20th European Conference
on Artificial Intelligence
(ECAI-12)

Proceedings

3rd International Workshop on Combinations of Intelligent Methods and Applications (CIMA 2010)

August 28, 2012
Montpellier, France

Ioannis Hatzilygeroudis
Vasile Palade
Editors

20th European Conference on Artificial Intelligence

Proceedings

3rd International Workshop on Combinations of Intelligent Methods and Applications (CIMA 2012)

August 28, 2012
Montpellier, France

Ioannis Hatzilygeroudis, Vasile Palade
Editors

Table of Contents

Workshop Organization

Chairs-Organizers

Ioannis Hatzilygeroudis
University of Patras, Greece

Vasile Palade
University of Oxford, UK

Program Committee

Salha Alzahrani, Taif University, Saudi Arabia
Plamen Agelov, Lancaster University, UK
Soumya Banerjee, Birla Institute of Technology, India
Nick Bassiliades, Aristotle University of Thessaloniki, Greece
Kit Yan Chan, Curtin University of Technology, Australia
Artur S. d'Avila Garcez, City University, UK
Constantinos Koutsojannis, TEI of Patras, Greece
George Magoulas, Birkbeck College, Univ. of London, UK
Toni Moreno, University Rovira i Virgili, Spain
Ciprian-Daniel Neagu, University of Bradford, UK
Jim Prentzas, Democritus University of Thrace, Greece
Roozbeh Razavi-Far, Universite Libre de Bruxelles, Belgium
Han Reichgelt, Southern Polytechnic State University, GA, USA
Jun Sun, Jiangnan University, China
George Tsihrintzis, University of Piraeus, Greece

Contact Chair

Ioannis Hatzilygeroudis
Dept. of Computer Engineering & Informatics
University of Patras, Greece
Email: ihatz@ceid.upatras.gr

Preface

The combination of different intelligent methods is a very active research area in Artificial Intelligence (AI). The aim is to create integrated or hybrid methods that benefit from each of their components. It is generally believed that complex problems can be easier solved with such integrated or hybrid methods.

Some of the existing efforts combine what are called soft computing methods (fuzzy logic, neural networks and genetic algorithms) either among themselves or with more traditional AI methods such as logic and rules. Another stream of efforts integrates case-based reasoning with soft-computing and more traditional AI or machine learning methods. Yet another integrates logic-based agent approaches with non-symbolic approaches. Some of the combinations have been quite important and more extensively used, like neuro-symbolic methods, neuro-fuzzy methods and methods combining rule-based and case-based reasoning. However, there are other combinations that are still under investigation, such as those related to natural language processing and the Semantic Web. In some cases, combinations are driven by theoretical aspects, but usually they are created in the context of specific applications.

The Workshop is intended to become a forum for exchanging experience and ideas among researchers and practitioners who are dealing with combining intelligent methods either based on first principles or in the context of specific applications. There were totally fourteen papers submitted to the Workshop. Each paper was reviewed by at least three members of the PC. We accepted eleven papers. Revised versions of the accepted papers (based on the comments of the reviewers) are included in these proceedings in alphabetic order (based on first author).

The papers involve various intelligent methods and application domains. Chang et al use a genetic algorithm to optimize a back propagation neural network, which implements an early warning model for water eutrophication. Dinu et al introduce the SLiM middleware, a symbolic layer in the context of a software agent, that can integrate various heterogeneous components. Guedersen and Sormo present a combination of temporal abstraction, case based reasoning and fuzzy rule-like alarms in the context of a system that gives advice to drilling engineers. Li et al use a neuro-fuzzy approach for gray scale image impulse noise filtering. Nascimento et al combine fuzzy clustering and ontology engineering aspects in analyzing and representing the activities of a computer science research organization. Pereira et al combine machine learning with fuzzy logic for generating teaching tasks adaptable to the needs of children with reading disabilities. Prentzas and Hatzilygeroudis present a rule-based explanation mechanism for neurules, a kind of integrated rules that integrate symbolic rules and neurocomputing. Razavi-Far et al use a PSO algorithm for validating appropriateness of the simulated results of the IWO algorithm application to a refrigeration controller optimization. Souza et al present a combination of EPSO and PSO algorithms over a cooperative approach of multiple swarms. Sun, Palade and Wu propose a combination of genetic programming and quantum-behaved particle swarm optimization for modeling and optimization of fermentation process. Finally, Sun, Wu and Sun integrate the advantages of global and local feature extraction methods in the contextual constraints based linear discriminant analysis for face recognition improvement.

We hope that this collection of papers will be useful to both researchers and developers. Given the success of the first three Workshops on combinations of intelligent methods and applications, we intend to continue our effort in the coming years.

Ioannis Hatzilygeroudis

Vasile Palade

An Early Warning Model of Water Eutrophication Based on a NeurO-Genetic Approach

Jinyi Chang, Zhenjiang Qian, Li Tang,Conghua Xie[1]

Abstract. With the rapid economic development in China, environmental protection is facing an unprecedented pressure. In order to enhance aquatic environment control effectively and deal with suddenly environmental pollution accident on social and economic development, this paper establishes a Back Propagation (BP) neural network to fit aquaculture feed, total phosphorus, total nitrogen, oxygen consumption, and other nutritious indicators of changes in the aquaculture bases in Shajiabang town of Changshu. Furthermore, we uses a genetic algorithm to achieve the optimization of the BP neural network and construct a early warning model of for water eutrophication. The model provides a technical support for water environment governance and public decision-making. At the same time, the model is used to make further analysis for the samples of Shajiabang new aquaculture base and forecast good results.

Keywords: Genetic Algorithm, Water Eutrophication, Early Warning Model, Neural Network

1 INTRODUCTION

In recent years, the scale of aquatic framing is gradually expanding in China. In order to increase the output of aquatic products, the farmers release a lot of artificial feed in the aquaculture waters, which causes serious eutrophication in water bodies. At present, the water of lakes which are more than 70% is in the eutrophication that the area, the strength and the microcystin content are in substantial growth [1]. For example, there were several severe outbreaks of algal blooms in the DianChi Lake, Han River emerged a very rare phenomenon of water bloom. Dianchi Lake, Taihu Lake and Chaohu Lake, which are called “three lakes” in China, are equally swamped with algal blooms in the trend and have been included in the national environmental management focus. Water eutrophication has become a major anthropogenic problem issue in both inland and coastal waters [2].

The contradiction between the pollution of aquaculture and the environment of water bodies is becoming increasingly conspicuous. Between the aquaculture and environment of water bodies, most people only focus on the active influence of water on aquaculture, but don’t care the negative impact of aquaculture. Security problems of water environment caused by water eutrophication aroused sufficient attention in the past ten years. Most researches on the pre-warning of water eutrophication of the inland rivers and lakes at home and abroad focus on the water eutrophication of the pathogenesis[3], and evolution trend prediction phases[4,5].

Early warning models of water eutrophication systems can improve the level of understanding between the cause and effect relationships that influence environmental ecosystems or organisms of interest or concern [6]. And it can assist in decision-making for environmental management. Therefore, it is necessary to early warn the problem of water eutrophication in water bodies. Chapra[7] presented a discussion of the water quality modeling process, which used the term calibration, i.e., initial modeling efforts wherein the modeler “varies the model parameters to obtain an optimal agreement between the model calculations and the dataset.” Arhonditsis[8] reported on a meta-analysis of 153 mechanistic biogeochemical models and considered differences between model calibration and model validation. Lee [9], for tracking and monitoring of marine aquaculture zones representative of chlorophyll, dissolved oxygen and other water and weather conditions change, established an automated real-time remote monitoring system. Gilbert [10] used the vector auto regression model on behalf of the endogenous variables to establish red tide simulation forecasting model, whose accuracy could reach 83%.Two kinds of methods were always used for the pre-warning of water eutrophication of the inland rivers and waters.

The first kind of method is establishing "alert level" or "alert value" of the outbreak of water eutrophication, as a basis for pre-warning. Chen[11] had made a concept "water bloom alert level", which was targeted mainly for the purposes of algal bio-density. Lu [12] put forward a “alert value” of water bloom concept and water bloom pre-warning vision on the Han River.

The second kind of method is using historical data to establish correspondence between the indicators of characterization and the causations of the outbreak of water eutrophication, giving pre-warnings of that whether there will be outbreaks of water eutrophication in the future. Zeng Yong [13] used ID3 decision tree method to divide the causes of the outbreak of water bloom into different intervals in the space, and predicted the scenarios of the outbreak of water bloom in LiuHai Lake in BeiJing by using a linear multiple regressions for different intervals. A pre-warning model for water bloom was developed by ZENG [1] using the Artificial Neural Networks(ANN) method; in order to adopt the emergency response measures and avoid erupting water bloom, and provide scientific basis for improving the water environment of river and lake system.

We will construct an early warning model of water eutrophication for the Shajiabang aquaculture bases. Shajiabang town with the total area of 33.13 square kilometers, inclusive of 12 administrative villages and one residential committee, beside the beautiful Yangchehu Lake and close to Shanghai-Nanjing. The town is rated as the token of the land of rice and fish in the south, with genial climate, rich soil, sweet water and abundant agricultural products. This town is in the famous red national tourist zone in China. The famous Yangcheng Lake hairy crabs in Shajiabang town are sold all around the world. The aquaculture area in the whole town is 18,000 mu (Chinese unit of area, 1/15 of a hectare) and the total output of aquatic products is 5.500 tons per year.

At the same time, elevated levels of nutrients, such as total nitrogen (TN) and total phosphorus (TP) through agricultural pollution, sewage discharge, coupled with high temperature commonly

[1]School of Computer Science & Engineering, Changshu Institute of Technology, Changshu 215500, China
Email: jinyichang@sina.com

enhanced the growth of algae, and sometimes even results in nuisance algal blooms[14] and serious water eutrophication.

In this paper, we propose an early warning model of eutrophication index of aquaculture waters. And, we analyze and fit the sample data acquired from the aquaculture bases in Shajiabang town of Changshu in China and make prediction by simulation effectively.

2 Related Works

Routine methods for analyzing the relationship between nutrient input and algal blooms or inland water productivity through field sampling and laboratory analyses have been well documented [15]. Nevertheless, these traditional approaches are limited in their suitability for monitoring water bodies with large surface areas due to the dynamic nature and patchy distribution of nutrients, algal blooms and suspended matter[16].

BP neural network[17] is a kind of multi-layer feed forward neural network based on error back propagation training algorithm. The BP neural network uses steepest descent method to learn rule, and constantly adjusts weight values and threshold values resulting in the minimization of quadratic error. During forward propagation, the order of propagation direction is from input layer, hidden layer to output layer, the status of neuron of each layer only impacts on the neuron of next layer. If the output layer cannot have expected output, it will start the process of an error back propagation. Through running these two processes by turns, the strategy of error function gradient descent is executed in weight vector space, the weight vector components are updated at each iteration search dynamically, and the network error function is minimized, so that the information extraction and memory storage are finished.

The neural network design by using genetic algorithm[18] can search the optimal structure which satisfies the problem requirement in the structure space according to the standard of performance evaluation, such as learning speed, generalization capability and structure complication. Genetic Algorithm (GA) [19] is a kind of global optimization algorithm. It uses the viewpoint of biology genetics, e.g., improving the adaptability of living things by the mechanisms of natural selection, genetics and mutation. The GA model includes five components: encoding, population initialization, individual selection, crossover, and mutation.

The structure of genetic algorithm can be described as follows:

(1) Generating some random neural network with the different structures, encoding each structure of network. Each code chain corresponds to one network structure. All code chains constitute a population;
(2) Training each network by using the different initial connection weights;
(3) Computing the error of neural network corresponding to each code chain, determining the adaptability of each individual by using strategies such as the error, generalization capability or structure complication;
(4) Selecting the individual with largest adaptability as farther;
(5) Processing current population by using crossover operator and mutation operator, so that generating new population;
(6) Repeating steps from 2 to 5 until some individuals in the groups can meet the requirements, i.e., the neural network corresponding to this individual is optimal.

3 BP NEURAL NETWORK CONSTRUCTION OF AQUACULTRUE WATER AREAS

The reason of water eutrophication is the increment of nutritive materials. Currently, water eutrophication is measured based on TP (total phosphorus), TN (total nitrogen), SD (Secchi disk depth) and COD (chemical oxygen demand) [20,21]. The increment of nutritive materials is greatly related to feeding stuff. The feeding stuff which release to aquaculture water mainly includes different fertilizers and compound feeds [22].

Table 1. Eutrophication classification standard of Taihu Lake

Index Class	Water Quality			
	COD(mg/L)	TN(mg/L)	TP(mg/L)	SD(m)
Oligotrophic	0.48	0.08	0.005	8.00
Lower-mesotrophic	0.96	0.16	0.010	4.40
Mesotrophic	1.80	0.31	0.023	2.40
Upper-mesotrophic	3.60	0.65	0.050	1.30
Trophic	7.10	1.20	0.110	0.73
Hypertrophic	14.00	2.30	0.250	0.40
Eutrophic	27.00	4.60	0.560	0.22
Hypereutrophic	54.00	9.10	1.230	0.12

The fisheries station in Changshu town conducts water sampling of aquaculture base from four sampling spots every day. The SD is measured in site. Besides, the values of total TP, TN and COD are measured through analysis of water samples in lab. In the same time, the aquaculture operator records the number of released feeding stuff. We get all data in recent five years from them. And we use the classification standard of Taihu Lake in China in Table1.

We construct a BP neural network of this aquaculture water area by using the number of released feeding stuff as input matrix of BP neural network and using the sampling values of total TP, TN, SD and COD as output matrix of network.

Because the BP network includes error and a S-type hidden layer plus a linear output layer, it can approximate any rational function. We use a three-layer BP network model to approximate to the training data. The number of reasonable hidden layer nodes is 2. The hidden layer uses Sigmoid transfer function and output layer uses linear transfer function.

Suppose there are n inputs and m outputs in the network, and S neurons in the hidden layer, the output of the hidden layer is b_j , the threshold value of the hidden layer is θ_j, the threshold value of the output layer isθ_k , the transfer function of the hidden layer is f_1, the transfer function of the output layer is f_2, the weight from input layer to hidden layer is w_{ij} , the weight from hidden layer to output layer is w_{jk} . Then we can get the output of the network y_k , the desired output is t_k , and the output of jth neuron of the hidden layer is:

$$b_j = f_1\left(\sum_{i=1}^{n} w_{ij} x_i - \theta_j\right)(i = 1,2,\cdots,n;\ j = 1,2,\cdots,s) \quad (1)$$

Calculating the output y_k of the output layer, this is:

$$y_k = f_2\left(\sum_{j=1}^{s} w_{jk} b_j - \theta_k\right)(j = 1,2,\cdots,s;k = 1,2,\cdots,m) \quad (2)$$

Defining the error function by the network actual output, that is:

$$e = \sum_{k=1}^{m} (t_k - y_k)^2 \quad (3)$$

The network training is a process of continual readjustment between the weights and the threshold, in order to make the network error reduce to a pre-set minimum or stop at a preset training step. Then input the forecasting samples to the trained network, and obtain the forecasting results.

4 NN MODEL OPTIMIZATION WITH GA

Before optimizing the generated neural network by using the genetic algorithm, the output data of neural network described above need to be converted into adaptive matrix suitable for optimization of genetic algorithm. The transformation algorithm is described as follows.

```
Algorithm 1: Transformation Algorithm
(1) function t_tmp=anntoga(p_tmp)
(2) global net; global is_request_max;
(3) global minmax_t;
(4) min_t=minmax_t (1);
(5) max_t=minmax_t (2);
(6) t_tmp=sim(net, p_tmp);
(7) if is_request_max
(8) t_tmp=t_tmp-min_t+(max_t-min_t); else
(9) t_tmp =- t_tmp +max_t+(max_t-min_t);
```

The matrixes minmax_p and minmax_t are the maximum and minimum value in the input layer and output layer of the BP neural network, respectively. The function anntoga computes its value by using the generated neural network, which will be called in the program of genetic algorithm, and the function value should be within the range of minmax_t. The function sim implements the relationship of, which is the result of neural network training. At the same time, whether calculating maximum or minimum value depends on the identity of is request_max, and the output value of network is converted into the adaptive value that is suitable for optimization of genetic algorithm.

After the neural network establishes the model relationship, it looks for a suitable input matrix of network by using genetic algorithm, so that the output matrix of network will achieve the maximum (or minimum), i.e., the optimization of neural network by using the genetic algorithm. The optimization algorithm can be expressed as follows: Our algorithm that uses GA to optimize the connection weights of BP neural networks is as follows and the flow chart of the algorithm is as the Fig. 1 shows:

```
Algorithm 2: BP Neural Networks Optimization by
GA
(1) RandomGenerate(P[0]);
(2) pANN = new ANN;
(3) Set iGeneration = 1;
(4) REPEAT
(5)  For i = 1, 2, 3,…, Size
(6)  pANN→SetANNWeight(P[iGeneration]);
(7)  Fitness[i] = C - pANN→ComputeError();
(8)  NewP = Select, Crossover, Mutate;
(9)  Set iGeneration = iGeneration+1;
(10)  P[iGeneration] = NewP;
(11)  UNTIL halting criteria are satisfied;
(12)  Local search with the BP algorithm.
```

The above genetic algorithm includes the processes of encoding, selection, crossover, and mutation. The encoding function encodes the input data, and the corresponding function decoding is a decoding function. Then, the neural network is optimized after limited number of loop times of selection operator function selection, crossover operator crossover and mutation operator function mutation.

The key problems of the algorithm 2 and are as follows:

Coding Strategy: There are two ways to encode the connection weights and the threshold in the neural network. One is binary encoding, the other is real encoding. Binary encoding is that each weight is expressed by fixed length 0, 1 string. Real encoding is to express each weight with a real number, it overcomes the abuse of binary encoding, but it need to re-design the operators, such as the crossover and mutate operators, such as the crossover and mutate operators.

Determine the fitness function: Using GA to evolve the weights of neural networks, if the network architecture is fixed, generally, the network with big error, the fitness is small.

Evolution process: Determine the global search operator of the algorithm, such as selection, crossover and mutation operator, also can design special operator.

Train the network: Because GA is good at searching large-scale, complex, non-differentiable and multimodal spaces; it doesn't need the gradient information of the error function. On the other hand, it doesn't need to consider whether the error function is differentiable, so some punishment may be added to the error function, so as to improve the network commonality, and reduce the complexity of the network.

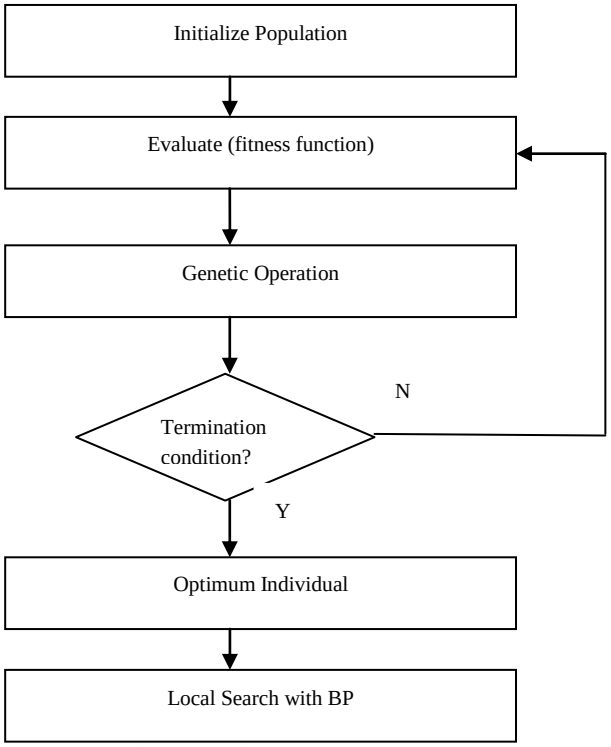


Figure1. The flow chart of using GA to optimize BP algorithm

Because GA is good at searching large-scale, complex, non-differentiable and multimodal spaces; it doesn't need the gradient information of the error function. On the other hand, it doesn't need to consider whether the error function is differentiable, so some punishment may be added to the error function, so as to improve the network commonality, and reduce the complexity of the network.

5 EXPERIMENTAL RESULTS

We use the dataset acquired from the aquaculture bases in Shajiabang town of Changshu in recent five years to do experiments. The datasets are shown as Table 2. The numbers of the samples, attributes and classes of each dataset are showed. All algorithms are written by Matlab2009. Use 200 samples of the data set in each year to train and all the remains to test the neural network.

Table2. Experiment dataset

Dataset	Samples	Attributes	Classes
2011	300	5	8
2010	320	5	8
2009	310	5	8
2008	290	5	8
2007	310	5	8

The experiments preparation is conducted as follows.

(1) Normalize the original data.

(2) GA used real encoding. The length of the chromo is the total number of connection of the neural networks, the parameter settings are as follows:

Population size: 40

Crossover rate: 0.8

Mutation rate: 0.015

Crossover operator: Single-point crossover

Mutation operator: Gaussian mutation

Selection mechanism: Roulette wheel selection

Use the elitism strategy (reserve the 4 fittest individuals to the next generation).

Use linear fitness scaling.

(3) The neural network has only one hidden layer which has ten neurons, the transfer functions of the hidden layer and the output layer both are sigmoid functions. The learning rate of the BP algorithm is 0.15; the BP algorithm is without any improvement.

After training 6500 times, the corresponding average absolute errors (AAE) of training samples, examination samples and test samples are 0.1198, 0.1257 and 0.1176, respectively; average relative errors(ARE) are 3.14%, 3.21% and 4.76%, respectively; root mean square errors (RMSE) are 0.1542, 0.1556 and 0.151, respectively. The correlation coefficients are 0.9987, 0.9973 and 0.9980. These results show that the generated neural network described above not only achieves good fitting effect on training samples, but also have a high generalization and prediction effect on examination samples and test samples.

The traditional BP method in our dataset is compared with our method. The average AAE, ARE and RMSE of both methods are shown in Table3. From the table3, we can know that our method is better than the traditional BP method. The main reason is that the GA algorithm can achieve a better parameter values of the BP neural network.

Table3. Experiment Results

	AAE	ARE	RMSE
Our method	0.1210	3.70%	0.1536
Traditional BP Method	0.3105	6.89%	0.4127

6 CONCLUSION

The outbreak of water eutrophication is sudden, and the question arises the need for early warning model. This paper combines the BP neural networks with GA to construct the early warning model for water eutrophication. It shows an example of neural network in the model of water quality assessment, which contributes to the scientific evaluation of water eutrophication indexes in aquaculture waters. Through the early warning of water eutrophication, emergency response measures can be taken timely.

The study also helps the public administrators to find the problem of eutrophication during the aquaculture timely, therefore, it is beneficial to instruct the environmental treatments for local or regional scales.

ACKNOWLEDGMENTS

The authors would like to acknowledge the reviewers whose constructive comments helped to better focus this paper.This work is financially supported by the Natural Science Foundation of Jiangsu Province in China (No. SBK201240533).

REFERENCES

[1] ZENG W.H，SONG Q.L, LIU H.C.Research on ANN-based Pre-warning Water Bloom Model of LiuHai Lake in Beijing.Procedia Environmental Sciences ,2010: 625～635.

[2] Diaz, R.J., Rosenberg, R.,. Spreading dead zones and consequences for marine ecosystems. Science. 2008. 926–929.

[3] WANG H.P. The water bloom mechanism of hydrological factors in Han River. Resources and the Environment of Yangtze River Valley, 2004,13(3): 281-285.

[4] Jason B, Friedrich R. Knowledge discovery for prediction and explanation of blue-green algal dynamics in lakes by evolutionary algorithms. Ecological Modeling, 2001, 146: 253–262.

[5] Hugh W, Friedrich R. Towards a generic artificial neural network model for dynamic predictions of algal abundance in freshwater lakes.Ecological Modeling, 2001, 146: 69-84.

[6] James J. Fitzpatrick. Assessing skill of estuarine and coastal eutrophication models for water quality managers.Journal of Marine Systems.2009.76: 195–211.

[7] Chapra, S.C. Surface Water-Quality Modeling, McGraw-Hill Series in Water Resources and Environmental Engineering. McGraw-Hill, New York, New York. 1997.

[8] Arhonditsis, G.B., Brett, M.T.Evaluation of the current state of mechanistic aquatic biogeochemical modeling. Mar. Ecol. Prog. Ser. 271,2004.13–26.

[9] Lee J.H.W, Hodgkiss I.J, Wong K.T.M et al. Real time observations of coastal algal blooms by an early warning system. Estuarine, Coastal and Shelf Science, 2005, 65: 172-190

[10] Gilbert C S, Li W K, Kenneth M Y. Leung Modeling algal blooms using vector autoregressive model with exogenous variables and long memory filter. Ecological Modeling, 2007, 200: 130-138.

[11] Chen H.H. Control and management of water bloom in Australia. Environmental leader: 1995, 28(5): 32-33

[12] Lu D.Y.Investigation and study on the burst water bloom in Han River downstream. Environment scientific research,2000,13(2): 27-30

[13] Zeng Y, Yang Z.F, Liu J.L. Pre- warning model of urban lake water bloom With "LiuHai" in Beijing as an example. Water Science,2007,18(1):79-85.

[14] Kosten S, Huszar V.M, Becares E, et al. Warmer climates boost cyanobacterial dominance in shallow lakes. Global Change Biol 2012;18:118–26.

[15] Richardson TL, Lawrenz E, Pinckney JL.Spectral fluorometric characterization of phytoplankton community comparision using the algae online analyzer. Water Res 2010;44:2461–72.
[16] Moses WJ, Gitelson AA, Berdnikov S. Operational MERISbased NIR-red algorithms for estimating chlorophyll-a concentrations in coastal waters—the Azov Sea case study. Remote Sens Environ 2012;121:118–24.
[17] Chen S.Y.Use of Engineering Fuzzy Sets, BP Neural Network and Genetic Algorithm for Intelligent Decision-Making .The 6th World Congress onIntelligent Control and Automation, 2006:3052 - 3056 .
[18] Jin X.C. Survey and Research on Environments of Chinese Lakes (1980-1985) [M]. Beijing: China Environmental Science Press,1990: 145-152.
[19] Armida D.F, Giuliano A, Caterina G.Caco-2 cell permeability modelling: a neural network coupled genetic algorithm approach.Journal of Computer Aided Molecular Design, 2007(21)4: 207-221.
[20] Liu G.Z. Toxic Cyanobacteria in Water [M]. Beijing: China Environmental Science Press, 2005：127-129.
[21] Zhong G., Rao W.B, Zhou C.M. Artificial Neural Network and Its Application Technology [M]. Beijing: Science Press, 2007：132～168.
[22] Zhou M., Sun S.D. Genetic Algorithms: Theory and Applications [M]. Beijing: National Defense Industry Press, 2002:5-10.

A symbolic layer for autonomous component-based software agents

Razvan Dinu[1] and **Tiberiu Stratulat**[2] and **Jacques Ferber**[3]

Abstract. In order to handle complex situations, an autonomous agent needs multiple components ranging from simple input/output modules to sophisticated AI techniques. Integrating a high number of heterogeneous components is a non-trivial task and this paper discusses the use of a symbolic layer to address this issue. After an overview of existing techniques and their limitations this paper proposes a new approach through a generalized hyper-graph model in which the interaction of different components is modeled through a triggering mechanism based on patterns. Finally, the paper shows how a flexible symbolic middleware can be built and a few examples are presented.

1 Introduction

In order to keep up with the increasingly complex real-world problems, autonomous software agents need to integrate more and more components that range from simple input/output modules to sophisticated AI techniques. As the number of components increases the integration itself becomes an issue which unfortunately has been neglected until recent years. More and more researchers agree that *"the question about the inner workings of the pieces themselves holds equal importance to the question about the nature of the various dynamic glues that hold the pieces together"* [15].

When integrating multiple components, two levels of integration can be distinguished: *generic* and *specific*. The generic level is concerned with general mechanisms such as *how* components communicate with each other and *how* they exchange data. The specific level is concerned with the details of integrating components X_1, X_2, ..., X_n, of specific types, such as *when* component X_i calls a function of component X_j, *what* data should X_i provide, *when* should X_i send the data to X_j, etc.

Usually, in a running software agent, the generic level takes the form of a middleware and provides primitives for data and control flow to different specific levels. Such a middleware has to provide solutions to three main challenges: *communication*, *data sharing* and *global control*. The communication challenge is concerned with how different components can reach each other and how can they use each other's functionalities. The data sharing is concerned with how components can provide data (content) to other components. Global control is concerned with how all the interactions between components are handled and how a coherent global behaviour of the agent can be achieved.

One traditional technique for generic integration of multiple components is the *blackboard system* in which a set of experts, also called knowledge sources (KS), are constantly monitoring a blackboard searching for an opportunity to apply their expertize. Whenever they find sufficient information on the blackboard they apply their contribution and the process continues [6]. Unlike other techniques that implement formal models, the blackboard approach was designed to deal with ill-defined complex interactions. One of the first applications of the blackboard system was the speech understanding HEARSAY-II system [7] in which multiple components used a shared blackboard to create the required data structures.

Another generic integration technique is based on message passing and usually uses a publish-subscribe mechanism in which components subscribe to different types of messages and whenever a message arrives it is forwarded to corresponding modules. A message-based communication protocol for AI that has been gaining in popularity in recent years is the OpenAIR protocol managed by mindmakers.org [2].

CORBA is a well known standard by OMG [5], according to which components written in multiple computer languages and running on multiple computers are exposed as objects and their interaction is performed by method invocation. CORBA is very used as system integration in humanoid robotics, see for instance the simulator OpenHRP [10].

All of the above techniques provide more or less solutions to the three challenges mentioned earlier. Blackboards clearly provide means for data sharing, enables communication between components indirectly, but the control component is usually a simple scheduler and it does not help much in assuring a coherent global behaviour of the agent. On the other hand, message-passing focuses on communication and object-oriented techniques on communication and somewhat data sharing. Both leave global control entirely up to the interacting components.

Both improvements and hybrid solutions have been proposed for the above techniques. For example, whiteboards [16] consist of a blackboard with (i) a general-purpose message type, (ii) ontologically defined message and data stream types and (iii) specification for routing between system components. They also add an explicit temporal model thus providing more specialized solutions for communication and data sharing challenges. Also, the GECA Framework (Generic Embodied Conversational Agent) uses a hybrid solution in which multiple blackboards are used to perform message-passing based on message types [11].

Our opinion is that components integration would be much easier if we had an integration technique based on a more expressive data model and which provided better support for different *patterns of global control*.

By pattern of global control we understand the most abstract model that can be used to explain the behaviour of the software agent.

[1] University of Montpellier 2, LIRMM
[2] University of Montpellier 2, LIRMM
[3] University of Montpellier 2, LIRMM

A classical example of such a pattern, especially used in robotics, is the Brooks subsumption architecture [4]. In this approach components are structured into layers and those situated at higher levels are capable of altering the input and inhibiting the output of components at lower levels.

Another very widely used pattern of global control, especially in multi-agent systems, is BDI (Beliefs Desires Intentions) [14]. The software agent maintains a set of beliefs based on which desires are created. A desire which the agent has decided to pursue becomes an intention and a plan is chosen to achieve the desired goal.

More sophisticated patterns of global control come from the agent architectures domain. For example the INTERRAP agent architecture [13] uses three control layers: Behaviour-based layer, Local Planning layer and Cooperative Planning layer. Each layer has its own world model and includes subcomponents for situation recognition, planning and scheduling.

When we say that the generic integration middleware should support patterns of global control such as the ones mentioned above we are not saying that the patterns should be entirely implemented inside the middleware. But rather, the middleware should contain only part of the pattern and should smoothly integrate with components implementing key aspects of the control pattern (for instance a planning engine).

This paper focuses on the generic level of integration and proposes a middleware model that enables easier and more straightforward integration of different AI and non-AI components of an autonomous software agent. The next section introduces our approach and sections 3, 4 and 5 introduce our new symbolic model for generic integration and also perform a preliminary evaluation of its performance. Section 6 presents an implementation for smart phones based on the Android platform and finally, section 6 and 7 present our comments and conclusions.

2 Approach

As it has been outlined in the previous section, the main shortcomings for current approaches concern the *data sharing* and *global control* challenges. Our approach is an extension of the blackboard model which addresses exactly these two challenges.

2.1 Data sharing challenge

Firstly, we propose that the blackboard uses a more expressive symbolic data model rather than just isolated bits of typed data. The chosen symbolic structure is inspired by the generalized hyper-graph model proposed by [3]. Hyper-graphs generalize normal graphs by allowing an edge to contain more than two nodes and a directed hyper-graph considers edges as ordered sets (tuples). We are interested in a generalization of directed hyper-graphs in which an edge can contain both nodes and other edges. This represents the generalized hyper-graph model we're using and it will be described in more details in the next section. However, we will be using a different terminology that makes more sense in the context of symbolic representations: *symbols* instead of nodes and *links* instead of hyper-edges.

Secondly, we extend the generalized hyper-graph structure with a *map* which associates each symbol of the hyper-graph with another symbol. Finally, we allow each symbol to have some attached information, which can be typed or not.

A generalized hyper-graph, the information associated with the symbols and the map of symbols form a SLiM structure (Symbol Link Map). From now on, we will use the capital version (SLiM) to refer to the model and the lower letter version (*slim*) as a shorthand for "SLiM structure" which refers to a concrete structure.

One related work which uses a hyper-graph model close to ours is [12]. They use a directed hyper-graph and integrate a typing system in which a node has a handle, a type, a value and a target set. The main differences in our model are the lack of the typing system and the addition of the symbolic map which, as it will be shown in future sections, can be used to create a typing system. However, they show how such a hyper-graph structure can be efficiently implemented and used as central database especially in AI applications. The OpenCog project [9] is also an illustrative example of hyper-graphs usage in AI projects. These works show the increasing interest of using the flexible hyper-graphs structures in AI.

2.2 Global control challenge

In order to address the issue of *global control* we inspired ourselves from the patternist philosophy of mind whose main premise is "the mind is made of patterns". In this perspective a mind is a collection of patterns associated with persistent dynamical processes capable of achieving different goals in an environment. For a quick overview of the patternist philosophy of mind and also a different way of applying it in the context of AI we recommend [8].

We define a pattern as a particular type of slim and we show how a set of patterns can be efficiently matched using an automaton. Next, we propose an interaction mechanism between components based on patterns that uses a central SLiM structure which can by accessed and modified by any component. Each component can register two types of patterns: data patterns and capability patterns. Whenever a component modifies the central slim and a data pattern is found then the corresponding component is notified. Also, whenever a component requests the execution of something that matches a capability pattern then the corresponding component is notified.

As it will be detailed in the following section all these mechanisms provide a very flexible way of performing interaction between different components of a software agent and they can be packed into a symbolic middleware which can be used in conjunction with other agent frameworks.

3 The SLiM Model

This section formally introduces the SLiM model and also proposes a representation language to represent a slim.

3.1 Formal definition

Definition 1. Let S be a finite set of elements. T_S is the set of all tuples over S and it is inductively defined as:

- $T_0 = \{(0, \emptyset)\}$.
- $T_k = \{X \cup \{(k, s)\} | X \in T_{k-1}, s \in S\}$ for $k \geq 1$
- $T_S = \cup_{k=0}^{\infty} T_k$

The sole element of T_0 is called the empty tuple and will be denoted simply by $\emptyset$. An element $t \in T_k$ is called a tuple of length k. Instead of $t = \{(0, \emptyset), (1, s_1), ..., (k, s_k)\}$ we use the equivalent notation $t = (s_1, s_2, ..., s_k)$. We also use the notation $s \in t$ to mean $\exists j \geq 1$ such that $(j, s) \in t$.

Definition 2. Let the following:

i. S be a finite set of elements called *symbols*.

ii. l: $S \rightarrow T_S$ be a function called a *linking* function on S.
iii. i: $S \rightarrow I$ be a function called an *information* function on S, where I is a set of elements.
iv. m: $S' \rightarrow S$, where $S' \subset S$, be a partial function on S called a *map* on S.

Then the quadruple $< S, l, i, m >$ is called a SLiM structure or simply a *slim*. The elements of $s \in S$ for which $l(s) \neq \emptyset$ are also called *links* and if $x, y \in S$ and $m(x) = y$ we say that x is mapped to y.

Below are a few terminological definitions associated with the SLiM model.

Definition 3. A symbol $d \in S$ is *reachable* from a symbol $s \in S$ if and only if there exist $s_1, s_2, ..., s_n \in S$ such that $s_1 \in l(s), s_2 \in l(s_1), ..., s_n \in l(s_{n-1})$ and $d \in l(s_n)$.

Definition 4. A symbol $d \in S$ is *mappable* from a symbol $s \in S$ if and only if d is equal to s or there exist $s_1, s_2, ..., s_n \in S$ such that $m(s) = s_1, m(s_1) = s_2, ..., m(s_{n-1}) = s_n$ and $m(s_n) = d$.

Definition 5. A tuple $(s_1, s_2, ..., s_n) \in T_S$ is called an *implied link* if and only if there exist $x_1, x_2, ..., x_n, y \in S$ such that x_1 is mappable from s_1, ..., x_n is mappable from s_n and $l(y) = (x_1, x_2, ..., x_n)$. If $x_i = s_i$ for $i = 1, n$ then the link is called *explicit*.

Definition 6. A slim $< S, l, i, m >$ is called *acyclic* if and only if no symbol can be reached from itself.

3.2 Representation language

Before going any further we will introduce an abstract syntax for a representation language, called the *SLiM language*, that can be used to describe a slim. Given S and I the sets of symbols and information elements, the language is given by the following EBNF:

```
   slim  →  symbol+                          (1)
 symbol  →  [id|link][:[info|symbol]]?       (2)
   link  →  [id=]?{symbol+}                  (3)
     id  →  s ∈ S                            (4)
   info  →  x ∈ I                            (5)
```

(the curly brackets are part of the terminal alphabet of the SLiM language)

Before giving a few examples we will provide the semantics of the production rules. In order to do that we consider that the non-terminal nodes of the grammar `symbol`, `link` and `id` have a synthesized attribute "s" which holds the corresponding symbol $s \in S$:

Production rule			**Attribute rule**
symbol	→	id[...]?	symbol.s = id.s
symbol	→	link[...]?	symbol.s = link.s
link	→	id={symbol+}	link.s = id.s
link	→	{symbol+}	link.s = *use/new*
id	→	$s \in S$	id.s = s

The *use/new* keyword means that if there is already a link corresponding to the sequence of symbols on the right side then the attribute `link.s` uses the same id, otherwise it gets a random id from S not used by any other production rule. Below we will give the semantics of each of the right sides of production rules 2 and 3.

Right side	**Semantics**
{s_1 ... s_n}	$l(use/new) = (s_1.s, ..., s_n.s)$
id={s_1 ... s_n}	$l(id.s) = (s_1.s, ..., s_n.s)$
[id|link]:symbol	$m($[id|link].s$) =$symbol.s
[id|link]:info	$i($[id|link].s$) =$info

Here's an example of a slim described using the SLiM language:

```
here={my location}          (1)
here:{city Lyon}            (2)
{user said msg:"Hello"}     (3)
```

Let $< S, l, i, m >$ be the slim described in the above example. The symbols set is S = { `my`, `location`, `here`, `city`, `Lyon`, `user`, `said`, `msg`, `rand1`, `rand2` } and the information set is I = { `null`, `"Hello"` }. The first line creates a link between the symbols `my` and `location` and assigns the id `here`, which means $l($ `here`$) =$ (`my`,`location`). The second line creates a link between `city` and `Lyon` whose id is not important (and we can consider it to be `rand1` $\in S$) and maps the symbol `here` to it. This means $l($ `rand1`$) =$ (`city`,`Lyon`) and that $m($ `here`$) =$`rand1`. The third line creates a link between other three symbols and assigns some information to the last one, $i($ `msg`$) =$`"Hello"`. All other symbols $s \in S$ have $i(s) =$`null`.

The meaning of the first production rule is the union of all the symbols, information and mappings defined by the `symbol` production rules. We say that a SLiM representation (a string in the SLiM language) is *valid* if and only if there are no contradictions (i.e. a symbol being assigned two different information, a symbol being mapped to different symbols, multiple definitions of a link, etc.). From now on, we will use the SLiM language rather than the formal definition to describe SLiM structures.

One last observation concerns the use of curly brackets to create links. The language does not use parentheses or square brackets in order to avoid confusion with languages such as LISP or REBOL.

3.3 Modeling with SLiM

The SLiM model does not impose any particular semantics on the data being represented in a slim. It is a semi-structured, general purpose model based on a generalized hyper-graph structure. The actual schema that will be used in a slim will evolve dynamically and data integrity constraints can be enforced by different components using the slim. This type of flexible models is especially suitable for online environments.

4 Patterns

As it was discussed in section 2, the pattern concept is central to our approach. Below we will explain what a pattern is, how can multiple patterns be matched, and we perform a simple performance evaluation of the proposed matching mechanism.

4.1 Definition

Definition 7. `?` and `@` are two special symbols called the *any* and the *root* symbols.

Definition 8. A *pattern* is an acyclic slim $< S, l, i, m >$ with the following properties:

i. `?` $\in S$ and `@` $\in S$;
ii. `@` is mapped to a symbol, which is called the *root of the pattern*, and all other mappings, if any, are to `?`;

iii. the symbols mapped to `?` are called *generic* symbols and they are all reachable from the root of the pattern;
iv. it contains no information ($I = \{\emptyset\}$).

Before discussing the different properties from the above definition we will provide two examples, described using the SLiM language, so that the reader can have a better grasp of what patterns look like. Each pattern has been enclosed inside an additional set of curly brackets:

```
{
    {sound enabled}
    @: { notify user Message:? }
}
{
    online
    @: {User:? wants {listen album Album:?} }
    {User allowed music}
}
```

Figure 1. "Examples of patterns"

First of all, we are only interested in patterns at the symbolic level, disregarding the information attached to symbols, hence property iv). Secondly, the SLiM model is intended to represent data mainly through symbols and links and that is why patterns will be used to describe only parts of the symbolic hyper-graph. As a consequence, a pattern contains only mappings that have special meaning to the matching mechanism.

In a few words, a pattern is a generic way of describing a set of symbols and links. A pattern can contain regular symbols or *generic* symbols (those mapped to `?`) which act as place holders for regular symbols. For convenience, we can omit the "`@:`" for simple patterns. Also, we can write directly {`listen album ?`} if the name of the generic symbol is not relevant when presenting a pattern.

Intuitively the first example in figure 1 can be matched for example by the following slim:

```
some-message: "Hello User!"
  {sound enabled}
  {notify user some-message}
```

The generic symbol `Message` is a place holder for the `some-message` symbol. The link {`sound enabled`} exists as it is.

