72.81†
ABRBFN 2 [8]	2.01	34.71	95.34	84.96	71.5	84.11	84.13†	75.67†	72.12†
ABRBFN 3 [8]	2.05	35.6	94.9	87.14	70.48	81.72	84.32	76.82†	73.16†
k-*meansABRBFN* [11]	2.29	35.87	95.83	84.16	70.07	81.15	83.57†	73.69†	70.8†
k-*meansRBFN* [11]	2.21	36.4	93.9	82.03	68.3	78.62	81.4†	70.42†	69.48†
randomRBFN [11]	3.41	47.84	84.92	77.5	67.2	72.79	74.84†	62.15†	65.4†
MLP	2.11 [40]	40.62 [40]	96.7 [13]	84.6 [13]	77.2 [13]	84.5 [13]	83.75 [27]	70.5 [13]	71.5 [13]
Multiple linear regression	2.38 [40]	36.26 [40]	-	-	-	-	-	-	-
SVR/SVM	1.97 [40]	44.91 [40]	96.9 [13]	84.8 [13]	-	76.9 [13]	85.0 [33]	75.3 [13]	-
C 4.5	-	-	94.7 [13]	85.5 [13]	70.5	76.9 [13]	-	73.82 [7]	73.32

• Computational Experiments

The average Friedman test weights and the Bonferroni-Dunn's graphic corresponding to the obtained ranking.



• Conclusions

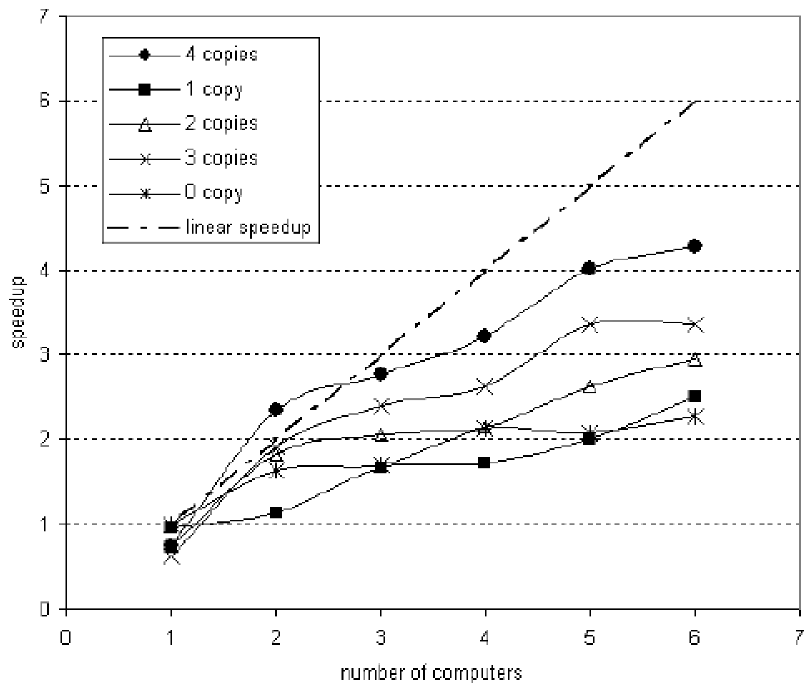
- Proposing an integrated and customized approach to designing RBFNs, where the resulting network “fits to measure” to the particular training dataset
- Important feature of the approach is that the clusters, location of prototypes, type of the transfer function and its parameters, and the output weights are integrated into a single optimization problem and are determined jointly and in parallel using a set of dedicated agents
- Proposing an approach having the following features making the RBFN design more flexible:
  - Structure of the RBNs is automatically initialized
  - Locations of centroids within clusters can be modified during the training process
  - Type of transfer function is determined during the training process
  - There is a possibility of producing a heterogeneous structure of the network

