UNIVERSITY OF SZEGED
Faculty of Science and Informatics
Department of Computer Algorithms and Artificial Intelligence

PhD School in Computer Science

A special class of fuzzy operators and its application in modelling effects and decision problems

Thesis of PhD Dissertation

József Dániel Dombi

Supervisor
Dr. János Csirik

Szeged, 2013

1 Introduction

Fuzzy sets were introduced by Lofti Zadeh in 1965 with the aim of reconciling mathematical modeling and human knowledge in the engineering sciences. Most of the building blocks of the theory of fuzzy sets were proposed by him, especially fuzzy extensions of classical basic mathematical notions like logical connectives, rules, relations and quantifiers.
Over the last decade fuzzy sets and fuzzy logic have become more popular areas for research, and they are being applied in fields such as computer science, mathematics and engineering. This has led to a truly enormous literature, where there are presently over thirty thousand published papers dealing with fuzzy logic, and several hundreds books have appeared on the various facets of the theory and the methodology. However, there is not a single, superior fuzzy logic or fuzzy reasoning method available, although there are numerous competing theories.
The Pliant system is a kind of fuzzy theory that is similar to a fuzzy system [2]. The difference between the two systems lies in the choice of operators. In fuzzy theory the membership function plays an important role, but the exact definition of this function is often unclear. In pliant systems we use a so-called distending function, which represents a soft inequality. In the Pliant system the various operators, which are called the conjunction, disjunction and aggregative operators, are closely related to each other.
The main contribution of this thesis will be to show how the Pliant system can be applied to a variety of problems in the real world. During my studies I was guided by pragmatism and utility. First, by creating a dynamic system, one can create a system like the Fuzzy Cognitive Map. Second, one can apply the Pliant system by introducing function approximation techniques, which have useful and practical aspects. And third, one can apply it in problems that use decision-making techniques.

2 Definitions and basic notations

Here, we provide the basic definitions and notations that are required to understand fuzzy logic and concrete applications of it in the real world.

2.1 Negation operator

Definition 1. *We say that $n(x)$ is a negation if $n\colon [0,1] \to [0,1]$ satisfies the following conditions:*

C1:	$n:[0,1]\to[0,1]$ *is continuous*	*(Continuity)*
C2:	$n(0)=1, n(1)=0$	*(Boundary conditions)*
C3:	$n(x)<n(y)$ *for* $x>y$	*(Monotonicity)*
C4:	$n(n(x))=x$	*(Involution)*

2.2 Conjunctive, disjunctive operator and modifiers

In the Pliant concept we characterize the operator class (strict t-norm and strict t-conorm) for which various negations exist and build a DeMorgan class. The fixpoint ν_* (or the neutral value ν) can be regarded as decision threshold. Operators with various negations are useful because the threshold can be varied.

Theorem 1. *$c(x,y)$ and $d(x,y)$ build a DeMorgan system for $n_{\nu_*}(x)$ where $n_{\nu_*}(\nu_*)=\nu_*$ for all $\nu_*\in(0,1)$ if and only if*

$$f_c(x)\,f_d(x)=1. \tag{1}$$

For proof see [3].

Definition 2. *The general form of the pliant system is*

$$o_\alpha(x,y)=f^{-1}\left((f^\alpha(x)+f^\alpha(y))^{1/\alpha}\right) \tag{2}$$

$$n_\nu(x)=f^{-1}\left(f(\nu_0)\frac{f(\nu)}{f(x)}\right) \qquad \textit{or} \tag{3}$$

$$n_{\nu_*}(x)=f^{-1}\left(\frac{f^2(\nu_*)}{f(x)}\right), \tag{4}$$

where $f(x)$ is the generator function of the strict t-norm operator and $f:[0,1]\to[0,\infty]$ continuous and strictly decreasing function.