In order to explain the role of the `@:` mapping we have to define exactly what is a match for a pattern.

4.2 Matching a pattern

Definition 9. A slim M is a match for a pattern P if and only if it can be obtained from P by:

1. replacing all mappings to `?` with mappings to other non-generic, possibly new, symbols;
2. replacing the occurrences of all generic symbols in links with the symbol they're mapped to;
3. removing the mapping from `@` and the symbols `?` and `@`.

Definition 10. Let X be a slim, P be a pattern and M be a match of P. We say that M is a match of P in X if and only:

i. every symbol in M which is not generic in P also exists in X;
ii. every link in M is implied in X.

For example, the following slim (M):

```
User: current-user
Album: s32
online
{current-user wants {listen album s32} }
{current-user allowed music}
```

is a match for the second example pattern in the following slim (X):

```
current-user: John
{s32 author Michael-Jackson}
{s32 title s41:"Bad"}
online
{current-user wants {listen album s32} }
{John allowed music}
```

We can easily see that the symbols such as `online`, `wants`, `music`, etc. exist directly in X and so do links such as {`listen album s32`} . On the other hand the link {`current-user allowed music`} is only implied in X (by definition 5) because `current-user` is mapped to `John` and we have the link {`John allowed music`}.

One important aspect of matching a pattern is the fact that the links described by the pattern must be implied in the slim we're searching, and not necessarily exist. This gives a lot of flexibility when modeling data we slim. We can choose to leave some links implicit and still be able to match them in patterns. A more in-depth discussion of implied links would be suitable but due to space limitation we will leave it to the reader to imagine how implied links can be used.

Searching a set of symbols and links that match a pattern in a slim can be a very time consuming task especially when generic symbols are used in more than one link. It is the equivalent of a join operation on a relational database which requires special indexing in order to be processed efficiently. That is why we divide a pattern in two, the root of the pattern and the rest, and we impose that the generic symbols are reachable from the root.

If the root is a link and some of the symbols inside the link are also links and so on, we have an ordered tree[4] of symbols determined by the root of the pattern. We will refer to this tree as *the tree of the pattern* (figure 2 shows the tree for the second example pattern).

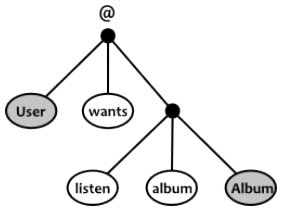


Figure 2. Example of "tree of a pattern"

Since all the generic symbols are reachable from the root, in order to find a match for a pattern we first have to find a match for the tree of the pattern. If one is found, all the other symbols and links can be easily checked since there are no other generic symbols. But even

[4] a tree in which the order of the sons is important.

finding a match for a tree in a slim can take a very long time so we will limit our approach to a particular case which will be used in our symbolic middleware presented in the next section.

The pattern matching problem we are addressing is the following: *given a symbol in a slim and a pattern tree, can it be matched starting at the given symbol (the root of the instance tree will correspond to the given symbol)*?

We will represent a pattern tree using the simplified notation (no `@` and no generic symbols names). For example, for the second example pattern we get {`? wants {listen album ?}` } . This will be called the normal string representation of a pattern tree. So, the question is, given such a representation and a symbol (which can be a link) in a slim, can the tree be matched with the given symbol as root? This can actually be done in liner time by reading the tree pattern representation from left to right and by performing a depth-first navigation of the symbol in parallel. The algorithm is left as an exercice to the reader.

4.3 Summary

In order to conclude this section we will summarize the main idea: we have a set of patterns and a symbol in a slim and we are interested to see if there's a pattern whose tree can be matched with its root at the given symbol; if so, then the rest of the pattern can be easily checked once we have the values of all the generic symbols in the pattern; if all other symbols and links exist then we have found a match for a pattern.

5 SLiM based integration middleware

Now we will show how the symbolic SLiM module introduced in the previous section can be used in what we call *symbolic integration middleware*.

5.1 Symbolic middleware

We consider a software agent as having two parts: a *crown* which contains many different components and a *trunk* which contains one or more middlewares. The only way components can interact is through a middleware situated in the trunk. The SLiM middleware, which is the proposed solution for components integration, is composed of:

- A *slim* which acts as shared blackboard. Every component is able to create new symbols, modify links, attach information to symbols, change mappings, etc.
- A *capabilities index* and a *triggers index*. They are two sets of patterns together with two automata capable of matching them.
- A *behaviour rules* set. A rule is an association between a trigger pattern and an *entry-slim* (a slim with a designated symbol called *entry point*).

Every component can register one or more capability or trigger patterns with the SLiM middleware. For example a text-to-speech component can register the capability pattern {`speak ? english`}. A natural language processing component can register the trigger pattern {`user said ?`}.

In order to illustrate the role of each type of pattern or rule we will explain the different modes in which the SLiM middleware can be used.

***tell* mode**. In this mode a component can access the shared slim and perform whatever changes it needs. For example in this mode the speech recognition module would add {`user said msg:'Hello world!'`} .

***do* mode**. In this mode a component asks the SLiM middleware to do something by providing an entry-slim. The middleware will try to match the given slim, starting from the entry point, by a capability pattern. If one is found then the component that registered the pattern will be notified and it will be provided the match. If no pattern is found and the entry point is a link then all the symbols in the link will be successively used as entry points, creating a sequence of *do*-s. For example a component can request {{`wait 5 seconds`} {`speak msg:'Hello you too!' english`}} to be done.

***trigger* mode**. Whenever a symbol is created or a link made, the triggers automaton will try to match it to an existing trigger pattern. If one is found and it is associated with a component then the component will be notified. However if it is associated with an entry-slim, through a behaviour rule, a *do* will be requested on the entry-slim. For example the rule {`user said ?`} → {`log to history ?`} logs what the user said to a history.

***ask* mode**. This mode is not discussed in this paper but it allows a module to query a slim structure in the same manner as [12].

Other details such as how symbols from a trigger pattern match are used in the entry-slim of a behavior rule, details of the *do* mode or the role of mappings, which some readers might consider important, are not discussed in this paper.

Let's take now the two example patterns provided in previous section (figure 1). The first one could be registered by a component capable of producing some sounds, perhaps depending on the type of the message. However, the pattern also contains the link {`sound enabled`} . We can imagine a convention like {`sound enabled`} when components can produce sounds and {`sound disabled`} when they can't (the volume is set to 0). So, if a component wants to send a notification to the user it will request a *do* on the following slim {`notify user msg12:"New email!"`}. At that point the capability index will try to match the given slim, starting from its entry point which is the whole link, and will find the example pattern 1 as being a match, the corresponding component will be notified and for example the user will hear a beep.

The second pattern is more complicated and it can correspond to a component capable of playing albums from internet for example. But this pattern will be registered as a trigger pattern and not a capability pattern. For example a speech recognition component combined with a natural language processing module can recognize that the user wants to listen to an album and it will create a link stating that fact. When the link is created the trigger automaton will analyze the link and if the user is online and is allowed to play music than a match is found and the player component gets notified. If no component would have registered the second example pattern than the link created by the natural language processing module would have no effect and maybe it will be removed by a component that deletes links unused for a certain amount of time.

These examples should give the reader an idea of how the interactions between components with different functions will happen by using the proposed SLiM middleware.

6 Test implementation

The described SLiM middleware has been implemented and tested on a mobile device using the Android platform [1] which uses a message-based integration mechanism.

The implemented application asks the user the name of a city and then searches a predefined list of hotels. By showing the user a few

options, and by integrating a simple clustering algorithm, the application is able to learn progressively which part of the city the user is interested in.

We had to integrate components already existing in the Android platform such as the text-to-speech engine, speech recognition, internet browsing and map view. For each of them we have created a wrapper that exposed the capabilities of each component through appropriate patterns such as {`show on map ?`} which looked for a link {`? address`} (where `?` is the same in both patterns) and then showed the address using the available map view.

The application behaved correctly and the components integration was very smooth.

7 Comments and limitations

We believe that integration of many of components is the key to making software agents smarter and make humans think of them as autonomous agents with which they can interact. Different works in AI and other connected domains are situated at different levels of abstraction and an integration middleware has to be able to deal with it. Also, the data that can be shared between different components is very diverse and an integration middleware would have to use a data model capable of handling this diversity. We believe the SLiM middleware, through the use of a very expressive data model and a flexible pattern matching mechanism, is a first step towards that.

One important limitation of the SLiM middleware comes from the fact that each created link or symbol has to be processed by the triggers index. This basically creates a bottleneck which means that slim updates cannot be performed at very high rates (on the android platform the maximum rate, as tested, is at about 1200 updates per second). This makes it not suitable for components that need interactions between them at a high rate. In that case, additional middlewares capable of handling such interactions should be used in the trunk together with the SLiM middleware. Alternatively, pattern checking can be performed in parallel on multiple cores which would also give better performance.

One interesting aspect of SLiM which has not been mentioned earlier is the fact that the SLiM language can be used as a scripting language for the interaction of components. We can have a default module which implements patterns for the usual control constructs we find in a scripting language (i.e. {`if Cond:? then Action: ?`}, {`for X:? in List:? do Action: ?`}, etc.). Together with the patterns registered by different components we will actually end up with a kind of domain specific language whose primitives are dictated by the capabilities of the agent.

Having a symbolic middleware like SLiM implemented in multiple agents, even developed by third parties, would create a more solid base for an ACL (Agent Communication Language). We can have the constructs of the language translated into symbolic representations which would then be executed by an agent using the registered components at a given time.

As it can be seen, having an explicit symbolic middleware in a software agent has many advantages and it opens very interesting perspectives.

8 Conclusions and future works

This paper proposes the integration of multiple components in a software agent through the use of a symbolic middleware based on the new SLiM model. Due to the expressivity of hyper-graphs and the flexibility of the proposed pattern matching mechanism this model is well suited for the integration of AI with non-AI components.

We provided a clear formal description of the SLiM model, a representation language that can be used to describe slims and a pattern matching mechanism. Based on them we proposed a model of a symbolic middleware which uses a slim at its core and two types of patterns, capability and trigger patterns.

The efficiency of the pattern matching algorithm and its small memory footprint show that the SLiM integration middleware is well suited for mobile platforms such as Android on which a test implementation was done.

The proposed approach combines the advantages of multiple generic integration techniques: a) the flexibility of a shared blackboard with an expressive hyper-graph based data model; b) a triggering mechanism based on patterns which generalizes the publisher-subscribe paradigm; c) *do* mechanism based on patterns similar to method invocation; d) straightforward integration through behaviour rules based on patterns.

Also, the use of an explicit symbolic middleware such as the SLiM middleware can lead to interesting application such as a component integration scripting language, easier implementation of agent communication languages and even the creation of lightweight or distributed agents.

REFERENCES

[1] Android. http://www.android.com., 2010.

[2] OpenAIR. www.mindmakers.org, 2011.

[3] H. Boley. Directed recursive labelnode hypergraphs: A new representation-language. *Artificial Intelligence*, 9(1):49 – 85, 1977.

[4] R. A. Brooks. Intelligence without representation. *Artificial Intelligence*, 47(1-3):139–159, 1991.

[5] h. CORBA, 2011.

[6] R. Engelmore and A. Morgan, editors. *Blackboard Systems*. Addison-Wesley, 1988.

[7] L. D. Erman and V. R. Lesser. The HEARSAY-II speech understanding system: Integrating knowledge to resolve uncertainty. *Computing Surveys*, 12:213–253, 1980.

[8] B. Goertzel. Patterns, hypergraphs and embodied general intelligence. *International Joint Conference on Neural Networks*, pages 451 – 458, 2006.

[9] B. Goertzel, H. de Garis, C. Pennachin, N. Geisweiller, S. Araujo, J. Pitt, S. Chen, R. Lian, M. Jiang, Y. Yang, and D. Huang. OpenCogBot: Achieving generally intelligent virtual agent control and humanoid robotics via cognitive synergy. *ICAI*, 2010.

[10] H. Hirukawa, F. Kanehiro, and S. Kajita. OpenHRP: Open architecture humanoid robotics platform. In R. Jarvis and A. Zelinsky, editors, *Robotics Research*, volume 6 of *Springer Tracts in Advanced Robotics*, pages 99–112. Springer Berlin / Heidelberg, 2003.

[11] H.-H. Huang, A. Cerekovic, I. Pandzic, Y. Nakano, and T. Nishida. Scripting human-agent interactions in a generic ECA framework. In *Applications and Innovations in Intelligent Systems XV*, pages 103–115. Springer London, 2008.

[12] B. Iordanov. HyperGraphDB: A generalized graph database. *First International Workshop on Graph Database*, 2010.

[13] J. Muller. The agent architecture INTERRRAP. In *The Design of Intelligent Agents*, volume 1177 of *LNCS*, pages 45–123. Springer Berlin / Heidelberg, 1996.

[14] A. S. Rao and M. P. Georgeff. BDI-agents: from theory to practice. In *Proceedings of the First Intl. Conference on Multiagent Systems*, San Francisco, 1995.

[15] K. Thrisson. Integrated A.I. systems. *Minds and Machines*, 17:11–25, 2007.

[16] K. R. Thrisson, T. List, C. Pennock, and J. Dipirro. Whiteboards: Scheduling blackboards for semantic routing of messages & streams. In *AAAI-05 Workshop on Modular Construction of Human-Like Intelligence*, pages 8–15, 2005.

An Architecture For Multi Dimensional Temporal Abstraction Supporting Decision Making In Oil-Well Drilling

Odd Erik Gundersen [1,2] and **Frode Sørmo** [3]

Abstract. We have developed an online decision support system that advice drilling engineers online based on data streams of real-time rig site measurements. This is achieved by combining multi dimensional abstraction for recognizing symptoms, fuzzy rule-like smart alarms and case based reasoning. Case-based reasoning compares the current situation in the well with past situatons stored in the case base that contains advices for how to solve similar problems. Some results of a commercial test are presented in a case story.

1 Introduction

Data is updated continuously in streams in a multitude of domains. Typical examples include health care in which patients are monitored continuously, stock market trading in which stock prices are recorded over time and aviation control. In this paper we will focus on the oil well drilling domain. Currently, over 3,800 oil-well drilling rigs are operating world-wide [3], and real-time data is collected from around 75% of these according to domain experts. Examples of parameters that are measured in real-time are stand-pipe pressure, total amount of mud pumped through the drill-string, drill-string rotations per minute and torque. Although a vast amount of data is produced, it is not used to its full extent. This data is primarily used for visual display, statistical analysis of performance and threshold alarming, and there exists a few systems that update physical models in real-time to get a better overview of the drilling environment [22], but these are not in wide-spread use. Monitoring is usually done manually by drilling engineers on the rig site, or in real-time operation centers that gather data from multiple rigs to a central location [6]. Typically, drilling engineers interpret these data continuously in shifts lasting up to 12 hours by staring at parameter graphs. Identifying situational events like symptoms of developing problems by manually inspecting data graphs is error prone, as it is a tiring task and it is hard to keep all previously identified symptoms in mind at all times.

Problematic oil-well drilling situations typically develop over time with symptoms that are recognizable by drilling experts. However, the symptoms behaves differently based on the context of the operation, like for instance with depth or the drilling tools used. Thus, threshold alarms, which are known to produce many false positive alarms , do not work well as users tend to ignore them if they go off often without any reason. An analysis of the data from operations in context of the actual operation will alleviate the drilling engineers and can shift the cognitive load from identifying situational elements to predicting the future status of the drilling situation and thus increasing the situational awareness helping drilling engineers to making better decisions [10].

Our main goal is to develop an online decision support system that provides relevant information to drilling engineers during oil well drilling operations using data streams of real-time rig site measurements and case-based reasoning. Case-based reasoning (CBR) is a methodology for reusing past experiences by comparing them to a current problem and then solves the current problem using the past experiences [2]. Experiences are stored in a case base as cases comprised of a problem description and a problem solution. A new experience is compared to the past cases in the case base. Then, the problem solutions of the most similar past cases are used to solve new experience. If no applicable cases are stored in the case base, the new experience will be retained with the proper solution, which promotes learning.

In this paper we present an architecture that supports the main goal. The main component in the architecture is an agent system that coordinates a set of agents in which each agent has a specific task related to enhance the understanding of the current situation. Tasks include recognizing different aspects of the current operation, such as which activity is performed and symptoms of problems, in addition to projecting the future based on the recognized symptoms. In order to recognize different aspects of the operation multi dimensional temporal abstraction is performed while CBR is utilized to project the future state of the operation and provide support in selecting the most appropriate remedial actions. Temporal abstraction is a term used when transferring data stream over time from a low level quantitative form to a high level qualitative form incorporating domain knowledge and context information [21], and hence avoiding simplistic threshold alarms that do not take sound reasons for parameters to change into account.

The architecture is implemented in DrillEdge, a commercial decision support system that is used by several large oil companies. In DrillEdge, different aspects of the current situation of the oil well drilling operation are visualized to alleviate the cognitive load of the drilling engineers monitoring the situation. Past problematic oil-well drilling situations and experiences related to how the drilling problem could be avoided are captured as cases, which are brought to the attention of the user when the current situation becomes similar to a past situation.

The rest of this paper is structured as follows. Some related work will be investigated in section 2. The architecture of the system will be presented in depth in section 3. In section 4 selected results are

[1] Norwegian University of Science and Technology, Trondheim, Norway, email: odderik@idi.ntnu.no
[2] Verdande Technology, Norway, email: odderik@verdandetechnology.com
[3] Verdande Technology, Norway, email: frode@verdandetechnology.com

presented, and finally in section 5 we will conclude with some final remarks and future work.

2 Related Work

Cately et al. discuss trends and challenges of multi dimensional temporal abstractions and data mining of medical time series in [7]. While they restrict their discussion to medical time series, several of the trends apply to the domain of oil well drilling as well. They identify six trends. Multi dimensional correlations between parameters exist in the oil well drilling domain, but while down hole tools record high frequency data, these are not communicated to the surface in real-time and can therefore not be used when supporting decisions made by humans. Another trend is that medical data mining systems and temporal abstractions should be applied to real-world clinical data. This clearly applies to real-world oil well drilling data as it often contains faulty readings and not only some occasional data points, but for long periods of time as the measuring tools are in though conditions. Null-hypothesis testing of data mining methods will not be required until data mining methods are used more widespread in the oil industry. The authors argue that data mining methods should be used to explore complex patterns that might hide parameter correlations from the users, which is true in the oil domain too. Data mining algorithms will push the knowledge bases towards synthesized knowledge bases, which will apply to the oil well drilling domain when data mining methods get more widespread. Finally, they identify a need for automatically transferring knowledge between data mining and temporal abstraction mechanisms, which is what we seek to achieve with the combination of pattern matching, fuzzy rules and CBR.

In [20], Stacey et al presents an architecture for multi dimensional temporal abstraction and its application to support neonatal intensive care. The goal was to go beyond threshold alarms causing many false positive alarms by introducing domain knowledge in form of an ontology and rules. The architecture was based on previously performed research in business event monitoring and performance measurement [14]. However, while the previous work focused on correlating workflow events, the neonatal intensive care research performs temporal abstraction on low level quantitative data from patient data streams logging parameters like blood pressure and oxygen saturation.

The temporal abstraction is performed by Patient Agents that execute rules on the patient data streams and fire alerts (events) if the conditions of the rules apply. Rules can be made manually by domain experts like doctors or mined automatically by an Analytical Processor. Such rules are supported as Smart Alarms in DrillEdge. However, because of the complexity of symptoms, rules will generate too many false alarms and thus we use Graph and Symptom Agents that are based on heuristic mathematical models designed by pattern matching experts in cooperation with oil-well drilling experts.

The alerts that are fired if a rule applies are received by an active ontology which triggers an alarm to the users of the system. The ontology is a central knowledge base which stores the rules and agent responses and allows temporal abstraction across multiple patient's data. A similar learning of patterns across operations is found in DrillEdge by introducing new cases representing new situations and experiences in the case base. While the active ontology performs this cross-correlating automatically, cases in Drilledge must be captured manually. Creek [1], a knowledge intensive CBR system in which ontologies are an integral part, is the roots of DrillEdge. Despite of this, the only remnants of the ontological reasoning is the ontology itself, which is used only for quality assuring the terminology used by the system.

Montani et al. present an architecture in which a CBR system configures and processes temporal abstractions produced from raw time-series data in [15, 18]. The problem description part of cases capture the context knowledge about the time series interpretation while the problem solution part configure temporal abstraction trends and states that are used to monitor the time-series data. So, the CBR system configures the temporal abstractions using context information in order to monitor and evaluate patients undergoing hemodialysis to provide medical personnel with an assessment of the situation. In comparison, DrillEdge provide drilling engineers with an assessment of the current situation of an oil-well drilling operation by relating the temporal abstractions (events) with the context.

Another domain with large amounts of time-series data is in the financial market. Barbosa and Belo present an architecture for currency trading in [4], which will forecast the direction of the currency value the next six hours, decide how much to trade for and whether to trade. The architecture has three parts: An agent system which forecasts the direction of several currencies, a CBR system that decides how much to trade for based on previous experiences and a rule-based system that decides whether to trade or not. Each agent in the agent-based system forecasts the value of one currency by using an ensemble of data mining algorithms. The data mining algorithms perform the temporal abstractions which is the input to the CBR system. Then based on how coherent the data mining algorithms was in the decision that the agent made, the CBR system decides how much to trade. Feedback on the accuracy of the forecasting is used provided to the CBR system to learn whether to trust the decision in similar situations later. Thus, the system is autonomous and can learn from its own experiences in contrast to DrillEdge which requires manual input of new cases. The architecture was later used to implement an agent task force for stock trading [5].

3 The Architecture

From a process view, the architecture has four layers, which are: 1) *Data acquisition*, 2) *Data interpretation*, 3) *Decision support* and 4) *Visualization*. The architecture is designed with online decision support and modularity in mind. In a decision support system, the reasons for providing a recommendation must be transparent for a user to trust it. The main advantage of case-based reasoning systems are that they provide the user with actual experiences, and this is why they are a good fit for decision support systems. When the situation becomes very similar to the situation experienced on the Macondo rig hours before the catastrophic accident, you get the attention of the users if you point this out rather than if you say that the situation is highly severe. Also, cases provide better means of explanation than rule-based systems, as the case can provide reasons for why it has been stored in the case base and what the learnings that should be taken from it are. The system is modular in that the system can grow with both symptoms and experiences without having to change what is already learned by the system. New agents recognizing new symptoms can be easily added and so can new cases. Also, the architecture can easily scale to monitor many operations concurrently.

3.1 Data Flow

Figure 1 shows how data flows through the system to the user. The data is gathered mainly from *Input Data Sources* such as streaming XML sources that use the oil well drilling industry standard called

WITSML and the operators setting up and configuring operations that are to be monitored by DrillEdge. The data is read an stored in the system as *Unified Data* that can be accessed and stored to by the *Data Analysis Agents*. A *Situational Description* that describes the current state in the oil well is generated and compared by the *Case-Based Reasoning* engine to past situations stored in the case base. The most similar cases are *Visualized* to the Drilling Engineer on the Case Radar as color coded dots according to their severity.

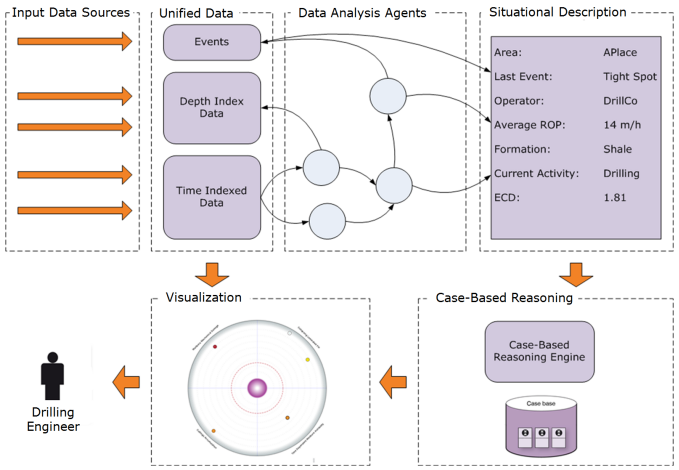


Figure 1. Data flow oriented representation of the DrillEdge architecture.

3.2 Agent Platform

The agent platform serves as an interface between the data and the agents in addition to control the execution of the agents. Agents depend both on the data streams and each other, and they are executed at every time step which is defined by the arrival of new data. Agents are executed in such a way that dependencies are maintained - i.e. an agent that depends on the output of another agent is always executed after the agent it depends on.

This agent architecture differs from the typical agent architecture found in the agent literature [16]. First, they do not communicate through a meta-level agent communication language like FIPA ACL [17] or KQML [11], but through a common data structure. Second, agent coordination is performed through a predefined partial ordering of dependencies based on requirements specified when coding the agents, and not through a multi-agent coordination strategy like negotiation [13]. Thirdly, the agents are not autonomous in the way that they run in separate threads or machines, but have their execution controlled by the agent platform. However, the agents are autonomous in that they control themselves when to generate a new symptom; they do not necessarily produce new values each time step.

Hence the architecture is similar to a blackboard system [8] in which the agents act as *Knowledge Sources* that communicate indirectly and anonymously through shared data structures (the time and depth tables) that act as the *Blackboard*. The agent platform resembles the *Control Component* that controls which agents to execute. It is common, however, that agent systems not necessarily are comprised of fully autonomously agents, as is shown in section 2, Related Work.

There exists three different categories of data analysis agents in DrillEdge, and these are *Graph agents*, *Symptom agents* and the *CBR agent*. All types of agents can read and use all data available to the system to perform their tasks. Some agents only use parameter values stored in the time or depth tables while others can use context information too. The agents and an example of dependence relations are illustrated in figure 2.

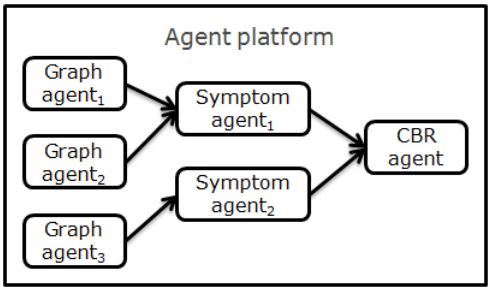


Figure 2. The agent platform showing the three different agent types: Graph agents, Symptom agents and the CBR agent.

Graph Agents: In general, graph agents produce a new data value for every time step and store these in dedicated columns in the depth or time tables. Graph agents are not necessarily dependent on any other agents, but some are. Mainly they produce values that are required by other agents to work properly. Some agents quality ensure parameters that are produced by others. These agents do not produce a completely new parameter, but ensures that calculated parameters are consistent and do not contain clearly erroneous values.

Symptoms agents: Symptom agents perform the multi dimensional temporal abstraction by monitoring a set of predefined parameters and generate events if symptoms of problems are detected. An example of a symptom is when gravel starts packing off around the drill-bit. In the worst case, this can lead to the drill bit getting stuck in the ground. For example, the symptom agent recognizing such pack-off tendencies perform multi dimensional temporal abstraction on several parameters, which are the amount of mud pumped in to the drill-string, standpipe pressure, torque and the weight on the drill-bit. In addition, it relies on context parameters like the dimension of the wellbore, casing depth and whether a mud motor is used when drilling. It is also dependent on the output from two graph agents, one that identifies whether the drill-bit is in the open hole or in the casing and the agent that recognizes which drilling activity is performed. Both the agents that the pack-off symptom agent is dependent on use several parameters in their calculations too, so clearly symptom detection is multi dimensional temporal abstractions.

Algorithm 1 lists a simplified pseudo code for the *PackOff* symptom agent, which generates a new pack off event at time t and depth d if the mudflow has been stable while the pressure has increased the last 60 seconds. The parameters $\vec{m}$ and $\vec{p}$ are vectors containing all values over the last 60 seconds of the mudflow and pressure parameters respectively. The methods $isStable()$ and $isIncreasing()$ can be implemented using regression analysis on moving windows of values that, for example, contains all values of a parameter over the last 60 seconds, such as $\vec{m}$ and $\vec{p}$.

As events are observed at a given time and depth in the well, both of the time and depth are stored in an event when one is generated along with a severity and the event type. Events serve as input to

Algorithm 1 Pack Off Symptom Agent

$d \leftarrow getCurrentDepth()$
$t \leftarrow getCurrentTime()$
$t_0 \leftarrow t - 60s$
$\vec{m} \leftarrow mudflow(t_0, t)$
$\vec{p} \leftarrow pressure(t_0, t)$
if $isStable(\vec{m}) \wedge isIncreasing(\vec{p})$ **then**
    return PackOff(t,d)
end if

the CBR agent, but they are also drawn in the depth and time view so that they can be viewed together with the parameters used to generate them. This is done to enhance the situation awareness of the drilling engineers monitoring the oil-well drilling operation in addition to provide transparency to the system.

CBR agent: The CBR agent contains a CBR system and captures the current situation continuously and compares it to cases stored in the case base. The current situation is represented by both events and contextual information. Events are stored in the case as depth and time sequences sorted on the distance from the drill-bit and distance from the current time respectively. Problematic situations typically occur when symptoms cluster together – either on depth or in time or both. This is why the cases contain sequences of events for both time and depth that are compared using sequence similarity measures [19]. Context information describing the circumstances that the symptoms occurred in is also important. Typically, some geological formations causes more problems than others.

The CBR agent searches for and retrieves cases from the case base and compares them to the current case. The comparison result is sorted on similarity. All past cases with a degree of similarity above a given threshold are visualized on a GUI element, the Case Radar, to alert and advice the user of past historic cases that are similar to the current situation. The closer a case is to the center of the radar, the more similar it is to the current situation. The edge of the radar is 50% similar so all cases on the radar are more than fifty percent similar to the current situation. By investigating the similar situations the user is advised on what others did and what should have been done in similar situations in the past. A new case is made by the drilling engineer if the current situation is not covered by any cases stored in the case base or if other advices apply to this situation. The system learns when new cases are added to the case base. The result of the case comparison is written by a graph agent to the time table so that the users can monitor the behavior of cases on the time view.

Figure 3 shows a screen capture from the DrillEdge Client that visualizes the current state of an oil well drilling operation. The deph column with the drill-bit at the bottom of the wellbore and pack-off events distributed along the drill string is showed to the left, and the radar is depicted with two cases indicating a stuck pipe situation to the right. The sectors in the radar indicate different problem types that are covered by the cases in the case base. If a case enters the radar, the current situation is getting similar to the situation captured in the past case. Thus it can be inferred that the current situation develops into a similar problematic situation and mitigating actions must be taken. The closer a case is to the center of the radar, the more similar the current situation is to the past situation.

3.3 Case-Based Reasoning Cycle

Figure 4 illustrates the CBR process in DrillEdge. Real-time data (1) is interpreted by graph and symptom agents and events are generated (2). Events together with other data are captured to form the input

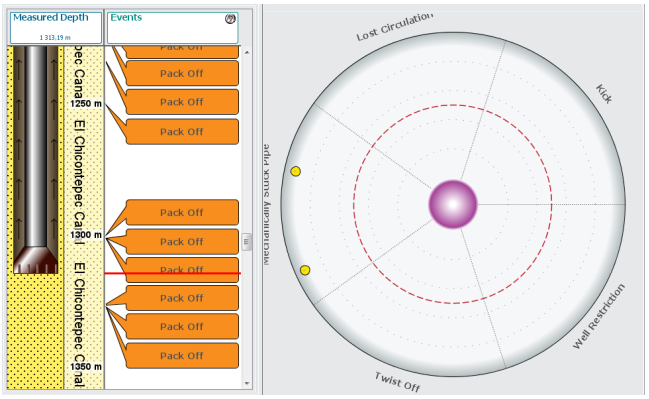


Figure 3. Screen capture from the DrillEdge client showing the depth column and the Radar.

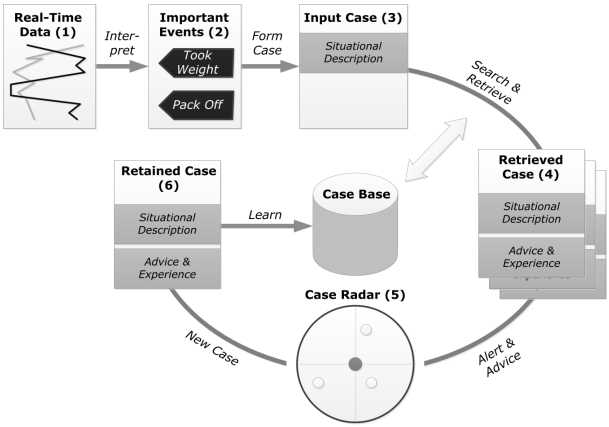


Figure 4. Case-Based Reasoning Cycle: From real-time data to a retained case.

cases specifying the current situation (3). The case base is searched and past cases are retrieved from the case base (4). The past cases are plotted on the Case Radar to alert and advice the users (5). New cases can be captured and the retained case (6) is learned when stored in the case base.

Capturing and learning a new case is not done in real-time as quite a lot of experience is required to capture proper cases, so this is not a task that can be performed by any drilling engineer monitoring an oil well. Lots of man hours are used on quality assurance of whether cases match well in similar situations as well as the actual advices that are given in the case.

3.4 Alarm Center

A Smart Alarm is another way to perform temporal abstraction on the streaming time-series data. Smart Alarms can be compared to fuzzy rules that react to one or more parameters. As cases, smart alarms are plotted on the radar. Smart alarms will behave more predictable than cases as the designer of the smart alarm will specify the limits for when the smart alarm will appear in the radar and when it will be in the center. The smart alarms are administered by the CBR agent and

can be viewed as synthetic cases designed for very local needs.

In the simplest form a smart alarm can react to the proximity of the hole to a formation at a target depth. This can be valuable information to the drilling engineers as certain problems can be expected when reaching certain formations. A more advanced smart alarm can appear on the radar when a formation is expected to be reached in a given time interval. For example, the smart alarm will appear on the radar when it is less than one hour left until the formation is reached. Another type is one that appears on the radar if a parameter gets above a lower limit and reaches the center at a upper limit in a given formation.

Algorithm 2 illustrates a smart alarm that fires when the drill bit is in shale and the weight on bit (WOB) parameter is inside predefined lower and upper limits. The method will put the smart alarm on the rim of the radar if the parameter is right inside the lower limit and on the center if the parameter is at the upper limit. *getRadarThreshold()* returns the current radar threshold, which can be set dynamically, but currently is 50%, while *getCurrentFormation()* returns the formation at the current depth of the drill-bit. *getWeightOnBit()* returns the current value of the weight of bit parameter, while WOBInShale has an upper limit and a lower limit, which are returned by *getUpperLimit()* and *getUpperLimit()* respectively. Smart alarms have been used by several customers, and some results from a test with Marathon Oil Company are documented in [12].

Algorithm 2 Smart Alarm: Weight on Bit Limits in Shale

$t \leftarrow getRadarThreshold()$
$f \leftarrow getCurrentFormation()$
$w \leftarrow getWeightOnBit()$
$u \leftarrow getUpperLimit(WOBInShale)$
$l \leftarrow getLowerLimit(WOBInShale)$
if $isShale(f) \wedge l \leq w \wedge w \leq u$ **then**
    return $t + (1 - t) \cdot (w - l)/(u - l)$
else
    return 0
end if

3.5 Client Server Architecture

The DrillEdge server is a distributed system that can be deployed on server farms like the Amazon Elastic Computing Cloud (ECC) or VMWare clusters as well as large physical servers. The client is a Java application that is installed from a web page using Java Web Start. All data interpretation is done on the server cluster, so the main task of the client is to visualize the information analyzed by the server.

The server is implemented as a set of services, and some services, like the license and management services, provide administrative functions and only one of each of these are required in the cluster. The license service ensures that the cluster runs with a valid license while the management service enables power users to set up and configure operations and administrators to administrate users. The operation service analyses oil-well drilling operations and communicates with DrillEdge clients through a front end. Each oil-well drilling operation monitored by the system has its own dedicated operation service.

Figure 5 illustrates the client server architecture. The box stippled with gray lines contain a server cluster deployed on the Amazon ECC while the gray boxes illustrate physical or virtual machines that run services. Hence, the server cluster is comprised of four machines in

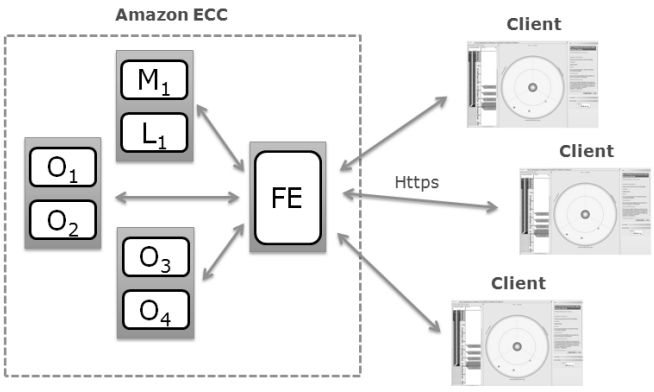


Figure 5. The client server architecture: *O* indicates a operation service while *L* and *M* refers to license and management services respectively. *FE* is the front end and the subscripts indicate different services. The gray, stippled line confines the server cluster and the gray boxes indicate physical machines on the Amazon ECC.

which two of them are running four operations, one machine is running the management and license server while the last one runs the front end service that communicates with the clients through HTTPS.

3.6 Operation Service

The main task of an operation service is to analyze the data streams from one concrete oil well drilling rig that has been assigned to the service. These data streams are measured on the rig, then aggregated and made available through a web service-based industry standard called WITSML [9]. The operation service is responsible for the data acquisition, and it has to communicate the results to the clients that users monitor the operations through. The agent platform is the most important part of the operation service.

Operation services are set up and configured by operators, which typically are drilling engineers responsible of ensuring that the service is set up correctly. The time streams update frequency ranges from 1 second to 1 minute. However, the lower the data rate, the less detailed patterns can be recognized. DrillEdge achieves best results when the range between data points is shorter than ten seconds, as internal tests have shown that the results degrade when longer. Depth data is measured by down hole tools, which collect measurements at high frequencies that are sampled and communicated to the surface at a rate of up to 40 bits per second. In addition to the streamed measurement data from the rig, the system requires context information that can be added to the operation service by operators. Context information includes static parameters like rig type and the configuration of the drilling tools, and the expected geology that is to be drilled through. When setting up the operation both the case base and alarm base will have to be configured by the operators too.

4 Results

DrillEdge monitored the first live well during the summer of 2008 as part of the conclusion of the research project it was developed as part of. During 2011 around 15 pilot projects were finished, partly by analyzing historical data and monitoring live wells. In December 2011, Petroleum Development Oman (PDO) started monitoring live wells

using DrillEdge, and in early January 2012 DrillEdge was running on more than 30 commercial wells concurrently. The current maximum number of operations monitored concurrently is 41, and it was reached in late February 2012 and continued for about a week.

Shell has run several tests both on historical and live data and deployed DrillEdge commercially in mid 2011. Shell experienced problems of twisting off the drill-pipe while drilling, and requested a solution that could predict twist off problems in advance. Twisting off the drill-string is a costly problem that often requires side tracking in order to avoid the parts of the drill-string that is left in the hole. Verdande Technology analyzed the data and found that long periods of maxing out the torque while drilling wore out the drill-string so that it finally twisted off. A symptom agent was developed to recognize when the torque was maxed out, and several cases were captured from historical data.

The solution was tested in a blind test using test data from a US land well while five cases was build from Middle East data. A total of 31 Maxed Out Torque events were fired, which resulted in three of the five Middle East cases appeared on the radar before the drill-string twisted off. The first case appeared on the radar two days before the twist off, potentially giving Shell long time to react to the problem. Cases were even captured from the US land blind test data and used to analyze the Middle East twist offs. Similar results were achieved. Other twist off tests were performed with varying results, but overall the tests were good enough for Shell to deploy DrillEdge commercially. In addition to the twist off tests, the stuck pipe solution was also put under pressure, but predicted stuck pipe six hours in advance. A detailed summary of the tests can be found in [23]. DrillEdge has been documented to accelerate learning in Shell [24].

5 Conclusion and Future Work

We have presented an architecture for multi temporal abstraction that supports decision making in oil well drilling. The architecture scales well and is shown to predict problematic situations hours in advance. For this, Verdande Technology was awarded the Meritous Award for Engineering Excellence for its DrillEdge software platform by E&P Magazine in March 2011.

We will research how data mining techniques can be applied to detect symptoms of problematic drilling situations so that we do not need to rely on manually designed heuristic mathematical models. In order to do this, we need to be able to track event accuracy without human interference. Currently, we are implementing a system for quality ensuring event recognition automatically. Also, we will investigate how to combine a rule-based system with the case-based reasoning system in order to let domain experts control the case matching to a higher degree. Another task is to move in a more knowledge intensive direction and use the capabilities of the ontology as a part of the reasoning process. Finally, we would like to investigate ways to automatically identify where to capture cases and thus avoiding time consuming manual work.

ACKNOWLEDGEMENTS

This research could not have been done without the fabulous teams at Verdande Technology. Some of you must be thanked specifically: Sigve Hovda, Tore Brede, Christian Schjølberg, and Martin Stige. Also, we would like to thank professor Agnar Aamodt for valuable comments and suggestions.

REFERENCES

[1] A. Aamodt, 'Knowledge-Intensive Case-Based Reasoning in CREEK', in *ECCBR*, eds., P. Funk and P.A González-Calero, volume 3155 of *Lecture Notes in Computer Science*, pp. 1–15. Springer, (2004).

[2] A. Aamodt and P. Plaza, 'Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches', *AI Communications*, **7**(1), 39–59, (1994).

[3] Baker Hughes, 'Baker Hughes Investor Relations: Overview and FAQ', (January 2012).

[4] R. Barbosa and O. Belo, 'Multi-Agent Forex Trading System', in *Agent and Multi-agent Technology for Internet and Enterprise Systems*, 91–118, (2010).

[5] R. Barbosa and O. Belo, 'An Agent Task Force for Stock Trading', in *PAAMS*, pp. 287–297, (2011).

[6] J.E. Booth, 'Real-Time Drilling Operations Centers: A History of Functionality and Organizational Purpose–The Second Generation', *SPE Drill & Compl*, **26**(2), 295–302, (2011).

[7] C. Catley, H. Stratti, and C. McGregor, 'Multi-dimensional temporal abstraction and data mining of medical time series data: Trends and challenges', in *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, pp. 4322 –4325, (aug. 2008).

[8] Daniel D Corkill, 'Collaborating Software: Blackboard and Multi-Agent Systems and the Future', in *Proceedings of the International Lisp Conference*, New York, New York, (October 2003).