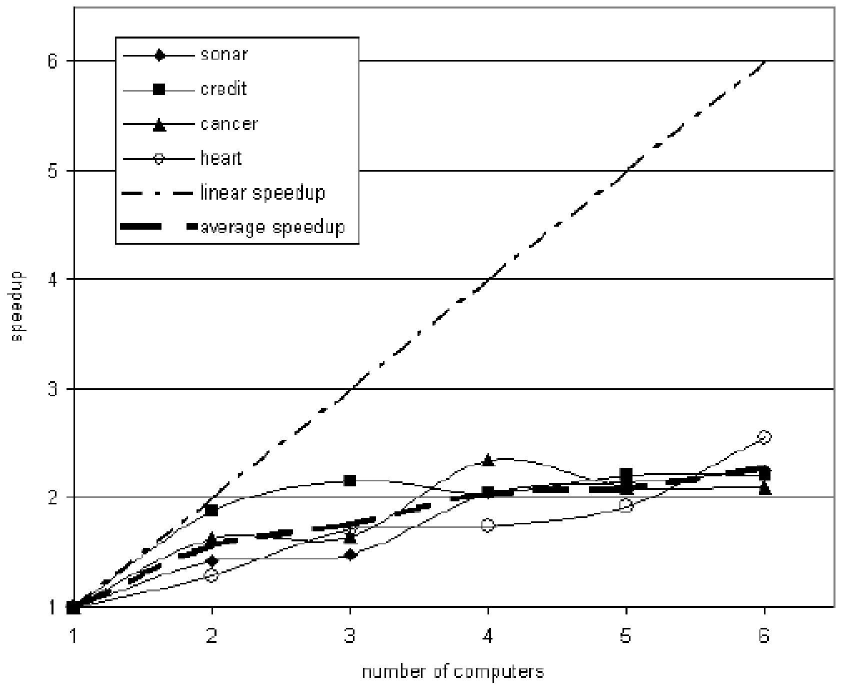
• Bibliography

- Czarnowski I., Jędrzejowicz P.: Designing RBF Networks Using the Agent-Based Population Learning Algorithm. **Cybernetics and Systems** (2014) - in print
- Czarnowski I., Jędrzejowicz P.: Agent-Based Population Learning Algorithm for RBF Network Tuning. In: Rutkowski L., Korytkowski M., Scherer R., Tadeusiewicz R., Zadeh A.L., Zurada J.M. (Eds.): Artificial Intelligence and Soft Computing, ICAISC 2013, **LNAI 7894**, pp. 41-51 Springer 2013, Berlin-Heidelberg
- Czarnowski I., Jędrzejowicz P.: Agent-Based Approach to the Design of RBF Networks. **Cybernetics and Systems** 44(2-3): 155-172 (2013)
- Czarnowski I., Jędrzejowicz P.: An Approach to Cluster Initialization for RBF Networks. In: Frontiers of Artificial Intelligence and Applications, **IOS Press** 2012.
- Czarnowski I., Jędrzejowicz P.: An agent-based approach to ANN training. **Knowledge-Based Systems** 19, Elsevier 2006, p. 304-308
- Czarnowski I, Jędrzejowicz P.: Implementation and Performance Evaluation of the Agent-Based Algorithm for ANN Training. **KES Knowledge-Based and Intelligent Engineering Systems**. IOS Press, Vol. 14(1), 2010, p. 1-10

• Parallel Implementation of A-Team for NN

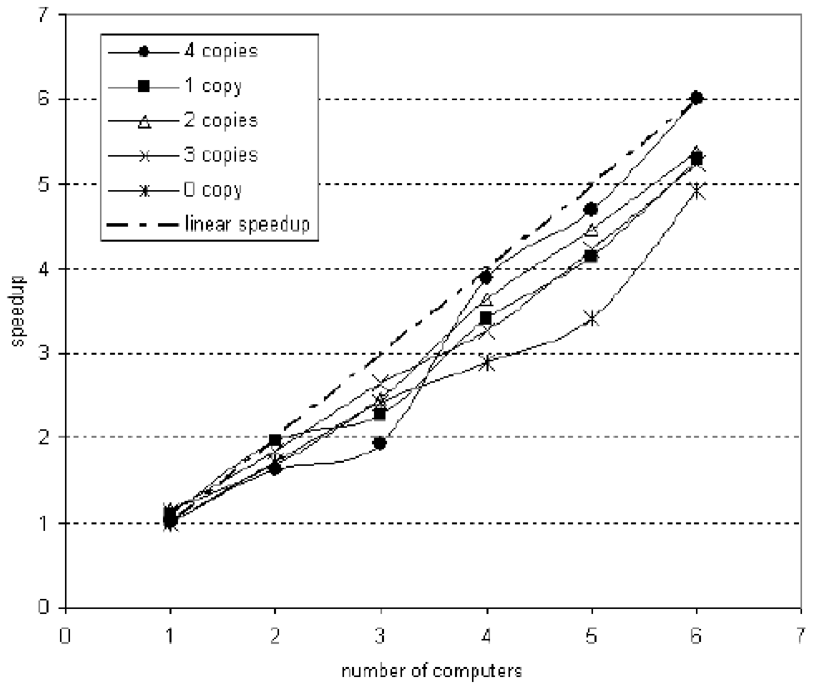
PC environment: Relations between the speed-up factor and the number of computers with more optimizing agents allowed (left box)
and between the speed-up factor and the number of computers, each with a single optimizing agent (right box). Data averaged over all problem types.