Definition 3. *The general form of the modifier operators in the pliant system is*

$$\kappa^{(\lambda)}_{\nu,\nu_0}(x)=f^{-1}\left(f(\nu_0)\left(\frac{f(x)}{f(\nu)}\right)^\lambda\right) \tag{5}$$

2.3 Aggregative operator

The term *uninorm* was first introduced by Yager and Rybalov [7]. Uninorms are a generalization of t-norms and t-conorms, got by relaxing the constraint on the identity element in the unit interval $\{0,1\}$. The paper of Fodor, Yager and Rybalov [4] is notable since it defined a new subclass of uninorms called representable uninorms.

The aggregative operator is defined [1] in the following way:

Definition 4. *An aggregative operator is a function* $a : [0,1]^2 \to [0,1]$ *with the properties:*

1. *Continuous on* $[0,1]^2 \backslash \{(0,1),(1,0)\}$
2. $a(x,y) < a(x,y')$ *if* $y < y', x \neq 0, x \neq 1$
   $a(x,y) < a(x',y)$ *if* $x < x', y \neq 0, y \neq 1$
3. $a(0,0) = 0$ *and* $a(1,1) = 1$ *(boundary conditions)*
4. $a(x,a(y,z)) = a(a(x,y),z)$ *(associativity)*
5. *There exists a strong negation n such that* $a(x,y) = n(a(n(x),n(y)))$ *(self-DeMorgan identity) if* $\{x,y\} \neq \{0,1\}$ *or* $\{x,y\} \neq \{1,0\}$
6. $a(1,0) = a(0,1) = 0$ *or* $a(1,0) = a(0,1) = 1$

The multiplicative form of the aggregative operator is

$$a_{\nu_*}(\mathbf{w},\mathbf{x}) = f_a^{-1}\left(f_a^{1-\sum_{i=1}^{n} w_i}(\nu_*)\prod_{i=1}^{n} f_a^{w_i}(x_i)\right) \tag{6}$$

2.4 Distending function

In fuzzy concepts the most powerful term is the membership function. Up until now the research community could not give an unambiguous definition of this term. In the Pliant concept we give one which is connected to the operator system. In the Pliant concept, we will introduce the distending function. We will use the notation

$$\delta(x) = truth(0 < x) \qquad x \in R$$

We can generalize this in the following way

$$\delta(g(\mathbf{x})) = truth\big(0 < g(\mathbf{x})\big) \qquad \mathbf{x}\varepsilon R^n$$

Instead of a strict relation, we will define a function which provides information on the validity of the relation.

In fuzzy logic theory, the membership function has a different interpretation. In the Pliant concept, the membership function is replaced by a soft interval. Its mathematical description is

$$\delta_{a,b}^{\lambda_1,\lambda_2}(x) = truth(a <_{\lambda_1} x <_{\lambda_2} b)$$

Definition 5. *In a pliant system if the initial conditions are*

$$\delta_{a,b}^{\lambda_1,\lambda_2}(a) = \nu_0 \qquad \delta_{a,b}^{\lambda_1,\lambda_2}(b) = \nu_0, \tag{7}$$

then the distending interval is

$$\delta_{a,b}^{\lambda_1,\lambda_2}(x) = f^{-1}\left(\frac{1}{A}\left(A_1 e^{-\lambda_1(x-a)} + A_2 e^{-\lambda_2(b-x)}\right)\right), \tag{8}$$

where

$$\begin{aligned} A &= \frac{1}{f(\nu_0)}\left(1 - e^{-(\lambda_1+\lambda_2)(b-a)}\right) \\ A_1 &= 1 - e^{-\lambda_2(b-a)} \\ A_2 &= 1 - e^{-\lambda_1(b-a)} \end{aligned} \tag{9}$$