[9] N.R. Deeks and Halland T., 'WITSML Changing the Face of Real-Time', in *Intelligent Energy Conference and Exhibition*, (2008).

[10] Mica R. Endsley, 'Toward a theory of situation awareness in dynamic systems', *Human Factors: The Journal of the Human Factors and Ergonomics Society*, **37**, 32–64(33), (1995).

[11] Tim Finin, Richard Fritzson, Don McKay, and Robin McEntire, 'KQML as an agent communication language', in *Proceedings of the third international conference on Information and knowledge management*, CIKM '94, pp. 456–463, New York, NY, USA, (1994). ACM.

[12] J. F. Jones, S. Tucker, and T. Sheehy, 'Well Test Highlights Benefits of Real-Time Infrastructure in Land Drilling Operations', *The American Oil & Gas Reporter*, (January 2011).

[13] V. R. Lesser, 'Reflections on the Nature of Multi-Agent Coordination and Its Implications for an Agent Architecture', *Autonomous Agents and Multi-Agent Systems*, **1**, 89–111, (1998). 10.1023/A:1010046623013.

[14] C. McGregor and J. Schiefer, 'A Web-Service based framework for analyzing and measuring business performance', *Inf. Syst. E-Business Management*, **2**(1), 89–110, (2004).

[15] S. Montani, A. Bottrighi, G. Leonardi, and L. Portinale, 'A CBR-Based, Closed-Loop Architecture for Temporal Abstractions Configuration', *Computational Intelligence*, **25**(3), 235–249, (2009).

[16] H. S. Nwana, 'Software Agents: An Overview', *Knowledge Engineering Review*, **11**, 1–40, (September 1996).

[17] P. D. O'Brien and R. C. Nicol, 'FIPA -Towards a Standard for Software Agents', *BT Technology Journal*, **16**, 51–59, (July 1998).

[18] L. Portinale, S. Montani, A. Bottrighi, G. Leonardi, and J. M. Juárez, 'A Case-Based Architecture for Temporal Abstraction Configuration and Processing', in *ICTAI*, pp. 667–676, (2006).

[19] M.M. Richter, 'Similarity', in *Case-Based Reasoning on Images and Signals*, ed., P. Perner, volume 73 of *Studies in Computational Intelligence*, 25–90, Springer, (2008).

[20] M. Stacey, C. McGregor, and M. Tracy, 'An architecture for multi-dimensional temporal abstraction and its application to support neonatal intensive care', in *Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE*, pp. 3752 –3756, (aug. 2007).

[21] A. Stein, M. A. Musen, and Y. Shahar, 'Knowledge acquisition for temporal abstraction', in *Proceedings of AMIA Annu Fall Symp*.

[22] S. Strø m, M. K. Balov, H. Kjø rholt, R Gaasø, E. Vefring, and R. Rommetveit, 'The Future Drilling Scenario', in *Offshore Technology Conference*.

[23] E. van Oort and K. Brady, 'Case-based reasoning system predicts twist-off in Louisiana well based on Mideast analog', *World Oil*, (April 2011).

[24] E. van Oort, J. Griffith, and B. Shneider, 'How to Accelerate Drilling Learning Curves', in *SPE/IADC Drilling Conference and Exhibition*, (2011).

A Neuro-Fuzzy Network Approach to Impulse Noise Filtering

Yueyang Li[1] and **Haichi Luo**[1,*] and **Jun Sun**[1]

Abstract. A neuro-fuzzy network approach to impulse noise filtering for gray scale images is presented. The network is constructed by combining four neuro-fuzzy filters with a postprocessor. Each neuro-fuzzy filter is a first order Sugeno type fuzzy inference system with 4-inputs and 1-output. The proposed impulse noise filter consists of two modes of operation, namely, training and testing (filtering). As demonstrated by the experimental results, the proposed filter not only has the ability of noise attenuation but also possesses desirable capability of details preservation. It significantly outperforms other conventional filters.

1 INTRODUCTION

Images are often corrupted by noise during the acquisition or transmission process. So noise cancellation/filtering is an important task in image processing, especially when the final product is used for edge detection, image segmentation, and data compression.

Image signals are composed of flat regional parts and abrupt changing areas, such as edges, which carry important information in visual perception. In the case of corruption by impulse noise, nonlinear techniques seem to perform better than linear ones, which tend to blur the edges and degrade the lines, edges, and other fine image details. So a great majority of filtering methods for the removal of impulse noise from images are based on median filtering techniques. The *standard median filter* (SMF) [1] is a simple nonlinear operation that outputs a median value of the pixels in the predefined filtering window to replace the center pixel of the window. The *weighted median filter* (WMF) [2] and the *center weighted median filter* (CWMF) [3] are extensions of the median filter, which give more weight to the appropriate pixels within the filtering window. These filters are spatially invariant operators, so they inevitably distort the uncorrupted pixels in image while restoring the corrupted pixels.

In the case of impulse noise removal, the aim of optimal filtering is to design noise reduction algorithms that would affect only corrupted image pixels, whereas the undistorted image pixels should be invariant under the filtering operation. So a number of algorithms, such as the *edge-detecting median filter* (EDMF) [4], the *progressive switching median filter* (PSMF) [5], the *multi-state median filter* (MSMF) [6], and the *signal-dependent rand-ordered mean filter* (SDROMF) [7], have been proposed to combine the median filter with a decision mechanism which attempts to determine whether the center pixel of a detecting window is corrupted or not. If the central pixel of the window is found to be corrupted by noise, it is replaced by the median filter output; otherwise, it is unchanged.

In addition to the conventional filters discussed above, a number of filtering methods based on neural networks and fuzzy systems have been proposed. Fuzzy systems are fundamentally well suited to model the uncertainty that occurs when both noise cancellation and detail preservation are required. The *fuzzy filter* (FF) [8] is presented which adopts a fuzzy logic approach for the enhancement of images corrupted by impulse noise. On the other hand, artificial neural networks have the ability to learn from examples. Therefore, neuro-fuzzy systems combining neural networks and fuzzy set theories can be employed as powerful tools for the removal of impulse noise from digital images [9, 10].

In this paper, we propose a neuro-fuzzy (NF) network approach to impulse noise filtering for gray scale images. The network is constructed by combining four NF filters with a postprocessor. Each NF filter is a *first order Sugeno type fuzzy inference system* [11] with 4-inputs and 1-output. Each NF filter evaluates a different relation between the median value of the pixels in a predefined window and the three appropriate pixels. The proposed impulse noise filter consists of two modes of operation, namely, training and testing (filtering). During training, each NF filter is trained individually and the internal parameters of each NF filter are adaptively optimized by training by using a simple artificial training image. During testing, the outputs of the four NF filters are fed to the postprocessor, which generates the final output. As demonstrated by the experimental results, the proposed filter not only has the ability of noise attenuation but also possesses desirable capability of details preservation. It is compared with several conventional impulse noise filters. Simulation results show that the proposed filter significantly outperforms other conventional filters.

2 ALGORITHM

2.1 The neuro-fuzzy network

Figure 1 shows the structure of the proposed NF network. The network is constructed by combining four NF filters with a postprocessor. Each of the four NF filters is a *first order Sugeno type fuzzy inference system* with 4-inputs and 1-output.

As to each pixel of the input image, the four inputs of each NF filter, x_1, x_2, x_3 and x_4, are obtained by employing selecting data blocks and can be described as follows:

[1] Key Laboratory of Advanced Process Control for Light Industry, Jiangnan University, Wuxi, China, email: jnamandaluo@gmail.com

1. Three pixels, p_1, p_2, and p_3, are obtained in 3-by-3 pixel filtering window centered around the current operating pixel shown in Figure 2. Four different pixel neighborhood topologies are employed which correspond to four selecting data blocks shown in Figure 1.
2. The median value m is obtained from all pixels in a predefined window centered around the current operating pixel.
3. The four inputs of the NF filter can be defined as follows:

$$\begin{aligned} x_1 &= p_1 - m \\ x_2 &= p_2 - m \\ x_3 &= p_3 - m \\ x_4 &= m \end{aligned} \tag{1}$$

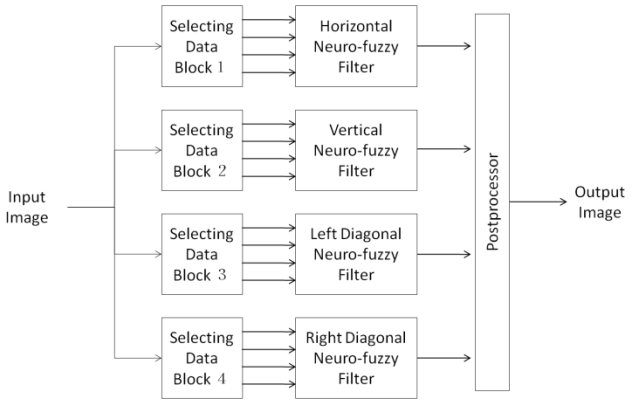


Figure 1. Structure of the neuro-fuzzy network

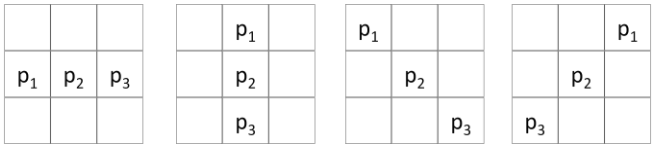


Figure 2. Four pixel neighborhood topologies: (a) Horizontal direction; (b) Vertical direction; (c) Left diagonal direction; (d) Right diagonal direction.

2.2 The neuro-fuzzy filter

Each of the four NF filters is a *first order Sugeno type fuzzy inference system* with 4-inputs and 1-output. The internal structures of the four NF filters are identical to each other. Each input has three *generalized bell* type membership functions and the output has a linear membership function. Since the NF filter has 4 inputs and each input has 3 membership functions, the rule base contains a total of 81 (4^3) rules, which are as follows:

Rule 1: if (x_1 is M_{11}) and (x_2 is M_{21}) and (x_3 is M_{31}) and (x_4 is M_{41}) then $y_1 = d_{11}\,x_1 + d_{12}\,x_2 + d_{13}\,x_3 + d_{14}\,x_4 + d_{15}$

Rule 2: if (x_1 is M_{11}) and (x_2 is M_{21}) and (x_3 is M_{31}) and (x_4 is M_{42}) then $y_2 = d_{21}\,x_1 + d_{22}\,x_2 + d_{23}\,x_3 + d_{24}\,x_4 + d_{25}$

Rule 3: if (x_1 is M_{11}) and (x_2 is M_{21}) and (x_3 is M_{31}) and (x_4 is M_{43}) then $y_3 = d_{31}\,x_1 + d_{32}\,x_2 + d_{33}\,x_3 + d_{34}\,x_4 + d_{35}$

Rule 4: if (x_1 is M_{11}) and (x_2 is M_{21}) and (x_3 is M_{32}) and (x_4 is M_{41}) then $y_4 = d_{41}\,x_1 + d_{42}\,x_2 + d_{43}\,x_3 + d_{44}\,x_4 + d_{45}$

Rule 5: if (x_1 is M_{11}) and (x_2 is M_{21}) and (x_3 is M_{32}) and (x_4 is M_{42}) then $y_5 = d_{51}\,x_1 + d_{52}\,x_2 + d_{53}\,x_3 + d_{54}\,x_4 + d_{55}$

Rule 6: if (x_1 is M_{11}) and (x_2 is M_{21}) and (x_3 is M_{32}) and (x_4 is M_{43}) then $y_6 = d_{61}\,x_1 + d_{62}\,x_2 + d_{63}\,x_3 + d_{64}\,x_4 + d_{65}$

⋮

Rule 81: if (x_1 is M_{13}) and (x_2 is M_{23}) and (x_3 is M_{33}) and (x_4 is M_{43}) then $y_{81} = d_{81,1}\,x_1 + d_{81,2}\,x_2 + d_{81,3}\,x_3 + d_{81,4}\,x_4 + d_{81,5}$

where M_{ij} denotes the jth membership function of the ith input, y_k denotes the output of the k th rule, i = 1, 2, 3, 4, j = 1, 2, 3, k = 1, …, 81.

The input membership functions are *generalized bell* type:

$$M_{ij}(x_i) = \frac{1}{1 + \left| \dfrac{x_i - c_{ij}}{a_{ij}} \right|^{2b_{ij}}} \tag{2}$$

The output Y of each NF filter is the weighted average of the individual rule outputs y_k. The weighting factor w_k of each rule is the multiplication of four input membership values. Hence, the weighting factors w_1, …, w_{81} of the rules, and the output Y of each NF filter, can be calculated as follows:

$$\begin{aligned} w_1 &= M_{11}(x_1) \times M_{21}(x_2) \times M_{31}(x_3) \times M_{41}(x_4) \\ w_2 &= M_{11}(x_1) \times M_{21}(x_2) \times M_{31}(x_3) \times M_{42}(x_4) \\ w_3 &= M_{11}(x_1) \times M_{21}(x_2) \times M_{31}(x_3) \times M_{43}(x_4) \\ w_4 &= M_{11}(x_1) \times M_{21}(x_2) \times M_{32}(x_3) \times M_{41}(x_4) \\ w_5 &= M_{11}(x_1) \times M_{21}(x_2) \times M_{32}(x_3) \times M_{42}(x_4) \\ w_6 &= M_{11}(x_1) \times M_{21}(x_2) \times M_{32}(x_3) \times M_{43}(x_4) \\ &\vdots \\ w_{81} &= M_{13}(x_1) \times M_{23}(x_2) \times M_{33}(x_3) \times M_{43}(x_4) \end{aligned} \tag{3}$$

$$Y = \frac{\sum_{k=1}^{81} w_k y_k}{\sum_{k=1}^{81} w_k} \tag{4}$$

The outputs of the four NF filters are fed to a postprocessor, which calculates the average value of the four NF filters. Then the final output of the NF network is obtained.

2.3 Training procedure

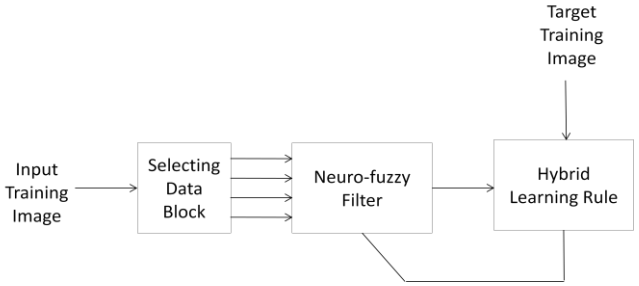


Figure 3. Structure of training of an individual neuro-fuzzy filter.

(a) (b)

Figure 4. Artificial training images: (a) Original image (Target training image in Figure 3); (b) Impulse noise image (Input training image in Figure 3).

The four NF filters operate on the same 3-by-3 pixel filtering window and they are trained individually. The internal parameters of each NF filter are adaptively optimized by training by using a simple artificial training image. Figure 3 shows the structure of training of an individual NF filter.

The images shown in Figure 4 (a) and Figure 4 (b) are employed as the target and the input training images, respectively. The original image (target training image) is a 64-by-64 pixel image that is generated in a computer [9]. The impulse noise image (input training image) is obtained by corrupting the original image by impulse noise of 30% noise density.

The internal parameters of NF filter are optimized by using the *hybrid learning rule* [11] to reduce the error. The antecedent parameters are optimized by using the gradient descent algorithm while the consequent parameters are solved by the least squares algorithm.

2.4 Testing (Filtering) procedure

Once the training of the four NF filters are completed, they are combined with the postprocessor to construct the NF network shown in Figure 1. In the testing (filtering) procedure, the restoration of the noisy input image can be described as follows:

1. As to each pixel of the input image, a 3-by-3 pixel filtering window centered around the current operating pixel is obtained. The four inputs of each NF filter are obtained by employing a corresponding selecting data block.
2. Each NF filter individually generates an output value for the current operating pixel and the four outputs of the NF filters are fed to the postprocessor, which calculates the average value of the four NF filters. The average value is thus the final output of the NF network and represents the restored value of the corresponding pixel in the output image.
3. The filtering procedure is repeated until all the pixels of the noisy input image are restored. Then the output image shown in Figure 1 is the restored image.

3 EXPERIMENTAL RESULTS

Extensive experimental results are reported in this section to demonstrate the performance of the proposed impulse noise filter. The Baboon image with size 256×256 are tested and it is corrupted by impulse noise ranging from 3% to 80%. Both quantitative and qualitative evaluations have been presented to demonstrate the superior filtering performances of proposed filter.

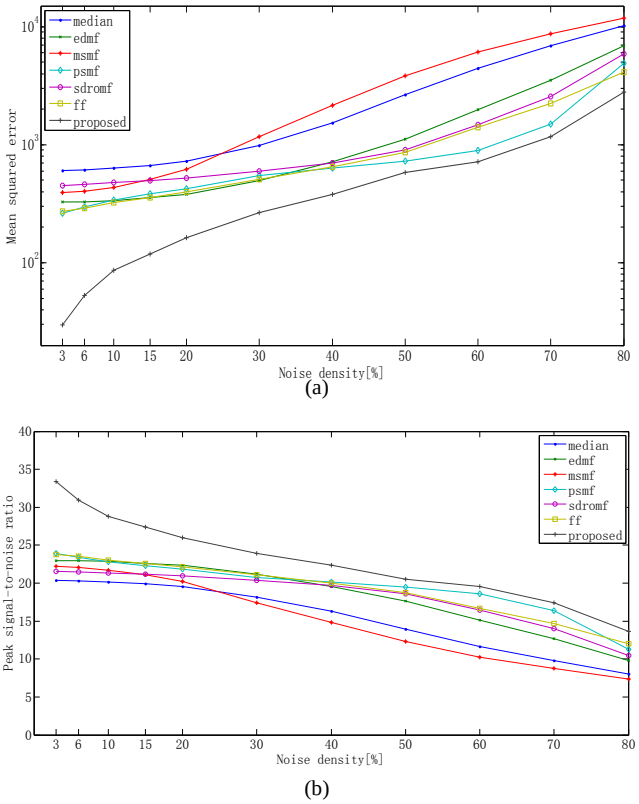


Figure 5. Filtering performance of the proposed filter compared with the conventional filters. The testing image Baboon is corrupted by impulse noise ranging from 3% to 80%. (a) Mean squared error (MSE); (b) Peak signal-to-noise ratio (PSNR).

Two quantitative measures, *mean squared error* (MSE) and *peak signal-to-noise ratio* (PSNR), are used to assess the filtering performance. Note that the smaller value for MSE and the bigger value for PSNR mean better filtering performance. The performance of the proposed filter is compared with that of the other conventional filters, namely the *standard median filter* (SMF) [1], the *edge-detecting median filter* (EDMF) [4], the *progressive switching median filter* (PSMF) [5], the *multi-state median filter* (MSMF) [6], the *signal-dependent rand-ordered mean filter* (SDROMF) [7], and the *fuzzy filter* (FF) [8]. The results are reported in Figure 5. The proposed filter, in terms of the two quantitative measures, clearly outperforms the conventional filters.

In addition to the quantitative evaluation presented above, a qualitative evaluation is necessary since the visual assessment of the processed images is ultimately the best subjective measure of the effectiveness of any method. Figure 6 depicts the superior filtering results of the proposed filter for the Baboon image corrupted by 40% impulsive noise in comparison with the other conventional filters. Figure 6(i) shows the restored image using proposed filter. It is obvious that the proposed filter has the excellent ability of noise attenuation because there is almost no impulse noise left. Compared with the restored images using other conventional filters, the proposed filter also has the desirable capability of details preservation as indicated on the beard part of the Baboon image.

From the experimental results reported above, it can be easily seen that the proposed new filter provides significantly good

results in the testing image corrupted by different percentages of impulse noise, and outperforms the other conventional filters under consideration.

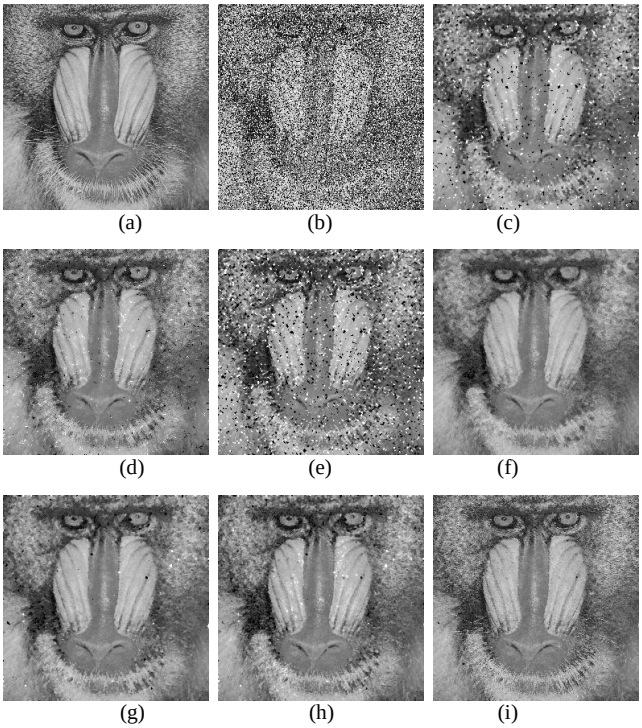


Figure 6. Filtering results for the testing image Baboon. (a) Original image. (b) Image corrupted by 40% impulse noise. (c) Restored image using SMF. (d) Restored image using EDMF. (e) Restored image using MSMF. (f) Restored image using PSMF. (g) Restored image using SDROMF. (h) Restored image using FF. (i) Restored image using proposed filter.

4 CONCLUSION

The paper presents a novel impulse noise filter for gray scale images based on a neuro-fuzzy network. The network is constructed by combining four NF filters with a postprocessor. As supported by the simulation results, the proposed filter compares favorably with conventional techniques in the capabilities of noise attenuation and details preservation, in both quantitative and qualitative measures.

ACKNOWLEDGEMENTS

This work is supported by the 111 Project (B12018) and the Fundamental Research Funds for the Central Universities (No.JUSRP211A38).

REFERENCES

[1] Pratt, W.K., *Digital Image Processing*, New York: Wiley Interscience, 1978.

[2] Yli-Harja, O., J. Astola, and Y. Neuvo, 'Analysis of the properties of median and weighted median filters using threshold logic and stack filter representation', *Signal Processing, IEEE Transactions on*, **39**(2), 395-410, (1991).

[3] Ko, S.J. and Y.H. Lee, 'Center weighted median filters and their applications to image enhancement', *Circuits and Systems, IEEE Transactions on*, **38**(9), 984-993, (1991).

[4] Shuqun, Z. and M.A. Karim, 'A new impulse detector for switching median filters', *Signal Processing Letters, IEEE*, 9(11), 360-363, (2002).

[5] Zhou, W. and D. Zhang, 'Progressive switching median filter for the removal of impulse noise from highly corrupted images', *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, 46(1), 78-80, (1999).

[6] Tao, C. and W. Hong Ren, 'Space variant median filters for the restoration of impulse noise corrupted images', *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, 48(8), 784-789, (2001).

[7] Abreu, E., et al., 'A new efficient approach for the removal of impulse noise from highly corrupted images', *Image Processing, IEEE Transactions on*, 5(6), 1012-1025, (1996).

[8] Russo, F. and G. Ramponi, 'A fuzzy filter for images corrupted by impulse noise', *Signal Processing Letters, IEEE*, 3(6), 168-170, (1996).

[9] Yuksel, M.E. and A. Basturk, 'A simple generalized neuro-fuzzy operator for efficient removal of impulse noise from highly corrupted digital images', *AEU - International Journal of Electronics and Communications*, 59(1), 1-7, (2005).

[10] Li, Y., F.-L. Chung, and S. Wang, 'A robust neuro-fuzzy network approach to impulse noise filtering for color images', *Applied Soft Computing*, 8(2), 872-884, (2008).

[11] Jang, J.-S.R. and C.-T. Sun, *Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence*, Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1997.

Representing Research Activities in a Hierarchical Ontology

Susana Nascimento[1] and **Trevor Fenner**[2] and **Boris Mirkin**[3]

Abstract. The present work is motivated by the problem of analyzing the activities of an organization and representing them in a taxonomy of the domain as its hierarchical ontology. We focus on representing the research activities of a Computer Science research organization in terms of the ACM-CCS taxonomy. We derive research topic clusters according to the similarity derived on the basis of profiling of the topics covered by the researchers working in the organization and then place them onto the hierarchical tree. Each of the steps is performed by using our original algorithms: one-by-one spectral-additive fuzzy clustering SAF and a recursive algorithm PARL for mapping fuzzy clusters to higher ranks of the taxonomy. The latter minimizes the weighted sum of penalties for the chosen "head subjects", and corresponding "gaps" and "offshoots".

1 INTRODUCTION

Ontology is a convenient concept to handle the semantic contents of data computationally. Additionally, ontologies permit to build more interoperable systems due to the possibility of reusing and sharing knowledge. A popular ontology structure is a taxonomy (an "is a" or "part of" hierarchy). Usually, a taxonomy provides a consensual conceptualization of a given domain [1]. Many ontologies – most of them being hierarchical – have been developed and collected into repositories such as Wordnet [2], a general linguistic ontology for the English Language; the ACM Computing Classification System (ACM-CCS) [3], a four-layer system for mathematical, numerical, and engineering aspects of computing; a vast number of biological ontologies such as the Gene Ontology(GO) [4], and more recently a set of taxonomies comprising the SNOMED CT, the 'Systematized Nomenclature of Medicine Clinical Terms' [5].

Much effort has focused on building rules for ontological reasoning and querying by utilizing the inheritance relation supplied by the ontology's taxonomy in the presence of different data models [6, 7, 8]. These do not attempt at approximate representations but just utilize additional possibilities supplied by the ontology relations. A specific type of ontology usage is in using its taxonomy nodes for interpretation of clustering results based on knowledge extracted from the ontology [9, 10, 11].

The present work is motivated by the problem of analyzing and representing the activities of an organization in a taxonomy of the domain. To be specific, we focus on representing the research activities of a Computer Science research organization in terms of the ACM-CCS taxonomy. We cluster research topics according to their similarity derived on the basis of profiling of the ACM-CCS topics covered by the researchers working in the organization. Such a topic cluster would represent an elementary unit of the department's research as a whole. The set of clusters can be used for placement of the organization within the ACM-CCS taxonomy. Since a topic cluster is not necessarily consistent with the taxonomy structure, we consider it as a query topic set (whose elements correspond to leaves of the taxonomy) to be represented and interpreted in terms of related nodes of the taxonomy. To this end, we developed a recursive algorithm for minimizing a penalty function, summing penalties for the chosen "head subjects" together with penalties for related "gaps" and "offshoots". This provides a parsimonious representation of the research activities by mapping them to higher ranks of the taxonomy.

Each topic cluster is constructed using a novel fuzzy clustering algorithm based on an additive model of the topics similarities and using a sequential spectral method [12]. This algorithm combines additive clustering [13, 14], spectral clustering [15], and relational fuzzy clustering [16, 17]. The sequential nature of the algorithm provides a number of model-based stop conditions that allow us to determine an indicator of the number of clusters present in the data.

The goal of this paper is to develop further the proposed methods both in respect of their foundations and their application to the activity analysis. Specifically, we propose a similarity data generator to test the spectral fuzzy clustering method. Our lifting method [18, 19] is extended here to a fuzzy version to deal with fuzzy topic clusters. These are applied to synthetic and real world data.

The paper is organized as follows. In Section 2 we describe the parsimonious lifting method, and illustrate its mechanism with an example. Section 3 describes the additive fuzzy clustering model and corresponding spectral method. We analyse the ability of the fuzzy spectral clustering algorithm to recover a cluster structure from the data using a similarity data generator built on the proposed additive clustering model (Section 3.1). A real world case study for representing the research activities of a research center in terms of the ACM-CCS is discussed in Section 4. Section 5 concludes the paper.

2 PARSIMONIOUS LIFTING METHOD

Let us consider a rooted hierarchy T with I the set of its leaves. Each interior node $t \in T$ corresponds to a subject that generalizes the topics corresponding to the subset of leaves $I(t)$, that is the leaves of the subtree $T(t)$ rooted at t, which will be referred to as the leaf-cluster of t.

[1] Department of Computer Science and Centre for Artificial Intelligence (CENTRIA), Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, 2829-516 Caparica, Portugal, email: snt@fct.unl.pt

[2] Department of Computer Science and Information Systems, Birkbeck University of London, London WC1E 7HX, UK, email: trevor@dcs.bbk.ac.uk

[3] Department of Computer Science and Information Systems,Birkbeck University of London, London WC1E 7HX, UK, and National Research University Higher School of Economics, Moscow, RF, email: mirkin@dcs.bbk.ac.uk

A fuzzy set on I is a mapping u of I to the non-negative real numbers assigning a membership value $u(i) \geq 0$ to each $i \in I$. We refer to the set $S_u \subset I$, where $S_u = \{i : u(i) > 0\}$, as the support of u.

Given a taxonomy T and a fuzzy set u on I, let us assume that there exists an interior node of the tree T that covers S_u, up to small errors; such a node can be considered as an "head subject" (h). Two types of possible errors are "gaps" and "offshoots" as illustrated in Figure 1. A gap is a node g in the subtree $T(h)$ rooted at h such that $I(g)$ is disjoint from S_u but $I(parent(g))$ is not disjoint from S_u. An offshoot is a leaf $i \in S_u$ which is not covered by h, i.e., $i \notin I(h)$. A gap under the head subject h is interpreted as a loss of the concept h by the topic set u. On the other hand, establishing a node h as a head subject is referred to as a gain.

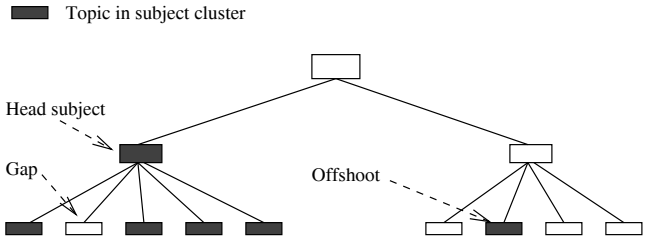


Figure 1. Three types of features in lifting a topic set within taxonomy.

Since no taxonomy perfectly reflects the activities conducted in an organization, some topic sets u may refer to general subjects that are not captured by the structure of T. This means that two or more head subjects are needed to cover them. This way, the pair (T, u) can be considered as an 'interpretation query'. Consider a set H of nodes of T that covers the support S_u, that is, each $i \in S_u$ either belongs to H or is a descendant of a node in H, i.e. $S_u \subseteq \bigcup_{h \in H} I(h)$. Set H is a possible result of the query (T, u). Nodes in H will be referred to as 'head subjects' if they are interior nodes of T or 'offshoots' if they are leaves. A node $g \in T$ is a gap for H if it is a 'gap' for some $h \in H$. A penalty value $p(H)$ will be defined so that the result of an interpretation query will be a set H with minimum penalty.

Given a fuzzy topic set u over I, a set of nodes H will be referred to as a *u-lift* if: (a) H covers S_u, that is, $S_u \subseteq \bigcup_{h \in H} I(h)$, and (b) all nodes in H are unrelated, that is, $I(h) \cap I(h') = \emptyset$ for all $h, h' \in H$ such that $h \neq h'$. That is, the set of offshoots in H is $H \cap I$. Let us denote by $G(h)$ the set of all gaps bellow a head subject $h \in H$. The set of gaps of H is the union of $G(h)$ over all head subjects $h \in H$.

We associate a penalty with H, so that only the most parsimonious lifts are to be considered. A parsimonious lift should have as small a number of head subjects, gaps and offshoots as possible. To reflect the relative importance of each of these, we introduce *penalty rates*, λ and γ, for gaps and offshoots, respectively. A head subject is assigned rate 1.

The u membership values on leaves are propagated to the interior nodes T, and aggregated consistently. For this, we assume the normalization condition which is compatible with our fuzzy clustering method:

(Q) Quadratic condition

$$\sum_{i \in I} u^2(i) = 1$$

Notice that a crisp set $S \subseteq I$ can be considered as a fuzzy set with the non-zero membership values defined according to the normalization principle. Specifically, for each interior node $t \in T$, its membership weight is defined as:

(Q) $u(t) = \sqrt{\sum_{i \in I(t)} u(i)^2}$

Under this definition, the weight of a gap is zero. The membership weight of the root is 1. We consider that the weight $u(t)$ of node t influences not only its own contribution, but also contributions of those gaps that are children of t. Therefore, the contribution to the penalty value of each of the gaps g of a head subject $h \in H$ is weighted according to the membership weight of its parent, as defined by $v(g) = u(parent(g))$. The gap contribution of h is defined as $V(h) = \sum_{g \in G(h)} v(g)$. For a crisp query set S this is just the number of gaps of h.

Therefore, our penalty function $p(H)$ for a u-lift H is defined as:

$$p(H) = \sum_{h \in H - I} u(h) + \lambda \sum_{h \in H - I} \sum_{g \in G(h)} v(g) + \gamma \sum_{h \in H \cap I} u(h), \quad (1)$$

The problem we address is to find such a u-lift H that minimizes the penalty $p(H)$. Such a parsimonious lift will be the annotation resulting as a response to the topic set u considered as a query.

2.1 PARL Algorithm

We define a pre-processing step to remove irrelevant nodes from tree T by pruning it as follows: i) label with 0 all nodes t whose clusters $I(t)$ do not overlap S_u; ii) remove from T all nodes that are children of 0-labeled nodes since they cannot be gaps. Then all interior nodes $t \in T$ are annotated by extending the leaf membership to membership weights (Q). Those nodes in the pruned tree that have a zero weight are gaps; they are assigned with a v-value which is the u-weight of its parent.

The PARL algorithm applies to a pruned hierarchy and it proceeds recursively from the leaves to the root as follows.

For each node t, PARL computes two sets, $H(t)$ and $L(t)$, containing those nodes in $T(t)$ at which gains and losses of head subjects occur. The respective penalty is computed as $p(t)$. [An assumption of the algorithm is that no gain can happen after a loss ("independent gains"). Therefore, $H(t)$ and $L(t)$ are defined assuming that the head subject has not been gained (nor, therefore, lost) at any of t's ancestors.]

I **Initialisation**
At each leaf $i \in I$:
if $u(i) > 0$, define $H(i) = \{i\}$, $L(i) = \oslash$ and $p(i) = \gamma u(i)$;
if $u(i) = 0$, define $H(i) = \oslash$, $L(i) = \oslash$ and $p(i) = 0$.

II **Recursion**
Suppose a node $t \in T$ has a set of children W, with each child $w \in W$ assigned a pair $H(w), L(w)$ and associated penalty $p(w)$. We have to consider the following two cases depending on whether:

(a) The head subject has been gained at t, so the sets $H(w)$ and $L(w)$ at its children $w \in W$ are not relevant.
Then $H(t)$, $L(t)$ and $p(t)$ are defined by:
$H(t) = \{t\}$;
$L(t) = G(t)$;
$p(t) = u(t) + \lambda V(t)$

(b) The head subject has not been gained at t.
So, at t we combine the children's H- and L-sets as follows:
$H(t) = \bigcup_{w \in W} H(w)$,

$L(t) = \bigcup_{w \in W} L(w)$,
$p(t) = \sum_{w \in W} p(w)$.

To obtain a parsimonious lift, we choose whichever of (a) and (b) has the smaller value of $p(t)$. We then recursively proceed to higher rank nodes.

III **Output**

Accept the values at the root:
$H = H(root)$ - the set of head subjects (interior nodes) and offshoots (leaves),
$G = L(root)$ - the set of gaps of H,
$p(root)$ - the penalty.

These give a full account of the parsimonious scenario obtained for the query u. Indeed, the optimal lifting interprets the lifted topic set within the ontology used with the minimum number of additional events, viz. "head subject", "gap" and "offshoot" correspondingly weighted.

2.2 ILLUSTRATIVE EXAMPLE

Consider the tree on Fig. 2a along with a fuzzy set comprised of four leaves, A.1, A.2, B.1, B.2, and membership values normalized according to the quadratic option (Q). Fig. 2b presents the pruned version of the tree in which the nodes are annotated by membership values, in thousandths and the gap weights V are shown at the higher rank nodes.

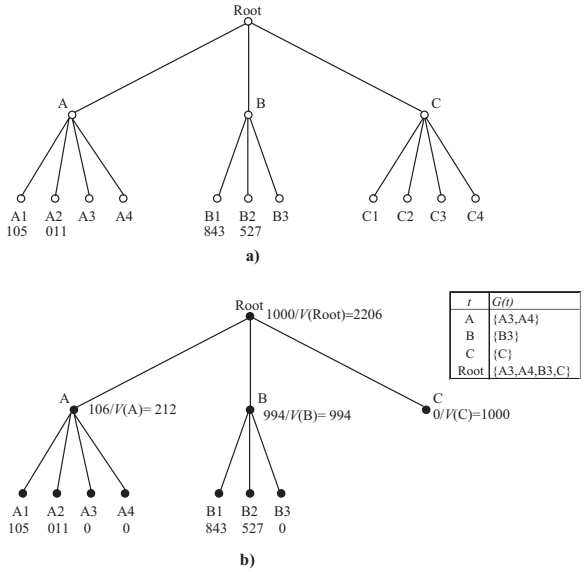


Figure 2. Illustrative taxonomy T with a fuzzy topic set membership values assigned to leaves, in thousandths (part (a); part (b) presents T after pruning and annotating with membership and gap values

Tables 1 and 2 represent sequential recursion steps of the lifting algorithm at the nodes A, B (Table 1) and the Root (Table 2).

Table 2 shows that No Gain solution at the root relates to a smaller penalty, 1.297, so that the query results in the lifting to node B, leaving A1 and A2 as offshoots, and B3 as the only gap.

Table 3 presents a number of different membership values and penalty parameters leading to different query outcomes for the taxonomy presented on Figure 2a. The membership values are not normalized for more convenient reading. The results are presented in the

Table 1. Recursion lifting steps at nodes A and B of the pruned tree for the query at Figure 2; the penalty values are $\gamma = 0.9$, $\lambda = 0.2$

i		$H(i)$	$L(i)$	$p(i)$
$A1$		$\{A1\}$	$\oslash$	$0.9 \times 0.105 = 0.095$
$A2$		$\{A2\}$	$\oslash$	$0.9 \times 0.0105 = 0.009$
$A3$		$\oslash$	$\oslash$	0
$A4$		$\oslash$	$\oslash$	0
$B1$		$\{B1\}$	$\oslash$	$0.9 \times 0.843 = 0.759$
$B2$		$\{B2\}$	$\oslash$	$0.9 \times 0.527 = 0.474$
$B3$		$\oslash$	$\oslash$	0
C		$\oslash$	$\oslash$	0
t		$H(t)$	$L(t)$	$p(t)$
A	Gain	$\{A\}$	$\{A3, A4\}$	$0.106 + 0.2 \times 0.212 = 0.148$
	No Gain	$\{A1, A2\}$	$\oslash$	$0.095 + 0.009 = 0.104$
B	Gain	$\{B\}$	$\{B3\}$	$0.994 + 0.2 \times 0.994 = 1.193$
	No Gain	$\{B1, B2\}$	$\oslash$	$0.759 + 0.474 = 1.233$

Table 2. Recursion lifting steps at the root for the query at Figure 2 following the results presented in Table 1, with penalties $\gamma = 0.9$ and $\lambda = 0.2$

t		$H(t)$	$L(t)$	$p(t)$
A		$\{A1, A2\}$	$\oslash$	0.104
B		$\{B\}$	$\{B3\}$	1.193
C		$\oslash$	$\oslash$	0.0
t		$H(Root)$	$L(Root)$	$p(Root)$
$Root$	Gain	$\{Root\}$	$\{A3, A4, B3, C\}$	$1 + 0.2 \times 2.206 = 1.441$
	No Gain	$\{A1, A2, B\}$	$\{B3\}$	$0.104 + 1.193 = 1.297$

Table 4. One should notice that all possible outcomes are covered by the eight query sets and corresponding penalty values. The query illustrated in Figure 2 and Tables 1 and 2 is case 2 in Table 3. Changing the gap rate λ to 0.1 leaves the result the same (case 4), B is the only genuine head subject, which is no wonder because the major membership values are concentrated at B. If we reshuffle the membership values to shift the weight to A, A indeed becomes the head subject– see case 5. Yet such an outcome is not a certainty, as case 3 clearly demonstrates - a very small increase in membership values towards B, still leaving the bulk of membership with A, makes B the head subject again, against the odds – just because of a slight asymmetry in the structure of A and B nodes. Cases 6 and 7 show that the same membership values as in case 3, but with somewhat changed gap or offshoot rates can lead to a different output: both A and B, or even the root, as the head subjects.

Table 3. Different queries and distinct penalty rates for the illustrative taxonomy in Figure 2a; the membership values are further normalized according to condition (Q).

#	$u(A1)$	$u(A2)$	$u(B1)$	$u(B2)$	γ	λ
1	0.8	0.5	0.1	0.01	0.9	0.2
2	0.1	0.01	0.8	0.5	0.9	0.2
3	0.8	0.5	0.1	0.1	0.9	0.2
4	0.1	0.01	0.8	0.5	0.9	0.1
5	0.8	0.5	0.1	0.01	0.9	0.1
6	0.8	0.5	0.1	0.1	1.1	0.2
7	0.8	0.5	0.1	0.1	0.9	0.1
8	0.1	0.1	0.8	0.5	0.9	0.1

3 ADDITIVE FUZZY CLUSTERS USING A SPECTRAL METHOD

Let I be the set of leaves of a hierarchical taxonomy T such as the ACM-CCS. Then individual members (or projects) of a research organization can be represented with fuzzy membership profiles over the leave subjects of the taxonomy. Given an individual-to-subject profile matrix F, a similarity matrix $Y = (y_{ii'})$, with $i, i' \in I$ can

Table 4. Lifting outputs for the illustrative taxonomy on Figure 2a taking different queries as shown in Table 3.

#	$H(Root)$	$L(Root)$	$P(Root)$
1	$\{A1, A2, B1, B2\}$	$\{\}$	1.34
2	$\{A1, A2, B\}$	$\{B3\}$	1.30
3	$\{A1, A2, B\}$	$\{B3\}$	1.40
4	$\{A1, A2, B\}$	$\{B3\}$	1.20
5	$\{A, B1, B2\}$	$\{A3, A4\}$	1.30
6	$\{A, B\}$	$\{A3, A4, B3\}$	1.56
7	$\{Root\}$	$\{A3, A4, B3, C\}$	1.31
8	$\{Root\}$	$\{A3, A4, B3, C\}$	1.23

be defined as $Y = F^T F$ so that $y_{ii'}$ is the inner product of subject columns i and i'. These subject-to-subject similarity values are assumed to express some relational hidden patterns represented by fuzzy clusters[4].