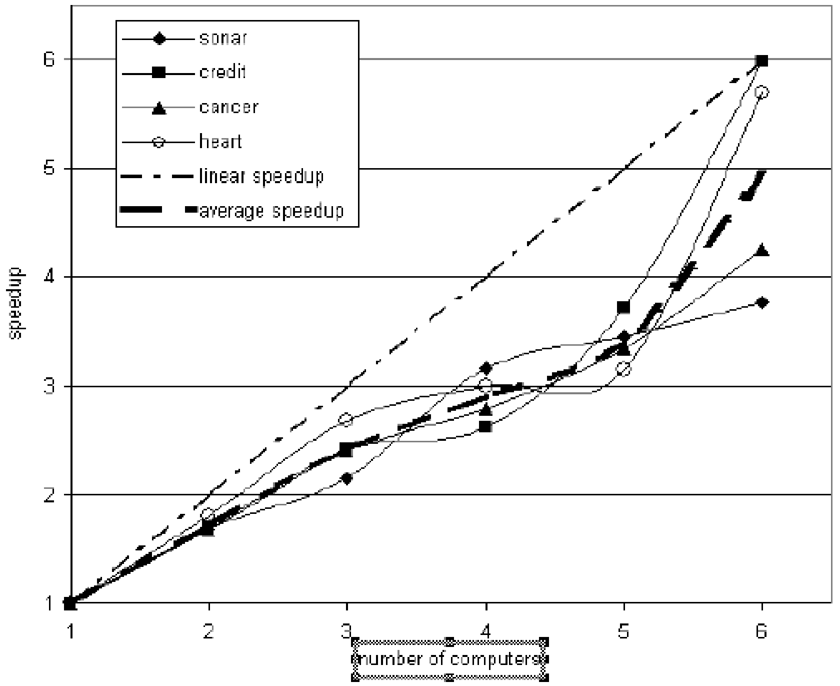




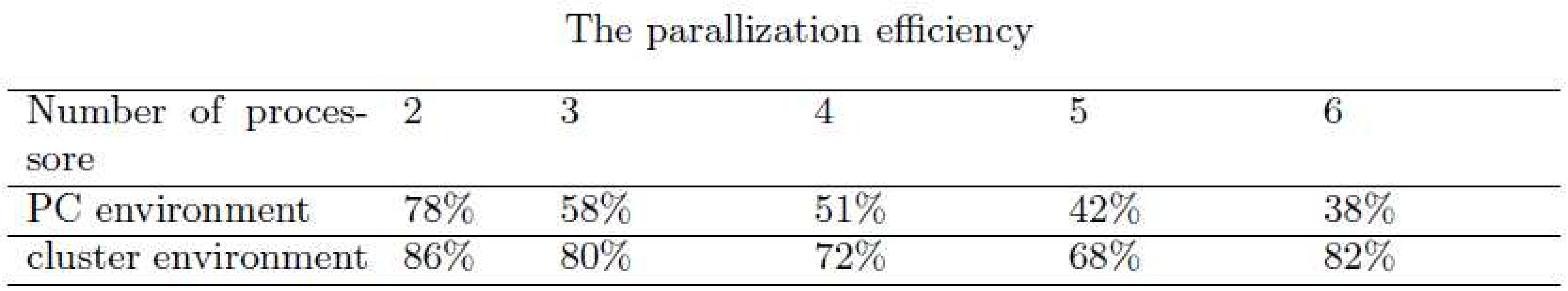
• Parallel Implementation of A-Team for NN

Cluster environment: Relations between the speed-up factor and the number of computers with more optimizing agents allowed (left box) and between the speed-up factor and the number of computers, each with a single optimizing agent (right box). Data averaged over all problem types.





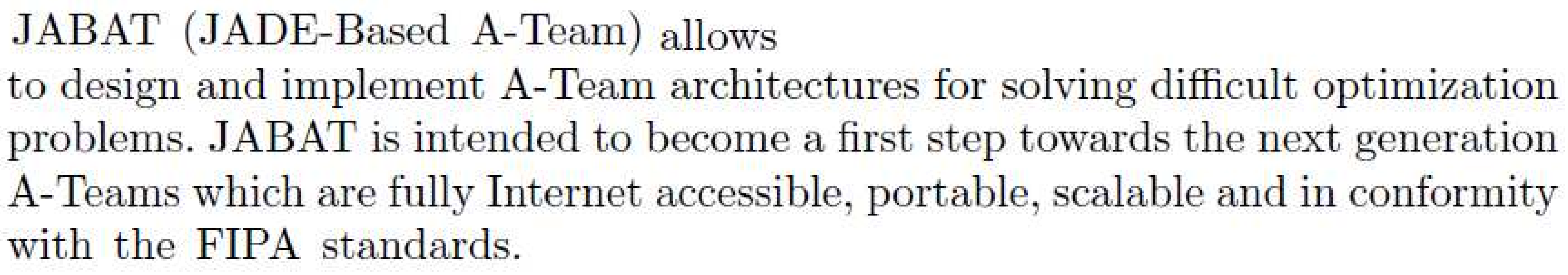
• Parallel Implementation of A-Team for NN