2.5 The Pliant system

The operators of the Pliant system are

$$c(\mathbf{x}) = \frac{1}{1 + \left(\sum_{i=1}^{n} w_i \left(\frac{1-x_i}{x_i}\right)^{\alpha}\right)^{1/\alpha}} \tag{10}$$

$$d(\mathbf{x}) = \frac{1}{1 + \left(\sum_{i=1}^{n} w_i \left(\frac{1-x_i}{x_i}\right)^{-\alpha}\right)^{-1/\alpha}} \tag{11}$$

$$a_{\nu_*}(\mathbf{x}) = \frac{1}{1 + \left(\frac{1-\nu_*}{\nu_*}\right)\prod_{i=1}^{n}\left(\frac{1-x_i}{x_i}\frac{1-\nu_*}{\nu_*}\right)^{w_i}} \tag{12}$$

$$n(x) = \frac{1}{1 + \left(\frac{1-\nu_*}{\nu_*}\right)^2 \frac{x}{1-x}}, \tag{13}$$

$$\kappa_{\nu}^{(\lambda)}(x) = \frac{1}{1 + \frac{1-\nu_0}{\nu_0}\left(\frac{\nu}{1-\nu}\frac{1-x}{x}\right)^{\lambda}}$$

where $\nu_* \in]0,1[$, with generator functions

$$f_c(x) = \left(\frac{1-x}{x}\right)^{\alpha} \qquad f_d(x) = \left(\frac{1-x}{x}\right)^{-\alpha}, \tag{14}$$

where $\alpha > 0$. The operators c, d and n fulfil the DeMorgan identity for all ν, a and n fulfil the self-DeMorgan identity for all ν, and the aggregative operator is distributive with the strict t-norm or t-conorm.

3 Results of the dissertation

3.1 Decision making

I applied the Pliant system in the Grid environment as a decision support algorithm. In the first part of the related chapter in the dissertation, I introduced the history and unit elements of the Grid environment. The Grid Meta-Broker itself is a standalone Web-Service unit element that can serve both users and grid portals, and it has a direct connection with brokers. The novel enhanced scheduling solutions allows a higher level, interoperable brokering by utilising existing resource brokers of different grid middleware. The Grid Meta-Broker gathers and utilises meta-data about brokers from various grid systems to establish an adaptive meta-brokering service. To improve the scheduling performance of the Meta-Broker we need to send the job to the broker that best fits the requirements; and it executes the job without failures in the shortest possible execution time. Each broker has *four properties* that the algorithm can rely on: a success counter, a failure counter, a load counter and the running jobs counter.

Here, I introduced several new scheduling components for this Meta-Broker. These algorithms utilise the Broker's properties for making decisions. The best one, called Decision Maker, uses Pliant functions with a random generation (which in the figure is called Decision 5) in order to select a good performing broker for user jobs even under conditions of high uncertainty. This algorithm defines a score number for each broker and uses the generator function to select a broker, and it uses the kappa function to determine the broker's score number. The best algorithm performs the following steps.

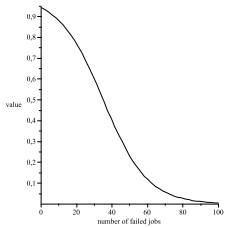

Figure 1: Normalising the failed jobs counter using Sigmoid function

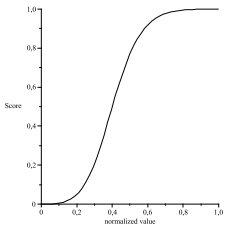

Figure 2: Normalized parameter values using the Kappa function

Because the Pliant system is defined in the $[0, 1]$ interval, we need to *normalize* the input value. This algorithm uses the sigmoid function to normalise the input values.

We should also *emphasize* that the closer the value is to one, the better the broker is, and if the value is close to zero, it means that the broker is not good. For example if the failure counter is high, both normalization algorithms should give a value close to zero because it is not a good thing if the broker has a lot of failed jobs (see Figure 1). The opposite is true for the success counter.

In the next step, we can modify the normalised property value by using the same Kappa function (see Figure 2). We can also define the expected value of the normalisation via the ν and λ parameters.

