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

Contents

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.

During 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 kind of fuzzy theory that is similar to a fuzzy system [21]. 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 aggregation 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, we can create a system like the Fuzzy Cognitive Map. Second, we can apply the Pliant system by introducing function approximation techniques, which have useful and practical aspects. And third, we can apply it in problems that use decision-making techniques.

This thesis is organised as follows. In Chapter 1, we briefly review fuzzy set theory and operators that are needed to understand fuzzy logic and its applications. We describe the most important properties of the negation operator, the t-norm operator, t-conorm operator and aggregative operator. Next, we explain the connection between modalities and hedges. Here, we

will also present the most important definitions and theorems.

In Chapter 2 we present the Pliant system, which is a subset of fuzzy logic. We give a definition of the Pliant system, and then we describe the various operators of this system. We also introduce the distending function that will be used later when we make function approximations.

In the second part of the thesis we present some problems where the Pliant system may be readily applied. In Chapter 3, we use the Pliant system in decision-making situations. Here, we describe the Grid system, and we discuss the main problem in grids, which is to decide which Grid system should execute the actual job. We create several algorithms that are used for decision making, then test them in a simulation environment. We show that the algorithm that uses Pliant logic performs the best. The results of Chapter 3 were published in [29, 47].

In Chapter 4 we apply the Plaint system in the problem of function approximation. We describe the basic approximation technique that may be inappropriate in some sense. We define two different kinds of techniques based on the distending function. This algorithm has several good properties e.g. we can modify only a part of the function and elements of the algorithm has a semantic meaning. The results of Chapter 4 were published in [28, 45].

The last chapter deals with the Cognitive Map. Here, we present the Fuzzy Cognitive Map, which was first proposed by Bart Kosko. We also describe a weakness of this system, and we propose a new technique that is called the Pliant Cognitive Map. We explain how to build the Cognitive Map and we also describe a framework that was developed by us. After, we describe a real problem and evaluate the PCM for it. The results of Chapter 5 were published in [42, 44].

ACKNOWLEDGEMENTS

First of all I would like to thank my supervisor, Prof. János Csirik, for his guidance and advice, and for letting me work in an interesting and encouraging environment at the Department of Computer Algorithms and Artificial Intelligence.

I would also like to thank to my father for his advice and supporting my work by offering useful suggestions. He also never let me lose sight of my final goal.

I would also like to thank David P. Curley for scrutinising and correcting this thesis from a linguistic point of view.

I would like to thank my wife Krisztina, my daughter Henriett and my son András for their endless love, support and patience during the long road needed to complete this thesis. Last, but not least I wish to thank my parents, my grandparents, and my sisters for their constant love and support.

Chapter 1

Elements of fuzzy systems

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 fuzzy set theory were proposed by him, especially fuzzy extensions of classical basic mathematical notions like logical connectives, rules, relations and quantifiers.
During 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 hundred 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.
Before we introduce the Pliant system, we have to define basic elements of the fuzzy sets. First, we define the negation operator and its properties, followed by a definition of the t-norm operator and t-conorm operator. Next we define the aggregation operator, which is also called a uninorm in the literature. After, we will present hedges and modalities, and explain the connection between them. Finally we introduce the general form of the modifiers.

1.1 Negation operators

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)*

From C1, C2 and C3, it follows that there exists a fix point $\nu_* \in [0,1]$ of the negation where

$$n(\nu_*) = \nu_* \tag{1.1}$$

So another possible characterisation of negation is when we assign a so-called decision value ν for a given ν_0; i.e. a point (ν, ν_0) can be specified such that the curve must intersect. This tells us something about how strong the negation operator is.

$$n(\nu) = \nu_0 \tag{1.2}$$

If $n(x)$ has a fix point ν_*, we use the notation $n_{\nu_*}(x)$ and if the decision value is ν, then we use the notation $n_\nu(x)$. If $n(x)$ is employed without a suffix, then the parameter has no importance in the proofs. Later on we will characterise the negation operator in terms of the ν_*, ν_0 and ν parameters.

For the strong negation, two representation theorems are known. Trillas [74] once showed that every involutive negation operator has the following form

$$n(x) = f^{-1}(1 - f(x)), \tag{1.3}$$

where $f : [0,1] \to [0,1]$ is a continuous strictly increasing (or decreasing) function. This generator function corresponds to the nilpotent operators (nilpontent t-norms). For the strictly monotonously increasing t-norms, another form of negation operator given in [24] is

$$n(x) = f^{-1}\left(\frac{1}{f(x)}\right), \tag{1.4}$$

where $f : [0,1] \to [0,\infty]$ is a continuous, increasing (or decreasing) function and f is the generator function of the strict monotone t-norm or t-conorm.

We can express these negation operators in terms of their neutral values to get a new form of the negation operator.