We consider a relational fuzzy cluster to be defined by: (1) a membership vector $\mathbf{u} = (u_i)$, such that $0 \leq u_i \leq 1$ for all $i \in I$, and (2) an intensity $\mu > 0$ that expresses the extent of significance of the pattern corresponding to the cluster. The intensity applies as a scaling factor to $\mathbf{u}$ so that it is the product $\mu\mathbf{u}$ that expresses the hidden pattern rather than its individual co-factors. Given a value of the product μu_i, to separate μ and u_i, we consider that the scale of the membership vector $\mathbf{u}$ is constrained on a constant level by the normalization condition $\sum_i u_i^2 = 1$; the remaining factor defines the value of μ. Also, to admit a possible pre-processing transformation of the given similarity matrix Y, we denote the matrix involved in the process of clustering as $A = (a_{ii'})$.

We define the additive fuzzy clustering model by K fuzzy clusters that reproduce the input similarities $a_{ii'}$ up to an error, such as:

$$a_{ii'} = \sum_{k=1}^{K} \mu_k^2 u_{ki} u_{ki'} + e_{ii'}, \tag{2}$$

where $\mathbf{u}_k = (u_{ki})$ is the membership vector of cluster k, μ_k its intensity ($k = 1, 2, ..., K$), and $e_{ii'}$ is the residual similarity not explained by the model.

The item $\mu_k^2 u_{ki} u_{ki'}$ in (2) is the product of $\mu_k u_{ki}$ and $\mu_k u_{ki'}$ expressing the impacts of i and i', respectively, in cluster k. This value adds up to the others to form the similarity $a_{ii'}$ between topics i and i'. The square of the cluster's intensity μ_k^2 will be referred to as the cluster's weight.

To fit the model (2) corresponds to minimize the sum of all $e_{ii'}^2$ by least-squares. Within that, we apply the one-by-one principal component analysis strategy for finding one cluster at a time by minimizing the corresponding one-cluster criterion

$$E = \sum_{i,i' \in I} (b_{ii'} - \xi u_i u_{i'})^2 \tag{3}$$

with respect to the unknown positive ξ weight and fuzzy membership vector $\mathbf{u} = (u_i)$, given similarity matrix $B = (b_{ii'})$.

In the beginning, matrix B is taken to be equal to matrix A. Each found cluster (μ, $\mathbf{u}$) is subtracted from B, so that the residual similarity matrix applied for obtaining the next cluster is defined as $B - \mu^2 \mathbf{u}\mathbf{u}'$. This way, A is additively decomposed according to formula (2) and the number of clusters K can be determined in the process.

The optimal value of ξ at a given $\mathbf{u}$ is given by

$$\xi = \frac{\mathbf{u}' B \mathbf{u}}{(\mathbf{u}'\mathbf{u})^2}, \tag{4}$$

which is non-negative if B is positive semidefinite.

Substituting the defined ξ in equation (3), one arrives at $E = S(B) - \xi^2 (\mathbf{u}'\mathbf{u})^2$, where $S(B) = \sum_{i,i' \in I} b_{ii'}^2$ is the similarity data scatter.

Denoting the last item as

$$G(\mathbf{u}) = \xi^2 (\mathbf{u}'\mathbf{u})^2 = \left(\frac{\mathbf{u}' B \mathbf{u}}{\mathbf{u}'\mathbf{u}}\right)^2, \tag{5}$$

the similarity data scatter is decomposed as $S(B) = G(\mathbf{u}) + E$ where $G(\mathbf{u})$ is the part of the data scatter that is explained by cluster (μ, $\mathbf{u}$), and E, the unexplained part. Therefore, an optimal cluster is to maximize the explained part $G(\mathbf{u})$ in (5) or its square root

$$g(\mathbf{u}) = \xi \mathbf{u}'\mathbf{u} = \frac{\mathbf{u}' B \mathbf{u}}{\mathbf{u}'\mathbf{u}}, \tag{6}$$

which is the Rayleigh quotient: its maximum value is the maximum eigenvalue of matrix B, which is reached at its corresponding eigenvector, in the unconstrained problem.

Therefore, the spectral clustering approach can be applied to maximize equation (6). This corresponds to find the maximum eigenvalue λ and corresponding normed eigenvector z for B, $[\lambda, z] = \Lambda(B)$, and take its projection to the set of admissible fuzzy membership vectors. The normalization condition $\sum_i u_i^2 = 1$ simplifies the criterion (6) and leads to the spectral solution.

Several criteria for halting the process of sequential extraction of fuzzy clusters follow from the above. The process stops if either of the conditions holds:

- S1 The optimal value of ξ (4) for the spectral fuzzy cluster becomes negative.
- S2 The contribution of a single extracted cluster to the data scatter becomes less than a pre-specified $\tau > 0$ threshold.
- S3 The residual data scatter becomes smaller than a pre-specified $\epsilon > 0$ proportion of the original similarity data scatter.
- S4 A pre-specified number K_{max} of clusters is reached- in some real world problems that information can be set

The described one-by-one Spectral Additive Fuzzy cluster extraction method is referred to here as SAF. Since SAF method extracts clusters one-by-one, in the order of their contribution to the data scatter, the algorithm is supposed to be oriented at cluster structures at which the clusters contribute differently: the more the differences, the better. We refer to this supposed property of the data as the property of differing contributions.

3.1 SAF ANALYSIS of CLUSTERING RECOVERY

An experimental study was conducted to explore the ability of SAF algorithm to recover a cluster structure from similarity data generated based on model (2) perturbed with different levels of Gaussian noise.

3.1.1 The Fuzzy Core Cluster Data Generator

A similarity data generator following the additive model (2) was developed. As usual in fuzzy clustering, we assume that each entity has one "core" cluster to which it belongs most. The data generation process starts with the generation of the "core" clusters.

[4] The membership vector u assigned to each fuzzy cluster corresponds to a potential query set to be interpreted by the PARL algorithm

Given the size N of an entity set I, and the number of clusters K, the proposed Fuzzy Core Cluster Data Generator (FCC DG), generates a $N \times N$ similarity data matrix G according to the underlying model $Y = U\Lambda U^T$, as follows:

$$G = U\Lambda U^T + \alpha E, \tag{7}$$

where:

- $N \times K$ fuzzy membership matrix U is randomly generated using a fuzzy "core" clusters generating procedure.
- Positive real valued $K \times K$ diagonal weight matrix Λ with diagonal positive values λ_k of the cluster weights equal to $\lambda_k = \mu_k^2$ is defined according to model (2). Since the vectors $\mathbf{u}_k$ in (2) are assumed normed, the weights take in the norms of the generated vectors $\mathbf{u}_k$. To test the supposed property of different contributions of the SAF, the weights are also made proportional to $(K-k+1)^\beta$, for $k = 1, 2, \ldots, K$, so that the greater the $\beta > 0$, the greater the difference. Therefore, the weights are defined by $\lambda_k = (K-k+1)^\beta * \|\mathbf{u}_k\|$.
- Elements of $N \times N$ error matrix E are independently generated from a Gaussian distribution $N(0,1)$, and then symmetrized so that $e_{ii'} = (e_{ii'} + e_{i'i})/2$.
- The value $\alpha \in [0,1]$ is the parameter that controls the level of error introduced into the model $Y = U\Lambda U^T$.

A fuzzy cluster structure is built by conventionally relaxing a crisp K-partition, where each k-th fuzzy cluster $\mathbf{u}_k$ is build having the corresponding crisp cluster C_k as its core in such a way that the maximum membership values u_{ik} will be at entities $i \in C_k (k = 1, \ldots, K)$ while the other components of $\mathbf{u}_k$ are close to 0.

Given the number K of core clusters covering the entire data set, I, the data generator builds each core cluster by filling it with fuzzy membership values, such that: (a) the membership values of k-th fuzzy cluster $\mathbf{u}_k$ are very high at k-th core (e.g. $u_{ik} > 2/3$ for $i \in C_k$); and (b) the fuzzy clusters form a fuzzy partition so that $\sum_k u_{ik} = 1$ at each entity $i \in I$. After all the membership vectors $\mathbf{u}_k$ are generated, the norms of $\mathbf{u}_k$'s are computed and assigned as factors in the clusters' weights, in order to "adjust" them to the additive fuzzy cluster model. Then, the final membership matrix has its membership vectors $\mathbf{u}_k$ normalized.

3.1.2 Analysing SAF with FCC Simulated Data

A pool of datasets have been generated in three groups corresponding to three different numbers of clusters: $K = 3, 4, 5$. The experiments were cross-combined according to the following settings: (i) Total number of entities of the data set $N = 50, 200, 400, 700$; (ii) α values of the standard deviation of noise, $\alpha = \{0, 0.05, 0.1\}$. (iii) For each value of K, 10 distinct datasets have been generated for each tuple (N, α, β). In our preliminary experiments, we observed that the ability to recover a cluster structure significantly decreases for the values of $\alpha > 0.1$.

Table 5 shows the means/std and modal values of the recovered number of clusters by the SAF algorithm. The best value in each row is marked with (*).

For $K = 3, 4, 5$ one can see that when the β value increases from $\beta = 0.0$ to $\beta = 1.0$ the percentage of data sets for which the correct number of clusters is recovered also increases. The only exception occurs for $K = 5, N = 200$, where the best values are achieved for $\beta = 0.5$. In all the cases, the best working stop condition of the SAF algorithm is condition $S2$.

Table 5. FCC DG - Summary data of the percentage avg/std of correct extracted clusters and mode of the number of extracted clusters for std of added Gaussian noise=[0, 0.1] for SAF in best conditions for $K = \{3, 4, 5\}$

		SAF					
		$\beta = 0.0$		$\beta = 0.5$		$\beta = 1.0$	
	N	(%)	M	(%)	M	(%)	M
$K = 3$	50	50.0/0.0	3	62.5/9.6	3	85.0/5.8*	3
	200	60.0/0.0*	3	32.5/20.6	2	60.0/0.0*	3
	400	30.0/21.6	3	62.5/15.0	3	80.0/0.0*	3
	700	17.5/17.1	2	40.0/35.6	2	65.0/19.1*	3
$K = 4$	50	47.5/9.6	4	60.0/8.2	4	70.0/18.3*	4
	200	50.0/35.6	4	50.0/0.0	4	65.0/5.8*	4
	400	27.5/18.9	5	55.0/10.0	4	72.5/5.0*	4
	700	17.5/20.6	1	67.5/5.0	4	77.5/5.0*	4
$K = 5$	50	40.0/21.6	5	60.0/8.2	5	67.5/5.0*	5
	200	37.5/26.3	5	52.5/5.0*	5	40.0/8.2	5
	400	45.0/46.5	5	50.0/0.0	5	65.0/10.0*	5
	700	25.0/23.8	1	35.0/5.8	6	42.5/5.0*	5

To measure the similarity between generated and found partitions we apply the Adjusted Rand Index (ARI) whose avg/std are presented in Table 6. The higher ARI values are achieved for data sets generated with $\beta = 1.0$ for the data sets with $K = 3$ and $K = 4$ clusters. However, for $K = 5$, the best values are achieved at $\beta = 0.5$, in contrast to the expected property of different contributions.

Table 6. FCC DG - Summary Table for ARI avg/std for std of added Gaussian noise=[0, 0.1] for all algorithms in best conditions for $K = \{3, 4, 5\}$

		SAF		
	N	$\beta = 0.0$	$\beta = 0.5$	$\beta = 1.0$
$K = 3$	50	0.88/0.14	0.84/0.21	**0.90/0.19***
	200	0.74/0.19	0.70/0.21	**0.81/0.18***
	400	0.87/0.10	0.87/0.10	**0.91/0.11***
	700	0.79/0.16	0.70/0.19	**0.80/0.20***
$K = 4$	50	0.92/0.07	0.91/0.07	**0.93/0.1***
	200	0.92/0.09	0.91/0.09	**0.94/0.11***
	400	0.87/0.14	0.91/0.14	**0.93/0.13***
	700	0.84/0.15	**0.93/0.08***	0.83/0.17
$K = 5$	50	0.92/0.08	**0.95/0.08***	0.78/0.23
	200	0.89/0.12	**0.93/0.09***	0.83/0.17
	400	0.87/0.22	**0.95/0.06***	0.88/0.15
	700	0.89/0.13	**0.94/0.06***	0.84/0.13

4 A REAL WORLD CASE STUDY

We consider a two-stage process by combining SAF and PARL methods in order to represent and interpret the individual member/project profiles in terms of the ACM-CCS taxonomy. First stage finds fuzzy clusters of the taxonomy subjects according to the working of the organization. Second stage maps each of the clusters to higher ranks of the taxonomy in a parsimonious way.

Table 7 presents one of the two fuzzy clusters obtained on the data from a survey conducted in a Computer Science research centre[5] involving 16 respondents and covering 46 ACM-CCS topics, by applying the SAF algorithm. This cluster is mapped to and parsimoniously lifted by the PARL algorithm over the ACM-CCS taxonomy.

For illustrative purposes, we consider the crisp version of this cluster over the ACM-CCS taxonomy taken as the topic query set to be solved by the PARL algorithm setting with offshoot and gap penalty rates of $\gamma = 0.8$ and $\lambda = 0.15$, respectively. The gap penalty rate is 0.15 because the numbers of children in ACM-CCS hierarchy are on average about 10 and we would like that the case when three of

[5] Centre for Artificial Intelligence (CENTRIA) of Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa

Table 7. A fuzzy cluster of research activities undertaken in a research centre by SAF

Member	Code	ACM-CCS Topic
0.6991	I.5.3	Clustering
0.3512	I.5.4	Applications in I.5 PATTERN RECOGNITION
0.2744	J.2	PHYSIC. SCIENCES AND ENG. (Applications in)
0.1992	I.4.9	Applications in I.4 IMAGE PROC. AND COMP. VISION
0.1992	I.4.6	Segmentation
0.1972	H.5.1	Multimedia Information Systems
0.1748	H.5.2	User Interfaces
0.1748	H.5.3	Group and Organization Interfaces
0.1669	H.1.1	Systems and Information
0.1669	I.5.1	Models in I.5 PATTERN RECOGNITION
0.1651	H.1.2	User/Machine Systems
0.1445	I.5.2	Design Methodology (Classifiers)
0.1365	H.5.0	General in H.5 INF. INTERFACES AND PRESENTATION
0.1365	H.0	GENERAL in H. Information Systems

the children do belong to the head subject, then they should be lifted to the parent, whereas two children should remain as they are. Indeed, two children belonging in the head subject would not be lifted to the parental node because then the total gap penalty 8*0.15=1.2 would be greater than the decrease of head subject penalty 0.8*2-1=0.6. The value of offshoot penalty rate is taken slightly less than that of the head subject (setting as 1.0) as we considered that lifting a head subject from the leaves should bear a slightly greater penalty.

The application of the PARL algorithm leads to the following representation of the (crisp) cluster: "head subjects"– *H.-Information Systems* and *I.5-PATTERN RECOGNITION*), their "gaps"– (*H.2-DATABASE MANAGEMENT*, *H.3-INFORMATION STORAGE AND RETRIEVAL*, *H.4-INFORMATION SYSTEMS APPLICATIONS*), and "offshoots" – (*I.4.6- Segmentation*, *J.2- PHYSICAL SCIENCES AND ENGINEERING*, and *I.4.9- Applications in I.4 IMAGE PROCESSING AND COMPUTER VISION*). Such a representation over the ACM-CCS taxonomy is shown in Figure 3.

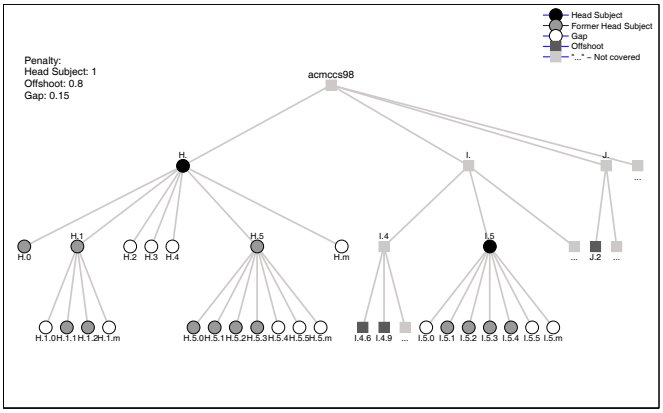


Figure 3. Visualization of the optimal lift of the cluster in Table 1 in the ACM-CCS tree; the irrelevant tree leaves are not shown for the sake of simplicity.

A similar result is obtained when taking the original fuzzy cluster (Table 7) as the topic query, taking the quadratic condition (Q) to extend the membership leaf node weights. However, to achieve this, the gap penalty has to be decreased to 0.055 to balance the decreased fuzzy membership values. Further decrease of the gap penalty rate to 0.05 would make make the "Root" the "head subject".

5 CONCLUSION

We propose a hybrid methodology for data representation and interpretation in terms of a taxonomy tree. The methodology combines: (i) a spectral fuzzy clustering method that allows to cluster research topics according to their thematic similarities without taking into account the topology of the taxonomy; (ii) a recursive method that lifts the clusters mapped onto the taxonomy to higher ranked nodes of the tree, leading to a parsimonious representation of the clusters. The interpretive ability of this method comes from the topology based concepts of "head subjects", "gaps" and "offshoots", their penalty weights, and the definitions of the aggregated membership weights.

For future research, the following issues are of prime importance. Even though the spectral fuzzy algorithm has shown to be a good indicator of the number of clusters present in the data due to its model-based cluster extraction stop conditions, the spectral method needs further investigation in order to satisfy the property of differing contributions. With respect to the lifting method, the issue of defining the "right" penalty weights remains to be addressed.

REFERENCES

[1] A. Gómez-Pérez, M. Fernández-López, O. Corcho, *Ontological Engineering: With Examples from the Areas of Knowledge Management, E-commerce and the Semantic Web*, Springer, 2004.

[2] G. A. Miller, 'WordNet: a lexical database for English', *Communications of the ACM*, **38**(11), 39–41, (1995).

[3] The ACM Computing Classification System (1998), `http://www.acm.org/class/1998/ccs98.html`.

[4] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J.M. Cherry, *et al.*, 'Gene ontology: Tool for the unification of biology. The Gene Ontology Consortium', *Nature Genetics*, **25**(1): 25-9. doi:10.1038/75556. PMC 3037419. PMID 10802651, (2000).

[5] SNOMED CT (2011) `http://www.connectingforhealth.nhs.uk/systemsandservices~/data/uktc/snomed` (Cited March 2011)

[6] D. Beneventano, N. Dahlem, S. El Haoum, A. Hahn, D. Montanari and M. Reinelt, 'Ontology-driven semantic mapping', *Enterprise Interoperability III*, Part IV, Springer, 329–341, (2008).

[7] S. Sosnovsky, A. Mitrovic, D. Lee, P. Prusilovsky, M. Yudelson, V. Brusilovsky and D. Sharma, 'Towards integration of adaptive educational systems: mapping domain models to ontologies', In D. Dicheva, A. Harrer, R. Mizoguchi (eds.), *6th International Workshop on Ontologies and Semantic Web for E-Learning (SWEL'2008)*, 60–64, (2008).

[8] A. Cali, G. Gottlob and A. Pieris, 'Advanced processing for ontological queries', *Proceedings of the VLDB Endowment*, **3**(1-2), 554–565, (2010).

[9] J. Liu, W. Wang and J. Yang, 'A framework for ontology-driven subspace clustering', *Proceedings of the 10th ACM SIGKDD international conference on knowledge discovery and data mining (SIGKDD)*, 623–628, (2004).

[10] D. Dotan-Cohen, S. Kasif and A. Melkman, 'Seeing the forest for the trees: using the gene ontology to restructure hierarchical clustering', *Bioinformatics*, **25**(14), 1789–1795, (2009).

[11] M. Peleg, N. Asbeh, T. Kuflik and M. Schertz, 'Onto-clust-A methodology for combining clustering analysis and ontological methods for identifying groups of comorbidities for developmental disorders', *Journal of Biomedical Informatics*, **42**(1), 165–175, (2009).

[12] B. Mirkin, S. Nascimento, "Additive spectral method for fuzzy cluster analysis of similarity data including community structure and affinity matrices", *Information Sciences*, 183, 16-34, (2012).

[13] R.N. Shepard and P. Arabie, 'Additive clustering: representation of similarities as combinations of overlapping properties', *Psychological Review*, **86**, 87–123, (1979).

[14] B. Mirkin, 'Additive clustering and qualitative factor analysis methods for similarity matrices', *Journal of Classification*, **4**, 7–31, (1987); Erratum 6, 271–272, (1989).

[15] U. von Luxburg, 'A tutorial on spectral clustering', *Statistics and Computing*, **17**, 395–416, (2007).

[16] J. Bezdek, J. Keller, R. Krishnapuram, T. Pal, *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*, Kluwer Academic Publishers, 1999.

[17] R.K. Brouwer, 'A method of relational fuzzy clustering based on producing feature vectors using FastMap', *Information Sciences*, **179**, 3561–3582, (2009).

[18] B. Mirkin, S. Nascimento, L.M. Pereira "Cluster-lift method for mapping research activities over a concept tree", In: J. Koronacki, S.T. Wierzchon, Z.W. Ras, J. Kacprzyk (eds.), *Recent Advances in Machine Learning II, Computational Intelligence Series*, Vol. 263, Springer, 245–258, (2010).

[19] B. Mirkin, S. Nascimento, T. Fenner, L. M. Pereira 'Building Fuzzy Thematic Clusters and Mapping Them to Higher Ranks in a Taxonomy', *International Journal of Software and Informatics*, **4**(3), 257–275, (2010).

A Fuzzy System for Educational Tasks for Children with Reading Disabilities

Adalberto Bosco C. Pereira[1] and **Leonardo B. Marques**[2] and **Dionne C. Monteiro**[1] and **Gilberto Nerino de Souza**[1] and **Clay Palmeira da Silva**[1].

Abstract. This paper proposes a computational approach that aims the use of Artificial Intelligence in Education (AIED) to aid teachers, psychologists and educationalists in the learning process of reading. This approach aims at generating teaching tasks which can be accordingly adapted to the individual needs of each student. While the tasks are being executed, a Machine Learning (ML) system will collect and process data to allow an analysis of the student's learning process for each individual word in reading and writing abilities. The fuzzy system will propose an appropriate task based on the data collected by the ML. The output of the fuzzy system is an adapted task that will motivate the children in execution of the tasks.

1 INTRODUCTION

Digital games have an important space in the lives of children, teenagers and adults, and are now one of the fastest growing sectors in the media and entertainment industry. In addition, games have been combined with a wide range of fields such as unusual simulations, Games for Health, Game-Based Learning (GBL) [1] Artificial Intelligence in Education (AIED) [2].

Good learning requires teachers and students to combine their efforts. In the case of students, their interest in studying is of fundamental importance [3] in order to ensure they can achieve their potential to the maximum. Regarding to teachers, it can be concluded that any attempt to improve students' achievements must be based on the acquisition of an effective teaching behavior [4]. In other words, they should give students appropriate guidance.

Usually gamification applies to non-game applications and processes, in order to motivate people to adopt them or to influence how they are used. Gamification works for: i) making technology more engaging, ii) encouraging users to engage in desired behaviors, iii) showing a path to mastery and autonomy, iv) helping to solve problems and not being a distraction, and v) taking advantage of humans' psychological predisposition to engage in gaming. The technique can encourage people to perform task that they ordinarily consider tedious, such as completing surveys, shopping, filling out tax forms, or reading web sites.

For several years, some games have incorporated automatic generation to create levels, missions and space that are designed to increase the lifetime of their games. There are games that have a form of challenge called 'endless', where the player is exposed to challenges at levels that are generated indefinitely until the game is lost. This approach is adopted for several areas in games and in many features such as: i) 2D textures, ii) 3D models, iii) music, iv) levels, v) story, vi) mission and so on. These studies are of increasing importance in the process of developing computer games [5].

The purpose of this work is to test a Fuzzy Logic solution applied to the current task of teaching reading in a digital game that is adaptive to the individual needs of each player. That is, if the new tasks of education are suited to pre-existing literacy skills of the student, this means that this task must not be either too easy or too difficult. The data collected during the execution of the teaching tasks will be pre-processed and analyzed with the aid of a Machine Learning (ML) system, which will provide data to Fuzzy System to evaluate and consider the best choice for proposing a task. The output of the Fuzzy System is determined by the data required for task generation, which in turn will be transformed into an adapted level of the game.

2 RELATED WORK

The "Gerenciador de Ensino Individualizado por Computador" (GEIC – Computer Individualized Education Manager) [6] approaches the problem of ensuring the dynamic generation of content for educational use, by providing software for programming procedures based on choice teaching tasks. It allows the creation of Teaching Units that combine various teaching tasks, and represent discrete attempts to provide choice tasks. The objective of this work is to aid teachers in teaching reading and writing abilities to children with learning difficulties. Further details of this program will be given in Section Three of this paper.

Azevedo in [7] discusses methods of teaching which, although of satisfactory standard, still present different efficiency levels for each student. This is discussed in examining the computational and educational resources and their degree of adaptability to the individual needs of each student.

Studies in the field of AIED [2] by B.du Benedict, state that the individualization of teaching instruction can be effective and is regarded by the author as the "Holy Grail of AIED". This paper compares the educational differences in AI systems (AIED) with conventional educational systems used in the classroom or traditional methods of Computer-Assisted Instruction (CAI).

The learning program called GEIC [6], was transformed into a game called ALE-RPG [8]. However, it should be stressed that the structured progress that the game kept in the GEIC proved to be a

[1] Laboratory of Applied Artificial Intelligence – Institute of Exact and Natural Sciences – Federal University of Para, Brazil, email: adalberto@ufpa.br, dionne@ufpa.br, gilbertojr@ufpa.br, claypalmeira@ufpa.br

[2] Laboratory for the Study of Human Behavior – Federal University of Sao Carlos, Brazil, email: leobmarques@gmail.com

little static. The progression of the player through the teaching tasks does not allow a fully customized advance to be made, since it is conditioned by the need to learn all the words of each teaching unit. The non-acquisition of any word component of the teaching units delays the teaching progress of the next units.

There are games like Diablo [9], Torchlight [10], Spore [11] and MineCraft [12] that use automatic generation levels when the player starts a new game, but keeps the missions and game objectives distributed randomly on the generates map. However, all this randomness follows 'brute force' algorithms, without any predetermined scale of difficulty.

The approach of D. Dormans [5] investigates strategies to generate levels of action-adventure games that are divided into two individual structures, so that they generate missions first and then spaces. The different types of generative grammar are analyzed in a search for the one that best fits.

L. Xiangfeng [1] uses Fuzzy Cognitive Maps to design Game-Based Learning (GBL). The goal is to use the Hebbian learning rule to increase learning capacity by employing the game data and Unbalance Degree to establish the lack of prior knowledge.

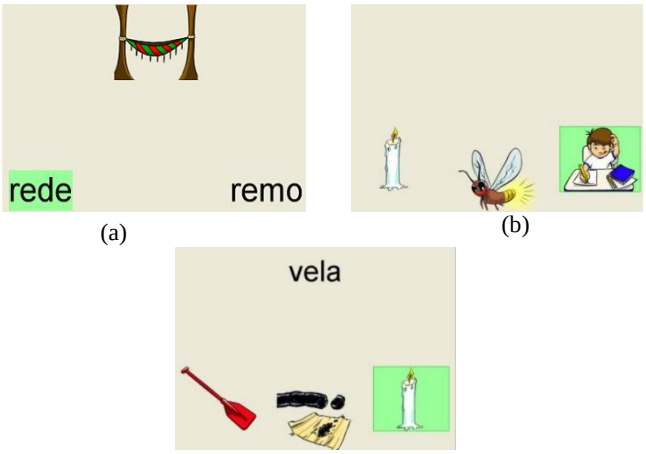
3 TEACHING PROGRAM

Since the 1980s, researchers in Brazil in the field of behavioral psychology have been refining a program to help children with a previous history of school failure to learn to read. This procedure, called "Aprendendo a Ler e a Escrever em Pequenos Passos" (ALEPP - Learning to Read and Write in Small Steps) [13], is mainly concerned with detecting and overcoming problems that are found in children who have literacy difficulties in an efficient manner. It also provides tasks that teach the basic components of reading in a personalized manner.

The GEIC is a remote software that allows the ALEPP curriculum [13] to be applied. But it is not yet adapted enough, because it is composed of static tasks, grouped into Teaching Units that have been previously determined by specialists.

The Matching to Sample (MTS) procedure is used for teaching reading relations programmed into ALEPP [13] through GEIC [6], Figure 1. This procedure is used to teach relations between printed words, pictures and dictated words. In this task, one stimulus (the sample) must be matched to the correct comparison.

Each task contains *n* comparisons, ranging from one to three, where one acts as a stimulus by referring to the sample on top of the screen. Finally, the last feature is the definition of the words that correspond to the model stimulus and stimulus choices. The tasks are subdivided into types of stimulus combinations such as AB, BC and CB. In Figure 1 (b) the sample stimulus is the sound that corresponds to the correct alternative.



(a) (b)

(c)

Figure 1. MTS Task Model. AB (a) Type CB (b) and type BC (c).

The tasks allow you to create relationships between the stimuli of different modalities. They are relations that are taught between the dictated words (A), representative figure of this word (B) and the printed word (C). The tasks that establish these relations are listed as tasks of type AB (dictated word-picture), CB (printed word-picture) and BC (picture-printed word).

3.1 Performance Assessment in Reading

In the current ALEPP assessment system, the GEIC automatically loads the teaching tasks assessments [15]. The GEIC can be set individually at the end of each teaching session, if the student has repeated the last block of tasks or advanced to the next block. This software allows a minimum number of correct answers to be set in a session and uses these successes as a criterion for advancement.

However it is not a fine evaluation in terms of the words learned. The final session assessment is unable to identify the different kinds of errors for each word taught. The ideal situation is to evaluate correctness for each word. For example, in trials of the word "pipe" as a sample, it shows that the student learned the word, how he knows this word, and then a task is set with a focus on learning how to read or write, so that the student can learn to read or write that word.

4 OUR APPROACH

In order to adopt an approach to assist instructors, the general purpose of the project is initially aimed at creating one AIED to teach reading and writing and then incorporate it into a digital game. AIED is composed of two intelligent systems, ML and Fuzzy System, which will act together, as shown in the diagram in Figure 2. They represent the Machine Learning system that is not covered in this work, but developed separately in parallel. Both the systems will be executed during gameplay of several mini-games.

The interaction between the student and the game, will be initially by computers, using mouse and keyboard. In the future, we intended to use tablets with the Android operational system. Both systems (computers and tablets) will be composed of several minigames with gamefied teaching duties.

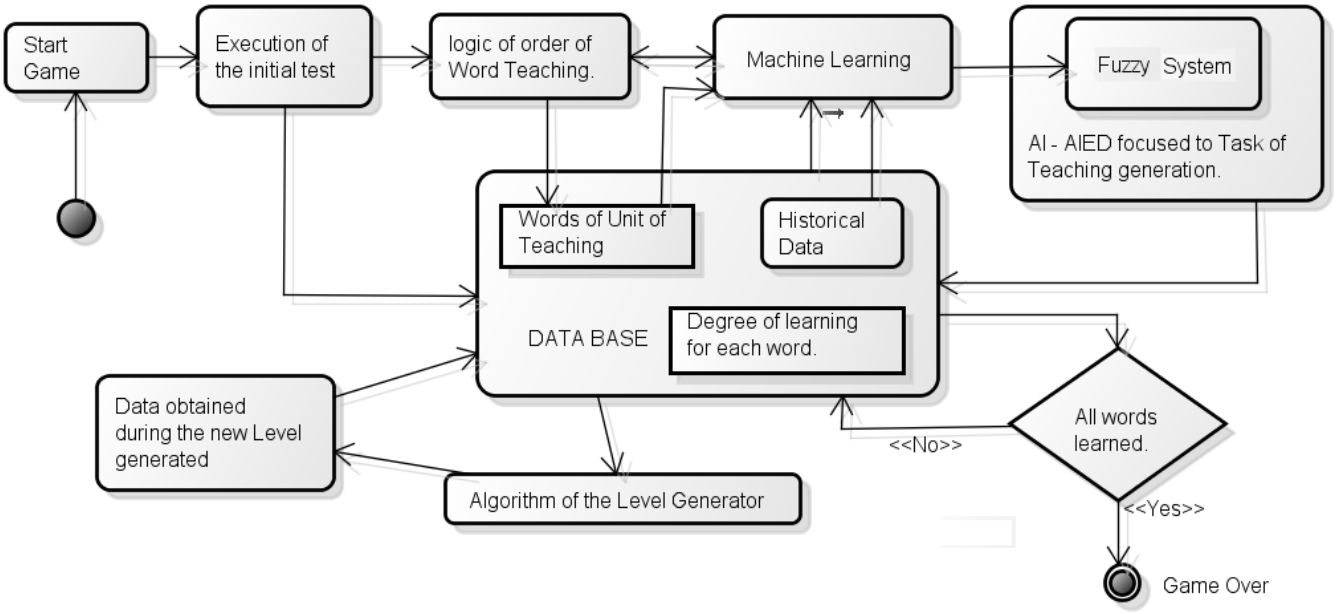


Figure 3. Flow diagram of the Approach.

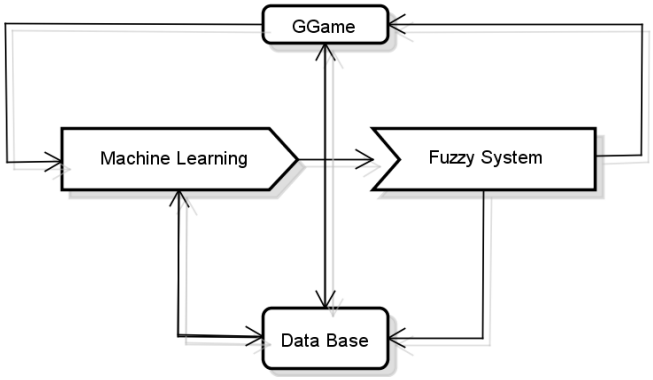


Figure 2. Macro view of the project.

The system is a digital game that begins with an initial pretest of static tasks to generate the minimum data required to enable the Machine Learning system to correctly analyze the student. After the information is obtained during gameplay, for each word that has been processed, a new task is generated by Fuzzy System and will be stored in the database. The level of the game is created by using the features of the generated task. The sequence of tasks follows that of Logical Teaching. For the specialists, there is a preferred order for each word being taught, and new words will enter the teaching tasks gradually, depending on the degree of literacy of the word.

The ML is responsible for evaluating the student, calculating his knowledge level, and providing the probability of success and difficulty of a given task. Done so, such data is passed to the fuzzy system in order that it can correctly generate an adapted teaching task.

For each Teaching Unit there are fifteen words that have to be taught. The ML will decide the level of knowledge and evaluate if the student has learned every single word. As in the case of the execution of new ML generated tasks, they will always be calculated on the basis of this new information regarding the degree of knowledge of a particular word. The words that are determined as literacy appear less frequently, and when every word has been learned, the game is over. Figure 3 shows a diagram of the operation of this proposal.

5 DESCRIPTION OF THE FUZZY SYSTEM

The main objective of this paper is to make use of data generated by ML correctly in order to generate an adjusted task. Fuzzy Logic was chosen to carry this out for the following reasons: it has a distinct capacity to express the vagueness and uncertainty of the knowledge it represents [16]; it is able to model a system close to logical grammatical rules; it ensures a better approximation to the knowledge of a specialist through semantic representations and linguistic terms; and it operates by choosing few rules and working with imprecise terms [17].

The data provided by ML are fuzzified and separated into fuzzy groups. For each task feature, one fuzzy inference occurs. At the end of the system, the output is the completion of the task, as shown in Figure 4.

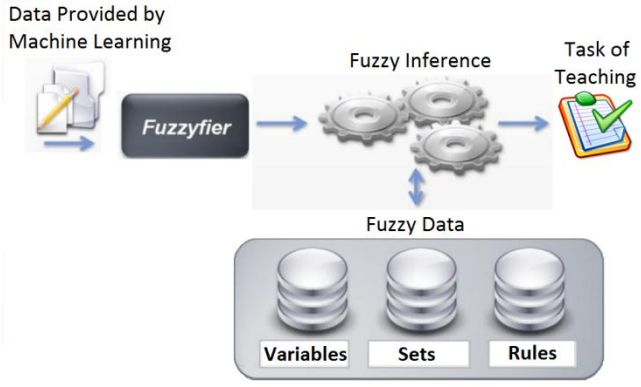


Figure 4. Macro view of the Fuzzy System.

5.1 Fuzzification

All the data processed and generated by ML, are abstracted and normalized in three fuzzy sets corresponding to a numerical range, from 0 to 100% and a degree of pertinence ranging from 0 to 100% [17], as shown in Figure 5. The fuzzy values represented in this approach correspond to the descriptors for the trapezoidal and triangular functions illustrated in Figure 5. In the present work all the fuzzy sets have these same values of classification. The representation of fuzzy set partitioning for a linguistic variable, showing the vertical and horizontal axes referring to the degree of pertinence values and fuzzy, respectively.

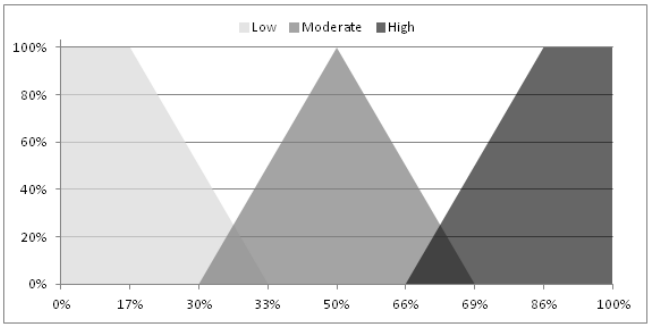


Figure 5. Graphical representation of fuzzy set.

This Fuzzification is done individually for each input variable of a Task Type, Number of Comparisons and Incorrect Words of a particular model of word, described in Table 1.

Table 1. Input variables of the fuzzy system.

Input Variable
PTT: Probability of hit with determined Task Type.
TTT: Hit Rate of Task Type.
PNC: Probability of hit with determined number of Comparisons.
TNC: Hit Rate of number of Comparisons.
PPI: Probability of hit with determined incorrect word.
TPI: Hit Rate with determined Incorrect Word.

5.2 Rule Sets

The architecture of Fuzzy Logic provided here is mapped out in a set of rules reflecting the ideas of specialists about the Study of Human Behavior project [18], as well as the interviews that were conducted for this study and the works outlined above.

The set of rules for this work is designed to create the appropriate learning tasks in an efficient manner, and also aims at encouraging the students, ensuring the game is kept enjoyable and stimulating.

If the tasks that are generated are too difficult for the player and he starts to miss too much, the player will not learn and may be discouraged and lose interest in the game. On the other hand, if there is too little difficulty for a player, he may also become discouraged, and will not be able to play the game to its full potential, and thus be mining the teaching process, which can delay his learning. From this perspective, the aim is to reach a balance between degrees of difficulty for each student.

The rules that are set (with the assistance of specialists) attempt to treat the input variables so that the choices of output variables, which are the characteristics of the new task, have an appropriate degree of difficulty. In addition, a further aspect of this logic is to ensure that the same feature does not appear too often and, therefore, prevents the tasks from becoming repetitive.

Each task performed by the ML player will pass on updated data about the student's progress, which means that the rules of the fuzzy system will always generate an appropriate task based on the updated data.

5.3 Fuzzy Inference

The fuzzy inference system utilized in this work uses the Mamdani model [19]. It corresponds to the algorithm of fuzzified information processing in accordance with linguistic rules [20] which are defined by specialists and research studies referred to. Table 2 correlates the input variables with the output in logical terms and the "if-then" form in causal terms. In the initial testing, the following rules were obtained and are arranged in Table 3 (a), (b) and (c) as follows: "**If** a variable column 1 = X **and** variable column 2 = Y **then** variable column 3 = Z".

Table 2. Output variables.

Output Variable
DTT: Need for Task Type.
DNC: Need for number of comparisons.
DPI: Need for incorrect word.

Table 3. Rules: (a) type of task, (b) number of comparisons and (c) misspelled words.

PTT	TTT	DTT
Low	Low	Low
Low	Moderate	Low
Low	High	Moderate
Moderate	Low	Moderate
Moderate	Moderate	High
Moderate	High	Moderate
High	Low	Moderate
High	Moderate	High
High	High	Low

(a)

PNC	TNC	DNC
Low	Low	Low
Low	Moderate	Low
Low	High	Moderate
Moderate	Low	Moderate
Moderate	Moderate	High
Moderate	High	Moderate
High	Low	Moderate
High	Moderate	High
High	High	Low

(b)

PPI	TPI	DPI
Low	Low	Low
Low	Moderate	Moderate
Low	High	Moderate
Moderate	Low	Low
Moderate	Moderate	High
Moderate	High	Moderate
High	Low	Moderate
High	Moderate	Low
High	High	Low

(c)

As an example that points to a rule set for the task type BC: "**If** the Probability of the Task Type BC **is** Low (pertinence 85%) **and** the Hit Rate of Task Type **is** High (63% pertinence) **then** the Need for Task Type BC **is** Medium (pertinence 74%)", as shown in Figure 6.

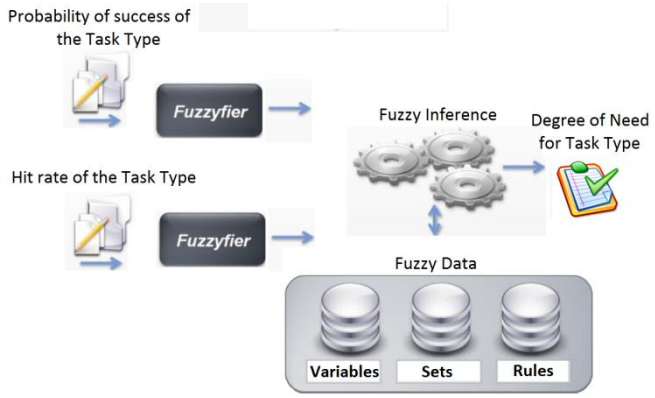


Figure 6. Application of fuzzy logic to the task type.

5.4 Decision-making

Each activation of the rules is analyzed separately, and then the linguistic value of fuzzy set assigned as Highest is chosen as the best option for each feature of the new task that is being generated.

The logic presented above is applied to establish Type of Task, the number of incorrect comparisons and choice of words. For the words choices, instead of only one value, the algorithm that chooses the words returns a list with *n* words, where *n* is previously defined by the fuzzy logic responsible for deciding the number of comparisons.

There is no need, in this work, to defuzzify the output variables because the choice is given by the highest degree of pertinence as stated previously. This leads to the generation of task features for the subsequent level of the game. It should be decided which task type is better to choose.