To *calculate* the score value, we can make use of the conjunctive or aggregation operator. After running some tests, we found that we got better results if we used the aggregation operator. In this step, the result is always a real number lying in the $[0, 1]$ interval and then we multiply it by 100 to get the broker's score number.

To further enhance the scheduling we developed a *training process* that can be executed before the simulation in order to initialise the first and second properties. This process sends a small number of jobs with various properties to the brokers and sets the successful and failed jobs number at the BPDLs of the brokers. We evaluated our algorithms in a grid-simulation environment based on GridSim, and performed simulations with real-world workload samples.

The simulation results for the algorithms with training can be seen in Figure 3. As we mentioned earlier, we used a training process to initialize the performance values of the brokers before job submissions. In this way, the quality of the decisions for the first round of jobs can be improved. Upon

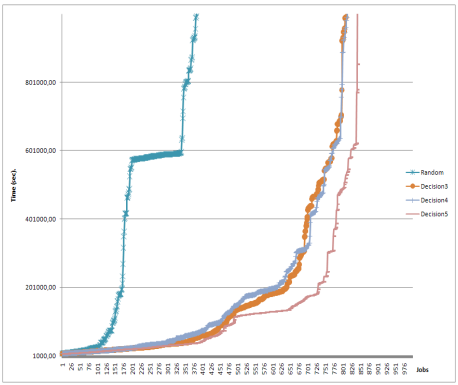

Figure 3: Simulation results for three decision algorithms with training compared with the random decision maker

examining the results, Decision4 still performs about the same as Decision3, but Decision5 clearly *outperforms* the other two.

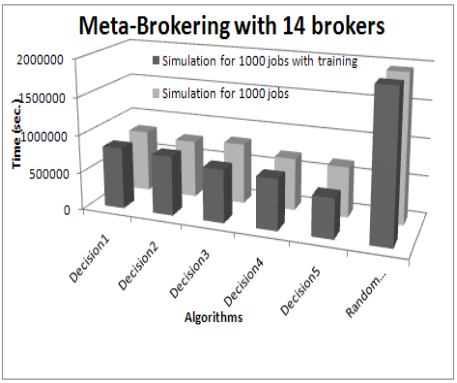

Figure 4: Simulation in the main evaluation environment

In Figure 4, we provide a graphical summary of the various evaluation phases. The columns show the average values of each evaluation run with the same parameter values. The results clearly demonstrate that the more intelligence (more sophisticated methods) we put into the system, the better the performance.

The evaluation results accord with our expected utilisation gains; namely, the enhanced scheduling provided by the revised Decision Maker resulted in

a more efficient job execution.

3.2 Function approximation

In Chapter 4, I described a simple approximation technique that may be inappropriate in some sense. Then I developed a new type of non-linear regression method that is based on the distending function and provides a natural description of the function. In the approximation method, the distending function could be applied in two different ways. During the creation of the distending function in the first method we could use the peak of the function, and in the second method we could use the length of an interval. Next, I defined a function and showed how to create a function with this technique. Because the aggregation has a neutral value, we have to transform the interval into $[0,\nu]$ or $[\nu,1]$. I will define positive and negative effects using the distending interval. That is,

$$P_{a_1,a_2}^{\lambda_1,\lambda_2}(x) = \frac{1}{2}\left(1+\gamma\sigma_{a,b}^{\lambda_1,\lambda_2}(x)\right) \tag{15}$$

$$N_{a_1,a_2}^{\lambda_1,\lambda_2}(x) = \frac{1}{2}\left(1-\gamma\sigma_{a,b}^{\lambda_1,\lambda_2}(x)\right), \tag{16}$$

where the scaling factor $\gamma \in [0,1]$ controls the intensity of the effect.

We can use the impulse function to interpolate the function. To create the function we will use the aggregation operator. If λ_1 and λ_2 are not too large, then we can get a smooth approximation.