For the strict monotone operators

$$n_{\nu_*}(x) = f^{-1}\left(\frac{f^2(\nu_*)}{f(x)}\right) \tag{1.5}$$

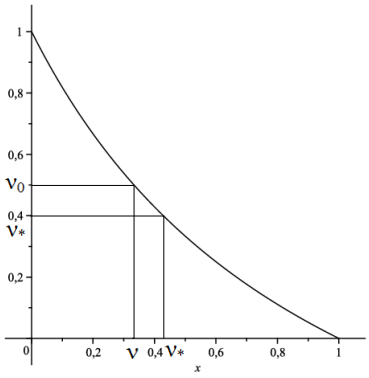

Figure 1.1: The shape of the negation function

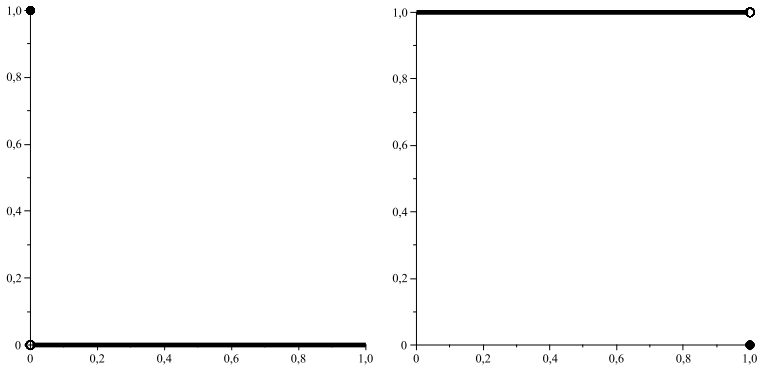

Figure 1.2: n_0 strict and n_1 non-strict negation

The other form of the negation operator in terms of ν_0 and ν (corresponding to (1.4)), is

$$n_\nu(x) = f^{-1}\left(f(\nu_0)\frac{f(\nu)}{f(x)}\right) \tag{1.6}$$

In the following we will use (1.5) and (1.6) to represent the negation operator because here we are just considering strict monotone operators.

In Figure 1.1, we explain the meaning of the ν_*, ν, ν_0 values and we sketch the shape of the negation function.

Definition 2. *If $\nu_1 < \nu_2$, then $n_{\nu_1}(x)$ is a stricter negation than $n_{\nu_2}(x)$.*

Definition 3 (Drastic negation)**.** *We call $n_1(x)$ and $n_2(x)$ a drastic negation when*

$$n_1(x) = \begin{cases} 1 & if \quad x \neq 1 \\ 0 & if \quad x = 1 \end{cases} \qquad n_0(x) = \begin{cases} 1 & if \quad x = 0 \\ 0 & if \quad x \neq 0 \end{cases}$$

$n_0(x)$ is the strictest negation, while $n_1(x)$ is the least strict negation. This is a non-continuous negation, so it is not a negation in the original sense (see Figure 1.2).

Theorem 4. *The negation operators* $n_\nu(x) = f^{-1}\left(f(\nu_0)\frac{f(\nu)}{f(x)}\right)$, $n_{\nu_*}(x) = f^{-1}\left(\frac{f^2(\nu_*)}{f(x)}\right)$ *have the following properties:*

a. They are continuous.

b. They are strictly monotonous and decreasing.

c. The correspondence principle is valid:

$$n_\nu(0)=1, \qquad n_\nu(1)=0, \qquad n_{\nu_*}(0)=1, \quad n_{\nu_*}(1)=0$$

d) The involutive property holds:

$$n_\nu(n_\nu(x))=x, \quad n_{\nu_*}(n_{\nu_*}(x))=x.$$

e) The neutral value property is valid:

$$n_\nu(\nu)=\nu_0, \quad n_{\nu_*}(\nu_*)=\nu_*.$$

Proof. It is trivial using the representation form of the negation operator.

□

In fuzzy theory, we utilise two types of negation operator. These are

$$\textbf{Yager:} \quad n_m(x)=\sqrt[m]{1-x^m} \tag{1.7}$$

$$\textbf{Hamacher, Sugeno:} \quad n_a(x)=\frac{1-x}{1+ax} \tag{1.8}$$

We can express the parameters of the negation operator in terms of its neutral values $n(\nu_*)=\nu_*$. So we have

$$\nu_*=n(\nu_*)=\sqrt[m]{1-\nu_*^m} \text{ and } m=-\frac{\ln(2)}{\ln(\nu_*)}$$

Then the Yager negation operator has the form

$$n_{\nu_*}(x)=\left(1-x^{-\frac{ln2}{ln\nu_*}}\right)^{-\frac{ln\nu_*}{ln2}} \tag{1.9}$$

In a similar way, for the Hamacher negation operator,

$$n_\nu(x)=\frac{1}{1+\frac{1-\nu_0}{\nu_0}\frac{1-\nu}{\nu}\frac{x}{1-x}}, \quad n_{\nu_*}(x)=\frac{1}{1+(\frac{1-\nu_*}{\nu_*})^2\frac{x}{1-x}} \tag{1.10}$$

This form of negation operator can be found in [22].