6 RESULTS

The work is still in its initial phase and to validate the operation and efficiency of the proposal, the Fuzzy System algorithms were implemented.

The ALEPP and GEIC teaching programs are currently being tested at schools in the city of São Carlos, SP - Brazil. This database, which has records of the teaching program for each child, was used to yield the ML results. It generated data for the initial tests in the fuzzy system. These tests were conducted under supervision of specialists, as well as on the basis of a results analysis.

The simulations were performed on three groups of students: i) Students with Learning Deficit, ii) Students with Gradual Learning and iii) Students with Consolidated Learning.

The system was tested in a teaching unit containing fifteen words used by GEIC and with the same teaching structure, being them: *bolo* (cake), *tatu* (armadillo), *vaca* (cow), *bico* (beak), *mala* (bag), *tubo* (tube), *pipa* (kite), *cavalo* (horse), *apito* (whistle), *luva* (gloves), *tomate* (tomato), *vovô* (grandpa), *muleta* (crutch), *fita* (tape) *e pato* (duck).

Of the fifteen words that correspond to the teaching session called pre-test, five were analyzed about the literacy. These five words are: *bolo* (cake), *tatu* (armadillo), *apito* (whistle), *tomate* (tomato) *e muleta* (crutch). Were also analyzed by experts the fifteen adaptative tasks generated by the fuzzy system.

To validate the system, three experts in the field of behavioral analysis of the UFSCar were submitted to a questionnaire in order to classify the ML evaluation of learning of the each word. On the rating scale used (1 = Poor, 2 = Regular, 3 = Good, 4 = Very Good, 5 = Excellent) the experts' rating were "Good". According to that, the following characteristics were evaluated: i) Degree of Learning and ii) if it has been learned or not.

After this analysis, an interview was made with the same experts. Has been discussed about the generation of the new task that was considered suitable to the "Good" rating scale.

The system was validated by specialists, and usually led to the creation of a task that could be the same task designed by a specialist.

7 CONCLUSION AND FUTURE STUDIES

This work is part of a project that proposes the use of Intelligent Agents attached to computer games and targeted at the learning of reading and writing. This part of the system being researched aims at exploring the question of decision-making when there is uncertainty about the real need for a particular person to perform a given task type.

The first results were obtained by processing data from the database project where there is a record of how the GEIC program was implemented. With this information it was possible to simulate results to allow the ML to convey on information to the Fuzzy System, and then it was validated in a satisfactory way by specialists in the field of psychology. However, it is still necessary to make adjustments to the variables, address other tasks, examine other fuzzy variables and also cover the tests directly in the computer games with the students running the system in real time.

It is concluded that the project is viable, with positive results encouraging us to continue expanding the research to enable the game to teaching writing and seeking improvements for it.

Despite obtaining satisfactory results, there is still a need to carry out tests in a complete game running in real time. Future developments will focus on employing the fuzzy system to generate writing tasks, and thus teach the player to read and write simultaneously. It is also expected that there can be an extension of the analysis by the ML. At the same time we must include AIED in the game that is under development. The game that is currently

under development is being implemented in a Game Engine called Unity3D. This tool allows various games to be generated for platforms such as PC, Xbox360, Web HTML, Flash, iOS and Android. The goal is to ensure that the end of the game runs on a tablet with Android.

REFERENCES

[1] L. Xiangfeng, W. Xiao and Z. Jun 'Guided Game-Based Learning', *Published by IEEE Transactions on Learning Technologies*, (2010).

[2] B.du Benedict. 'What does the "AI" in AIED buy?*' Printed and published by IEE*, Savoy Place, London WC2R 0BL, U.K., 1997.

[3] S. Tobias, "Interest, Prior Knowledge, and Learning," Rev. of Educational Research,vol. 64, no. 1, pp. 37-54, 1994

[4] J. Brophy, "Teacher Influences on Student Achievement," Am. Psychologist,vol. 41, no. 10, pp. 1069-1077, 1986.

[5] D. Dormans and B. Sander, *'Generating Missions and Spaces for Adaptable Play Experiences",* Published by *IEEE Transactions on Computational Intelligence AI in Games*, 2011.

[6] L. B. Marques, R. G. Meio, R. M. Maria, '*Manual do Usuário de Programas de Ensino via GEIC*' - Volume 1: '*Aprendendo a Ler e Escrever em Pequenos Passos'*. São Carlos, 2011.

[7] M. A. Azevedo, M. L Marques, '*Alfabetização hoje*'. São Paulo: Cortez, 2001.

[8] E. S. Sarmanho, E. B. Sales, D. M. Cavalcante, L. B. Marques, '*Um Jogo com Reconhecedor de Voz para o Ensino de Crianças com Dificuldade de Aprendizagem em Leitura e Escrita'*. Published by *Semish*, 2011.

[9] E. Schaefer, D. Brevik, M. Schaefer, E. Sexton, and K. William, "Diablo," 1996", Blizzard North.

[10] T. Baldree, "Torchlight," 2009, Runic Games.

[11] W.Wright, "Spore," 2008, Maxis.

[12] M. Person and J. Bergensten, "Minecraft," 2009, Mojang.

[13] J. C. Rose, D. G. Souza, A. L. Rossito, T. M. S. Rose, '*Aquisição de leitura após história de fracasso escolar: equivalência de estímulos e generalização*'. In: *Psicologia: Teoria e Pesquisa*, p.451-69. 1989.

[14] de Souza, D. G., de Rose, J. C., Faleiros, T. C., Bortoloti, R., Hanna, E. S., & McIlvane, W. J. Teaching Generative Reading Via Recombination of Minimal Textual Units: A Legacy of Verbal Behavior to Children in Brazil. *Revista Internacional De Psicologia Y Terapia Psicologica - International Journal of Psychology and Psychological Therapy*, *9*(1), pp 19–44, 2009.

[15] T. S. Reis, D. G. Souza, J. C. Rose, '*Avaliação de um programa para o ensino de leitura e escrita*'. In: *Estudos em Avaliação Educacional*, 20, p.425-50. 2009.

[16] C. D. Pedro, J. O. Adriano, 'Aprendizado de Regras Nebulosas em Tempo Real para Jogos Eletrônicos'. *XI Brazilian Symposium of Multimedia Systems and Web. Games – II Brazilian Workshop of Games and Digital Entertainment,* 2003.

[17] P. B. Moratori, M. V. Pedro, L. M. B. Manhaes, C. Lima, A. J. O. Cruz, E. B. Ferreira, and L. C. V. de Andrade. 'Analysis of the Stability of a Fuzzy Control System Developed to Control a Simulated Robot,' Fuzzy Systems, 2005. *The 14th IEEE International Conference on Fuzzy*, pp.726-730, 25-25 May 2005.

[18] L.B. MARQUES, '*Variáveis Motivacionais no Ensino de Leitura: O jogo como recurso complementar*', ed. São Carlos: UFSCar, 2009.

[19] Mamdani, E.H., "Advances in the linguistic synthesis of fuzzy controllers," *International Journal of Man-Machine Studies*, Vol. 8, pp. 669-678, 1976.

[20] R. J. Timothy Fuzzy Logic with Engineering Applications, Third Edition", ISBN: 047074376X, Wi ey, 5, 117-148, 2010.

An Explanation Mechanism for Integrated Rules

Jim Prentzas[1] and **Ioannis Hatzilygeroudis**[2]

Abstract. Neurules are a type of integrated rules combining a symbolic and a connectionist representation. Each neurule is represented as an adaline unit. A neurule base consists of a number of autonomous adaline units (neurules). In contrast to other neuro-symbolic knowledge bases, naturalness and modularity is retained in neurule bases. In this paper, we present a neurule-based explanation mechanism to provide explanations for reached conclusions. The explanation mechanism is integrated in its nature by combining neurocomputing with symbolic processes. Therefore, the provided explanations are more natural compared to those offered by connectionist expert systems. As shown by experiments, the presented explanation mechanism provides explanations more efficiently compared to corresponding explanation mechanisms used in connectionist expert systems.

1 INTRODUCTION

Many approaches combining different Artificial Intelligence (AI) methods have been presented [1], [2], [3]. The presented approaches exploit the advantages of the combined methods.

Integrations of symbolic and connectionist approaches have given fruitful results [4], [5], [6], [7], [8]. One research direction regarding this type of integration uses prior domain knowledge to contribute in configuring a neural network, either by implementing the meaning function of a logic program (e.g. [9], [7]) or by helping in constructing the initial architecture of a neural network (e.g. [10], [11], [12]). Another research direction, related to connectionist expert systems, provides a type of integrated systems that represent relationships between concepts, considered as nodes of a neural network [13], [14], [15]. All the above approaches give pre-eminence to the connectionist component of the integration by mapping symbolic rules, concepts or processes to a neural network. By giving pre-eminence to connectionism, naturalness and modularity of symbolic rules as well as interactive inference and provision of explanations are either missing or greatly reduced.

Neurules are a type of integrated rules achieving a uniform and tight integration of a symbolic component (production rules) and a connectionist one (adaline unit) [16], [17], [18]. Neurules give pre-eminence to the symbolic component. Thus, the constructed knowledge base retains modularity and (to some degree) naturalness of symbolic rules. Neurules possess an interactive, integrated inference mechanism, which is proved to be more efficient than other actually pure connectionist mechanisms [19]. Methods performing efficient updates of a neurule base in order to reflect changes to the corresponding source knowledge have also been developed [20], [21].

In this paper, we present a neurule-based explanation mechanism. The given explanations are in the form of if-then rules and explain how a conclusion has been reached. We compare our explanation mechanism with the explanation mechanism of connectionist expert systems. Our provided explanations are more natural and also computationally less expensive.

The paper is organized as follows. Section 2 briefly presents neurules, mechanisms for their construction from training examples or existing symbolic rules and reasoning mechanisms. Section 3 discusses main aspects of the explanation mechanism. Section 4 presents experimental results and finally Section 5 concludes.

2 NEURULES: SYNTAX AND SEMANTICS

Neurules (: neural rules) are a kind of hybrid rules. Each neurule (Fig. 1a) is considered as an adaline unit (Fig.1b). The *inputs* C_i (i=1,...,n) of the unit are the *conditions* of the rule. Each condition C_i is assigned a number sf_i, called a *significance factor*, corresponding to the weight of the corresponding input of the adaline unit. Moreover, each rule itself is assigned a number sf_0, called the *bias factor*, corresponding to the bias of the unit. Each input takes a value from the following set of discrete values: [1 (true), -1 (false), 0 (unknown)]. The *output* D, which represents the *conclusion* of the rule, is calculated via the formulas:

$$D = f(\mathbf{a}), \quad \mathbf{a} = sf_0 + \sum_{i=1}^{n} sf_i \; C_i \qquad (1)$$

where **a** is the *activation value* and *f(x)* the *activation function*, which is a threshold function:

$$f(a) = \begin{cases} 1 & \text{if } a \geq 0 \\ -1 & \text{otherwise} \end{cases}$$

The output can thus take one of two values, '-1' and '1', representing failure and success of the rule respectively.

The general syntax of a neurule (in a BNF notation, where '{}' denotes zero, one or more occurrences and '<>' denotes non-terminal symbols) is:

<rule>::= (<bias-factor>) if <conditions> then <conclusions>
<conditions>::= <condition> {, <condition>}
<conclusions>::= <conclusion> {, <conclusion>}
<condition>::= <variable> <l-predicate> <value>
(<significance-factor>)
<conclusion>::= <variable> <r-predicate> <value> .

In the above definition, <variable> denotes a variable, that is a symbol representing a concept in the domain, e.g., 'sex', 'pain' etc,

[1] Democritus University of Thrace, School of Education Sciences, Dept. of Education Sciences in Pre-School Age, Laboratory of Informatics, Nea Chili, 68100 Alexandroupolis, Greece, email: dprentza@psed.duth.gr
[2] University of Patras, School of Engineering, Dept. of Computer Engineering & Informatics, 26500 Patras, Greece, ihatz@ceid.upatras.gr

in a medical domain. <l-predicate> denotes a symbolic or a numeric predicate. The symbolic predicates are {is, isnot}, whereas the numeric predicates are {<, >, =}. <r-predicate> can only be a symbolic predicate. <value> denotes a value. It can be a symbol or a number. <bias-factor> and <significance-factor> are (real) numbers. The significance factor of a condition represents the significance (weight) of the condition in drawing the conclusion. For the sake of simplicity, in the following, we consider 'is' and 'isnot' as the only <l-predicate> and <r-predicate>, without loss of generality, since the others can be somehow expressed via them.

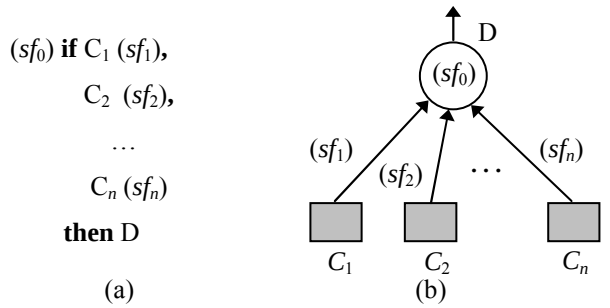


Figure 1. (a) Form of a neurule (b) corresponding adaline unit

Neurules are constructed either from empirical data (i.e. training examples) [17] or from existing symbolic rules [16]. In either process, an adaline unit is initially assigned to each of the intermediate and final conclusions. Each unit is individually trained via the Least Mean Square (LMS) algorithm. If the patterns in the training set of a neurule form a non-separable set, special techniques are used. In that case, more than one neurule having the same conclusion are produced. When neurules are produced from symbolic rules, each neurule usually corresponds to (or merges) a set of symbolic rules.

We use the following notation for a neurule R_k:

- $sf_0^{R_k}$: the bias factor of R_k.
- $C_i^{R_k}$: the ith condition of R_k, ($C_i^{R_k} \equiv$ "$V_i^{R_k}$ is $v_i^{R_k}$ ").
- $V_i^{R_k}$: the variable involved in $C_i^{R_k}$.
- $sf_i^{R_k}$: the significance factor of $C_i^{R_k}$.
- D^{R_k} : the conclusion of R_k ($D^{R_k} \equiv$ "$V_d^{R_k}$ is $v_d^{R_k}$ ").
- $V_d^{R_k}$: the variable involved in D^{R_k} .
- $v_d^{R_k}$: the value involved in D^{R_k} .

Let us consider a finite set of *domain variables* V = $\{V_i\}$, $1 \leq i \leq m$, which represent the concepts of the problem domain involved in making inferences. Each variable V_i can take values from a (corresponding) set of discrete values $S_{V_i} = \{v_{ij}\}$, $1 \leq j \leq k$. *Input variables* are those whose values are given by the user as input to the system. The set of input variables is denoted by V_{INP}. *Intermediate variables* represent concepts related to intermediate conclusions that are conclusions inferred via some rules, so their values depend on the values of input variables or other intermediate variables. The set of intermediate variables is denoted by V_{INT}. *Goal* (or *output*) *variables* are related to final conclusions, so their values depend on the values of input or intermediate variables and constitute the output of the system. The set of goal variables is denoted by V_G. Intermediate and goal variables are called *inferable variables*. The set of inferable variables V_{INF} is defined as: $V_{INF} = V_{INT} \cup V_G$. Of course: V_{INP}, V_{INT}, V_G and $V_{INF} \subset$ V. We define *goal (or output) neurules* the ones having $V_d^{R_k} \in V_G$. The set of goal neurules is denoted by R_G.

Evaluation of conditions that refer to the same variable is done at the same time. We call those conditions 'sibling' conditions:

A condition C_i can evaluate to TRUE, FALSE or UNKNOWN. Conditions are evaluated based on either the contents of the working memory (*WM*) or the user responses or firings of rules. WM contains *fact assertions*:

$$WM = \{(F_i, ass(F_i)), i=1, n\}$$

where $F_i \equiv$ "V_{F_i} is v_{F_i}" is a fact and $ass(F_i) \in$ {TRUE, FALSE, UNKNOWN} is the assertion value related to it. Fact assertions are either given by the user, as initial input data or during an inference course, or produced by the system, as intermediate/final conclusions during inference.

Evaluation of a condition, when based on WM contents, is done via the following rules:

- A condition C_i evaluates to COND-VALUE, denoted by $ass(C_i)$ = COND-VALUE, if there is a fact assertion (F_i, $ass(F_i)$) in WM with $F_i \equiv C_i$ and $ass(F_i)$ = COND-VALUE. COND-VALUE $\in$ {TRUE, FALSE, UNKNOWN}.

At any time, of an inference process, we distinguish between two sets of conditions for a neurule R_k: the *set of evaluated conditions* $C_E^{R_k}$ and the *set of unevaluated conditions* $C_U^{R_k}$. $C_E^{R_k}$ refers to conditions that have already been evaluated and the results of their evaluation have been taken into account in deducing the conclusion of R_k, whereas $C_U^{R_k}$ to the rest ones. We denote by $value(V_i)$ the value acquired for variable V_i during inference. Whenever a user is asked to evaluate a condition $C_i \equiv$ "V_i is v_i" (where $V_i \in V_{INP}$) he/she is actually called to assign a value $value(V_i)$ to the corresponding variable V_i, where $value(V_i) \in S_{Vi} \cup$ {UNKNOWN}. We denote by $assv(C_i^{R_k})$ the contribution of a condition to the activation value of a neurule R_k and define it as follows:

$$assv(C_i^{R_k}) = \begin{cases} 1 & \text{if} \quad ass(C_i^{R_k}) = \text{TRUE} \\ -1 & \text{if} \quad ass(C_i^{R_k}) = \text{FALSE} \\ 0 & \text{if} \quad ass(C_i^{R_k}) = \text{UNKNOWN} \end{cases}$$

The neurule-based inference engine is also integrated, based on a backward chaining strategy [19]. The choice of the next neurule to be considered during inference is based on a neurocomputing measure (i.e. *firing ratio*). Conclusions are reached based on the values of the condition variables and the weighted sums of the conditions. A neurule *fires* if the output of the corresponding adaline unit is computed to be '1' after evaluation of its conditions. A neurule is said to be *blocked* if the output of the corresponding adaline unit is computed to be '-1' after evaluation of its conditions. In both cases the neurule is considered *evaluated*. The set of fired rules during an inference process is denoted by R_F, whereas that of blocked rules by R_B. Also, the set of evaluated rules is denoted by R_E and that of unevaluated by R_U. To facilitate inference, conditions of neurules are organized according to the descending order of their significance factors. When a neurule is examined during inference, certain heuristics are applied to avoid evaluation of all its conditions [19].

3 EXPLANATION MECHANISM

The explanation mechanism justifies inferences by producing a set of if-then rules, explaining how conclusions were reached. Explanation rules are derived from evaluated neurules by selecting some of their evaluated conditions and by making possible changes to their predicates as well as to predicates of the neurules' conclusions. Conclusions of explanation rules correspond to final conclusions reached and to intermediate conclusions contributing to drawing final conclusions. Conditions of explanation rules correspond to a subset of the input and intermediate variables participating in drawing the conclusions.

For explanation purposes, the conditions of an evaluated neurule are distinguished in *positive* and *negative conditions*. This distinction is based on the positive or negative contribution of the conditions in evaluating the corresponding neurules.

Definition 1. An evaluated neurule's R_k set of positive conditions is defined as:

$$posConds(R_k)=\begin{cases}\left\{\begin{array}{l}C_i^k:\left(ass\left(C_i^k\right)=TRUE\wedge sf_i^k>0\right)\\ \vee\left(ass\left(C_i^k\right)=FALSE\wedge sf_i^k<0\right)\end{array}\right\}, \text{if } R_k\in R_F\\ \left\{\begin{array}{l}C_i^k:\left(ass\left(C_i^k\right)=TRUE\wedge sf_i^k<0\right)\\ \vee\left(ass\left(C_i^k\right)=FALSE\wedge sf_i^k>0\right)\end{array}\right\}, \text{if } R_k\in R_B\end{cases}$$

Definition 2. An evaluated neurule's R_k *set of negative conditions* is defined as:

$$negConds(R_k)=\begin{cases}\left\{\begin{array}{l}C_i^k:\left(ass\left(C_i^k\right)=TRUE\wedge sf_i^k<0\right)\\ \vee\left(ass\left(C_i^k\right)=FALSE\wedge sf_i^k>0\right)\end{array}\right\}, \text{if } R_k\in R_F\\ \left\{\begin{array}{l}C_i^k:\left(ass\left(C_i^k\right)=TRUE\wedge sf_i^k>0\right)\\ \vee\left(ass\left(C_i^k\right)=FALSE\wedge sf_i^k<0\right)\end{array}\right\}, \text{if } R_k\in R_B\end{cases}$$

Conditions that are unevaluated, unknown or negative are not included in explanation rules, as non contributing. Furthermore, some of the positive conditions may be also not included, based on the fact that they are not necessary in evaluating the neurule. More specifically, the conditions of an explanation rule are the ones having the most positive contribution in evaluating the corresponding neurule possibly with changes to their predicates. We call such conditions the *necessary positive conditions* of an evaluated neurule R_k and denote the corresponding set by *necSet*(R_k).

For each $C_i^k \in necSet\ (R_k)$, the following changes are made to the corresponding condition in the explanation rule produced from R_k:

- Conditions in the explanation rule do not contain significance factors so sf_i^k is omitted.
- If $ass\left(C_i^k\right)= FALSE$, the predicate of the condition changes from 'is' to 'isnot' and vice versa.

Let $C_i'^k$ be the condition in the explanation rule corresponding to $C_i^k \in necSet\ (R_k)$, produced after making the aforementioned change(s) to C_i^k. The conclusion of an explanation rule produced from an evaluated neurule R_k corresponds to the conclusion D^k of R_k by possibly changing it. More specifically:

- If R_k is fired, the conclusion of the produced explanation rule is the same as D^k.
- If R_k is blocked, the predicate of the conclusion changes from 'is' to 'isnot' and vice versa.

Let D'^k be the conclusion produced from D^k after making the aforementioned change(s).

Definition 3. The explanation rule HR_k produced from an evaluated neurule R_k is an if-then rule without sf_0^k and whose condition set and conclusion are defined as follows:

$$conds\ (HR_k)=\left\{C_i'^k : C_i^k \in necSet\ (R_k)\right\}$$
$$concl\ (HR_k)=D'^k$$

The following process is followed in order to produce an explanation rule HR_k from an evaluated neurule R_k:

1. Set $necSet\ (R_k)=\left\{C_i^k : C_i^k \in posConds\ (R_k)\right\}$,
   $unnecSet\ (R_k)=\{\ \}$
   $$pos\_sum = sf_0^k + \sum_{C_i^k \in posConds\ (R_k)} sf_i^k assv\left(C_i^k\right),$$
   $$neg\_sum = \sum_{C_i^k \in negConds\ (R_k)} sf_i^k assv\left(C_i^k\right) + \sum_{ass\left(C_i^k\right)=UNKNOWN} sf_i^k assv\left(C_i^k\right) + \sum_{C_i^k \in C_U^{R_k}} \left|sf_i^k\right|$$
2. While ($|pos\_sum|>|neg\_sum|$) and ($|necSet(R_k)|>1$) do
   Let
   $$C_i^k : C_i^k \in necSet\ (R_k) \wedge sf_i^k = \min_{C_i^k \in necSet\ (R_k)} \left\{\left|sf_i^k\right|\right\}$$
   $$pos\_sum = pos\_sum - assv\left(C_i^k\right) sf_i^k$$
   $$neg\_sum = neg\_sum + \left|sf_i^k\right|$$
   If ($|pos\_sum|>|neg\_sum|$) then
   $$necSet\ (R_k) = necSet\ (R_k) / \left\{C_i^k\right\}$$
   $$unnecSet\ (R_k) = unnecSet\ (R_k) \cup \left\{C_i^k\right\}$$
3. For each $C_i^k \in necSet\ (R_k)$ do
   For each $C_j^k \in necSet\ (R_k)$ with $i \neq j$ do
   If $\left(V_i^k \equiv V_j^k\right)$ and $\left(predicate\ \left(C_i^k\right)='is'\right)$ then
   If $\left(predicate\ \left(C_j^k\right)='is'\right)$ and
   $ass\left(C_i^k\right)= TRUE$ and $ass\left(C_j^k\right)= FALSE$ then
   $$necSet\ (R_k) = necSet\ (R_k) / \left\{C_j^k\right\}$$
   Else If $\left(predicate\ \left(C_j^k\right)='isnot'\right)$ and
   $ass\left(C_i^k\right)= TRUE$ and $ass\left(C_j^k\right)= TRUE$ then
   $$necSet\ (R_k) = necSet\ (R_k) / \left\{C_j^k\right\}$$
   Else If $\left(predicate\left(C_j^k\right)='isnot'\right)$ and
   $ass\left(C_i^k\right)= FALSE$ and $ass\left(C_j^k\right)= FALSE$ then
   $$necSet\ (R_k) = necSet\ (R_k) / \left\{C_i^k\right\}$$
4. Set $conds\ (HR_k)=\left\{C_i'^k : C_i^k \in necSet\ (R_k)\right\}$,
   $concl\ (HR_k)=D'^k$

5. Return HR_k

It should be pointed out that step 3 is necessary in the above process because when a set of sibling conditions is contained in $necSet(R_k)$ not all of these conditions have something meaningful to offer.

Following is an outline of the explanation mechanism. We use the function *max_fr_rule*(R) which returns the neurule in R that has the maximum absolute activation value. For each condition in an explanation rule containing an intermediate variable, an if-then explanation rule is produced based on an evaluated neurule having that condition as its conclusion. The predicate of the condition is the one before its possible change from 'is' to 'is-not' and vice versa took place. More than one neurule may have the condition as their conclusion. If the condition was evaluated to true, only one of these neurules has its output evaluated to '1' and is the chosen one. If the condition was evaluated to false, then the outputs of all neurules having it as their conclusion were evaluated to '-1'. In this case, the neurule with the maximum absolute activation value will be chosen.

1. For each $R_k \in R_G \cap R_F$ do
   1.1. Produce HR_k
   1.2. For each $C_i'^k \in conds(HR_k)$ with $V_i^k \in V_{INF}$ do
      1.2.1. If $ass(C_i^k) = TRUE$ then
         Let $R_j \in R_F$ with $V_d^j \equiv V_i^k$, $v_d^j \equiv v_i^k$ and
         $predicate\ (D^j) \equiv predicate\ (C_i^k)$
      Else If $ass(C_i^k) = FALSE$ then
         $TempSet = \{ R_j : R_j \in R_B \wedge V_d^j \equiv V_i^k \wedge v_d^j \equiv v_i^k \wedge predicate\ (D^j) \equiv predicate\ (C_i^k) \}$
         Let R_j = *max_fr_rule*(*TempSet*)
      1.2.2. Produce HR_j
         Execute Step 1.2 recursively for HR_j

4 EXAMPLES

We use as an example to illustrate the explanation mechanism the one presented in [14]. It contains training data dealing with acute theoretical diseases of the sarcophagus. There are six symptoms (Swollen feet, Red ears, Hair loss, Dizziness, Sensitive aretha, Placibin allergy), two diseases (Supercilliosis, Namastosis) whose diagnoses are based on the symptoms and three possible treatments (Placibin, Biramibio, Posiboost). Also, dependency information is provided. We used the dependency information to construct the initial neurules and the training data provided to train them. The produced knowledge base, which contains five neurules (NR1-NR5), is illustrated in Fig. 2. An equivalent knowledge base forming a multilevel network is presented in [13], [14].

Let us suppose that inference runs given the initial data {'Symptom is HairLoss' (TRUE)}. The final results of the inference process are the following:

- Fired neurules: {NR5, NR3, NR1}
- Blocked neurules: {NR4, NR6}
- WM: {'Symptom is HairLoss' (TRUE),'Treatment is Biramibio' (FALSE), 'Symptom is SwollenFeet' (FALSE), 'Symptom is RedEars' (FALSE), 'Disease is Supercilliosis' (TRUE), 'Symptom is PlacibinAllergy' (FALSE), 'Treatment is Placibin' (TRUE), 'Treatment is Posiboost' (TRUE)}.

NR1: (-0.4) **if** Symptom is RedEars (-0.8),
Symptom is SwollenFeet (3.6),
Symptom is HairLoss (3.6)
then Disease is Supercilliosis

NR2: (1.4) **if** Symptom is Dizziness (4.6),
Symptom is SensitiveAretha (1.8),
HairLoss is true (1.8)
then Disease is Namastosis

NR3: (-2.2) **if** Symptom is PlacibinAllergy (-5.4),
Disease is Supercilliosis (4.6)
Disease is Namastosis (1.8),
then Treatment is Placibin

NR4: (-4.0) **if** Symptom is HairLoss (-3.6),
Disease is Namastosis (3.6),
Disease is Supercilliosis (2.8)
then Treatment is Biramibio

NR5: (-2.2) **if** Treatment is Biramibio (-2.6),
Treatment is Placibin (1.8)
then Treatment is Posiboost

NR6: (-2.2) **if** Treatment is Placibin (-1.8),
Treatment is Biramibio (1.0)
then Treatment is Posiboost

Fig. 2. An example knowledge base

We now present the explanation that will be produced by the explanation mechanism for the above inference example. Given that there are two outputs, 'Treatment is Posiboost' and 'Treatment is Placibin', the explanation mechanism should provide explanations for them (here we present only for the first).

In order to generate an explanation rule for the first output, the explanation mechanism will examine neurule NR6 whose output is 'Treatment is Posiboost' and was evaluated to '1' (TRUE). The explanation rule extracted is the following:

EXR1: **if** Treatment isnot Biramibio,
Treatment is Placibin
then Treatment is Posiboost.

The tracing of the generation of the above explanation rule is as follows:

nec-set(NR6): {'Treatment is Biramibio', 'Treatment is Placibin'}
unnec-set(NR6): { }
pos-sum: -2.2 + 2.6 + 1.8 = 2.2
neg-sum: 0
Since (|pos-sum| > |neg-sum| and |nec-set(NR6)|>1):
pos-sum: 2.2 - 1.8 = 0.4
neg-sum: 1.8
Since (|pos-sum| < |neg-sum|):
nec-set: {'Treatment is Biramibio', 'Treatment is Placibin'}.
unnec-set: { }

Due to the fact that 'Treatment is Biramibio' was evaluated to false, we change its predicate from 'is' to 'isnot'.

Due to the fact that the explanation rule EXR1 contains an intermediate variable, a corresponding explanation rule should be generated for it (with conclusion 'Treatment isnot Biramibio'). To this end, the blocked neurule NR4 is examined and the following explanation rule is generated:

EXR2: **if** Symptom is HairLoss
then Treatment isnot Biramibio

The tracing of the generation of the above explanation rule is as follows:

nec-set: {'HairLoss is true'}

Since nec-set has only one condition, it remains as it is. Furthermore, since NR4 was blocked, we change the conclusion predicate of its derived explanation rule from 'is' to 'isnot'.

5 EXPERIMENTAL RESULTS

We have used four datasets to test the neurule-based explanation mechanism. The first dataset (called ACUTE) is the one used as an example in the previous section and is taken from [13], [14]. The dataset is incomplete. It consists of 8 input data patterns out of 64 possible. For most of the input combinations no final conclusion can be drawn since all the output cells of the connectionist knowledge base become false and all the output rules of the corresponding neurule base become blocked. We give results for 27 input combinations for which final conclusion(s) can be drawn that is, one or more output cells of the connectionist knowledge base become true and one or more output rules of the corresponding neurule base fire. The second dataset (called LENSES) is taken from the UCI Machine Learning Repository (MLR) [22] and regards a database for fitting contact lenses. This dataset is complete and contains 24 input patterns each consisting of four input and one output attribute (variable). The third dataset (called CAR) is also from the UCI MLR. However, we also used the dependency information provided by the donators of the dataset [23], which is not included in the MLR. The above datasets were chosen to satisfy at least one of the requirements set: (a) be categorical, (b) there is dependency information available.

Table 1 presents some experimental results comparing the explanation process associated with neurules with the corresponding explanation process used in connectionist expert systems. The explanations produced are the ones justifying the conclusions reached by the neurule-based inference mechanism and the inference mechanism used in connectionist expert systems. Comparison is made in terms of the number of explanation rules produced and the time required to produce them. Column 'EXRUL-NUM' contains the mean number of explanation rules produced. Column 'TIME' contains the mean required computational time to produce the explanation rules.

As can be seen from the table, the number of explanation rules produced by our explanation mechanism is less than those produced by the connectionist expert system. So, the time required is less too. Furthermore, our rules are more natural than those of connectionist expert systems. The above disadvantages of the connectionist expert system are mainly due to the use of some extra cells, the so-called 'random cells' in constructing it [14]. Those cells are used in the explanation rules too, but since they have no concepts assigned to them are meaningless.

Table 1. Experimental results for the explanation mechanism

Datasets	*Neurules*		*Connectionist expert system*	
	EXRUL-NUM	*TIME (msec)*	*EXRUL-NUM*	*TIME (msec)*
ACUTE (27)	3.07	0.0125	4.30	0.0156
LENSES (24)	1	0.0104	2.20	0.0208
CAR-DEP (1728)	4.30	0.0438	7.40	0.0831

6 CONCLUSIONS

In this paper, we present the explanation mechanism of neurules, a type of integrated rules, integrating symbolic rules with neurocomputing. In contrast to other neuro-symbolic methods, neurules follow a different approach in combination of the symbolic and the connectionist approach. More specifically, connectionism is incorporated within the symbolic framework, not the other way round. The neurule-based explanation mechanism is also integrated leading into more natural explanations. A further advantage of the neurule-based explanation mechanism compared to the explanation mechanism used in connectionist expert systems is that less explanation rules and computational time are required for the provision of explanations. Our future work is directed to providing other types of explanations. Such types of explanations, among others, involve why a value of a specific input variable is asked and why a specific inferable variable has not acquired a specific value during inference.

REFERENCES

[1] L.R. Medsker, *Hybrid Intelligent Systems*, Kluwer Academic Publishers, second printing, 1998.

[2] I. Hatzilygeroudis and J. Prentzas (Eds.), *Combinations of Intelligent Methods and Applications*, Smart Innovation, Systems and Technologies, Vol. 8, Springer–Verlag, Berlin, 2011.

[3] J. Prentzas and I. Hatzilygeroudis, 'Combinations of Case-Based Reasoning with Other Intelligent Methods', *International Journal of Hybrid Intelligent Systems*, **6**, 189-209, (2009).

[4] K. McGarry, S. Wermter and J. MacIntyre, Hybrid Neural Systems: From Simple Coupling to Fully Integrated Neural Networks. *Neural Computing Surveys*, **2**, 62–93, (1999).

[5] I. Hatzilygeroudis and J. Prentzas, 'Neuro-Symbolic Approaches for Knowledge Representation in Expert Systems', *International Journal on Hybrid Systems*, **1**, 111–126, (2004).

[6] S. Bader, and P. Hitzler, *Dimensions of Neural-Symbolic Integration – A Structured Survey*, 167–194, We Will Show Them: Essays in Honour of Dov Gabbay, 1, International Federation for Computational Logic, College Publications, 2005.

[7] A. Garcez, L.C. Lamb and D.M. Gabbay, *Neural-Symbolic Cognitive Reasoning*, Springer–Verlag, Berlin, 2008.

[8] B. Hammer and P. Hitzler (eds.), *Perspectives of Neural–Symbolic Integration*, Springer-Verlag, Berlin, 2007.

[9] S. Bader, P. Hitzler and S. Holldobler, 'Connectionist Model Generation: A First-Order Approach', *Neurocomputing*, **71**, 2420-2432, (2008).

[10] G. Towell and J. Shavlik, 'Knowledge-Based Artificial Neural Networks', *Artificial Intelligence*, **70**, 119–165, (1994).

[11] L.M. Fu, 'Knowledge-Based Connectionism for Revising Domain Theories', *IEEE Transactions on Systems, Man, and Cybernetics*, **23**, 173–182, (1993).

[12] T.-H. Teng, Z.-M. Tan and A.-H. Tan, *Self-Organizing Neural Models Integrating Rules and Reinforcement Learning*, 3771–3778, IEEE International Joint Conference on Neural Networks, 2008.
[13] S.I. Gallant, 'Connectionist Expert Systems', *Communications of the ACM*, **31**, 152–169, (1988).
[14] S.I. Gallant, *Neural Network Learning and Expert Systems*, MIT Press, Cambridge, MA, 1993.
[15] A.Z. Ghalwash, 'A Recency Inference Engine for Connectionist Knowledge Bases', *Applied Intelligence*, **9**, 201–215, (1998).
[16] I. Hatzilygeroudis and J. Prentzas, 'Neurules: Improving the Performance of Symbolic Rules', *International Journal on AI Tools*, **9**, 113–130, (2000).
[17] I. Hatzilygeroudis and J. Prentzas, 'Constructing Modular Hybrid Knowledge Bases for Expert Systems', *International Journal on AI Tools*, **10**, 87–105, (2001).
[18] J. Prentzas, I. Hatzilygeroudis, *Neurules – A Type of Neuro-Symbolic Rules: An Overview*, 145–165, Combinations of Intelligent Methods and Applications, Smart Innovation, Systems and Technologies, 8, Springer–Verlag, Berlin, 2011.
[19] I. Hatzilygeroudis and J. Prentzas, 'Integrated Rule Based Learning and Inference', *IEEE Transactions on Knowledge and Data Engineering*, **22**, 1549–1562, (2010).
[20] J. Prentzas and I. Hatzilygeroudis, 'Rule-Based Update Methods for a Hybrid Rule Base', *Data and Knowledge Engineering*, **55**, 103–128, (2005).
[21] J. Prentzas and I. Hatzilygeroudis, 'Incrementally Updating a Hybrid Rule Base Based on Empirical Data', *Expert Systems*, **24**, 212–231, (2007).
[22] A. Asuncion and D.J. Newman, *UCI Machine Learning Repository* [http://www.ics.uci.edu/~mlearn/ MLRepository.html]. School of Information and Computer Science, University of California, Irvine, CA, 2007.
[23] M. Bohanec and B. Zupan, *AI Lab Datasets* [http://magix.fri.uni-lj.si/blaz/hint/datasets.htm]. Faculty of Computer and Information Science, University of Ljubljana, Slovenia, 1997.

Optimizing the Performance of a Refrigeration System Using an Invasive Weed Optimization Algorithm

Roozbeh Razavi-Far [1] and **Vasile Palade** [2] and **Jun Sun** [3]

Abstract. This paper presents a study and the obtained results on the performance optimization of a large refrigeration system in steady state conditions. It is shown that, by using adequate knowledge on plant operation, the plant wide performance can be optimized with respect to a small set of variables. For this purpose, an appropriate performance function is defined. A derivative free optimization technique based on the invasive weed optimization (IWO) algorithm has been used to optimize the parameters of the local controllers in the system. The performance of the IWO algorithm, both in terms of optimality of the results and speed of convergence, is compared with particle swarm optimization (PSO) algorithm. Simulation results have been used to validate the proposed approach.

1 INTRODUCTION

Complex plants, such as large supermarket systems, include many sub-systems that dynamically interact together. Plant-wide performance optimization is not guaranteed by just local tuning of individual controllers of the subsystems.

Various optimization techniques have been extensively used to tune the controller parameters to achieve an improved performance [1, 21]. However, these techniques are generally gradient-based and mostly focused on local performance optimization.

An alternative is to use derivative-free search algorithms. These algorithms directly utilize the performance function and constrain values to steer towards the optimal solution. Recently, genetic algorithms [6], particle swarm optimization [11], ant colony optimization [4], simulated annealing [15] and tabu search [16] have been widely used for global optimization in different engineering applications [2, 19].

The Invasive Weed Optimization (IWO) algorithm is a bio-inspired numerical optimization algorithm that simulates the behavior of weeds in nature when colonizing and finding a suitable place for growth and reproduction [13]. Here, the IWO algorithm is used to find the optimal parameters for the local controller with respect to a plant-wide performance function. A comparative evaluation of the IWO algorithm and particle swarm optimization (PSO) algorithm, on this problem, is carried out in order to validate the appropriateness of the simulation results obtained.

[1] Department of Control Engineering and System Analysis, Université Libre de Bruxelles (ULB), 50 Av. F.D. Roosevelt, CP 165/55, B-1050 Brussels, Belgium, (e-mail: roozbeh.razavi-far@ulb.ac.be) and Danfoss A/S, A/C Control, Denmark

[2] Department of Computer Science, University of Oxford, Wolfson Building, Parks Road, Oxford OX1 3QD, UK, (e-mail: vasile.palade@cs.ox.ac.uk)

[3] Department of Computer Science and Technology, Jiangnan University, No. 1800, Lihu Avenue, Wuxi, Jiangsu 214122, China

The rest of the paper is organized as follow. The description of the refrigeration system under optimization is presented in section 2. To perform the optimization task, an appropriate performance function is proposed in section 3 along with the problem formulation. The IWO algorithm, its main properties and pseudocode are presented in section 4. In section 5, the main features and algorithm of a standard PSO are briefly explained. Simulation setup and comparative results are shown in section 6. Conclusions are drawn in section 7.

2 SYSTEM DESCRIPTION

This section provides a short description of the vapor-compression cycle in a refrigeration system. Then, the relevant dynamics of the system, seen from a control perspective, are described. In the vapor-compression cycle a refrigerant is circulating between two heat exchangers. One in the cold storage where the refrigerant absorbs energy by evaporating, and one in a hot reservoir, typically the surroundings, where energy leaves the refrigerant during condensing [12].

The temperature at which the refrigerant evaporates / condenses is called the saturation temperature, T_{sat}. For the cycle of heat transfers in the refrigeration system to work, T_{sat} in the evaporator must be lower than the temperature in the cold storage, while it, in the condenser, must be higher than the temperature of the surroundings. To achieve this a compressor is inserted between the evaporator and the condenser since T_{sat} of any liquid or gas depends on the pressure. Hence, evaporator and condenser are referred to as the low and high pressure parts respectively.

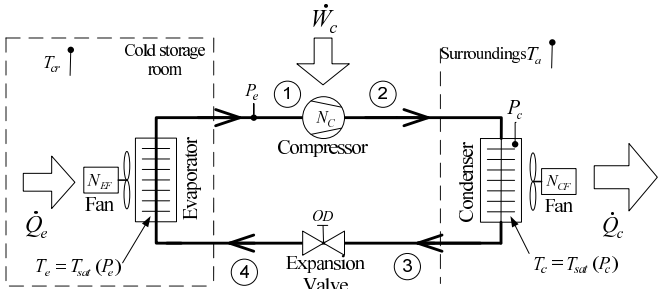


Figure 1. The basic layout of a refrigeration system [12].