Next, I defined a function and showed how to decompose a function using this technique. In general, the function here will be defined by coordinates. Now, we will use a function with a dense sampling procedure. In each example we will use 100 equidistant coordinates on the given interval. Now choose a function $F : R \to [0,1]$ to be approximated. Our task is to decompose it into effects. This can be done via our distending function (approximation) or impulse function (interpolation) procedures. First, the usual step is to smooth the function $F(x)$.

Algorithm for using the distending function

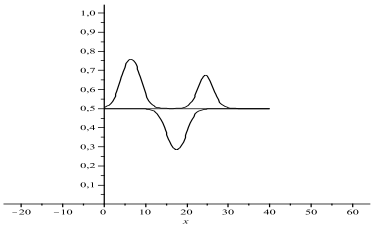

Figure 5: Optimal components

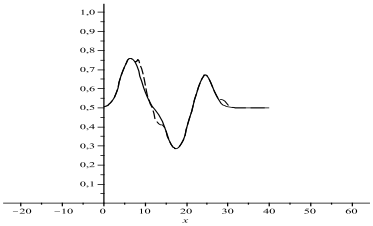

Figure 6: The function and its approximation

1. Let us find the the local minimum and maximum of the function $F(x)$

$$F(c_i) = A_i \qquad \text{such that}$$
$$F(x) < A_i \quad \text{if} \quad x \in (c_i - \varepsilon, c_i + \varepsilon)$$

$$F(c_j) = A_j \qquad \text{such that}$$
$$F(x) > A_j \quad \text{if} \quad x \in (c_i - \varepsilon, c_i + \varepsilon)$$

2. Let us define the $[a_i, b_i]$ intervals

$$a_1 = c_1 - \frac{c_1 + c_2}{2}, \quad b_1 = \frac{c_1 + c_2}{2}, \ldots$$
$$a_n = \frac{c_{n-1} + c_n}{2}, \quad b_n = c_n + \frac{c_{n-1} + c_n}{2},$$

where

$$c_1 < c_2 < c_3 < \ldots < c_k$$

3. Let us define the initial values of λ_{i_1} and λ_{i_2} by

$$\lambda_{i_1} = \frac{f(c_i) - f(a_i)}{c_i - a_i} \qquad \lambda_{i_2} = 2\frac{f(b_i) - f(c_i)}{b_i - a_i}$$

4. Let us build the initial values of the function and get

$$G_{a,b}^{\lambda_1,\lambda_2}(x) = \sum_{i=1}^{n} \delta_{a_i,b_i}^{\lambda_{i_1},\lambda_{i_2}}(x)$$

See Figure (5).

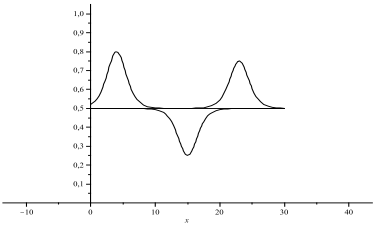

Figure 7: Main and optimal effects in the interpolative case for the function

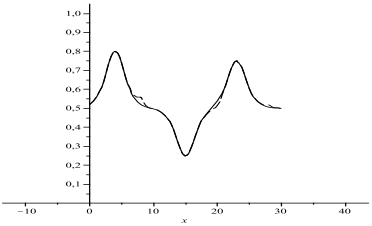

Figure 8: The function and its interpolative approximation

5. Now find the optimal solution of the $a_i, b_i, \lambda_{i_1}, \lambda_{i_2}$ values with the suggested initial values.

$$\min_{\mathbf{a},\mathbf{b},\underline{\lambda}_1,\underline{\lambda}_2} \sum \left(G_{\mathbf{a},\mathbf{b}}^{\underline{\lambda}_1,\underline{\lambda}_2}(x_i) - F(x_i) \right)^2$$

It is not easy to minimize this because a given minimum may not be the global minimum. However, because $G_{\mathbf{a},\mathbf{b}}^{\underline{\lambda}_1,\underline{\lambda}_2}(x)$ is a continuous function of its parameters and the initial values are well chosen, we can get good results.