The parallization efficiency

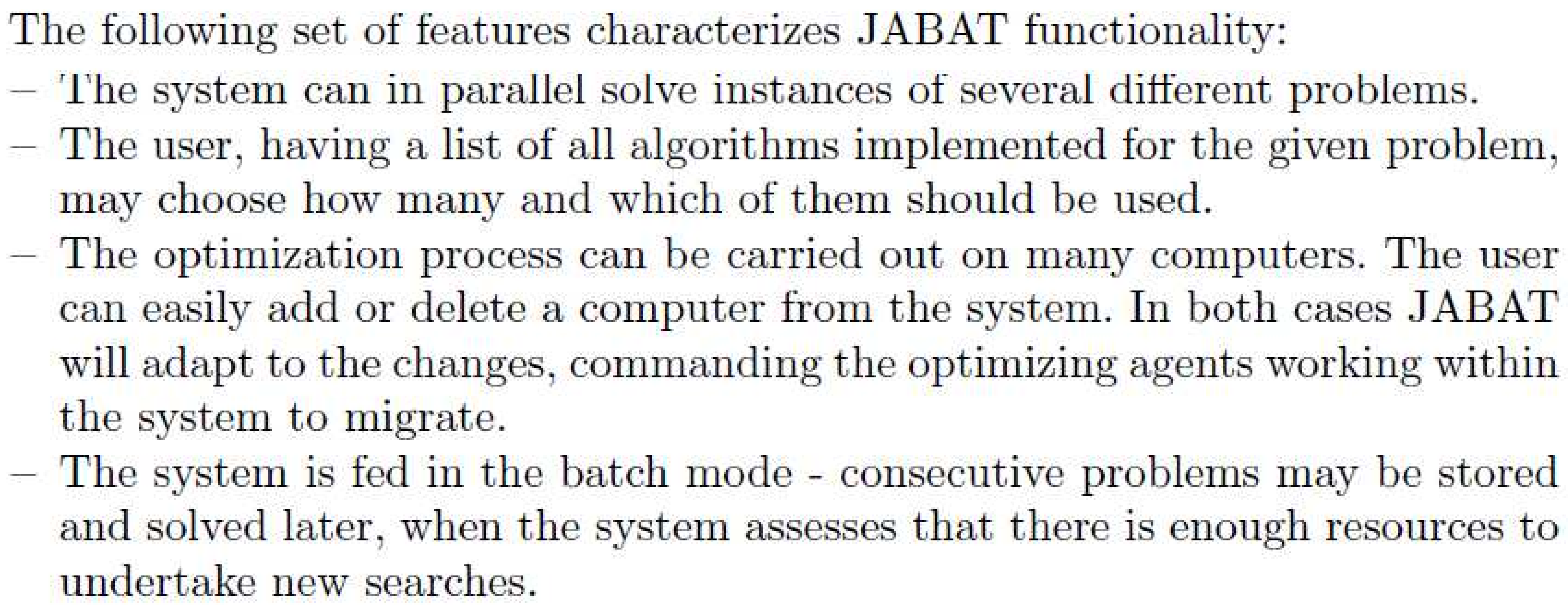
Number of procesore	2	3	4	5	6
PC environment	78%	58%	51%	42%	38%
cluster environment	86%	80%	72%	68%	82%

• JABAT


JABAT (JADE-Based A-Team) allows
to design and implement A-Team architectures for solving difficult optimization problems. JABAT is intended to become a first step towards the next generation A-Teams which are fully Internet accessible, portable, scalable and in conformity with the FIPA standards.

Barbucha, D., Czarnowski, I., Jędrzejowicz, P., Ratajczak, E., Wierzbowska, I.: *JADE-Based A-Team as a Tool for Implementing Population-Based Algorithms*. In: Y. Chen, A. Abraham, Intelligent Systems Design and Applications, ISDA, Jinan Shandong China, IEEE Los Alamos, pp. 144–149 (2006)

• JABAT

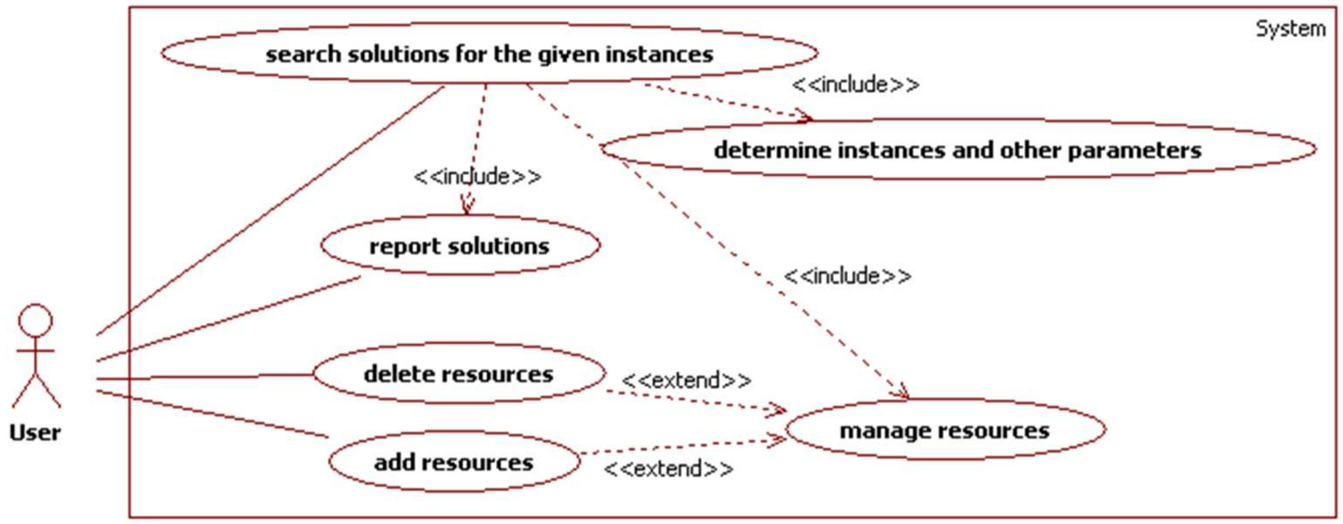


The following set of features characterizes JABAT functionality:

– The system can in parallel solve instances of several different problems.
– The user, having a list of all algorithms implemented for the given problem, may choose how many and which of them should be used.
– The optimization process can be carried out on many computers. The user can easily add or delete a computer from the system. In both cases JABAT will adapt to the changes, commanding the optimizing agents working within the system to migrate.
– The system is fed in the batch mode - consecutive problems may be stored and solved later, when the system assesses that there is enough resources to undertake new searches.