Fig. 1 shows the layout of the refrigeration system with four states of the process marked in the figure and Fig. 2 shows the correspondence between pressure and specific enthalpy in the refrigerant for

the entire process with markings of the same four states. Both figures are from [12]. When the refrigerant enters the evaporator as a liquid it starts evaporating. Thus, the first section of the evaporator contains a mixture of liquid and gas and is called the two-phase region. In order not to damage the compressor, it is very important that no liquid flows out of the evaporator outlet, and for that reason, an important control objective is to make sure that the two-phase region ends before the evaporator outlet such that the last section of the evaporator only contains gas. This section is called the superheat region. In the

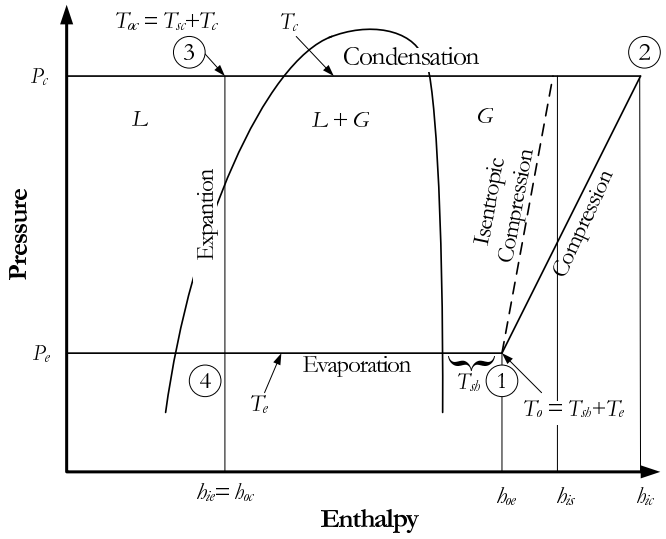


Figure 2. h - $\log(p)$-diagram for the vapour-compression cycle. The numbers correspond to the states in figure 1. Subindices : $i =$ inlet, $o =$ outlet, $e =$ evaporator, $c =$ condenser. L and G denotes Liquid and Gas respectively [12].

followings, we provide the model of relevant parts that are used for control purposes. The following references [7, 18, 17, 8] have been used, to derive the dynamic equations for the model.

2.1 The expansion valve

The refrigerant mass flow through the valve can be modeled as [8]:

$$\dot{m}_i = OD\alpha\sqrt{P_c - P_e} \tag{1}$$

In the above equation, $\dot{m}_i$ refrigerant mass flow rate at evaporator inlet, P_c and P_e are condensing and evaporating pressure respectively, α is the heat transfer coefficient, and OD is the opening degree of the expansion valve ($0 \leq OD \leq 1$). The mass flow $\dot{m}(kg/s)$ characteristics can be experimentally determined as a function of opening degree OD, and presented by the following polynomial function [8]:

$$\begin{aligned} \dot{m}_i &= f_{OD}(OD) \\ &= a_1 + a_2 OD + a_3 OD^2 + a_4 OD^3 + a_5 OD^4 \end{aligned} \tag{2}$$

For the chosen case the parameters are given as $a_1 = -0.009689, a_2 = 0.2236, a_3 = -0.4178$, $a_4 = 0.4546$, and $a_5 = -0.1573$ [8].

2.2 The compressor

The mass flow is modeled in Eqn. (3), since it is assumed that the compressor can act as an ideal pump, with the volume of V_{comp} [18].

$$\dot{m}_o = \rho_g \, V_{comp} f_{comp}, \tag{3}$$

where $\dot{m}_o$ stands for refrigerant mass flow rate at evaporator outlet, f_{comp} denotes the compressor speed (in frequency), and ρ_g is the density of the refrigerant in gas form. The suction pressure P_e is considered to be proportional to the vapor density in all operating conditions [18]. Therefore, the mass flow in equation (3) can be reformulated by means of a constant α_{comp} as:

$$\dot{m}_o = \alpha_{comp} P_e f_{comp} \tag{4}$$

2.3 The evaporator dynamics

For the design and optimization of controller parameters, an elaborate description of the evaporator dynamics is not needed since, in real life, such detailed models do not exist. To attain a proper model, the control signal OD is mapped to the superheat temperature Sh. This I/O mapping, is done by carrying out an OD sweep on the refrigeration system while keeping the air temperature $T_{air,in}$, the compressor frequency f_{comp}, and the condenser pressure P_c constant.

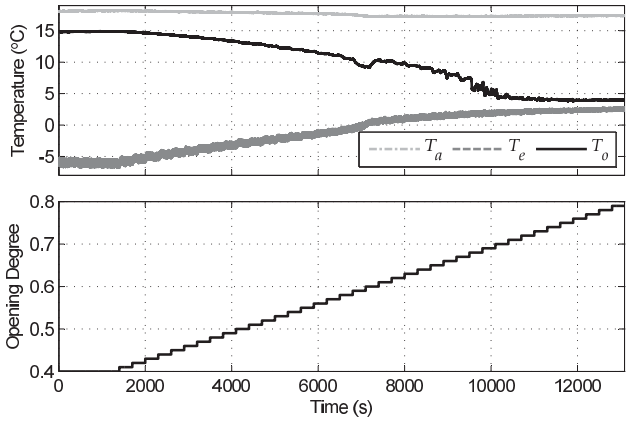


Figure 3. The temperature profiles as a function of a sweep on the expansion valves opening degree (OD). The temperatures are air temperature T_a, Outlet temperature T_o, and evaporation temperature T_e [8].

The dynamic behaviors of the air temperature T_a, outlet temperature T_o, and evaporation temperature T_e with respect to a gradual increase in opening degree OD of the expansion valve are presented in Fig. 3 [8]. Earlier studies, in [8], revealed that there is a considerable change in the system gain. This system nonlinearity is appropriately described, in [22], by the inverse trigonometric function $atan$ expressed as:

$$y = 7\left(-atan(10\pi\frac{u-50}{50}) + \frac{\pi}{2}\right) \tag{5}$$

In the above equation, u and y stand for the input and output respectively, and the rest is arbitrary scaling [22]. In addition, the transfer function of the outlet temperature T_o to opening degree OD,

in the Laplace domain, is explained by a first-order-plus-dead-time (FOPDT) process model [8]:

$$H(s) = \frac{K_p e^{-Ls}}{\tau s + 1}, \tag{6}$$

where K_p stands for the lumped system gain, L denotes the time delay, and τ is the system time constant. The transfer function parameters vary with respect to the change in operating points [8].

3 PROBLEM FORMULATION

The dynamic behavior of the evaporator unit varies due to the change in the operating conditions at the compressor unit. Therefore, it would be interesting to investigate whether it is possible to optimize the parameters of the evaporator's superheat controller, from the perspective of a global performance function. Inspired from [8], the global performance function is defined as:

$$J = \sum_{k=0}^{N} \left(\underbrace{q_1 \frac{OD_k^2}{OD_{max}^2} + q_2 \frac{(Sh_k - Sh_{ref})^2}{Sh_{ref}^2}}_{\text{Evaporator}} + \underbrace{q_3 \frac{f_{comp,k}^2}{f_{comp_0}^2}}_{\text{Compressor}} \right) \tag{7}$$

where q_1, q_2, and q_3 are appropriate positive weight constants. The first two terms in the right-hand side of the equation are normally used for evaluation/design of appropriate controller for superheat control. The last term relates to the energy consumption of the compressor. The performance function J is, thus, a global performance function. Since a change in the compressor operating conditions has an impact on the performance of the controller, it is desired therefore, to find the optimal controller parameters that also minimize this impact. The problem is defined as:
Find a set of controller parameters that minimize the global performance function J over all operating conditions. Using *derivative-based* optimization methods can become problematic due to the inherent non-smoothness in the performance function, which can make the computation of finite differences very inaccurate. The non-smoothness is caused by discrete switchings in the compressor speed.

4 INVASIVE WEED OPTIMIZATION (IWO)

At the beginning, before explaining the IWO algorithm, the key terms [13, 10] are introduced here as follow:

Seed: each individual in the colony that includes a value for each variable in the optimization problem prior to fitness evaluation.

Fitness: a value represents the merit of the solution for each seed.

Weed/Plant: each evaluated seed grows to a flowering plant or weed in the colony. Therefore, growing a seed to a plant corresponds to evaluating an individual's fitness.

Colony: the search space and indicates all agents or seeds.

Population size: the number of plants in the colony.

Maximum weed population: a predefined parameter corresponds to the maximum allowed number of weeds in the colony posterior to fitness evaluation.

The following steps are considered to simulate the colonizing behaviour of weeds [13, 10]:

1- **Search space definition**: Initially, the number of parameters that need to be optimized has to be defined, hereafter denoted by D. Next, for each parameter in the D-dimensional search space, a minimum and maximum value are assigned.

2- **Population initialization**: A limited number N_0 of initial seeds, $W = \{w_1, w_2, \ldots, w_{N_0}\}^T$ are being randomly dispread through the defined search space. Consequently, each seed catches a random position in the D-dimensional search space [13].

3- **Fitness estimation**: A fitness value assigned to each initial seed by the fitness function, defined to represent the goodness of the solution [10]. Here, initial seeds grow up to flowering plants.

4- **Ranking and reproduction**: The flowering plants are firstly ranked based on their assigned fitness values with respect to others. Subsequently, flowering plants can reproduce new seeds with respect to their rank in the colony. In other words, the number of seeds produced by each plant can increase linearly from the minimum possible seeds production S_{min}, to its maximum, S_{max} based on their own, the lowest, and the highest fitness of the colony (all plants). Then, the plants with higher fitness which are more adapted to the colony can produce more seeds that solve the problem better [10]. Fig. 4 illustrates this procedure [13]. The number of seeds to be created by each plant is computed as follow:

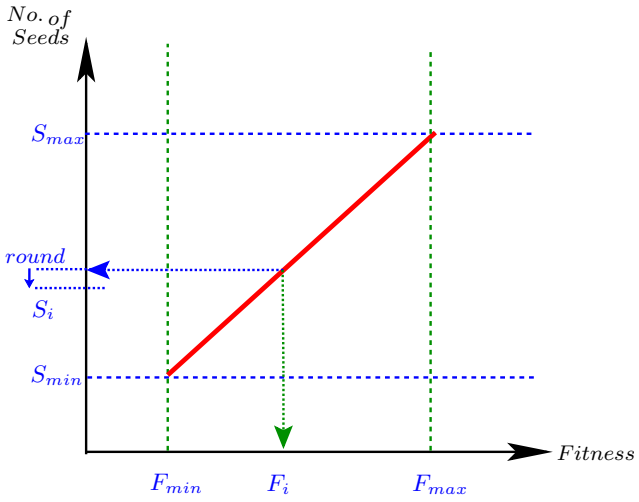


Figure 4. Seed reproduction procedure [13].

$$S_i = \left\lfloor \frac{F_i - F_{min}}{F_{max} - F_{min}} (S_{max} - S_{min}) + S_{min} \right\rfloor \tag{8}$$

where F_i is the fitness of $i-th$ plant. F_{min} and F_{max} stand for the lowest and highest fitness in the weed population. Therefore, this step guarantees cooperation of every weed in the reproduction procedure. Note that S_{max} and S_{min} are predefined parameters of the algorithm and adjusted according to the structure of the problem.

5- **Spatial dispersal**: Randomness and adaptation is provided in this part of the algorithm [13]. Here, the seeds are being randomly scattered through the search space by using normally distributed numbers with zero mean and adaptive standard deviations [13] as follow:

$$w_s[\kappa] = w[\kappa] + \mathcal{N}(0, \sigma_{iter}^2) \tag{9}$$

where, $w[\kappa]$ indicates the $\kappa - th$ variable of a solution vector in the current iteration, and $w_s[\kappa]$ shows the $\kappa - th$ variable of its $s - th$ seeds. The standard deviation σ_{iter} at the present time step can be

INPUTS: N_0, S_{min}, S_{max}, $iter_{max}$, $\sigma_{initial}$, σ_{final}, n, P_{max} and J
GENERATE a random population of N_0 individuals from a set of feasible solutions $W = \{w_1, w_2, \ldots, w_{N_0}\}^T$
DO FOR $iter = 1, 2, ..., iter_{max}$
  EVALUATE the fitness function for each individual in W.
  COMPUTE the maximum and minimum fitness in the colony F_{max} and F_{min}.
  DO FOR each individual w_i
    COMPUTE number of seeds for w_i, $S_i = \lfloor (F_i - F_{min})\,(S_{max} - S_{min})\,/\,(F_{max} - F_{min}) + S_{min} \rfloor$.
    RANDOMLY distribute seeds over the search space with normal distribution $\mathcal{N}\left(0, \sigma_{iter}^2\right)$ around the parent plant w, with zero mean and an adaptive standard deviation:

$$\sigma_{iter} = \sigma_{final} + (\sigma_{initial} - \sigma_{final})\,(iter_{max} - iter)^n\,/(iter_{max})^n$$

    ADD the generated seeds to the solution set, W.
  END DO
  IF $(|W| = N) > P_{max}$, Then:
    SORT the population W in descending order of their fitness.
    TRUNCATE population of weeds with smaller fitness until $N = P_{max}$;
  END IF
END DO
BEST solution is the plant w_{best} with minimum fitness in the last population.

Figure 5. The pseudo-code for the IWO algorithm [14].

computed adaptively according to the following equation [13]:

$$\sigma_{iter} = \frac{(iter_{max} - iter)^n}{(iter_{max})^n}\,(\sigma_{initial} - \sigma_{final}) + \sigma_{final} \quad (10)$$

where $\sigma_{initial}$ and σ_{final} denote to the pre-defined initial and final standard deviations, respectively. $iter_{max}$ indicates the maximum allowed number of iteration cycles and n is the nonlinear modulation index [13] assigned by the user. The σ_{iter} can be reduced from the $\sigma_{initial}$ to the σ_{final} with different velocities in accordance with the chosen nonlinear modulation index, n.

Initially, the whole search space can be explored by the algorithm due to the high value of initial standard deviation $\sigma_{initial}$. Then, the standard deviation σ_{iter} is gradually reduced by increasing the number of iterations, to focus the search around the local minima or maxima to find the global optimum. This gradual reduction guarantees to collect only fitter plants and to discard plants with lower fitness. The produced seeds, along with their parents are considered as the potential solutions for the next population.

6- **Competitive exclusion**: After passing a number of iterations, the population size reaches its pre-defined maximum (P_{max}) by fast reproduction and consequently a mechanism for discarding the plant with low fitness will be activated. To this end, the seeds and their parents are ranked together and those with higher fitness survive and subsequently reproduce new seeds in the next iteration.

7- **Termination condition**: Survived plants reproduce new seeds with respect to their fitness rank in the colony. The procedure is repeated at step 3 until either the maximum allowed number of iterations has been reached or the fitness criterion met [10].

A pseudocode version of the IWO algorithm is given in Fig. 5 [14].

5 Particle Swarm Optimization (PSO)

PSO is an evolutionary search process for stochastic optimization based on the social learning metaphor [11]. The PSO algorithm mimics the social behavior in flocks of birds when they are flying, by simulating a population of potential solutions or particles 'so-called swarms' in a multidimensional search space [5]. These randomly initialized particles freely fly across the predetermined search space and update their own position and velocity according to their best experience, the best experience of the entire group 'population' and balancing exploration and exploiting [9].

The aim of the PSO algorithm is to find a set of particles 'solutions' that minimize an objective function J. In the PSO algorithm, the so-called swarm includes a set of particles $\mathcal{P} = \{p_1, p_2, \ldots, p_{SW}\}$, where SW stands for the swarm size. Each particle's position stands for a candidate solution to minimize the fitness function J. At each time step t (iteration index in the optimization context), particle p_l contains a position $\vec{x}_l^{\,t}$ and a velocity $\vec{v}_l^{\,t}$ associated to it.

The PSO algorithm is begun by initialization on which random positions are generated for the particles, within an initialization region. The velocities are initialized also to zero or to small random values to preserve the particles inside the predetermined search space during the first iterations [3].

The velocities and positions of the particles are iteratively updated based on the following movement equations (11,12), until a stopping criterion has been met [3].

$$\vec{v}_l^{\,t+1} = \psi^t \vec{v}_l^{\,t} + c_1 \vec{\varphi}_1^{\,t}(\vec{b}_l^{\,t} - \vec{x}_l^{\,t}) + c_2 \vec{\varphi}_2^{\,t}(\vec{g}_l^{\,t} - \vec{x}_l^{\,t}) \quad (11)$$

$$\vec{x}_l^{\,t+1} = \vec{x}_l^{\,t} + \vec{v}_l^{\,t+1} \quad (12)$$

where $\vec{b}_l^{\,t}$ is the best position of the l^{th} particle (with respect to J) discovered thus far, $\vec{g}_l^{\,t}$ is the global best position of the swarm until t^{th} iteration, and l stands for the particle's index, $l = 1, \ldots, SW$. c_1 and c_2 are the acceleration factors. φ_1^t and φ_2^t are diagonal matrices in which the main diagonal entries are uniformly distributed random numbers in the range $[0, 1)$ and iteratively updated at each iteration. ψ^t is the time-varying inertia weight [20]. The inertia weight linearly

decreases in each iteration, based on the following:

$$\psi^t = (\psi_{max} - \psi_{min})\frac{t_{max} - t}{t_{max}} + \psi_{min} \quad (13)$$

where ψ_{max} and ψ_{min} are the maximum and minimum values of the inertia weight, respectively, and t_{max} is the maximum allowed number of iterations. The particles move in the search process based on their own best position and the global best position, in a cooperative manner, until best results can be found [3]. Here, all particles of the swarm are fully connected together, and thus, all particles are considered as neighbors. A pseudocode version of the PSO algorithm is given in Fig. 6 [3].

INPUTS: J, t_{max}, ψ_{max}, ψ_{min}, c_1, c_2 and SW
SET $t = 0$
DO FOR $l = 1, 2, ..., SW$
  INITIALIZE random particle's position $\vec{x}_l^t$
  INITIALIZE random particles' velocity $\vec{v}_l^t$
  CALCULATE fitness value for each particle
  INITIALIZE the local best $\vec{b}_l^t$
  INITIALIZE the global best $\vec{g}_l^t$
END DO
WHILE $t = 1, 2, ..., t_{max}$
  DO FOR $l = 1, 2, ..., SW$
    GENERATE random matrices $\vec{\varphi}_1^t$, $\vec{\varphi}_2^t$ within $[0, 1)$
    UPDATE inertia weight ψ^t according to Eq.(13)
    UPDATE particles' velocity $\vec{v}_l^t$ according to Eq.(11)
    UPDATE particle's position $\vec{x}_l^t$ according to Eq.(12)
    UPDATE the fitness value of each particle
    UPDATE particle's local best $\vec{b}_l^t$
    UPDATE swarm's global best $\vec{g}_l^t$
  END DO
SET $t = t + 1$
END WHILE
OUTPUT: Best solution found.

Figure 6. The pseudo-code for the PSO algorithm [3].

6 SIMULATION SETUP AND RESULTS

In this section, the invasive weed optimization algorithm is used for plant-wide optimization of the supermarket refrigeration system and compared with PSO. The simulation setup and results are presented in this section.

6.1 Simulation setup

The optimization approach has been tested on a simulation model of a supermarket refrigeration system. The simulation setup is presented in Fig. 7.

The IWO algorithm has been applied to optimize the integration time, T_n, and the gain, K_p, of the PI controller. Some constraints are needed for T_n and K_p, since the feasible set for each variable can not overlap together. These constrains have been applied to the search space according to the knowledge about the controller behavior.

The key parameters of the IWO algorithm (e.g. $\sigma_{initial}$, σ_{final} and n) have a crucial affect on its own convergence. The careful tuning of these parameters can guarantee having a proper value of the

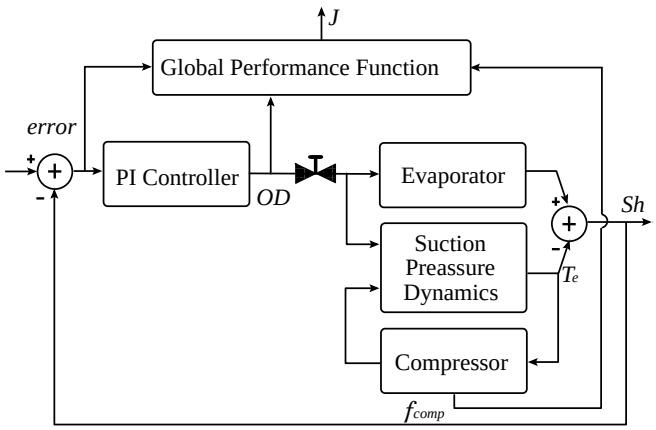


Figure 7. Simplified simulation setup

standard deviation at each iteration, σ_{iter}. The maximum number of plants in the colony P_{max} should also be selected a priori based on the achievable performance of the algorithm. The key parameters of concern and their selected values to start the IWO simulation are listed in table 1.

Table 1. The IWO and PSO parameters

Quantity	Symbol	Value
Number of initial plants	N_0	5
Maximum number of iterations	$iter_{max}$	200
Problem dimension	D	2
Maximum number of plants	P_{max}	30
Maximum number of seeds	S_{max}	5
Minimum number of seeds	S_{min}	0
Nonlinear modulation index	n	3
Initial value of standard deviation	$\sigma_{initial}$	1
Final value of standard deviation	σ_{final}	0.05
Acceleration factors	c_1, c_2	2
Maximum inertia weight	ψ_{max}	0.9
Minimum inertia weight	ψ_{min}	0.4
Swarm size	SW	30
Maximum number of PSO's iterations	t_{max}	200

The maximum number of simulation runs by the IWO algorithm, $SR_{max} = N_0 \cdot (S_{max} + 1) + P_{max} \cdot (iter_{max} - 2)$ equals to 5970. The simulation is performed for 6000 seconds, to guarantee a steady state performance function.

6.2 Results

In Fig. 8, the minimum fitness value J_{min} or the best performance of the refrigeration system is plotted for each iteration of the IWO algorithm. The optimal values proposed by the IWO algorithm for the controller parameters after 200 iterations, $T_n = 58.7295$ and $K_p = -0.1536$, lead to a reasonable performance, 1439466.

To evaluate the performance of the IWO algorithm, the simulation results are compared with the results of the PSO algorithm, introduced in Fig. 6. The key parameters of the PSO algorithm for the best results used for simulation are listed in table 1.

The PSO algorithm produces the results, J_{min}, shown on Fig. 8 after 200 iterations. The use of initial values for the algorithms

can decrease the number of iterations and consequently reduce the convergence time to find the optimal solution, however the parameters of the both algorithms are properly chosen to provide an identical setup for a fair comparison. The optimal values proposed by the PSO algorithm for the controller parameters after 200 iterations, $T_n = 61.7482$ and $K_p = -0.1465$, lead to a performance, 1547500. The results of IWO and PSO algorithms are compared in table 2 by means of the minimum fitness value J_{min}, maximum fitness value J_{max}, mean fitness value J_{mean} and standard deviation J_σ of the population in the last iteration of the simulation.

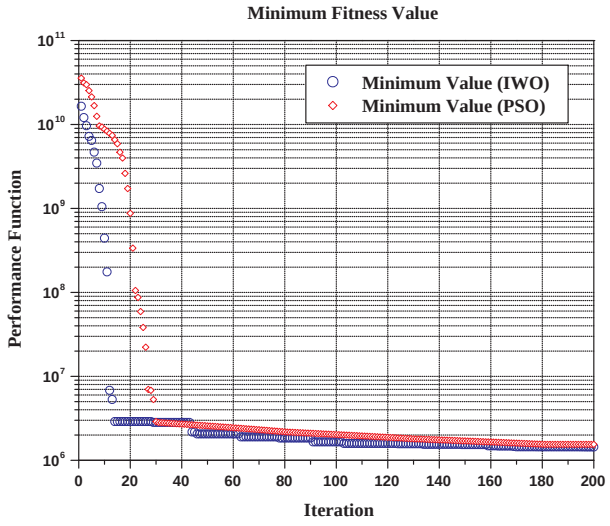


Figure 8. Iterations versus the minimum values of the performance function J_{min}

Table 2. Comparison of J_{min}, J_{max}, J_{mean} and J_σ of the population in the last iteration of IWO and PSO simulations

	J_{min}	J_{max}	J_{mean}	J_σ
IWO	1439466	1596000	1475000	89563
PSO	1547500	1706716	1634000	103693

As shown, in table 2 and Fig. 8, the IWO algorithm outperforms the PSO. J_{min} is dramatically decreased in few iterations by both the IWO and the PSO algorithms, and both algorithms converge rapidly, however the IWO has slightly better results than PSO (table 2). Here, the comparison was done in order to validate the results obtained by the IWO algorithm in this particular control problem are sensible. However, both algorithms and their hybrid schemes can be used in many other optimization problems from engineering, and not only.

7 CONCLUSION

This paper focused on the application of the IWO and PSO algorithms for the optimization of the global system performance of a supermarket refrigeration system. An appropriate performance function was firstly introduced. Then, the IWO and PSO optimization algorithms were used to find the optimal parameters of the local controllers with respect to the defined global performance function. A fair comparison was finally presented to validate the simulation results. A proper choice of the key parameters and initial values is of paramount importance in the performance of the IWO algorithm.

8 ACKNOWLEDGMENTS

The authors thank Dr. Roozbeh Izadi-Zamanabadi, for support in system description and problem formulation.

REFERENCES

[1] K. J. Åström and T. Hägglund, *Advanced PID Control*, ISA - Instrumentation, Systems and Automation Society, 2006.
[2] D. W. Boeringer and D. H. Werner, 'Particle swarm optimization versus genetic algorithms for phased array synthesis', *IEEE Transactions On Antennas And Propagation*, **52**, 771–779, (2004).
[3] M. Dorigo, M. A. Montes de Oca, and A. Engelbrecht, 'Particle swarm optimization', *Scholarpedia*, **3**(11), 1486, (2008).
[4] M. Dorigo, V. Maniezzo, and A. Colorni, 'Ant system: Optimization by a colony of cooperating agents', *IEEE Transactions On Systems, Man, And Cybernetics*, **26**, 1–3, (1996).
[5] R. Eberhart and Y. Shi, 'Comparing inertia weights and constriction factors', in *Proceedings of the Congress on Evolutionary Computing*, pp. 84–89, (2000).
[6] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison Wesley, Reading MA, 1989.
[7] X-D. He, S. Liu, H. H. Asada, and H. Itoh, 'Multivariable control of vapor compression systems', *HVAC&R RESEARCH*, **4**(3), 205–230, (1998).
[8] R. Izadi-Zamanabadi, K. Vinther, H. Mojallali, H. Rasmussen, and J. Stoustrup, 'Evaporator unit as a benchmark for plug and play and fault tolerant control', in *8th IFAC Safeprocess*, Mexico City, (2012).
[9] K.O. Jones, 'Comparison of genetic algorithms and particle swarm optimization for fermentation feed prole determination', in *Int. Conf. on Computer Systems and Tecnologies*, (2006).
[10] S. Karimkashi and A. A. Kishk, 'Invasive weed optimization and its features in electromagnetics', *IEEE Transactions On Antennas And Propagation*, **58**(4), 1269–1278, (2010).
[11] J. Kennedy and R. Eberhart, 'Particle swarm optimization', in *Proceedings of the IEEE Int Conf Neural Networks*, pp. 1942–1948, (1995).
[12] L. F. S. Larsen, *Model Based Control of Refrigeration Systems*, Ph.D. dissertation, Aalborg University, 2005.
[13] A. R. Mehrabian and C. Lucas, 'A novel numerical optimization algorithm inspired from weed colonization', *Ecological Informatics*, **1**, 355–366, (2006).
[14] A. R. Mehrabian and A. Yousefi-Koma, 'Optimal positioning of piezoelectric actuators on a smart fin using bio-inspired algorithms', *Aerospace Science and Technology*, **11**, 174182, (2007).
[15] R. H. J. M. Otten and L. P. P. P. van Ginnekent, *The Annealing Algorithm*, Boston/Dordrecht/London: Kluwer Academic, 1989.
[16] D.T. Pham and D. Karaboga, *Intelligent Optimization Techniques*, Springer, 2000.
[17] H. Rasmussen and L. F. S. Larsen, 'Non-linear and adaptive control of a refrigeration system', *IET Control Theory and Applications*, **5**, 364–678, (2011).
[18] H. Rasmussen, C. Thybo, and L. F. S. Larsen, 'Nonlinear superheat and evaporation temperature control of a refrigeration plant', in *the IFAC Workshop on Energy Saving Control in Plants and Buildings*, (2006).
[19] R. Razavi-Far, H. Davilu, V. Palade, and C. Lucas, 'Model-based fault detection and isolation of a steam generator using neuro-fuzzy networks', *Neurocomputing Journal*, **72**, 2939–2951, (2009).
[20] Y. Shi and R. Eberhart, 'A modified particle swarm optimizer', in *Proceedings of IEEE International Conference on Evolutionary Computing*, pp. 69–73, (1998).
[21] C.A. Smith and A.B. Corripio, *Principles and Practice of Automatic Process Control, second edition*, Wiley, 2 edn., 1997.
[22] K. Vinther, H. Rasmussen, R. Izadi-Zamanabadi, and J. Stoustrup, 'Single temperature sensor based evaporator filling control using excitation signal harmonics', in *Proceedings of the IEEE Multi-conference on Systems and Control*, Dubrovnik, Croatia, (2012).

A New Cooperative Evolutionary Multi-Swarm Optimizer Algorithm Based on CUDA Parallel Architecture Applied to Solve Engineering Optimization Problems

Daniel Leal Souza[1,2], **Otávio Noura Teixeira**[1,3], **Dionne Cavalcante Monteiro**[2] **and Roberto Célio Limão de Oliveira**[3]

Abstract. This paper presents a new *Cooperative Evolutionary Multi-Swarm Optimization Algorithm* (CEMSO-GPU) based on CUDA parallel architecture applied to solve engineering problems. The focus on this approach is: The use of the concept of master/slave swarm with a mechanism of sharing data; and, the parallelism method based on the paradigm of General Purpose Computing on Graphics Processing Units (GPGPU) with NVIDIA-CUDA architecture. All these improvements were made aiming to produce better solutions in fewer iterations of the algorithm and to improve the search for best results. The algorithm was tested for some well-known engineering problems (ATD, WBD and SRD-25) and the results compared to other approaches.

1 INTRODUCTION

In recent years, the evolution of computational resources made possible some significant advances in the techniques of search and optimization. [1] affirms that these advances have brought success in the construction of more efficient solutions, also in studies of metaheuristics, which guide the heuristics to obtain optimal solutions applied in N-dimensional search space's problems. These problems vary from a wide range of practical applications, including industry and management of basic supplies such as electric energy, gas and petroleum refining, for example [2].

One of the most well known metaheuristics with large use in engineering field is Particle Swarm Optimization (PSO) [8]. Among its variants, Evolutionary Particle Swarm Optimization (EPSO) proposed in [3] stands out for including the set of operations of evolutionary strategies such as replication, mutation and selection applied in an PSO environment.

Another type of variation of PSO algorithm involves the concept of multiple population, consisting of a master swarm and slave swarms. [4] uses the model of search partitioning in genetic algorithms to PSO, in order to reduce the deterioration of performance while dimensionality of search space increases. In general, most of these metaheuristics use computing systems with parallel and distributed architecture under an inter-communicable environment between two or more swarms. The ultimate goal is a more efficient search and the exchange of information obtained by each of them with order to compare and refine the search based on the results already obtained by the neighboring clusters.

In the matter of the improvement of hardware resources, it´s possible to highlight the evolution of Graphics Processing Units (GPUs). Since 2003, the many-cores processors, strongly represented by GPU, had shown greater superiority in terms of speed, especially operations involving floating point data [5]. Currently, many computing solutions, especially for applications in the scientific area, are developed by taking advantage of thousands of cores available in the multiprocessors found in a board video, due to technologies such as NVIDIA's Compute Unified Device Architecture (CUDA) and Open Computing Language (OpenCL) [5].

The implementation of Evolutionary Particle Swarm Optimization Algorithm in a cooperative and multi-swarm model applied under GPGPU paradigm (general-purpose computing on graphics processing units) has a number of benefits, since each element can be treated by a thread, which tends to contribute in reduction of execution time, in addition to the potential increase in performance of search and optimization.

This paper presents an algorithm based on both EPSO and PSO mechanisms over a cooperative approach of master and slaves swarms implemented under CUDA architecture. This new algorithm is called *Cooperative Evolutionary Multi-Swarm Optimization on Graphics Processing Units Algorithm* (CEMSO-GPU).

[1]Laboratory of Natural Computing (LCN) – Area of Exact and Natural Sciences (ACET) – University Centre of Pará (CESUPA) – Belém - Brazil, E-Mail: daniel.leal.souza@gmail.com; onoura@gmail.com.

[2]Laboratory of Applied Artificial Intelligence – Institute of Exact and Natural Sciences (ICEN) – Federal University of Para (UFPA), Belém - Brazil, E-Mail: dionnecmonteiro@gmail.com.

[3]Post-Graduate Program in Electrical Engineering (PPGEE), Institute of Technology (ITEC) – Federal University of Para (UFPA), Belém - Brazil, E-Mail: limao@ufpa.br

For comparison and validation, we used three engineering problems with large usage in scientific literature: Air Tank Design (ATD); Welded Beam Design (WBD); Speed Reducer Design with 25 restrictions (SRD-25).

2 RELATED WORKS

Some publications involving the use of PSO on GPUs were found in the scientific literature. The most relevant works for the context of this paper are briefly discussed below:

- ***Collaborative Multi-Swarm PSO for Task Matching Using Graphics Processing Units [6]***: In this work, the authors expose an implementation of the cooperative and multi-swarm model for PSO which was executed over CUDA architecture for a task matching operation. This work uses a multi-swarm environment model, similar to [4]. The algorithm uses data of discrete and continuous type, and the entire process is carried out continuously and in the end, the authors apply rounding to discrete values. The results obtained in this study show considerable improvements in processing time and in the values obtained at the end of execution when compared to a PSO of a swarm.
- ***MCPSO - A Multi-population Cooperative Particle Swarm Optimization [14]***: This paper presents a multi-swarm and cooperative PSO, which is used as basis for CEMSO-GPU approach. The MCPSO algorithm is a master–slave model that consists of one master swarm and several slave swarms. The evolution of slave swarms is likely to amplify the diversity of individuals of the population and consequently to generate more promising particles for the master swarm. The master swarm updates the particle states based on both its own experience and that of the most successful particles in the slave swarms.
- ***GPU-based Asynchronous Particle Swarm Optimization [7]***: This work presents an approach for asynchronous execution of PSO on the GPU and exposes the problems of a synchronous implementation. In this implementation, the authors propose a data processing where each *thread* represents a specific variable from the particle, and it is treated in a loop. The obtained results, in relation to the sequential version show an increase of performance of approximately 300-fold in tests using functions such as Rastrigin. No tests with restrictive functions were presented in this paper.

We conducted an extensive search in literature regarding the publications involving the EPSO algorithm, however, no work with respect to implementation of the method created by [3] on GPUs under CUDA architecture and its use in an multi-swarm approach has been found.

3 A BRIEF DESCRIPTION OF CLASSIC AND EVOLUTIONARY PARTICLE SWARM OPTIMIZATION FOR CEMSO

3.1 Particle Swarm Optimization

PSO is a technique for stochastic optimization of nonlinear and continuous functions developed by James Kennedy and Russell Eberhart [8]. It was based from studies related to the "social behavior" observed in some species of birds. The PSO algorithm simulates a population of particles called "swarm", which operates inside of a search space predetermined [9] [10]. A particle is represented by two vectors storing values of position and speed, where the value of number of vector elements is equal to the amount of N variables corresponding to a N-dimensional search space. Based on a collective and cooperative exploration, the particles tend to progress in the search process as new and best results are found.

The movement of the particles is defined by the equations (1), (2) and (3), where equation (1) is the position, equation (2) is the velocity and equation (3) is the inertial weight.

$$V_i^{(t+1)} = wV_i^{(t)} + R_1C_1(b_i - X_i^{(t)}) + R_2C_2(b_g - X_i^{(t)}) \quad (1)$$

$$X_i^{(t+1)} = X_i^{(t)} + V_i^{(t+1)} \quad (2)$$

$$w = \frac{(k-1)}{(I_{MAX}-1)(-W_{MAX}+W_{MIN})+W_{MAX}} \quad (3)$$

In order to avoid particles to escaping from the search space, the position values of the particle are subjected to a damping boundary conditions. This feature is used only if the position value is outside the minimum or maximum limits. Another important improvement added to the boundary conditions is a new disturbance variable called correction constant (α), which helps to prevent particles may get trapped in local bests by multiplying a uniform random number between 0 and 1 with the α constant, following by an addition or subtraction with the maximum or minimum limit respectively. It is important to note that if the values of position of the particles exceed the limits, the speed is also changed to the negative value of its current value and multiplied by a random number generated between 0 and 1. The scheme for boundary conditions with correction constant on the particle's velocity and position are described by equations (4) and (5):

$$V_i^{(t+1)} = \begin{cases} -V_i^{(t+1)}R_1, & X_i^{(t+1)} < X_{MIN} \\ V_i^{(t+1)}, & X_{MIN} \le X_i^{(t+1)} \le X_{MAX} \\ -V_i^{(t+1)}R_1, & X_i^{(t+1)} > X_{MAX} \end{cases} \quad (4)$$

$$X_i^{(t+1)} = \begin{cases} X_{MIN} + R_2\alpha, & X_i^{(t+1)} < X_{MIN} \\ X_i^{(t+1)}, & X_{MIN} \le X_i^{(t+1)} \le X_{MAX} \\ X_{MAX} - R_2\alpha, & X_i^{(t+1)} > X_{MAX} \end{cases} \quad (5)$$

The particle's movement on damping boundary condition process is shown in Figure 1.

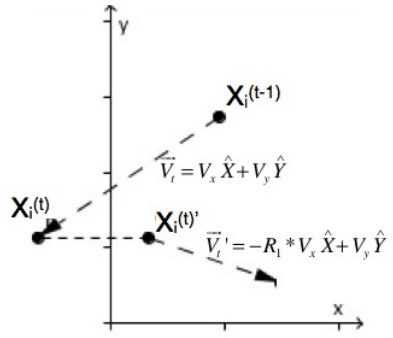


Figure 1. Example of a damping scheme for a two-dimensional problem

The variables for equations (1), (2), (3), (4) and (5) are described in Table 1.

Table 1. List of variables for PSO and its variants.

Variable	Description
t	Time Step
i	Particle's index
w	Inertia weight, with a linear decrease at each iteration
V	Particle's velocity
X	Particle's position
C1	Acceleration constant for individual interaction
C2	Acceleration constant for social interaction
b_i	Particle's local best
b_g	Swarm's global best
R_1 | R_2	Uniform random numbers between 0 and 1
W_{MAX} | W_{MIN}	Inertia weight's maximum and minimum values
I_{MAX}	Iteration's maximum value
k	Iteration index
X_{MAX} | X_{MIN}	Maximum and minimum values for particle's position
α	Correction constant for boundary conditions

The PSO algorithm used as a basis for implementation of the CEMSO-GPU algorithm is described in Algorithm 1:

Algorithm 1 Classic PSO

```
Initialize the population (velocity and positions);
Evaluate fitness for each particle in the swarm;
Initialize the local best (b_i);
Initialize the global best (b_g);
for k = 1 → I_MAX do
  Update inertia weight using equation (3);
  Update velocity using equation (1);
  Update position using equation (2);
  Apply velocity correction using equation (4);
  Apply position correction using equation (5),;
  Update fitness for each particle;
  Update particle's local best (b_i);
  Update swarm's global best (b_g);
end for
```

3.2 Evolutionary Particle Swarm Optimization

The EPSO algorithm, developed by [3], is a metaheuristic that adds the mechanism of evolutionary strategies (EE) to the PSO algorithm, where the operators of classical recombination are replaced by rules of motion of the particles. According to [3], EPSO can be classified in two forms, by given the mechanisms of the algorithm, it can be interpreted as a variant of the PSO or as a variant of EE.

The mechanisms of evolutionary strategy found in EPSO of [3] and [11] are: ***Replication*** of particles; ***mutation*** of the following weights: inertia weight, acceleration constants (C_1 ***and*** C_2) and the values of the global best; ***reproduction*** of new particles described by equation (6) with weights mutated; ***evaluation*** of new individuals; ***selection*** of the best individuals.

According to [3] and [11] the addition of the evolutionary mechanisms to PSO provides a search system more robust, since the mutation can produce a better result at runtime. The changes in the equation of velocity proposed by [3] are described below. It is important to note that the equations of inertia weight (3) and position (2) are not changed.

$$V_i^{(t+1)} = mw^*V_i^{(t)} + mC_{1(i)}^*(b_i - X_i^{(t)}) + mC_{2(i)}^*(b_g^* - X_i^{(t)}) \quad (6)$$

The differences between the equations for velocity (1) and (6) are absence of random number generators, in addition to the mutation process of the variables ***mw***, $mC_{1(i)}$, $mC_{2(i)}$ and the values of the global best (specified with the symbol '*'). The description of the mutation process are defined by equations (6), (7), (8), (9) and (10) respectively.

$$mw_{(i)}^* = w + (1 + \sigma N(0,1)) \quad (7)$$

$$mC_{1(i)}^* = C_1 + (1 + \sigma N(0,1)) \quad (8)$$

$$mC_{2(i)}^* = C_2 + (1 + \sigma N(0,1)) \quad (9)$$

$$b_g^* = b_g + (1 + \sigma_g N(0,1)) \quad (10)$$

The new variables found in equations (6), (7), (8), (9) and (10) are described in Table 2.