The results of this approximation are shown in Figure (6).

Algorithm for the impulse approximation

Let us find the maximum and minimum values of $F(x)$

$$c_1 < c_2 < c_3 < \ldots < c_k,$$

where c_i and c_{i+1} are the minimum and maximum (or maximum and minimum) points.

If $f(c_i) = A_i$, let the initial value of the approximation function be the following:

$$A_i = f(c_i) - \frac{1}{2}, \qquad \lambda_{1_i} = \frac{c_i - c_{i-1}}{A_i - A_{i-1}} \qquad \text{and} \qquad \lambda_{2_i} = \frac{c_{i+1} - c_i}{A_{i+1} - A_i}$$

See Figure (7).

The procedure used here is the same as that for the interval approximation.

In Figure (8), we plotted the results of applying this procedure.

This algorithm uses the BFGS method to get accurate results. I found that this method was fast (only a few iteration steps were required for the optimisation method), it was efficient and easy to use. Using this technique, it is possible to change only a part of the function, instead of the usual case where it is not possible to do so.

3.3 Cognitive Map

In the final chapter of the dissertation, I presented the Cognitive Map. Here, my intention was handle complex dynamic systems using the Pliant system. This concept is similar to the Fuzzy Cognitive Map, but the functions and the aggregation procedures are quite different. FCMs are hybrid methods that lie in some sense between fuzzy systems and neural networks. Knowledge is represented in a symbolic way using states, processes and events. Each piece of information has a numerical value. In Figure 9 we can see a typical FCM model, which is a directed graph.

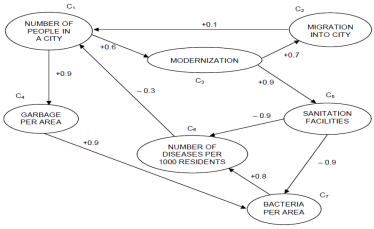

Figure 9: The FCM model

Our method is based on a continuous-valued logic and all the parameters have a semantic meaning. In the *Pliant Cognitive Map*, we can define influences. The sigmoid function naturally maps the values to the (0,1) interval. Positive (negative) influences can be built with the help of *two sigmoid functions and the conjunctive operator*. Hence, we get the generalized positive impulse function

$$c(t,u,v,a,b) = \frac{1}{1+ue^{-\lambda_1(t-a)}+ve^{-\lambda_2(t-b)}}$$

where u and v are weights. In Figure 3.3, we observe a basic influence. If the influence is neutral, we can represent it by a 1/2 value. If there are no influences, then we can continuously order the 1/2 values in the system. If we

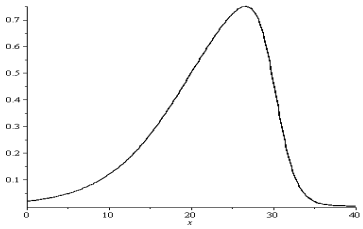

Figure 10: An average influence.

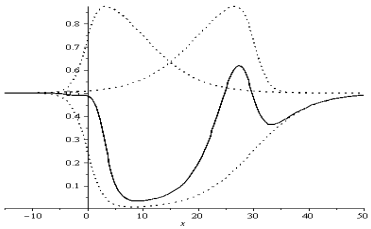

Figure 11: The aggregation of the influences.

want to model positive influences, we order a value which is larger than 1/2, and maximal value is 1. The negative influence is the negation of the positive influence. To create these influences, we will use the following transformations:

$$P(t,u,v,a,b) = \frac{1}{2}(1+c(t,u,v,a,b))$$

$$N(t,u,v,a,b) = \frac{1}{2}(1-c(t,u,v,a,b))$$

In Figure 3.3 we have plotted the aggregation of positive and negative effects.