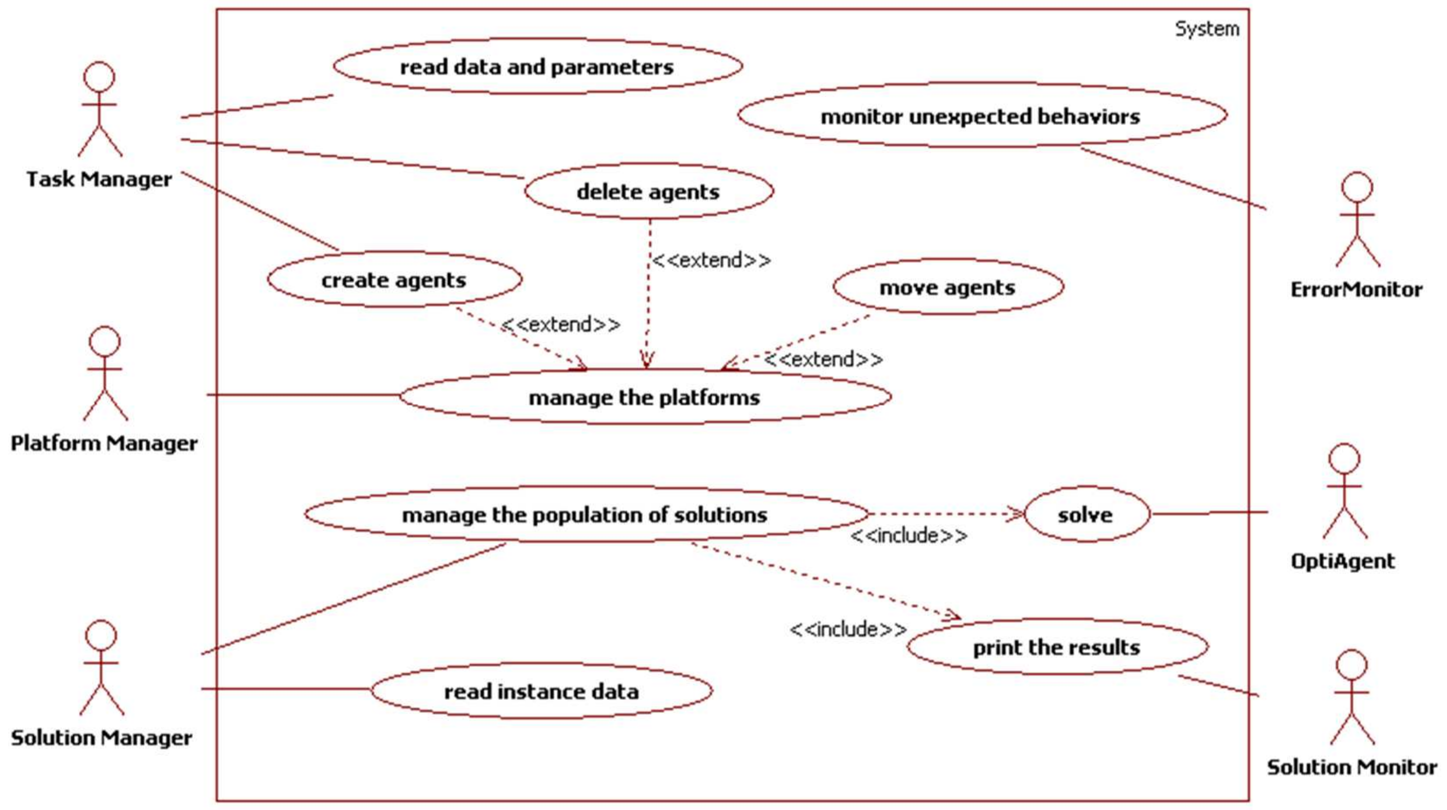
• JABAT

Use diagram of the functionality of JABAT



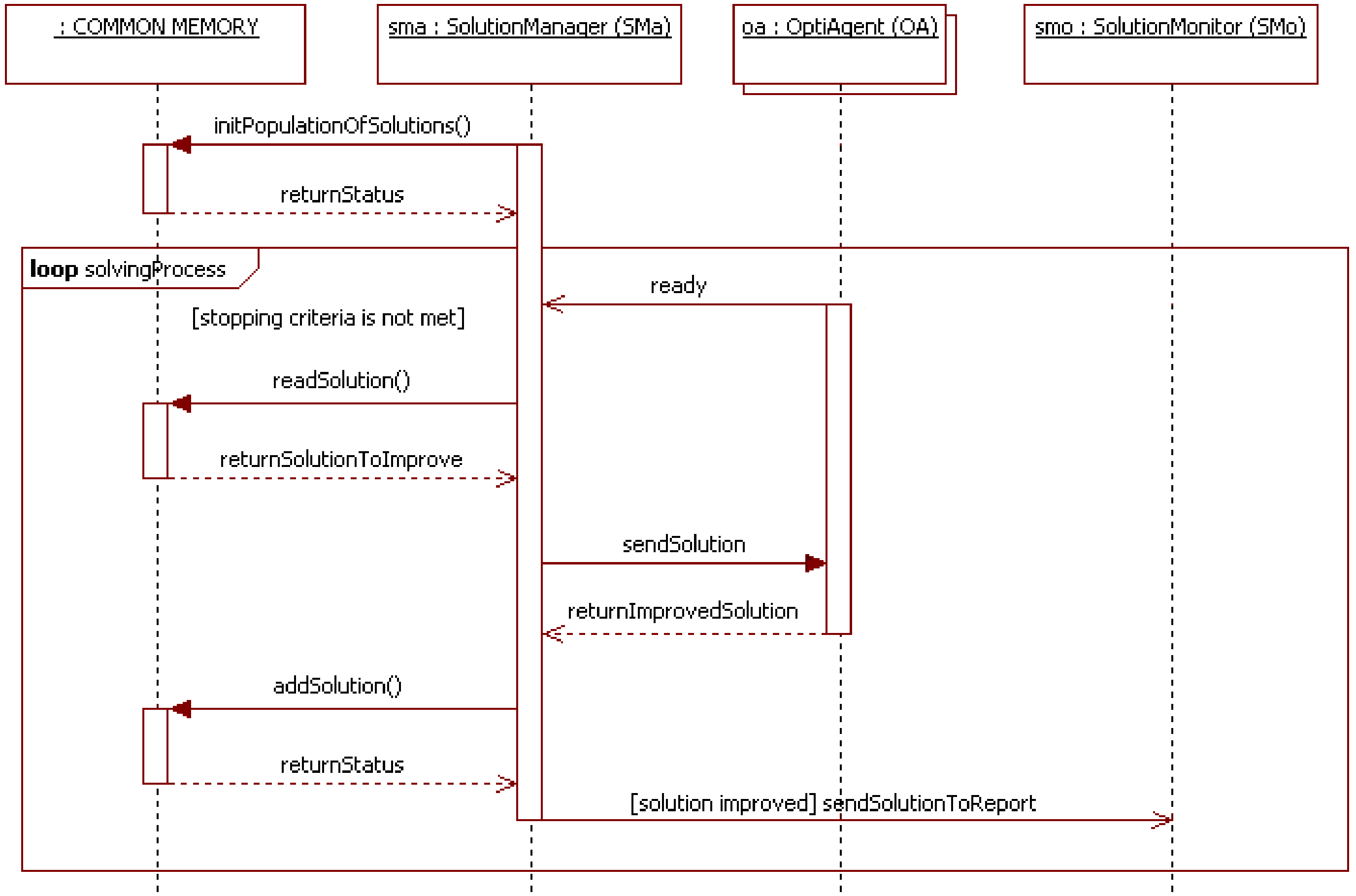
• JABAT

Use diagram of JABAT from the agents' point of view



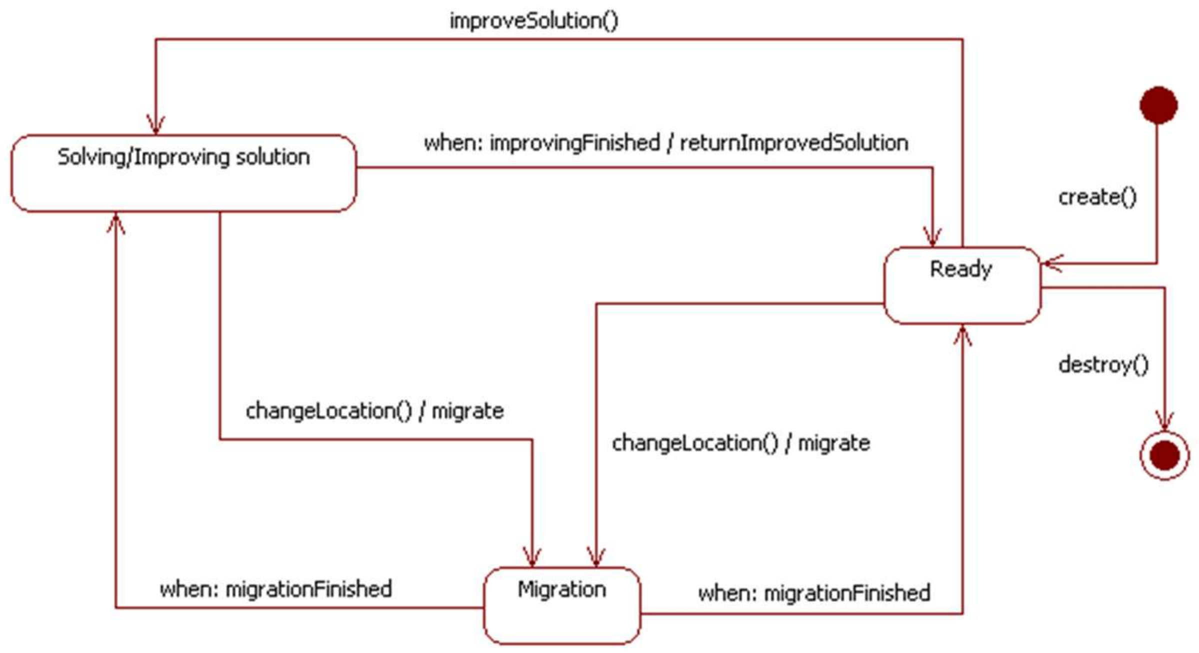
• JABAT

UML sequence diagram of the process of solving an instance of a problem