Table 2. List of variables for EPSO

Variable	Description
mw	Inertia weight, submitted to the mutation process
$mC_{1(i)}$	C_1 constant submitted to the mutation process
$mC_{2(i)}$	C_2 constant submitted to the mutation process
b_g	Swarm's global best
*	Marker of the variables subjected to the mutation process
σ_g	Disturbance constant for the global best
σ	Mutation parameter for ***mw***, $mC_{1(i)}$ and $mC_{2(i)}$
N(0,1)	Gaussian distribution of mean 0 and standard deviation 1

Given the description above, the customized EPSO algorithm (based on the work developed by [3]) used as the basis for implementation of CEMSO-GPU algorithm is described in the Algorithm 2:

Algorithm 2 Evolutionary PSO (EPSO)

```
Initialize the population (velocity and positions);
Evaluate fitness for each particle in the swarm;
Initialize the local best (b_i);
Initialize the global best (b_g);
for k = 1 → I_MAX do
  Update inertia weight using equation (3);
  for all particles in the swarm, do
    Update velocity from the original particles using equation (1);
    Update position from the original particles using equation (2);
    Apply velocity correction to the original particles using equation (4);
    Apply position correction to the original particles using equation (5);
    Update fitness for the original particles;
    Update particle's local best (b_i);
    Update swarm's global best (b_g);
    Replicate particle N times;
    Apply mutation for each weights (w; C1, C2) to all replicated particles;
    Update velocity from the replicated particles using equation (6);
    Update position from the replicated particles using equation (2);
    Apply velocity correction to the replicated particles using equation (4);
    Apply position correction to the replicated particles using equation (5);
    Update fitness for the replicated particles;
    Select the best particles;
    Update particle's local best (b_i);
    Update swarm's global best (b_g);
  end for all
end for
```

4 COMPUTE UNIFIED DEVICE ARCHITECTURE (CUDA)

CUDA architecture provides a set of extensions to the C language, which allow the implementation of parallel algorithms in video cards. GPUs with CUDA architecture have many cores which allows to run collectively thousands of independent small parts of processing, called *threads* [5].

The CUDA SDK (Software Development Kit) includes a compiler adapted for heterogeneous computing paradigm GPGPU (General Purpose Computing on Graphics Processing Units) and other tools capable of supporting heterogeneous applications, which have serial and parallel parts performed on the CPU (*host*) and GPU (*device*), respectively [12].

4.1 Thread Organization

To keep the organization in the executing data parallelism and data distribution effectively with control of memory access, CUDA uses three levels of organization: *thread*, *block* and *grid*. The explanation of each follows:

- ***Thread***: It is the basic unit of execution. Each *thread* is responsible for a copy of the program for a certain quantity of data. In some solutions, it is possible each *thread* stay responsible for an address of a vector or a matrix;
- ***Block*:** This structure stores a vector or a matrix (2D or 3D) of *threads*. All *blocks* have the same quantities of *threads*. Through this organization, it is possible to use synchronization of *threads* to *block* level;
- ***Grid***: It is the set of all *threads* that a *kernel* will use. A *Grid* is a vector or matrix of *blocks*. In multi-GPU environments, the *Grids* can not exchange information among themselves.

Figure 2 shows the schematic organization of *threads* and blocks on CUDA architecture, where the parallel application (executed into a function called *kernel*) uses a quantity of four blocks with five *threads* each.

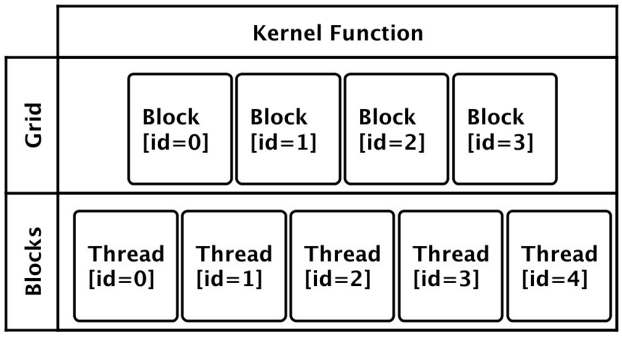


Figure 2. Scheme of *threads* and blocks. Adapted from [13]

5 COOPERATIVE MULTI-SWARM APPROACH FOR EPSO ON THE CUDA ARCHITECTURE (CEMSO-GPU)

The CEMSO algorithm is a result of the integration between the mechanisms found on the EPSO and PSO, acting under a multi-swarm approach.

It consists in an environment where two or more slave swarms cooperate with the master swarm the values of the particles classified as being the global best found to current iteration. The optimization process that occurs in the master swarm tends to enhance the global best of the slave swarms, and contribute in finding the global best related to the master swarm.

The use of CUDA architecture for CEMSO algorithm allows slave swarm to be executed in parallel, massively and with a low processing time. By comparison with other parallel and distributed computing methods, such as MPI (Message Parsing Interface), CUDA has the advantage of running many *threads* (SIMD taxonomy) on low cost computers with no need for clusters. In relation to the parallel scheme for slave swarms, the CUDA architecture provides a programming environment that makes possible to execute EPSO and PSO algorithms with high level of parallelism.

5.1 Multi-swarm Structure with Slave/Master Approach

The basic structure of the multi-swarm environment in CEMSO algorithm is based on the model found in the MCPSO algorithm [14]. This approach is organized so that the slave swarms are executed in parallel, without exchanging information among themselves, where they provide their best solutions (global best) to the master swarm. The cooperativity of data is applied only in the relationship between the master swarm and the slave swarms.

The master swarm updates its particles, taking into consideration the information of the best solution found among all slave swarms. Note that this information is integrated as a third component in the velocity update equation (more details in section 5.1.2). Figure 3 shows the architecture of CEMSO algorithm based on the model proposed by [14] and adapted for CUDA.

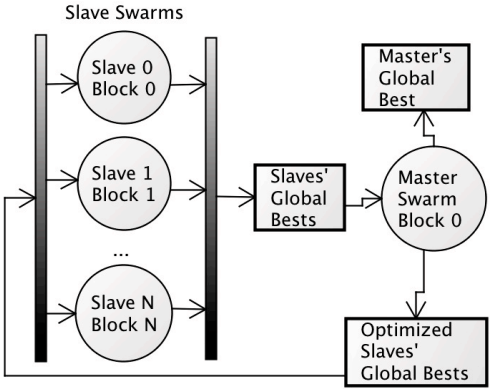


Figure 3. Schematic illustration of the master/slave model for the CEMSO-GPU algorithm. Modified from [14].

Based on the parallel implementation in CUDA, each *thread* is responsible for the manipulation of a particle (one *thread*, one particle (cardinality 1:1)), each block being responsible by a swarm (one *block*, one swarm (cardinality 1:1)).

Figure 4 shows the arrangement of particles and slave swarms in the CEMSO-GPU algorithm.

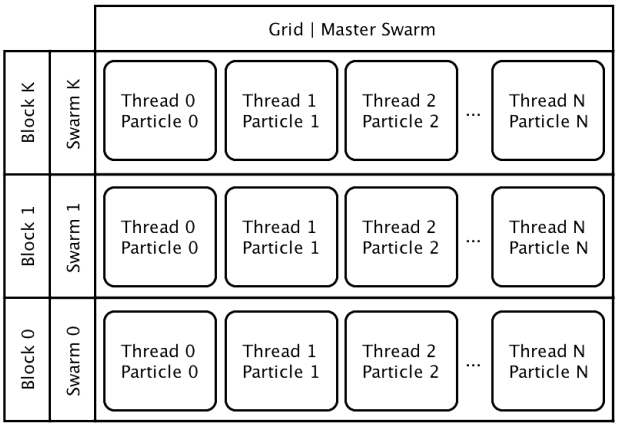


Figure 4. Scheme of the slave swarms and its particles in CUDA.

Figure 5 shows the arrangement of particles and master swarm in the CEMSO-GPU algorithm.

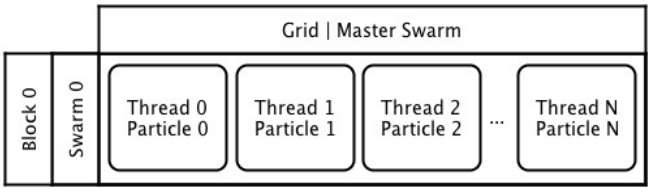


Figure 5. Scheme of the master swarm and its particles in CUDA.

5.1.1 Equations for Slave Swarms

The slave swarms in CEMSO-GPU algorithm operates in an parallel and independent way in the search and optimization of the problem and they all share the same amount of particles. Basically each swarm executes in parallel the process described in Algorithm 2 with some changes.

A few important points to be highlighted about the velocity equations are described below:

- The updating equations for velocity and position of original particles are not changed (equations (1) and (2) respectively);
- The velocity update equation applied to the replicas is also modified in order to add a uniform random number generator in a range between zero and one (R_1 and R_2). The addition of these variables allows to a higher exploration in the search space by the replicas.

Changes in the equation (6) for generated replicas in the slave swarms can be seen in equation (11).

$$V_i^{(t+1)} = mw^* V_i^{(t)} + R_1 mC^*_{1(i)}(b_i - X_i^{(t)}) + R_2 mC^*_{2(i)}(b_g^* - X_i^{(t)}) \quad (11)$$

5.1.2 Equations for Master Swarm

The use of the master swarm under the context of the CEMSO algorithm occurs after the parallel execution of slave swarms, where the values of the global best from every slave (b^S_g), as well as their values of velocity and fitness of each slave swarm are copied.

The update equations for velocity proposed in [14] are used in the master swarm, where is added the acceleration constant C_3 and the values of the best minimal/maximum global found by the slave swarms (b^S_g). It is important to emphasize that the equation for velocity used on replicated particles (equation (11)) was also adapted for the equation proposed by [14], where the mutated factors were added. Changes in the velocity equations are described below.

$$V_i^{(t+1)} = mw^* V_i^{M(t)} + R_1 C^*_{1(i)}(b_i^M - X_i^{M(t)}) + R_2 C^*_{2(i)}(b_g^{M*} - X_i^{M(t)}) + R_3 C^*_{3(i)}(b_g^S - X_i^{M(t)}) \quad (12)$$

$$V_i^{(t+1)} = w V_i^{M(t)} + R_1 C_1(b_i^M - X_i^{M(t)}) + R_2 C_2(b_g^M - X_i^{M(t)}) + R_3 C_3(b_g^S - X_i^{M(t)}) \quad (13)$$

As well as the acceleration constants $mC_{1(i)}$ and $mC_{2(i)}$, the new constant $mC_{3(i)}$ is also submitted to the mutation process. Equation (14) shows the mutation process for $mC_{3(i)}$.

$$mC^*_{3(i)} = C_3 + (1 + \sigma N(0,1)) \quad (14)$$

The new variables for CEMSO on equations (12), (13) and (14) are described in Table 3.

Table 3. List of variables for CEMSO

Variable	Description
C_3	Acceleration constant for master swarm's social interaction
$mC_{3(i)}$	C_3 constant submitted to the mutation process
$R_1 \mid R_2 \mid R_3$	Uniform random numbers between 0 and 1
b^M_i	Master swarm's local best
b^M_g	Master swarm's global best
b^S_g	Best global value obtained by the slave swarms

5.1.3 Pseudo-code

The implementation of CEMSO algorithm is described in Algorithm 3.

Algorithm 3 CEMSO-GPU

```
Allocate memory space for reading/writing on the GPU;
run in parallel for each particle of the slave swarms
  Initialize the population (velocity and positions);
  Evaluate fitness for each particle in the swarm;
  Initialize the local best (b_i);
  Initialize the global global (b_g);
  Sync threads (wait for all threads to finish);
  if thread index = 0 do for each block
    Initialize global best (b_g) of each slave swarm;
  end if
end run in parallel
Sync blocks (wait for all blocks to finish);
run in parallel for each particle of the master swarm
  Initialize master's local best (b^M_i) with values of b_g from each slave swarm;
  Sync threads (wait for all threads to finish);
  if thread index = 0 do
    Initialize master's global best (b^M_g);
  end if
end run in parallel
Sync blocks (wait for all blocks to finish);
for k = 1 → I_MAX do
  Update inertia weight using equation (3);
  run in parallel for each particle of the slave swarms
    Update velocity from the original particles using equation (1);
    Update position from the original particles using equation (2);
    Apply velocity correction to the original particles using equation (4);
    Apply position correction to the original particles using equation (5);
    Update fitness for the original particles;
    Update particle's local best (b_i);
    Sync threads (wait for all threads to finish);
    if thread index = 0 do for each block
      Update global best (b_g) of each slave swarm;
    end if
    Sync threads (wait for all threads to finish);
    Replicate particle N times;
    Apply mutation for each weights (w; C_1, C_2) from all replicated particles;
    Update velocity from the replicated particles using equation (11);
    Update position from the replicated particles using equation (2);
    Apply velocity correction to the replicated particles using equation (4);
    Apply position correction to the replicated particles using equation (5);
    Update fitness for the replicated particles;
    Select the best particles for the next iteration (slave swarm);
    Update particle's local best (b_i);
    Sync threads (wait for all threads to finish);
    if thread index = 0 do for each block
      Update global best (b_g) of each slave swarm;
    end if
  end run in parallel
  Sync blocks (wait for all blocks to finish);
  Update the best global value found in the slave swarms (b^S_g);
  Send to the master swarm every slave swarms' global best (b_g);
  run in parallel for each particle of the master swarm
    Update velocity from the original particles using equation (13);
    Update position from the original particles using equation (2);
    Apply velocity correction to the original particles using equation (4);
    Apply position correction to the original particles using equation (5);
    Update fitness for the original particles;
    Update particle's local best (b^M_i);
    Sync threads (wait for all threads to finish);
    if thread index = 0 do
      Update master's global best (b^M_g);
    end if
    Sync threads (wait for all threads to finish);
    Replicate particle N times;
    Apply mutation for each weight (w; C_1, C_2, C_3) from all replicated particles;
    Update velocity from the replicated particles using equation (12);
    Update position from the replicated particles using equation (2);
    Apply velocity correction to the replicated particles using equation (4);
    Apply position correction to the replicated particles using equation (5);
    Update fitness for the replicated particles;
    Select the best particles for the next iteration (master swarm)
    Update particle's local best (b^M_i);
    Sync threads (wait for all threads to finish);
    if thread index = 0 do
      Update master's global best (b^M_g);
    end if
  end run in parallel
  Sync blocks (wait for all blocks to finish);
end for
```

6 RESULTS

For comparison, the CEMSO-GPU algorithm was subjected to tests with three engineering problems widely used in the scientific literature: Air Tank Design (ATD); Welded Beam Design (WBD); Speed Reducer Design with 25 restrictions (SRD-25). The algorithm CEMSO-GPU is written in the CUDA-C programming language into parallel code which is then executed on the GPU.

The parameter values are: Maximum number of iterations = 1000; number of particles for each slave swarm = 40; number of particles for master swarm = 10; number of replicates per particle = 5; number of slave swarms = 10; correction constant for boundary conditions (α) = 0.22; disturbance constant to the global best (σ_g) = 0.005; parameter of strategies for mutation of the inertia and acceleration factors (σ) = 0.22; factors C_1 and C_2 = 2.05; acceleration factor for master swarm (C_3) = 2.02; number of executed tests per function = 20.

The hardware description is shown in Table 1.

Table 4. Hardware Setup

Hardware	Model	Specs
CPU	Intel Core i5	• **Clock**: 2.66 GHz; • **Cache**: 3 MB de Cache L3;
GPU	NVIDIA GeForce GT 330M	• **Number of CUDA cores**: 48; • **Clock**: 1.26 GHz; • **FLOPs/s**: 182 GFLOPs/s;

The results shown in Tables 2, 3 and 4 are the best values found in 20 executions for each problem.

6.1 Engineering Problems

1. Welded Beam Design (WBD)

Minimizing the manufacturing cost of a steel beam, subject to some restrictions, such as: Shear stress, efforts of beam in the bending, buckling load bar and deflection of the beam end and side constraints [16]. The parameters for this problem are: Thickness of the solder (***H***); beam width (***L***); thickness of the beam (***T***), length of the weld (***B***).

Figure 6 shows the layout of WBD problem.

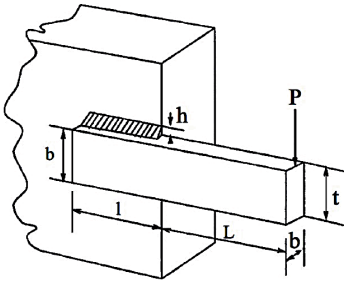


Figure 6. Scheme of Welded Beam Design (WBD). Source: [16].

Table 5. Comparison of the best solutions for WBD

Variables	CEMSO	[15]	[16]	[17]	[18]
X1(h)	0.205730	0.205735	0.171937	0.202369	0.205730
X2 (l)	1.517675	1.517678	4.122129	3.544214	3.470489
X3 (t)	9.036624	9.036624	9.587429	9.048210	9.036624
X4 (b)	0.205730	0.205730	0.183010	0.205723	0.205729
G1	-0.001953	-0.001953	-8.067400	-12.839796	0.000000
G2	-0.001953	-0.001953	-39.336800	-1.247467	0.000002
G3	0,000000	0,000000	-0.011070	-0.001498	0.000000
G4	-3.607646	-3.607646	-3.467150	-3.429347	-3.432984
G5	-0.080730	-0.080730	-0.236390	-0.079381	-0.080730
G6	-0.235540	-0.235540	-16.024300	-0.235536	-0.235540
G7	-0.000488	-0.000977	-0.046940	-11.681355	0.000001
Violations	0	0	0	N/A	1
fitness	1.458883	1.458885	1.664373	1.728024	1.724852

2. Speed Reducer Design With 25 Restrictions (SRD-25)

The first version of speed reduction system was proposed by [20]. It consists on minimizing the weight subject to certain restrictions, such as: Bending of stress of gear teeth, surface tension, transverse deviations of the stems and tensions on the axis. The variables of the problem are: the face width (***b***), module of teeth (***m***), number of teeth on pinion (***z***), length of shaft 1 between bearings (l_1), length of shaft 2 between bearings (l_2), diameter of shaft 1 (d_1), and diameter of shaft 2 (d_2). The objective is to minimize the total weight of the speed reducer [16][19].

Figure 7 shows the layout of SRD-25 problem.

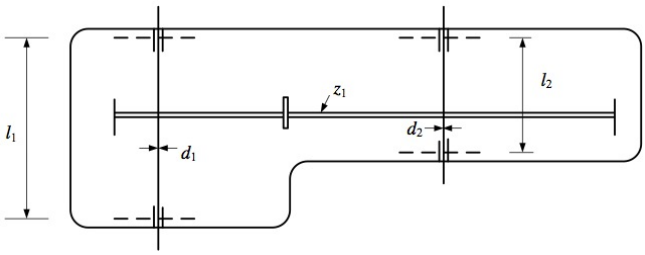


Figure 7. Scheme of Speed Reducer Design (SRD-25). Source: [19]

Table 6. Comparison of the best solutions for SRD-25

Variables	CEMSO	[19]	[20]
X1 (b)	3.241156	3.5197	3.5
X2 (m)	0.700015	0.7039	0.7
X3 (z)	17.000000	17.3831	17.0
X4 (l_1)	7.300010	7.3000	7.3
X5 (l_2)	7.800038	7.7152	7.3
X6 (d_1)	2.900001	3.3498	3.35
X7 (d_2)	5.000000	5.2866	5.29
G1	0.000000	-0.1095	-0.0739
G2	-0.133987	-0.2458	-0.1980
G3	-0.999994	-0.1095	-0.4990
G4	-0.999999	-0.9073	-0.9194
G5	-10.028872	0.0000	0.0001
G6	-132.342728	0.0000	-0.0020
G7	-0.702494	-0.6941	-0.7025
G8	-10.344292	0.0000	0.0000
G9	-0.810928	0.5833	-0.5833
G10	-0.197817	-0.2613	-0.2571
G11	-0.099679	-0.0223	-0.0278
G12	-0.000022	-0.0056	0.0000
G13	-0.124981	-0.1201	-0.1250
G14	0.000000	-0.0220	0.0000
G15	-0.392857	-0.3792	-0.3929
G16	-0.000001	0.0000	0.0000
G17	-0.120481	-0.1205	-0.1205
G18	-0.064107	-0.0538	0.0000
G19	-0.060236	-0.0705	-0.1205
G20	-0.000000	-0.1343	-0.1343
G21	-0.000000	-0.1411	-0.1410
G22	-26.500000	-0.0542	-0.0548
G23	-0.090909	-0.0388	-0.0382
G24	-0.143837	-0.0514	-0.0514
G25	-0.051287	0.0000	0.0574
Violations	0	1	2
fitness	2619.626465	3007.8	2985.2

3. Air Tank Design (ATD)

Minimizing the quantity of material used, which depends on the inner radius (***r***), the shell thickness (***s***), the shell length (***l***), and the head thickness (***h***). The volume of the tank has to be larger than the specified volume (constraint ***G1***), the thicknesses of the head and the wall have to satisfy the ASME code (***G2***, ***G3***), and there are constraints on the size of the tank (***G4***, ***G5***, ***G6***) [19].

Figure 8 shows the layout of ATD problem.

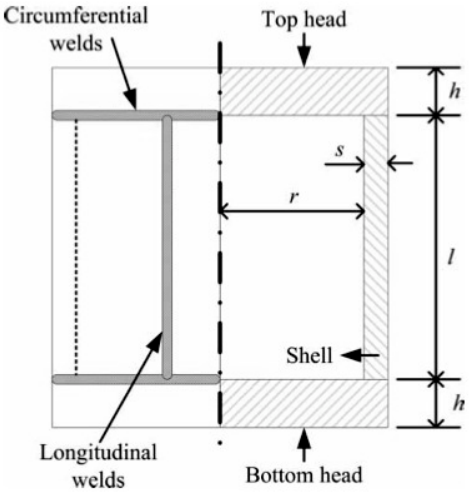


Figure 8. Scheme of Air Tank Design (ATD). Source: [19]

Table 6. Comparison of the best solutions for ATD

Variables	CEMSO	[19]
X1(h)	6.490450	13.67
X2 (l)	603.048584	610.00
X3 (r)	49.898628	105.18
X4 (s)	0.600016	1.01*
G1	-0.000035	-0.000000001
G2	-0.000559	0.0000917*
G3	-0.202475	0.00000000022*
G4	-0.983418	-0.98
G5	-0.663342	-0.29
G6	-0.011396	0.000000000783*
Violations	0	0
fitness	218122.468750	1380000 (1.38 x 10^6)

* In [19], constraints are satisfied with a tolerance of 0.01% of the initial values of the constraints.

7 CONCLUSION AND FUTURE WORKS

Based on the results obtained from the experiments evolving the engineering problems, the CEMSO-GPU algorithm showed the best results compared to the values obtained by other implementations. Although the focus of this work has been the implementation of the algorithm under CUDA, CEMSO can be easily ported to other parallel architectures such as Beowulf with MPI. Some key points about the experiments can be seen below:

- The improvement noted in results can be verified and attributed by following factors: 1) The use of boundary conditions with correction factor applied to a random variable with the rate of correction, where the particle that escapes of the limits of search returns to inner search space; 2) The execution of the master swarm in the particles tends to produce optimized values for each best global of the slave swarms; 3) The replicates produced by each particle allowed a better exploration of the search space;

- From a simplified viewpoint, the algorithm CEMSO can be classified as a hybrid metaheuristic between PSO and EPSO applied under a multi-swarm approach.

- The use of CUDA had a significant contribution to improvements on performance and feasibility of execution for multi-populations algorithms with a small time processing (average of 2.1472 seconds for WBD, 2.0155 seconds for ATD and 2.7235 seconds for SRD-25 among the twenty executions of each problem). In a final analysis, the GPGPU platform of massive parallelism opens possibility for testing involving large loads of processing data in a feasible runtime.

As some future works, we can highlight:

- Inclusion of mechanisms for social interaction based on games theory, inspired on the work developed by [16];

- More research related to other methods involving boundary correction;

- Study and implementation of mechanisms from other metaheuristics to CEMSO (i.e. Genetic Algorithms (GA), Artificial Immunological System Optimization (AISO), Ant Colony Optimization (ACO));

- A technical comparison analysis evolving other parallel methods and architectures (i.e. MPI, OpenMP, pThreads, etc.).

8 ACKNOWLEDGMENTS

This work is supported financially by Research Support Foundation of Pará (FAPESPA) and Federal University of Pará (UFPA).

REFERENCES

[1] C. J. A. Bastos Filho, M. P. Caraciolo, P. B. C. Miranda and D. F. Carvalho, "Multi Ring PSO.", *SBRN'2008 (the 10th Brazilian Symposium on Neural Networks)*, 111-116, (2008).

[2] H. S. Lopes and R. H. C. Takahashi, *Computação Evolucionária em Problemas de Engenharia* (in portuguese), Ed. OMNIPAX, 1st edn, (2011).

[3] V. Miranda and N. Fonseca, "EPSO - Evolutionary Particle Swarm Optimization, a New Algorithm with Applications in Power Systems.", *Transmission and Distribution Conference and Exhibition 2002: Asia Pacific. IEEE/PES*, 745-750 vol. 2, (2002).

[4] H. Van Den Bergh and A. P. Engelbrecht, "A Cooperative Approach to Particle Swarm Optimization.", *IEEE Transactions on Evolutionary Computation*, 225-239, (2004).

[5] D. B. Kirk, W. W. Hwu; *Programming Massively Parallel Processors: A Hands-on Approach*, Elsevier & Morgan Kaufman, 1st edn, (2010).

[6] S. Solomon, P. Thulasiraman and R. Thulasiraman, "Collaborative Multi-swarm PSO for Task Matching Using Graphics Processing Units", *GECCO '11 Proceedings of the 13th annual conference on Genetic and evolutionary computation*, 1563-1570, vol. 2, (2011).

[7] L. Mussi, Y. S. G. Nashed and S. Cagnoni, "GPU-based Asynchronous Particle Swarm Optimization", *GECCO '11 Proceedings of the 13th annual conference on Genetic and evolutionary computation*, 1555-1562 vol. 2, (2011).

[8] J. Kennedy and R. Eberhart, "Particle Swarm Optimization", *Proc. of the IEEE Int. Conf. on Neural Networks*, 1942-1948 (1995).

[9] R. Eberhart and Y. Shi, "Comparing Inertia Weights and Constriction Factors.", *Proceedings of the Congress on Evolutionary Computing*, p. 84-89, 2000.

[10] R. Eberhart and Y. Shi, "A Modified Particle Swarm Optimizer", *IEEE International Conference of Evolutionary Computation*, Anchorage, Alaska, 69-73, (1998).

[11] H. Leite, J. Barros and V. Miranda, "The Evolutionary Algorithm EPSO to Coordinate Directional Overcurrent Relay", *Developments in Power System Protection (DPSP 2010). Managing the Change, 10th IET International Conference*, 1-5, (2010).

[12] J. Sanders and E. Kandrot, *CUDA By Example: An Introduction to General-Purpose GPU Programming*, Addison-Wesley Professional, (2010).

[13] NVIDIA CORP, *NVIDIA CUDA C Programming Guide*, http://developer.download.nvidia.com/compute/DevZone/docs/html/C/doc/CUDA_C_Programming_Guide.pdf, (2012).

[14] B. Niu, Y. Zhu and X. He, "Multi-population Cooperative Particle Swarm Optimization", *Proc. European Conference on Artificial Life*, 874-883, (2005).

[15] D. L. Souza, G. D. Monteiro, T. C. Martins, O. N. Teixeira and V. A. Dmitriev, "PSO-GPU: Accelerating Particle Swarm Optimization In CUDA-Based Graphics Processing Units", *GECCO 2011*, ACM Digital Library, 837-838, (2011).

[16] O. N. Teixeira, W. A. L. L. Lobato, H. S. Yanaguibashi, R. V. Cavalcante, D. J. A. Silva and R. C. L. Oliveira, "Algoritmo Genético com Interação Social na Resolução de Problemas de Otimização Global com Restrições." (in portuguese), *Computação Evolucionária em Problemas de Engenharia* (in portuguese), Ed. OMNIPAX, 1st edn, 197-223, (2011).

[17] Q. He and L. Wang, "An Effective Co-evolutionary Particle Swarm Optimization for Constrained Engineering Design Problems", *Engineering Applications of Artificial Intelligence*, 89-99, (2007).

[18] E. Mezura-Montes and C. Coello Coello, "Useful Infeasible Solutions in Engineering Optimization With Evolutionary Algorithms", *Proceedings of the 4th Mexican International Conference on Artificial Intelligence, MICAI 2005, Lecture Notes on Artificial Intelligence No. 3789*, 652–662, (2005).

[19] Y. L. Hsu and T. C. Liu, "Developing a fuzzy proportional-derivative controller optimization engine for engineering design optimization problems", *Engineering Optimization*, 39:6, 679-700, (2007).

[20] J. Golinski, "An adaptive optimization system applied to machine synthesis", *Mech. Mach. Synthesis*, 8(4), 419–436, (1973).

Modeling and Optimization of Fermentation Processes with Genetic Programming and Quantum-Behaved Particle Swarm Optimization

Jun Sun[1] and Vasile Palade[2] and Xiaojun Wu[1]

Abstract. This paper proposes a novel method for modeling and optimization of fermentation process with a combination of genetic programming (GP) and quantum-behaved particle swarm optimization (QPSO). In this method, first, a GP algorithm is used to model the process, with the parameters of the model selected randomly within a given interval, while the population of models evolves. Then, the parameters of the model obtained by GP are tuned by a QPSO algorithm in order to increase the fitting accuracy. Finally, the values of the independent variables of the model representing the culture conditions are optimized by the QPSO in order to maximize the dependent variable, which generally represents the yield of the fermentation product. The proposed method is applied to the fermentation process of the hyaluronic acid (HA) production by *Streptococcus zooepidemicus*. The experimental results show the efficiency of the GP-QPSO approach in the modeling and optimization of this fermentation process.

1 INTRODUCTION

The goal of a fermentation process is to produce various substances in the pharmaceutical, chemical and food industries. Its performance depends on many factors, including pH, temperature, ionic strengthens, agitation speed, and aeration rate in the aerobic fermentation [1]. To achieve a best performance of a fermentation process, various modeling and optimization strategies have been developed, the most frequently used one being "one-at-a-time" strategy [2]. This approach, however, is not only time consuming, but also ignores the combined interactions between physiochemical parameters [3]. In contrast, the response surface methodology (RSM), which includes factorial design and regression analysis, seeks to identify and optimize significant factors to maximize the response (cell density, high yields of the desired metabolic products or enzyme levels in the microbial system). RSM produces a model, which mathematically describes the relationship that exists between the independent and dependent variables of the process. The most widely used simulating functions in the model developing stage of the RSM are second-order polynomials [4, 5]. The RSM has been widely applied in the modeling and optimization of biochemical processes [6, 7].

[1] Department of Computer Science and Technology, Jiangnan University, No. 1800, Lihu Avenue, Wuxi, Jiangsu Province 214122, China, email: sunjun_wx@hotmail.com; wu_xiaojun@yahoo.com.cn.

[2] Department of Computer Science, University of Oxford, Wolfson Building, Parks Road, Oxford OX1 3QD, UK, email: vasile.palade@cs.ox.ac.uk

In this paper, a method based on genetic programming (GP) and quantum-behaved particle swarm optimization (QPSO) is proposed for modeling and optimization of fermentation processes. In this method, GP is firstly used to model the fermentation process, with the parameters of the model (represented by an individual) being selected within a given interval during the evolution process of the GP. Then, the QPSO algorithm is employed to estimate the parameters of the model in order to improve fitting performance. After that, the QPSO is employed on the obtained model to optimize the culture conditions of the fermentation process, such that the fermentation production is maximized. The proposed GP-QPSO method is applied to model and optimize the hyaluronic acid (HA) production by *Streptococcus zooepidemicus*.

The rest of the paper is organized as follows. Section 2 provides a brief introduction to the GP method and the QPSO algorithm. The proposed GP-QPSO approach is presented in Section 3. Section 4 gives the experimental results for the application of the GP-QPSO method to the HA production. Finally, the paper is concluded in Section 5.

2 GENETIC PROGRAMMING AND QUANTUM-BEHAVED PARTICLE SWARM OPTIMIZATION

2.1 Genetic programming (GP)

As a type of evolutionary algorithms (EAs), where each individual is a computer program, genetic programming (GP) is a methodology inspired by biological evolution to find computer programs that perform a user-defined task [8-10]. It is essentially a machine learning technique used to optimize a population of computer programs according to a fitness landscape determined by a program's ability to perform a given computational task.

GP evolves computer programs, which are traditionally represented in memory as tree structures. Every tree node has an operator function and every terminal node has an operand, making mathematical expressions easy to evolve and evaluate. The evolution in GP, which is shown in the flow chart in Figure 1, proceeds in similar way to a standard GA, i.e. an initial population is generated at random and each individual is evaluated to find its fitness value, and then it is evolved by means of genetic operators as follows:

- Selection: Pairs of parent trees are selected based on its fitness for reproduction.

- Crossover: This process is performed by selecting a node at random then exchanging the associated sub-trees to produce a pair of off-spring trees.
- Mutation: This is performed by either replacing a node selected at random with its associated sub-trees generated randomly or changing its type.

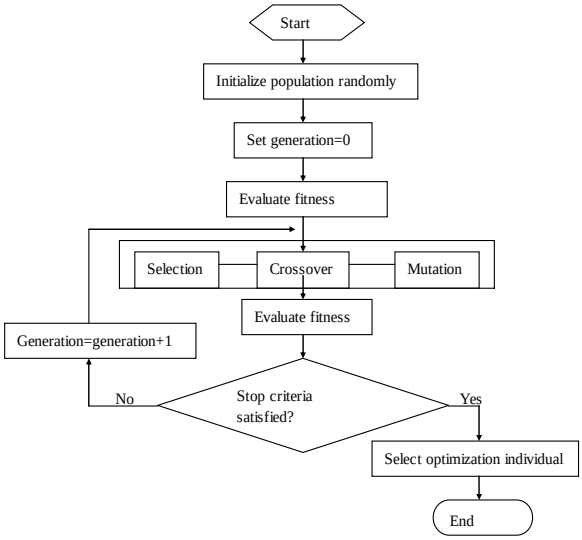


Figure 1. The flow chart of the GP algorithm

2.2 The QPSO algorithm

The PSO algorithm was proposed originally by Kennedy and Eberhart as a population-based optimization technique [11], motivated by the social behavior of bird flocks or fish schooling. In PSO, the potential solutions, called particles, fly through the problem space by following their own experiences and the current best particle. Many empirical studies showed that the PSO algorithm is comparable in performance with, and can be considered as an alternative to the well known GA approach [12].

Since the origin of the PSO in 1995, the algorithm has gained increasing popularity during the last decade due to its effectiveness in performing difficult optimization tasks at a cheap computational cost. There has been a large amount of work done on the PSO algorithm, which involves theoretical analyses, improvements, and applications of the algorithm [13-17].

In the PSO with *m* individuals, each individual is treated as a volume-less particle in the *D*-dimensional space, with the current position vector and velocity vector of particle *i* at the k^{th} iteration represented as $X_{i,k} = (X_{i,k}^1, X_{i,k}^2, \cdots, X_{i,k}^D)$ and $V_{i,k} = (V_{i,k}^1, V_{i,k}^2, \cdots, V_{i,k}^D)$. The particle moves according to the following equations:

$$V_{i,k+1}^j = w \cdot V_{i,k}^j + c_1 r_{i,k}^j (X_{i,k}^j - P_{i,k}^j) + c_2 R_{i,k}^j (X_{i,k}^j - G_k^j) \quad (1)$$

$$X_{i,k+1}^j = X_{i,k}^j + V_{i,k+1}^j \quad (2)$$

for $i = 1,2,\cdots m;\ j = 1,2\cdots, D$, where c_1 and c_2 are called acceleration coefficients. Parameter *w* is the inertia weight that can be adjusted to balance the exploration and exploitation of PSO [13]. Vector $P_{i,k} = (P_{i,k}^1, P_{i,k}^2, \cdots, P_{i,k}^D)$ is the best previous position (the position giving the best objective function value or fitness value) of particle *i* called *personal best* (*pbest*) position, and vector $G_k = (G_k^1, G_k^2, \cdots, G_k^D)$ is the position of the best particle among all the particles in the population and called *global best* (*gbest*) position. Without loss of generality, if we consider the following minimization problem:

$$\text{Minimize} \quad f(X),\ s.t.\ X \in S \subseteq R^D \quad (3)$$

where $f(X)$ is an objective function and S is the feasible space, then $P_{i,k}$ can be updated by

$$P_{i,k} = \begin{cases} X_{i,k} & \text{if} \quad f(X_{i,k}) < f(P_{i,k-1}) \\ P_{i,k-1} & \text{if} \quad f(X_{i,k}) \geq f(P_{i,k-1}) \end{cases} \quad (4)$$

G_k can be found by $G_k = P_{g,k}$, where $g = \arg\min_{1 \le i \le m}[f(P_{i,k})]$. The parameters $r_{i,k}^j$ and $R_{i,k}^j$ are sequences of two different random numbers distributed uniformly within (0, 1), which is denoted by $r_{i,k}^j, R_{i,k}^j \sim U(0,1)$. Generally, the value of $V_{i,k}^j$ is restricted in the interval $[-V_{\max}, V_{\max}]$.

The trajectory analysis [18] demonstrated the fact that convergence of the PSO algorithm may be achieved if each particle converges to its local attractor, $p_{i,k} = (p_{i,k}^1, p_{i,k}^2, \cdots p_{i,k}^D)$ defined at the coordinates

$$p_{i,k}^j = \varphi_{i,k}^j \cdot P_{i,k}^j + (1 - \varphi_{i,k}^j) \cdot G_k^j \quad (5)$$

where $\varphi_{i,k}^j = c_1 r_{i,k}^j / (c_1 r_{i,k}^j + c_2 R_{i,k}^j)$ with regard to the random numbers $r_{i,k}^j$ and $R_{i,k}^j$ in (1). In PSO, the acceleration coefficients c_1 and c_2 are generally set to be equal, i.e., $c_1 = c_2$, and thus $\varphi_{i,k}^j$ is a sequence of uniformly distributed random numbers in (0,1), i.e., $\varphi_{i,k}^j \sim U(0,1)$.

In QPSO [19-23], each single particle is treated as a spin-less one moving in quantum space. Thus, the state of the particle is characterized by a wave function ψ , where $|\psi|^2$ is the probability density function of its position. Inspired by convergence analysis of the particle in PSO, we assume that, at the k^{th} iteration, particle *I* flies in the *D*-dimensional quantum space with a δ potential well centered at $p_{i,k}^j$ on the j^{th} dimension ($1 \le j \le D$). Let $Y_{i,k+1}^j = | X_{i,k+1}^j - p_{i,k}^j |$, we can obtain the normalized wave function at iteration *k*+1

$$\psi(Y_{i,k+1}^j) = \frac{1}{\sqrt{L_{i,k}^j}} \exp(-Y_{i,k+1}^j / L_{i,k}^j) \quad (6)$$

which satisfies the bound condition that $\psi(Y_{i,k+1}^j) \to 0$ as $Y_{i,k+1}^j \to \infty$. $L_{i,k}^j$ is the characteristic length of the wave function. By the definition of the wave function, the probability density function is given by

$$Q(Y_{i,k+1}^j) = |\psi(Y_{i,k}^j)|^2 = \frac{1}{L_{i,k}^j} \exp(-2Y_{i,k+1}^j / L_{i,k}^j) \quad (7)$$

and thus the probability distribution function is

$$F(Y_{i,k+1}^{j}) = 1 - \exp(-2Y_{i,k+1}^{j} / L_{i,k}^{j}) \tag{8}$$

Using Monte Carlo method, we can measure the j^{th} component of position of particle i at the $(k+1)^{th}$ iteration by

$$X_{i,k+1}^{j} = p_{i,k}^{j} \pm \frac{L_{i,k}^{j}}{2} \ln(1/u_{i,k+1}^{j}) \qquad u_{i,k+1}^{j} \sim U(0,1) \tag{9}$$

where $u_{i,k+1}^{j}$ is a sequence of random numbers uniformly distributed within (0, 1). The value of $L_{i,k}^{j}$ is determined by

$$L_{i,k}^{j} = 2\alpha \mid X_{i,k}^{j} - C_{k}^{j} \mid \tag{10}$$

where $C_k = (C_k^1, C_k^2, \cdots, C_k^D)$ is called the mean best (*mbest*) position, defined by the average of the *pbest* positions of all particles, i.e., $C_k^j = (1/m)\sum_{i=1}^{m} P_{i,k}^{j}$ $(1 \le j \le D)$ [20]. Therefore, the position of the particle updates according to the following equation:

$$X_{i,k+1}^{j} = p_{i,k}^{j} \pm \alpha \mid X_{i,k}^{j} - C_{k}^{j} \mid \ln(1/u_{i,k+1}^{j}) \tag{11}$$

The parameter α in equations (10) and (11) is called contraction-expansion (CE) coefficient, which can be adjusted to balance the local and global search of the algorithm during the optimization process. The PSO with equation (11) is called Quantum-behaved Particle Swarm Optimization (QPSO) algorithm. The search procedure of the algorithm is outlined below.

Procedure of the QPSO:

Step 1: Initialize the current positions and personal best positions of all the particles;

Step 2: Execute the following steps;

Step 3: Compute mean best position C_k ;

Step 4: Properly select the value of α ;

Step 4: For each particle in the population, execute from step 5 to 7;

Step 5: Evaluate the objective function value $f(X_{i,k})$;

Step 6: Update $P_{i,k}$ and G_k ;

Step 7: Update each component the particle's position according to equation (5) and (11);

Step 8: While the termination condition is met, return to step 2;

Step 9: Output the results including global best position and its fitness value.

3 THE GP-QPSO METHOD

The GP-QPSO method for modeling and optimization of a fermentation process contains three execution steps. The first step is to use the GP algorithm to model the relationship between the independent and dependent variables of the process with the given experimental data. The terminal nodes for the GP are the independent variables. For example, when modeling the HA production process, the terminal nodes are the culture conditions, including agitation speed (X_1), aeration rate (X_2) and stirrer number (X_3). In order to determine the function nodes, we should take into account the physical meaning of the fermentation process. Generally speaking, only some simple operators are suitable, such as "+", "-", "*", "/", "square", and "cube". The periodical functions, including "sin" and "cos" are undesirable since fermentation is an asymptotic process, not a periodical one. The fitness function for the GP in the proposed approach is the error function given by

$$e(i,t) = \sum_{j=1}^{N_c} (s(i,j) - c(j))^2 \tag{12}$$

where $s(i, j)$ is the value of dependent variable computed using the expression representing the ith individual on the values of the independent variables in the j^{th} experiment; $c(j)$ means value of the dependent variable in the j^{th} experiment, and N_c denotes the number of the experiments. During the evolution of the GP, the parameters of the model (i.e., the mathematical expression given an individual) are randomly selected at given interval.

The second step for the GP-QPSO method is to optimize the parameters of the model obtained by the GP algorithm. Here, the dimension of the search space depends on the number of the parameters of the obtained model. It should be noted that the number of the parameters varies in each different run of the GP algorithm, which may generate completely different model. The fitness function in this step is the one given by (12).

The final step is to optimize the values of the independent variables by using the QPSO in order to maximize the dependent variable, which generally represents the concentration of a fermentation product. The purpose of this optimization task is to increase the production of the fermentation process by optimizing the culture conditions. The dimension of the search space in this step is the number of the dependent variables. The expression of the model is now served as the fitness function. The global best position found by the QPSO is thus the optimized values of the independent variables, and its fitness value is the maximized value of the dependent variable, i.e. the production of the fermentation product.