It is also possible to create an effect by using *sigmoid functions alone*. This has another meaning, which is useful when we do not know the size of the effect. So in this case we model the effect as an impulse. The domain is the same as before, so the neutral value is 1/2. To satisfy these requirements, all we have to do is to transform the sigmoid function into [0.5,1.0] if we want to create a positive impulse or [0.0,0.5] if we want to represent a negative impulse. To create an effect we will utilize the following transformation functions:

$$P(x,a,\lambda_1) = \frac{1}{2}(1+\sigma(x,a,\lambda_1))$$

$$N(x,a,\lambda_2) = \frac{1}{2}(1-\sigma(x,a,\lambda_2)),$$

where $\lambda_1 > 0$ and $\lambda_2 > 0$. It should be mentioned that if the value of the effect attains zero or one then the aggregation of the effect remains 1. So we need to transform the sigmoid function into something slightly smaller than 1 and slightly larger to 0. Here, we will use the $[0.15, 0.95]$ interval.

In Figure 13, we see the aggregation of positive and negative effects by just using sigmoid functions.

Construction of the PCM To simulate the system, all we have to do is to aggregate the influences. The aggregation operator provides a guarantee

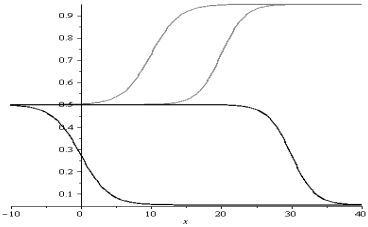

Figure 12: An average influence got by using sigmoids.

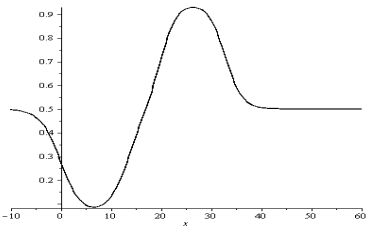

Figure 13: The aggregation of sigmoid influences.

that we will use influences in the right way. This means that the requirements of the simulation are fulfilled. The following steps should be carried out to simulate the system:

1. Collect the concepts.
2. Define the expectation values of the nodes (i.e. threshold values of the aggregations).
3. Build a cognitive map (i.e. draw a directed graph between the concepts).
4. Define the influences (i.e. whether they are positive or negative).

The iterative method:

1. Use the proper function or give a timetable for the input nodes.
2. Calculate the positive and negative influences using step 4.
3. Aggregate the positive and negative influences, where the ν_0 value of the aggregation parameter is the previous value of the concept C_j.

A framework was also described that was developed by the author. By using this framework, I provided some basic examples to illustrate how Pliant Cognitive Maps work. Then I demonstrated that this method could be used in a heat exchanger real-world environment. A heat exchanger is a standard device in the chemical and process industry [5]. This is a special tank where the temperature control is still a major challenge as the heat exchanger is used over a wide range of operating conditions. The system, which has a nonlinear behaviour, strongly depends on the flow rates and on the temperature of the medium. It is well known that the FCM can be used to model and

control the heat exchanger process [6]. I applied the PCM in this problem. Evaluating the results, I found that the values between the PCM simulation steps smoothly decrease, but required more simulation steps. In this example I used the same influence for each concept all the time, but it is also possible to change the strength of the influence and model real-world situations better.

4 Summary of the thesis points

The results achieved in the dissertation are summarized in three thesis groups.

4.1 Decision making

I applied the Pliant system in the Grid environment as a decision support algorithm. Here, I introduced several new scheduling components for this Meta-Broker. These algorithms utilize the Broker's properties for making decisions. The best one, called Decision Maker, uses Pliant functions with a random generation in order to select a good performing broker for user jobs even under conditions of high uncertainty. I evaluated my algorithms in a grid-simulation environment based on GridSim, and performed simulations with real-world workload samples. The evaluation results accord with our expected utilization gains; namely, the enhanced scheduling provided by the revised Decision Maker resulted in a more efficient job execution.