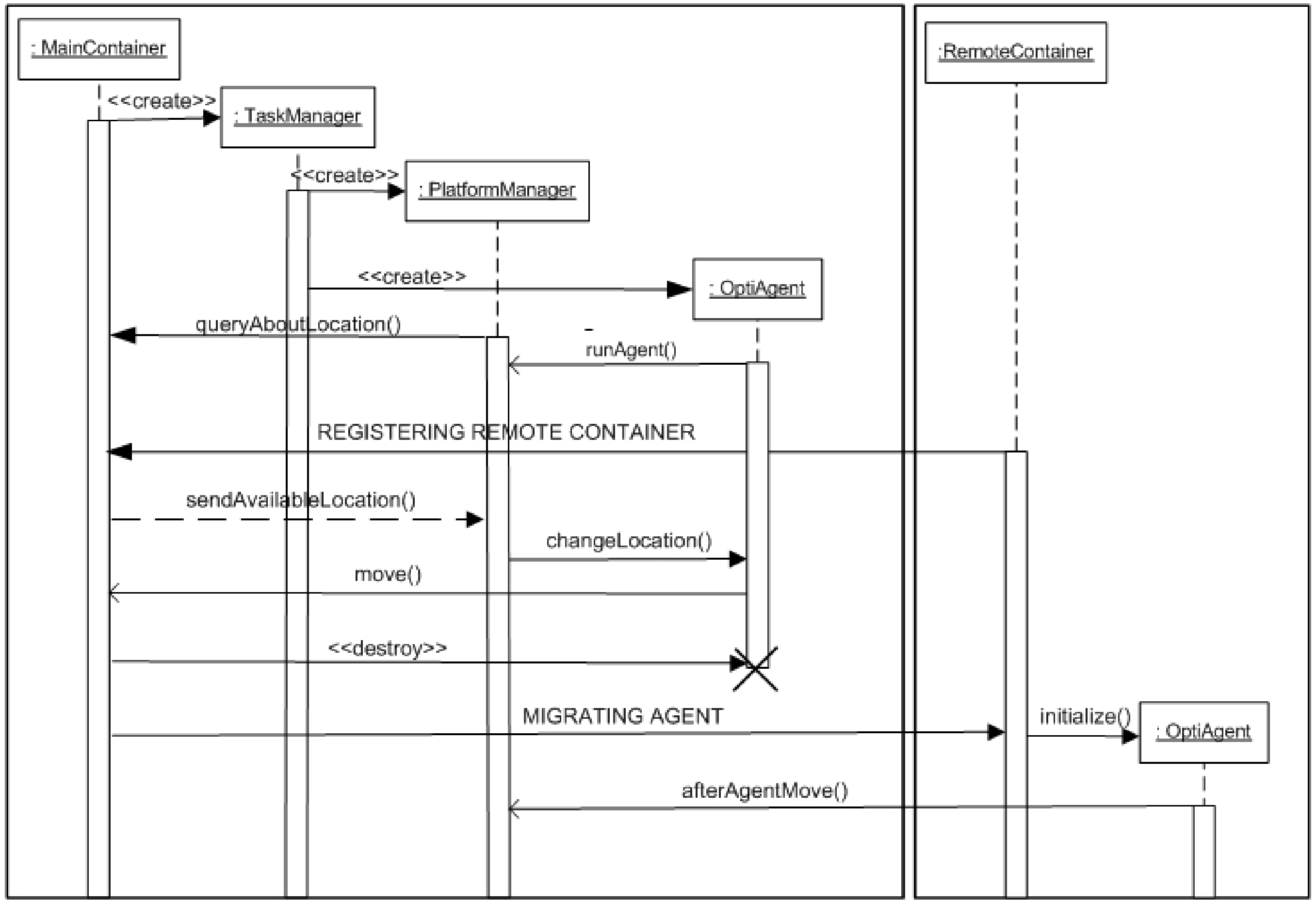
• JABAT

UML state transition diagram of the OptiAgent

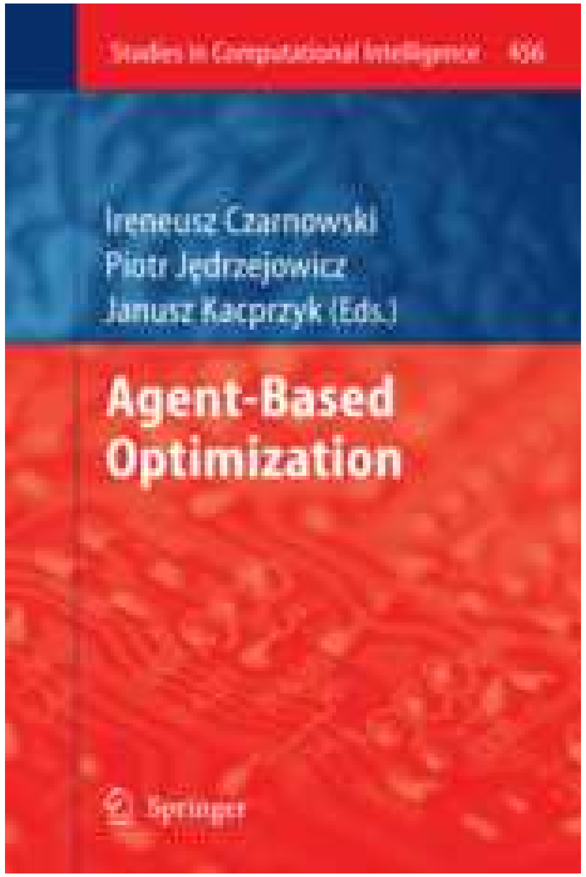

improveSolution()
Solving/Improving solution
when: improvingFinished / returnImprovedSolution
create()
Ready
destroy()
changeLocation() / migrate
changeLocation() / migrate
when: migrationFinished
Migration
when: migrationFinished

• JABAT

UML sequence diagram of the agents mobility in JABAT



• Agent-Based Optimization


Studies in Computational Intelligence 456

Ireneusz Czarnowski
Piotr Jędrzejowicz
Janusz Kacprzyk (Eds.)