4 EXPERIMENTAL RESULTS

The proposed GP-QPSO approach was applied to the HA production process. A performance comparison was also made between the RSM and the GP-QPSO.

4.1 Experimental data

To acquire real experimental data, a HA production by batch culture of *S. zooepidemicus* was carried out with an initial sucrose concentration of 70 g/l. One loop of cells from a fresh slant was transferred to 50 ml seed culture medium and cultured on a rotary shaker at 200 rpm and 37 ºC for 12 h. The seed culture was inoculated into a 7-l fermentor (Model KL-7l, K3T Ko Bio Tech, Korea) with a working volume of 4.0 l. The pH was automatically controlled at 7.0 by adding 5 mol/l NaOH solution. Three independent variables including agitation speed, aeration rate and stirrer number were considered as the culture conditions for the process optimization. The detailed level designs of agitation speed, aeration rate and stirrer number were listed in Table 1. The HA concentration was measured by the carbazole method based on uronic acid determination [24].

Table 1. The Box-Behnken experimental design with three independent variables for HA production

Experiment	Agitation speed (rpm)		Aeration rate (vvm)		Stirrer number		HA yield (g/l)
	X_1	Coding X_1	X_2	Coding X_2	X_3	Coding X_3	Y
1	200	-1	2.0	1	3	-1	4.9
2	300	0	1.5	0	4	0	5.2
3	400	1	2.0	1	5	1	4.1
4	200	-1	1.0	-1	3	-1	4.5
5	300	0	1.5	0	2	-1.682	4.3
6	400	1	1.0	-1	3	-1	4.3
7	100	-1.682	1.5	0	4	0	3.6
8	300	0	1.5	0	4	0	5.2
9	300	0	1.5	0	4	0	5.2
10	300	0	1.5	0	4	0	5.2
11	200	-1	1.0	-1	5	1	4.5
12	400	1	2.0	1	3	-1	4.7
13	500	1.682	1.5	0	4	0	3.9
14	200	-1	2.0	1	5	1	4.0
15	400	1	1.0	-1	5	1	4.6
16	300	0	1.5	0	4	0	5.2
17	300	0	1.5	0	4	0	5.2
18	300	0	1.5	0	6	1.682	4.8
19	300	0	0.5	-1.682	4	0	4.7
20	300	0	2.5	1.682	4	0	5.1

4.2 Results for RSM

By applying multiple regression analysis on the experimental data, the following second-order polynomial equation was developed that identifies the relationship of the HA production (Y) with agitation speed (X_1), aeration rate (X_2) and stirrer number (X_3):

$$Y = 5.1968 + 0.0223X_1 + 0.0346X_2 + 0.0263X_3 - 0.4918X_1^2 - 0.0853\ X_2^2 - 0.2090\ X_3^2 - 0.0012\ X_1X_2 - 0.0750X_1X_3 - 0.225X_2X_3 \quad (13)$$

From equations derived from the differentiation of equation (13), the optimal values of X_1, X_2 and X_3 (in the coded units) were found to be -1.4, -0.71 and -1.00, respectively. Correspondingly, we can obtain the maximum point of the model, which was 260 rpm for ~~of~~ agitation speed, 1.15 vvm for aeration rate, and 3 for stirrer number. The maximum predicted value of the HA production (Y) is 5.27 g/l.

4.3 Results for modeling with the GP algorithm

When the GP algorithm was used for modeling the fermentation process of the HA production, the terminal nodes of an individual represented the agitation speed (X_1), aeration rate (X_2) and stirrer number (X_3); the set of the function nodes was (+, -, *, /, square, cube); the population size was 40; the crossover rate was 0.8; the mutation rate was 0.2; and the number of the maximum generation is 100.

The GP algorithm was run for 10 times and the resulting model with the best fitness value 0.6325 is given by

$$Y = 5.1663 - 0.0179X_1 + 0.0774X_2 - 0.2346X_3 - 0.4844X_1^2 - 0.0985X_2^2 - 0.2008X_3^2 + 0.0220X_1^3 - 0.0048X_2^3 + 0.1323X_3^3 . \quad (14)$$

The convergence process the GP algorithm is visualized in Figure 2, which shows that the fitness value, namely the error converged to the best fitness value after 70 generations.

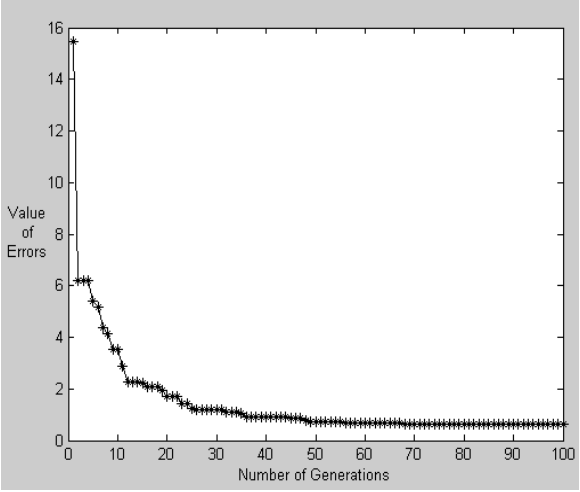


Figure 2. Convergence of the GP algorithm

4.4 Results for modeling with the GP-QPSO method

In the modeling of the fermentation process of the HA production with the GP-QPSO method, the QPSO algorithm was used to optimize the parameters of the model generated by the GP algorithm as described in equation (14). It can be seen that the number of the parameters of the model is 10. Therefore the dimension of the search space is 10 and the current and *pbest* positions of the i^{th} particle at the kth iteration of the QPSO are denoted as $X_{i,k} = (X_{i,k}^1, X_{i,k}^2, \cdots, X_{i,k}^{10})$ and $P_{i,k} = (P_{i,k}^1, P_{i,k}^2, \cdots, P_{i,k}^{10})$ respectively, with each component of the position vector representing a parameter (i.e., a coefficient) of the model. The objective function (i.e., the fitness function) is still the error function described by equation (12).

When employing the QPSO algorithm for the optimization task, we used 40 particles for the algorithm, which ran for 10 times, with each run executed for 300 iterations. The CE coefficient α of the QPSO decreased linearly from 1.0 to 0.5 on the course of the

running. The resulting model with the best fitness value 0.4809 out of 10 runs of the QPSO is described by

$$Y = 5.1968 - 0.0874X_1 - 0.1037X_2 - 0.3133X_3 - 0.4918X_1^2 - 0.0853X_2^2 - 0.2090X_3^2 + 0.0624X_1^3 + 0.0787X_2^3 + 0.1633X_3^3 \quad (15)$$

The convergence process of the GP-QPSO modeling is traced in Figure 3. It can be observed that the algorithm had a good convergence, with the fitness value converging to the best value 0.4809 after almost 130 iterations. Table 2 compares the performance of the RSM, GP and GP-QPSO by listing their fitness values, i.e., the errors of the models on the given experimental data. As evident from the results, the GP-QPSO methods generated the model with the best quality, for its error is the lowest among all the competitive models.

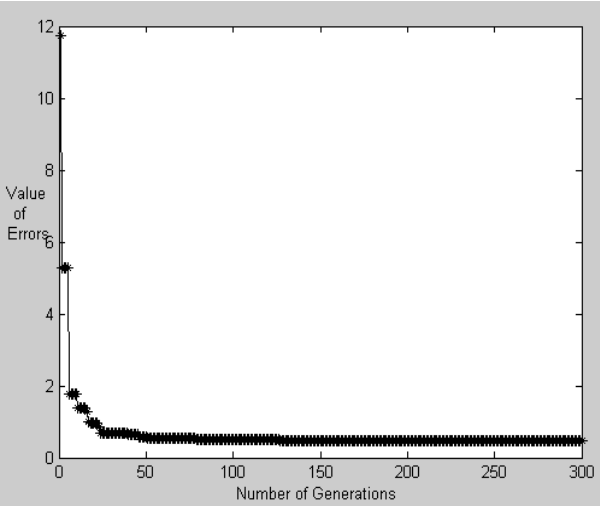


Figure 3. Convergence process of the GP-QPSO

Table 2. The comparison between the results of the GP, GP-QPSO and the RSM

Modeling Methods	Errors
RSM	0.6560
GP	0.6325
GP-QPSO	0.4809

4.5 Results on the optimization of the culture conditions

Based on the model produced by the GP-QPSO modeling method, we further optimize with the QPSO algorithm the independent variables, namely the culture conditions including the agitation speed (X_1), the aeration rate (X_2) and the stirrer number (X_3), in order to maximize the HA production. In this optimization procedure, the dimension of the search space is 3, the number of the culture conditions. Twenty particles were used for the optimization task and the CE coefficient decreased linearly from 1.0 to 0.5 over the running of the algorithm. The QPSO ran 10 times with each run lasting 100 iterations.

Table 3. Culture conditions and HA yield, optimized for the models obtained by the RSM and GP-QPSO methods.

Models	Agitation speed	Aeration rate	Stirrer number	HA yield (g/l)
RSM	260	1.15	3	5.27
GP-QPSO	278	1.65	3	5.57

The best fitness value, i.e., the best HA yield obtained out of 10 runs of the algorithm, was 5.57 g/l with the optimized agitation speed 278 rpm, aeration rate 1.65 vvm and stirrer number 3, as shown in Table 3. The optimized values of the independent variables of the model by the RSM are also provided in the table. It can be found that the optimization of the GP-QPSO based model resulted in higher HA yield. The convergence process of the QPSO in this search is shown in Figure 4.

We tested the optimized culture conditions of the model by using the GP-QPSO method with the same experimental conditions as those for acquiring the experimental data. We found that the HA yield was 5.55 with the agitation speed 278 rpm, aeration rate 1.65 vvm and 3 stirrer, which indicates that the model generated by the GP-QPSO fits well the fermentation process of the HA production.

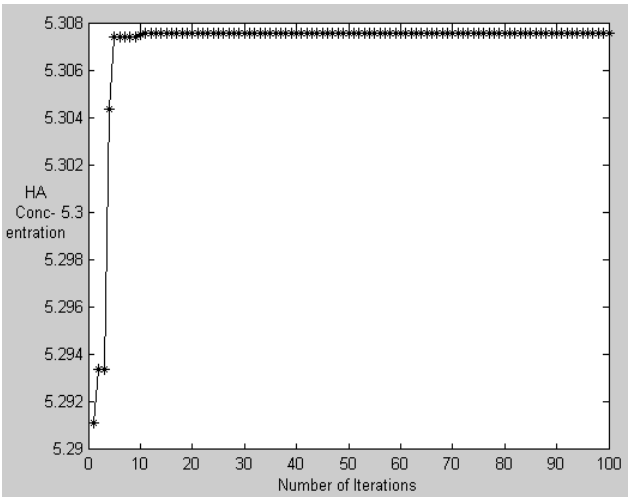


Figure 4. Convergence of the QPSO algorithm in the optimization of the culture conditions

5 CONCLUSION

In this paper, we proposed a GP-QPSO method for modeling and optimization of a fermentation process. This method firstly uses the GP algorithm to model the relationship between the independent variables and the dependent variable of the process. During the evolution of the GP, the parameters of the model represented by an individual are randomly selected within a given interval. The obtained model outputted by the GP is further improved by tuning its parameters with the QPSO algorithm. The independent variables of the resulting model, which represent the culture conditions of the fermentation process, are then optimized by the QPSO such that the dependent variable (i.e., the yield of the fermentation product) can be maximized.

The GP-QPSO approach was applied to the fermentation process of the HA production. With the given experimental data, the GP-QPSO was showed to fit the process better than the RSM method and the GP algorithm. It was also found that the optimization of the culture conditions of the model generated by the proposed method can lead to a better HA yield, which was verified by real experiments.

ACKNOWLEDGEMENTS

This work is partially supported by Natural Science Foundation of China (NSFC), under grant number 601190117 and 60975080, by Program for New Century Excellent Talents in University, and by the Natural Science Foundation of Jiangsu Province, China (Project Number: BK2010143).

REFERENCES

[1] M. Kennedy and D. Krouse, 'Strategies for improving fermentation medium performance: a review', *Journal of Industrial Microbiology and Biotechnology*, 23, 456-475, (1999).

[2] J.R. Dutta, P.K. Dutta, R.Banerjee, 'Optimization of culture parameters for extracellular protease production from a newly isolated *Pseudomonas sp*. using response surface and artificial neural network models', *Process Biochemistry*, 39, 2193-2198, (2004).

[3] J.H. Sim and A.H. Kamaruddin, 'Optimization of acetic acid production from synthesis gas by chemolithotrophic bacterium-*Clostridium aceticum* using statistical approach. *Bioresource Technology*, 99, 2724-2735, (2008).

[4] H. Ceylan, S. Kubilay, N. Aktas, N. Sahiner, 'An approach for prediction of optimum reaction conditions for laccase-catalyzed bio-transformation of 1-naphthol by response surface methodology (RSM)', Bioresource Technology, 99, 2025-2031, (2008).

[5] S.W. Chang, J.F. Shaw, K.H. Yang, S.F. Chang, C.J. Shieh, 'Studies of optimum conditions for covalent immobilization of *Candida rugosa* lipase on poly (gamma-glutamic acid) by RSM', *Bioresource Technology*, 99, 2800-2805, (2008).

[6] A. Kunamneni, S. Singh, 'Response surface optimization of enzymatic hydrolysis of maize starch for higher glucose production', *Biochemical Engineering Journal*, 27, 179–190, (2005).

[7] F.I. Ustok, C. Tari, N. Gogus, 'Solid-state production of polygalacturonase by *Aspergillus sojae* ATCC 20235', *Journal of Biotechnology*, 127, 322-334, (2007).

[8] W. Banzhaf et al., *Genetic Programming—An Introduction*, Morgan Kaufmann, San Francisco, 1998.

[9] J. R. Koza. *Genetic Programming*, volume I. MIT Press, New York, 1992.

[10] W. B. Langdon. *Data structures and genetic programming*, Advances in Genetic Programming 2. MIT Press, Cambridge, 1996.

[11] J. Kennedy, R. C. Eberhart, 'Particle swarm optimization', *in Proc. 1995 IEEE Int'l Conference on Neural Networks*, pp. 1942-1948.

[12] K.O. Jones, 'Comparison of genetic algorithm and particle swarm optimization', in *Proc. 2005 International Conference on Computer System and Technologies*, 2005, pp. IIIA1-6.

[13] Y. Shi, R. C. Eberhart, 'A modified particle swarm optimizer', in *Proc. IEEE Int'l Conf. Evolutionary Computation*, 1998, pp. 69-73.

[14] M. Clerc, 'The swarm and the queen: towards a deterministic and adaptive particle swarm optimization', in *Proc. Congress on Evolutionary Computation*, 1999, vol. 3, pp. 1951-1957.

[15] D. Bratton, J. Kennedy, 'Defining a standard for particle swarm optimization', in *Proc. of IEEE Swarm Intelligence Symposium*, 2007, pp. 120-127.

[16] J. Kennedy, "Bare bones particle swarms," in *Proc. 2003 IEEE Swarm Intelligence Symp.*, Indianapolis, IN, April 2003, pp. 80-87.

[17] J. Kennedy, 'Probability and dynamics in the particle swarm', in *Proc. 2004 Congr. Evol. Comput.*, vol. 1, June 2004, pp. 340-347.

[18] M. Clerc and J. Kennedy, 'The particle swarm-explosion, stability and convergence in a multidimensional complex space', *IEEE Transactions on Evolutionary Computation*, 6, 58–73, (2002).

[19] J. Sun, B. Feng and W.B. Xu, 'Particle swarm optimization with particles having quantum behavior', *in Proc. 2004 Congress on Evolutionary Computation*, pp.326-331.

[20] J. Sun, W.B. Xu and B. Feng, 'A global search strategy of quantum-behaved particle swarm optimization', *in Proc. 2004 IEEE conference on Cybernetics and Intelligent Systems*, pp.111-116.

[21] J. Sun, W Fang, X. Wu, V. Palade, W. Xu, 'Quantum-behaved particle swarm optimization: Analysis of the Individual Particle's Behavior and Parameter Selection', *Evolutionary Computation*, 20, (3), 349–393, (2012).

[22] J. Sun, X. Wu, V. Palade, W. Fang, C.-H. Lai, W. Xu, 'Convergence Analysis and Improvements of Quantum-behaved Particle Swarm Optimization', *Information Sciences*, 193, 81-103, (2012).

[23] J. Sun, W. Fang, V. Palade, X. Wu, W. Xu, 'Quantum-behaved particle swarm optimization with Gaussian distributed local attractor point', *Applied Mathematics and Computation*, 218(7), 3763-3775, (2011).

[24] H. Bitter, HM. Muir, 'A modified uronic acid carbazole reaction', Analytical Biochemistry, 4, 330-334, (1962).

Feature Extraction by Fusing Local and Global Contextual Constraints based Linear Discriminant Analysis for Face Recognition

Xiaoqi Sun[1] and Xiaojun Wu[1] and Jun Sun[1]

Abstract. In this paper, we present a novel scheme for feature extraction for face recognition by fusing local and global contextual constraints based linear discriminant analysis (LGCCLDA). The facial changes due to variation of pose, illumination, expression, etc., often appear locally in the face images. Therefore, global features extracted from the whole image fail to cope with these variations. To address these problems, our method first divides the original images into modular sub-images and then CCLDA is utilized to each of these sub-images as well as to the whole images to extract local and global discriminant features respectively. Moreover, CCLDA can take into the contextual constraints in images, which can provide useful information for classification. Experimental results obtained on ORL and XM2VTS show the effectiveness of our method.
.

1 INTRODUCTION

Human face recognition is one of the most valuable biometric identification methods, which involves image processing, pattern recognition, computer vision and others. It has been the hot topic over the past years. In terms of application, face recognition has two types, one of them is human identity recognition, which recognizes the identity of the person with face images and aims to solve " who are you " ; the other is face verification, which determines whether the one is the designated person and aims to solve whether you are the claimed one [1]. Feature extraction is the most fundamental problem in pattern recognition. For pattern classification, the purpose of feature extraction is to map the original data into a discriminative feature space in which the samples from different classes are clearly separated [2-5]. Principal Component Analysis (PCA) [6] and Linear Discriminant analysis (LDA) [7] are typical methods. However, in real applications, due to the high dimensionality of feature and usually small number of samples, the classical LDA always fails because of the small sample size (SSS) problem. To address this problem, extensive methods have been proposed in the literature [8-16].Moreover, the performance of the above methods degrades when dealing with images having local facial changes like facial expression, illumination condition, pose changes, etc. Due to these changes, only some face regions will vary and rest of the regions will remain the same. So R.Gottumukkal [17] proposed a modular PCA approach where face images are divided into smaller sub-images and then PCA method is applied to each of these sub-images. And the experimental results show that the modular PCA method has the ability to cope with these variations.

However, most of the existing subspace learning methods like PCA, LDA and so on, consider the pixels in image independently, not taking into account their spatial relationship [18].It is well known that images with certain pattern occupy specific manifold, which is constrained by contextual information in high-dimensional feature space. Therefore, contextual constraint in image is important for classification. One of the most successful work to model the contextual information is the Markov Random Fields (MRFs) [19-21] which derive the results by maximizing the posterior probability in Bayesian deduction framework. However, the optimization by MRF is somewhat computational expensive and is easy to converge into local minima that limits its application. Wang et al. [22] proposed a novel image matching distance considering the spatial information. However, they did not demonstrate how to integrate the contextual information into dimensionality reduction problem. In [18], Lei and Li proposed contextual constraints based linear discriminant analysis (CCLDA) which incorporates the contextual information into linear discriminant analysis. The main difference of CCLDA from the previous ones is that it takes into account the image contextual constraints during feature dimensionality reduction.

In this paper, we propose an improved algorithm for contextual constraints based linear discriminant analysis with application to face recognition. First, to preserve the local feature and reduce the computational complexity, a face image is partitioned into a set of sub-images with equal size. Then CCLDA is adopted to both original images and sub-patterns to extract the global and local features respectively

The rest of the paper is organized as follows. Feature extraction by fusing local and global contextual constraints based linear discriminant analysis is introduced in Section 2. Experimental results are proposed in Section 3 and conclusions are drawn in Section 4.

2 FEATURE EXTRACTION BY FUSING LOCAL AND GLOBAL CONTEXTUAL CONSTRAINTS BASED LINEAR DISCRIMINANT ANALYSI

The proposed method consists of three main steps: (1) partition face images into sub-patterns, (2) apply CCLDA to the original

[1] School of IoT Engineering, Jiangnan University, Wuxi 214122, China, email: wu_xiaojun@yahoo.com.cn

images for global feature extraction and sub-patterns for local feature extraction, (3) classification. We will describe the three steps in detailed.

2.1 Face image partition

In local matching based face recognition methods, a face image can be partitioned into a set of equally or unequally sized sub-images, depending on user's option. However, how to choose appropriate sub-image size which gives optimal performance is still an open problem. So without loss of generality, equal size partition is adopted in our study.

Let $Z=[z_1,z_2,\ldots,z_M]$ denote M face images belonging to C classes in the training set, and the size of each image z_i is $m\times n$. We first partition each face image into S equal size sub-images in a non-overlapping way, and then further concatenate them into corresponding column vectors with dimensionality of $\lceil m\times n/S \rceil$. After all training images are partitioned, the sub-image vectors at the same position of all face images are collected to form a specific sub-pattern's training set. Therefore, we can get S separate sub-pattern sets totally. This image partition process is illustrated in Figure 1.

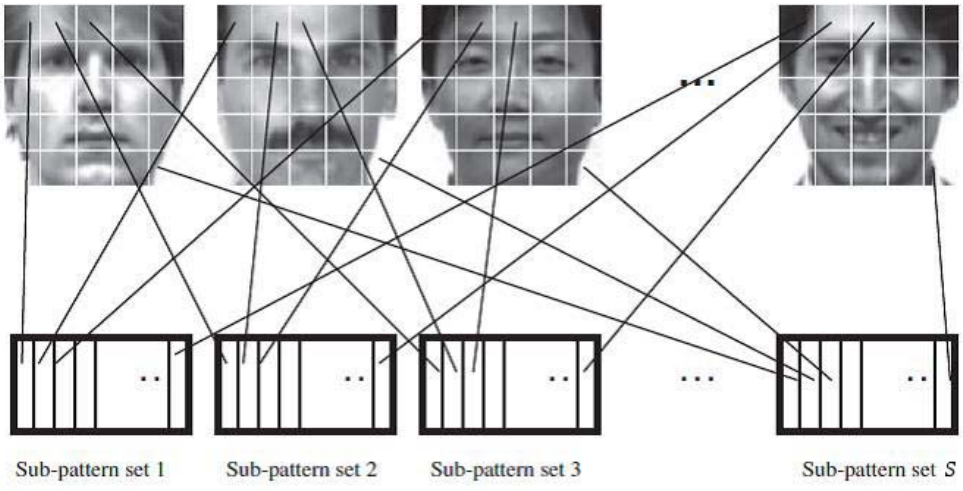


Figure 1. the process of image partition

2.2 Feature Extraction of Contextual Constraints based Linear Discriminant Analysis (CCLDA)

Let $sub-pattern^{(s)}=[z_1^s, z_2^s \ldots z_M^s]$ denote the M sub-images in the s-th sub-pattern set and where z_i^s denote the s-th sub-pattern vector of the i-th face image (i=1, 2… M; s=1, 2…S). A set of the original face images samples $\{z_i\}$ can be represented as an $K\times M$ matrix $Z=[z_1,z_2,\ldots,z_M]$, where K is the number of pixels in the images and M is the number of samples. Then we conduct the CCLDA method with each of the sub-pattern sets as well as the whole original images respectively.

CCLDA incorporates the contextual information into linear discriminant analysis. In CCLDA, intuitively, if the pixels are of the similar property or reflect the similar structure, the weights on them would have strong relationship; otherwise the weights on independent pixels would also be weakly related. Following this idea, a constraint $J_2(w)=\frac{1}{2}\sum_{ij}(w_i-w_j)^2 S_{ij}$ is imposed on traditional LDA to formulate the object of discriminant analysis as [18]:

$$J=\frac{w^T S_b w}{(w^T S_w w+\eta J_2(w))} \tag{1}$$

where

$$S_b=\frac{1}{M}\sum_{i=1}^{C} n_i(u_i-u)(u_i-u)^T \tag{2}$$

$$S_w=\frac{1}{M}\sum_{i=1}^{C}\sum_{j=1}^{n_i}(z_j^i-u_i)(z_j^i-u_i)^T \tag{3}$$

where n_i is the number of samples belonging to the i-th class, u_i is the mean vector of the i-th class and u is the total mean vector of all the samples.

Moreover, S_{ij} describes the similarity of pixels i and j, and η is a coefficient to balance the trade-off between the training discriminant power and contextual constraints. The constraints function $J_2(w)$ gives a high penalty when the weights of related pixels differ too much. Due to the symmetry of S_{ij} in general case, the contextual constraints $J_2(w)$ on weight image can be formulated using the matrix operations further as follows:

$$\begin{aligned} J_2(w)&=\frac{1}{2}\sum_{ij}(w_i-w_j)^2 S_{ij} \\ &=w^T L^w w \end{aligned} \tag{4}$$

where $L^w=D-S$ is the Laplacian matrix, and D is the diagonal matrix where $D_{ii}=\sum_j S_{ij}$. Thus, the objective of CCLDA can be

formulated as:

$$J = \frac{w^T S_b w}{(w^T S_w w + \eta w^T L^w w)} \tag{5}$$

The optimal projection w can be obtained by solving the following generalized eigenvalue problem:

$$S_b w = \lambda (S_w + \eta L^w) w \tag{6}$$

In our paper, the similarity matrix of weights are determined as follows[18]:

$$S_{ij} = \begin{cases} e^{\|f_i - f_j\|^2 / \sigma^2}, & if\ i\ and\ j\ are\ neighbors \\ 0, & otherwise \end{cases} \tag{7}$$

where f_i and f_j are the feature vectors extracted at position i and position j, respectively to describe the texture and spatial relationship between position i and position j. It should be noted the definition of weight similarity S is not limited. Different definitions are possible according to different problems.

2.3 Classification

Through CCLDA, the global transformation matrix W and each local transformation matrix w^s (s=1, 2… S) based on the sub-pattern$^{(i)}$ are obtained. And then the whole images and the sub-images in sub-patterns of training face images can be projected into low-dimensional subspace. In order to classify a test face, the unknown test face image x is classified with two individual classifiers.

Case1: Firstly, we divide x into S sub-patterns in the same way previously applied to the training images. Let $X = [x^1, x^2, ..., x^S]$ denotes the set of sub-patterns partitioned from x. Then the dimension-reduced test sub-pattern set can be derived by $v^s = (w^s)^T x^s$ (s=1, 2… S) and we get $V = [v^1, v^2, ..., v^S]$. For each v^s in low-dimensional set V, the Euclidean distance between the given feature and all dimension-reduced training sub-patterns in the s-th sub-pattern set is measured. So we have the distance vector for sub-pattern v^s as:

$$DM^s = [d(v^s, y_1^s), d(v^s, y_2^s), ..., d(v^s, y_M^s)] \tag{8}$$

Since the training samples $y_1^s, y_2^s, \ldots, y_M^s$ come from different face images, the nearest neighbor classifier can be used here, that is, we classify v^s to the class of y_i^s which has smallest $d(v^s, y_i^s)$ in DM^s, for i=1,2,…,M.

Considering all the sub-patterns in V, the final recognition result of the testing face image can be obtained by a majority voting method. The probability of the test image x belonging to the j-th class can be determined approximately:

$$p_j = \frac{1}{S} \sum_{i=1}^{S} q_i^j \tag{9}$$

where

$$q_i^j = \begin{cases} 1, & if\ the\ ith\ sub-pattern\ is\ classified\ to\ the\ jth\ person \\ 0, & otherwise \end{cases} \tag{10}$$

The final identity result of x is obtained by:

$$Identity(x) = \arg\max_j (p_j),\ j = 1, 2, ..., C \tag{11}$$

Case 2: for the whole testing sample x, the nearest neighbor classifier is also used and then we classify x to the class of y_i which has smallest distance with x, for i=1,2,…,M.

Finally, we fuse the two classifiers to recognize a face by one of the above possible case. That is to say: (1) If both the classifiers recognize the test image x as the same class C_j, we can choose either of them. (2)Otherwise, if the first classifier recognizes S/2 or more sub-images of an image x as class C_j, we classify a test face by case 1, otherwise, we adopt case 2 to classify a test image.

3 EXPERIMENTAL RESULTS AND ANALYSIS

In order to test the performance of the proposed algorithm in this paper, face recognition experiments have been conducted on both the XM2VTS database and the ORL database. In experiments, the database was divided into the two sets: training set and test set. We select several images from each person for training, whereas the rest of each individual are used for testing.

3.1 Parameter slection

There are mainly three parameters in our proposed method that affecting the performance of algorithm. One is the number of equal size sub-images from each image S, another are the contextual information regularized coefficient η in Eq.(1) and the value of σ in Eq.(7). The parameter σ is empirically set to be the average distance among these feature vectors extracted by grouping the pixel values at corresponding position from all the training images. And in [18] CCLDA can get its highest recognition rate at the point of η =0.0005. Therefore, in all the following experiments, the parameter η is set to 0.0005. For the parameter S, we try to examine the impact of different values on the performance of algorithm and then choose the best one.

Table1 and Table 2 respectively show the recognition rate and computational time of our proposed method and MPCA on XM2VTS and ORL face database as a function of S. In experiments, all the images are cropped to 32×32 and we select five images from each person for training.

Table 1. Recognition rate of different methods with different sub-images

(a) Recognition rate of different methods with different sub-images on XM2VTS

Method	**2*2**	**2*4**	**2*8**	**4*4**	**4*8**
MPCA(%)	83.28	86.55	87.46	87.23	85.08
LGCCLDA(%)	89.15	93.79	95.25	94.46	94.80

(b) Recognition rate of different methods with different sub-images on ORL

Method	**2*2**	**2*4**	**2*8**	**4*4**	**4*8**
MPCA(%)	92.00	87.50	83.00	79.00	70.00
LGCCLDA(%)	92.50	92.50	93.50	95.00	94.50

Table 2. computational time of different methods with different sub-images

(a) Computational time of different methods with different sub-images on XM2VTS

Method	2*2	2*4	2*8	4*4	4*8
MPCA(s)	137.219	214.437	394.172	412.281	784.672
LGCCLDA(s)	269.281	353.641	573.750	542.954	969.516

(b) Computational time of different methods with different sub-images on ORL

Method	2*2	2*4	2*8	4*4	4*8
MPCA(s)	7.203	7.5	11.14	10.954	18.25
LGCCLDA(s)	23.906	25.938	28.657	29.281	37.391

It is easy to see that our method can achieve better recognition performance than MPCA with different sub-images. Moreover, with the value of the parameter *S* increasing, the computational time also becomes longer. And our method can get higher result when the value of *S* is 16. To reduce the computational complexity and get better recognition performance, in all the following experiments, we partitioned all images into 4×4 sub-images. For MPCA, the images are partitioned into 2×2 sub-images in the same way.

3.2 Experiments on XM2VTS database

The XM2VTS database [25] is a multi-model database consisting of video sequences of talking faces recorded for 295 subjects at one month intervals. The data has been recorded in 4 sessions with 2 shots taken per session. From each session two facial images have been extracted to create an experimental face database of size 55×51. Figure 2 shows examples of images in XM2VTS.

Figure 2. Part of the images in XM2VTS

In this experiment, we compare the recognition performances of different methods on XM2VTS database, which are shown in Table 3 and Figure 3. And in LGCCLDA, images are partitioned into 4×4 sub-images, that is *S*=16; in MPCA, images are partitioned into 2×2 sub-images.

Table 1 Performance of different methods with different training set size on XM2VTS

Method	G2	G3	G4	G5	G6
PCA	61.98	73.02	74.75	81.02	82.03
MPCA	64.92	74.17	76.61	83.28	84.92
LDA	62.09	80.68	85.85	90.96	90.51
KPCA	58.94	72.75	74.07	79.89	80.00
KDA	64.92	80.75	86.69	92.88	92.37
CCLDA	66.21	91.39	91.02	91.86	92.22
LGCCLDA	86.38	91.80	92.54	94.46	95.76

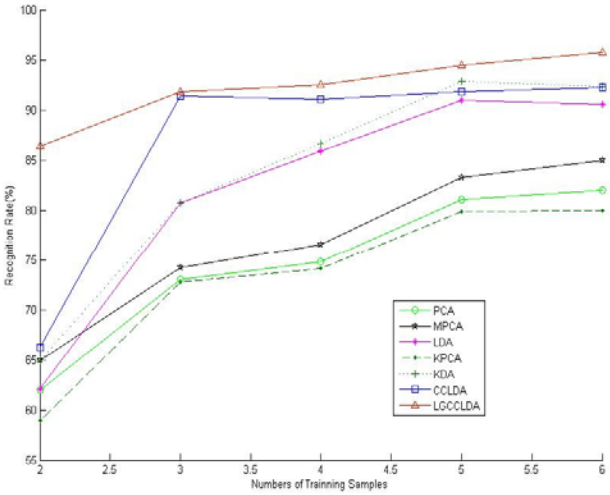


Figure 3 The performance of different methods with different training set size on XM2VTS

Both Table 3 and Figure 3 show the recognition accuracy of different methods on XM2VTS database. In the table 3, Gi refers to the number of each person for training is *i*. First, we can see that our method improves the performance of face recognition significantly and is more effective than other methods. Moreover, we can find that CCLDA and KDA can get high recognition rate when we select enough samples for training, which implies that both of them can deal with the facial changes due to variation of pose, illumination, expression effectively. However, our method can get the desired results even in less training samples, which implies that the performance of our method does not change rapidly with the change of training samples and is more robust.

3.3 Experiments on database of ORL

The ORL database [26] contains images from 40 individuals, each providing 10 different images. For some subjects, the images are taken at different times. The facial expressions and facial details also vary. The images are taken with a tolerance for some tilting and rotation of the face of up to 20 degrees. Moreover, there is also some variation in the scale of up to about 10 percent. All images are grayscale and normalized to a resolution of 92×112 pixels. Figure 4 shows examples of images in ORL.

In this section, we compare the performance of our proposed method with several other methods including PCA, MPCA, LDA, KPCA, KDA and CCLDA on ORL database, which are shown in Table 4 and Figure 5. And in LGCCLDA images are divided into 4×4 sub-images and in MPCA, images are divided into 2×2 sub-images.

Both Table 4 and Figure 5 show the recognition accuracy of different methods on ORL database. In the table 4, Gi refers to the number of each person for training is *i*. First, we can see that Table 4 and Figure 5 tell us the same story as in Table 3 and Figure 3. Moreover, we can find that our method can all get higher recognition accuracy than other methods in different training samples.

Figure 4. Part of the images in ORL

Table 4. Performance of different methods with different training set size on ORL

Method	G2	G3	G4	G5	G6	G7	G8
PCA	75.31	79.25	85.42	88.50	91.25	93.33	93.75
MPCA	76.25	78.57	87.92	92.00	93.13	95.00	95.00
LDA	74.06	80.36	87.92	90.00	91.87	95.00	95.00
KPCA	77.50	80.00	86.25	89.50	91.87	94.17	93.75
KDA	77.50	85.71	92.50	92.50	95.63	95.83	97.50
CCLDA	73.75	81.43	90.00	93.00	95.00	95.83	97.50
LGCCLDA	82.50	85.00	92.50	95.00	96.25	97.50	98.75

Both Table 4 and Figure 5 show the recognition accuracy of different methods on ORL database. In the table 4, Gi refers to the number of each person for training is i. First, we can see that Table 4 and Figure 5 tell us the same story as in Table 3 and Figure 3. Moreover, we can find that our method can all get higher recognition accuracy than other methods in different training samples.

The above experimental results show that the proposed method has more advantages for the face recognition than some other methods. However, because of the computational complexity, our method will take more time than CCLDA.

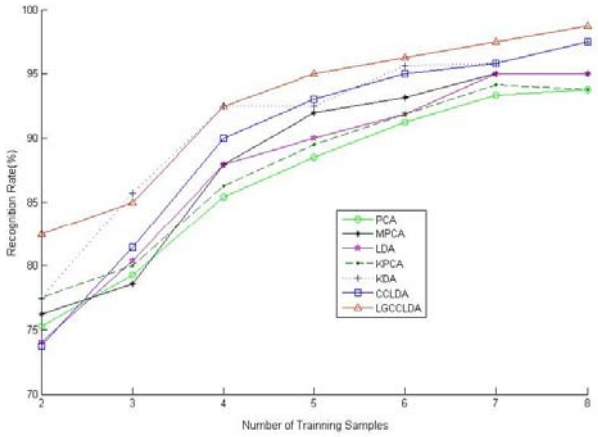


Figure 5. The performance of different methods with different training set size on ORL

4 CONCLUSIONS

In order to get better performance in face recognition, an improved algorithm is proposed based on the contextual constraints based linear discriminant analysis. The proposed method integrates the advantages of global and local feature extraction and takes into the spatial relationship of pixels in images. The experimental results show the effectiveness of our method. However, how to choose appropriate sub-image size which gives optimal performance is still an open problem. So our future work is to improve the effectiveness of choosing sub-images and find more efficient methodology to improve the performance in face recognition.

ACKNOWLEDGEMENTS

This work was supported in part by the following projects: 111 Project of Chinese Ministry of Education (Grant No. B12018), Key Grant Project of Chinese Ministry of Education (Grant No.: 311024), National Natural Science Foundation of P. R. China (Grant No.: 60973094, 61103128).

REFERENCES

[1] ROSENFELD, ‘A Survey: image analysis and computer vision. Computer Vision and Image Understanding’, 62(1), 33-93L, (1997).

[2] Y. Ng. Andrew, M. I. Jordan and Y. Weiss, ‘On spectral clustering: analysis and an algorithm’, Advances in Neural Information Processing Systems, 849-856, (2001).

[3] D. Bitouk, M. I. Miller and L. Younes, ‘Clutter invariant ATR’, IEEE Transactions on Pattern Analysis and Machine Intelligence., 27(5), 817-821, (2005).

[4] J. K. Moon, K. Kim and Y. Kim, ‘Design of missile guidance law via variable structure control’, Journal of Guidance, Control, and Dynamics, 24, 659-663, (2001).

[5] S.-G. Sun, H.-W. Park, ‘Invariant feature extraction based on radial and distance function for automatic target recognition’, In Proceeding of ICIP, 3, 345-348, (2002).

[6] I. T. Jolliffe, ‘Principal component analysis’, Springer-Verlag, New York, (1996).

[7] P.N. Belhumeur, J.P. Hespanha and D.J. Kriegman, ‘Eigenfaces vs fisherfaces: recognition using class specific linear projection’, IEEE Transactions on Pattern Analysis and Machine Intelligence., 19(7), 711-720, (1997).

[8] J.H. Friedman, ‘Regularized discriminant analysis’, Journal of the American Statiscal Association, 84, 165-175, (1989).
[9] L.F. Chen, H.Y.M. Liao, M.T. Ko, G.J. Yu, ‘A new LDA-based face recognition system which can solve the small sample size problem’, Pattern Recognition, 33 (1), 1713–1726, (2000).
[10] H. Li, T. Jiang, K. Zhang, ‘Efficient and robust feature extraction by maximum margin criterion’, IEEE Transactions on Neural Networks. 17 (1), 1157–1165, (2006).
[11] F. Song, D. Zhang, D. Mei, et al., ‘A multiple maximum scatter difference discriminant criterion for facial feature extraction’, IEEE Transaction on Systems, Man, and Cybernetics-Part B: Cybernetics, 33 (6), 1566–1599, (2007).
[12] X.-J. Wu, J Kittler, J.-Y. Yang, ‘An analytical algorithm for determining the generalized optimal set of discriminant vectors’. Pattern Recognition, 37(9), 1949-1952, (2004).
[13] W. Zheng, C. Zou, L. Zhao, ‘Weighted maximum margin discriminant analysis with kernels’, Neurocomputing, 67 (8), 357–362, (2005).
[14] X.-J. Wu, J Kittler, ‘A New Direct LDA (D-LDA) Algorithm for Feature Extraction in Face Recognition’, Proceedings of the 17th International Conference on Pattern Recognition, 548-555, (2004).
[15] X.S. Zhuang, D.Q. Dai, ‘Inverse Fisher discriminant criteria for small sample size problem and its application to face recognition [J]. Pattern Recognition. 2005, 38 (11), 2192–2194.
[16] X.S. Zhuang, D.Q. Dai. Improved discriminant analysis for high-dimensional data and its application to face recognition’, Pattern Recognition, 40 (5) , 1570–1578, (2007).
[17] R. Gottumukkal, Vijayan K. Asari, ‘An improved face recognition technique based on modular PCA approach’, Pattern Recognition Letters, 25, 429-436, (2004).
[18] Z. Lei, S.Z. Li, ‘Contextual constraints based linear discriminant analysis’, Pattern Recognition Letters, 32, 626-632, (2011).
[19] R. Huang, V. Pavlovic, D. Metaxas, ‘A hybrid face recognition method using markov random fields’, In: ICPR. Cambridge, UK, 157–160, (2004).
[20] S.C. Dass,, A. K. Jain, ‘Markov face models’, In: ICCV. Vancouver, 112–116, (2001).
[21] S. C. Dass, A. K. Jain, X. Lu, ‘Face detection and synthesis using Markov random field models’, In: ICPR. Quebec City, 12–116, (2002).
[22] L. Wang, Y. Zhang, J. Feng, ‘On the euclidean distance of images’, IEEE Trans. PAMI. 27 (8), 1334–1339, (2005).
[23] J.-Z. Wang, Z.-Q. Ma, B.-X. Zhang, etc., ‘A structure-preserved local matching approach for face recognition’, Pattern Recognition Letters, 32, 494-504, (2011).
[24] S. Chowdhury, J.K. Sing, D. K. Basu, etc., ‘A hybrid approach to face recognition using generalized two-dimensional fisher’s linear discriminant method’, Third International Conference on Emerging Trends in Engineering and Technology, 506-511, (2010).
[25] X.-J, Wu, J Kittler, etc., ‘On dimensionality reduction for client specific discriminant analysis with application to face verification’, In: LNCS 3338. Berlin: Springer, 305-312, (2004).
[26] J. Yang, D. Zhang, etc., ‘Two-dimensional PCA: a new approach to appearance-based face representation and recognition’, IEEE Transactions on Pattern Analysis and Machine Intelligence, 26(1), 131-137, (2004).