4.2 Function approximation

I described the basic approximation technique that may be inappropriate in some sense. I developed a new type of non-linear regression method that is based on the distending function and provides a natural description of the function. In the approximation method, the distending function could be applied in two different ways. During the creation of the distending function in the first method we could use the peak of the function, and in the second method we could use the length of an interval. I defined a function and showed how to create and decompose it with this technique. This algorithm uses the BFGS method to get accurate results. I found that this method was fast (only a few iteration steps were required for the optimisation method), it was efficient and easy to apply. With this technique, it is possible to change only a part of the function, instead of the usual case where it is not possible to do so.

4.3 The Cognitive Map

I ivestigated the Cognitive Map. Here, my intention was to handle a complex dynamic system using the Pliant system. This concept is similar to the Fuzzy Cognitive Map, but the functions and the aggregation procedures are quite different. It is based on a continuous-valued logic and all the parameters have a semantic meaning. I defined two different kinds of method to create an effect and I showed how to build the Pliant Cognitive Map. A framework was also described that was developed by the author. With this framework, I gave some basic examples to illustrate how Pliant Cognitive Maps work. I demonstrated that this method could be used in a real-world environment. Evaluating the results, I found that the values between the simulation steps smoothly decrease, but required more simulation steps. In this example I used the same influence for each concept all the time, but it is also possible to change the strength of the influence and model real-world situations better.

5 Author's publications related to the thesis

Thesis point 1

- 2008: Attila Kertész, József Dániel Dombi, József Dombi: Adaptive scheduling solution for grid meta-brokering, In Acta Cybernetica 2009, Volume 19, 105-123
- 2010: József Dániel Dombi, Attila Kertész: Scheduling solution for grid meta-brokering using the pliant system, In Second International Conference on Agents and Artificial Intelligence, Valencia-Spain (2010)

Thesis point 2

- 2009: J. Dombi, J. D. Dombi: Function decomposition using distending function In: Proceedings of IEEE International Workshop on Soft Computing Applications, Szeged-Hungary and Arad-Romania (2009)
- 2010: József Dániel Dombi, József Dombi: Create and decompose function by using fuzzy functions, In the 12th International Conference on Enterprise Information Systems, Funchal-Madeira-Portugal (2010)

Thesis point 3

- 2005: József Dombi, József Dániel Dombi: Cognitive Maps based on pliant logic In: International Journal of Simulations Systems, Science Technology Special Issue, April 2005, Volume 6, Number 6

- 2005: J. Dombi, J. D. Dombi: Pliant Cognitive Map using conjunctive operator In: Proceedings of IEEE International Workshop on Soft Computing Applications, Szeged-Hungary and Arad-Romania (2005)

References

[1] J. Dombi. Basic concepts for a theory of evaluation: the aggregative operator. *European Journal of Operations Research*, 10:282–293, 1982.

[2] J. Dombi. A general class of fuzzy operators, the demorgan class of fuzzy operators and fuzziness measures induced by fuzzy operators. *Fuzzy Sets and Systems*, 8:149–163, 1982.

[3] Jozsef Dombi. Demorgan systems with an infinitely many negations in the strict monotone operator case. *Information Sciences*, 181(8):1440 – 1453, 2011.

[4] R. R. Yager J. Fodor and A.Rybalov. Structure of uninorms. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 5(4):411–427, 1997.

[5] O. Nelles M. Fischer and R. Isermann. Knowledge-based adaption of neurofuzzy models in predictive control of a heat exchanger. *Soft Computing and Intelligent Systems*, pages 469–489, 2000.

[6] Chrysostomos D. Stylios and Petros P. Groumpos. Modeling complex systems using fuzzy cognitive maps. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 34:155–162, 2004.

[7] R.R. Yager and A. Rybalov. Uninorm aggregation operators. *Fuzzy Sets and Systems*, 80(1):111–120, 1996.