Agent-Based Optimization

Springer

Czarnowski, Ireneusz, **Jędrzejowicz**, Piotr, **Kacprzyk**, Janusz (Eds.), **Agent-Based Optimization,** Studies in Computational Intelligence, Vol. 456, 2013

• Summary

Machine Learning and Multiagent Systems as Interrelated Technologies

Ireneusz Czarnowski* and Piotr Jędrzejowicz

Abstract. The chapter reviews current research results integrating machine learning and agent technologies. Although complementary solutions from both fields are discussed the focus is on using agent technology in the field of machine learning with a particular interest on applying agent-based solutions to supervised learning. The chapter contains a short review of applications, in which machine learning methods have been used to support agent learning capabilities. This is followed by a corresponding review of machine learning methods and tools in which agent technology plays an important role. Final part gives a more detailed description of some example machine learning models and solutions where the paradigm of the asynchronous team of agents has been implemented to support the machine learning methods, and which have been developed by the authors and their research group. It is argued that agent technology is particularly useful in case of dealing with the distributed machine learning problems. As an example of such applications a more detailed description of the agent-based framework for the consensus-based distributed data reduction is given in the final part of the chapter.

1 Introduction

Contemporary definition sees machine learning as a discipline that is concerned with the design and development of algorithms that allow computers to learn behaviors based on empirical data. Data can be seen as examples that illustrate relations between observed objects. A major focus of machine learning research is to automatically learn to recognize complex patterns and make intelligent decisions based on data.

Ireneusz Czarnowski · Piotr Jędrzejowicz
Department of Information Systems, Gdynia Maritime University,
Morska 83, 81-225 Gdynia, Poland
e-mail: irek,pj@am.gdynia.pl

* Corresponding author.

I. Czarnowski et al. (Eds.): Agent-Based Optimization, SCI 456, pp. 1–28.
DOI: 10.1007/978-3-642-34097-0_1 © Springer-Verlag Berlin Heidelberg 2013

There are several ways the machine learning algorithm can profit from applying agent technology. Among them the following will be addressed in this paper:

- There are numerous machine learning techniques where parallelization can speed-up or even enable learning. Using a set of agents may, in such circumstances, increase efficiency of learning.
- Several machine learning techniques directly rely on the collective computational intelligence paradigm, where a synergetic effect is expected from combining efforts of various program agents.
- There is a class of machine learning problems known as the distributed machine learning. In the distributed learning a set of agents working in the distributed sites can be used to produce some local level solutions independently and in parallel. Later on local level solutions are combined into a global solution.

• Summary

Machine Learning and Multiagent Systems as Interrelated Technologies

Ireneusz Czarnowski* and Piotr Jędrzejowicz

Abstract. The chapter reviews current research results integrating machine learning and agent technologies. Although complementary solutions from both fields are discussed the focus is on using agent technology in the field of machine learning with a particular interest on applying agent-based solutions to supervised learning. The chapter contains a short review of applications, in which machine learning methods have been used to support agent learning capabilities. This is followed by a corresponding review of machine learning methods and tools in which agent technology plays an important role. Final part gives a more detailed description of some example machine learning models and solutions where the paradigm of the asynchronous team of agents has been implemented to support the machine learning methods, and which have been developed by the authors and their research group. It is argued that agent technology is particularly useful in case of dealing with the distributed machine learning problems. As an example of such applications a more detailed description of the agent-based framework for the consensus-based distributed data reduction is given in the final part of the chapter.

1 Introduction

Contemporary definition sees machine learning as a discipline that is concerned with the design and development of algorithms that allow computers to learn behaviors based on empirical data. Data can be seen as examples that illustrate relations between observed objects. A major focus of machine learning research is to automatically learn to recognize complex patterns and make intelligent decisions based on data.

Ireneusz Czarnowski · Piotr Jędrzejowicz
Department of Information Systems, Gdynia Maritime University,
Morska 83, 81-225 Gdynia, Poland
e-mail: irek,pj}@am.gdynia.pl

* Corresponding author.

I. Czarnowski et al. (Eds.): Agent-Based Optimization, SCI 456, pp. 1–[illegible].
DOI: 10.1007/978-3-642-34097-0_1 © Springer-Verlag Berlin Heidelberg 2013

Main focus of this review is using agent technology in the field of machine learning with a particular interest on applying agent-based solutions to supervised learning. Some references are also made with respect to applying machine learning solutions to support agent learning. The review allows to formulate the following observations:

- Machine learning and agent technology are becoming more and more interelated bringing an important advantages to both fields.
- Machine learning can be seen as a prime supplier of learning capabilities for agent and multiagent systems.
- Agent technology have brought to machine learning several capabilities including parallel computation, scalability and interoperability.
- Agent-based solutions and techniques when applied to machine learning have proven to produce a synergetic effect originating from the collective intelligence of agents and a power of cooperative solutions generated through agent interactions.

- Thank you for your attention!