Definition 5. *A negation $n_{\nu_1}(x)$ is stricter than $n_{\nu_2}(x)$, if $\nu_1<\nu_2$.*

1.2 T-norm, t-conorm

1.2.1 History of Triangular Norms

Where did the name *"t-norm"* originate and when? It appeared naturally in the study of generalized triangle inequalities for statistical metric spaces - hence the name triangular norm, or simply t-norm.

The name first appeared in a paper entitled Statistical Metrics [60] that was published on 27th october in 1942. A t-norm was supposed to act on the values of two distribution functions, hence on the unit square. Here is the original definition by Menger.

Definition 6. *A real-valued function T defined on a unit square is called a t-norm in the sense of Menger if*

(a) $0 \leq T(\alpha,\beta) \leq 1$

(b) T *is non-decreasing in either variable*

(c) $T(\alpha,\beta) = T(\beta,\alpha)$

(d) $T(1,1) = 1$

(e) *If* $\alpha > 0$ *then* $T(\alpha,1) > 0$.

Note that here the notation from the original sources is used, hence it may vary slightly later on.

In 1942 a t-norm was not supposed to be associative and the boundary conditions (a), (d), (e) were rather weak. Any t-norm (and also each t-conorm) satisfied the axioms (a)-(e). But for example, each convex combination of a t-norm T (and a t-conorm S) also satisfied axioms (a)-(e).

Another original idea of Menger was the simple t-norm T, which satisfies the additional condition (f) $0 < T(\alpha,\beta) < 1$ for $0 < \alpha\beta$.

We can see some kind of strictness in this condition. An Archimedean t-norm (t-conorm) is a simple t-norm in this sense if and only if it is strict.

Perhaps just this vagueness in the definition was the origin of some critical remarks by Wald [76] to Menger‘s approach and it impeded its development for several years.

Statistical matrix spaces in the forties and fifties were based on Wald‘s version of triangle inequality (this corresponds to the convolution of two distributions). So Menger’s approach was an ”overture”. The real starting point of t-norms came in 1960, when Berthold Schweizer and Abe Sklar, (two students of Menger) published their paper, Statistical Metric Spaces [70]. However, we recall a footnote of this paper ”as probably Menger informs us, even before the

paper was written, both he and Wald in a number of conversations had come to feel that the Wald inequality was in some respect too stringent a requirement to impose on all statistical metric spaces". In this thesis, the most common t-norms and t-conorms are introduced:

T_1: $T(a,b) = \max(a+b-1,0)$

T_2: $T(a,b) = ab$

T_3: $T(a,b) = \min(a,b)$

T_4: $T(a,b) = \max(a,b)$

T_5: $T(a,b) = a+b-ab$

T_6: $T(a,b) = \min(a+b,1)$,

where the notation follows Schweizer and Sklar [70]. This list was arranged in order of increasing "strength", where T'' is said to be stronger than T' (and T' weaker than T'') if $T''(a,b) \geq T'(a,b)$ for all (a,b) in the unit square with strict inequality for at least one pair (a,b). Schweizer and Sklar, motivated by some properties of statistical metric spaces, replaced the boundary conditions (a), (d) and (e) by the condition

(a') $T(a,1) = a, \quad T(0,0) = 0$

(note that $T(0,0)$ is superfluous).

This new condition implies $T \leq T_3 = \min$. Thus, under (a'), min is the strongest T (we have not yet assumed associativity). Similarly, the weakest T satisfying (a'), (b) and (c) was introduced, henceforth denoted by T_w, here

$$T_w(x,y) = \begin{cases} a & \text{if } x = a, y = 1 \text{ or } y = a, x = 1 \\ 0 & \text{otherwise} \end{cases}$$

With conditions (a'), (b) and (c) imposed on T, Schweizer and Sklar decided to add the associativity condition

(d') $T(T(a,b),c) = T(a,T(b,c))$,

which permits the extension of a triangular inequality in statistical metric spaces to a polygonal inequality.

Since 1960 a t-norm is always understood as an associative symmetric non-decreasing function on the unit square in the unit interval that fulfills the boundary condition $T(a,1) = a$; i.e. 1 is a neutral element of T.

Not long after, Schweizer and Sklar introduced several basic notions and properties. Namely, they introduced triangular conorms (briefly, t-conorms) as a dual concept of t-norms. For a given t-norm T, its dual t-conorm S is defined by

$$S(a,b) = 1 - T(1-a, 1-b).$$

They pointed out that the boundary condition is the only difference between the t-norm and t-conorm axioms: the t-norm boundary condition (a') is transformed to

(a'') $\quad S(1,1)=1, \quad S(a,0)=a$

(recall once again that $S(1,1)$ is superfluous).

Note that using axioms (a''), (b) and (d'), the definition of a t-conorm S does not depend on the notion of a t-norm. Further, the necessity of characterizing t-norms (and hence t-conorms) led Schweizer and Sklar to the study of associative functions (recall their paper entitled Associative functions and statistical triangle inequalities). So, in some sense, they went back to Abel [4] who showed that, under some natural conditions, the construction of a two-place function from a one place function always leads to an associative function.

Using the results of Abel, Aczél (1949) and other authors, they introduced the class of strict t-norms, which, in addition to (a'), (b), (c) and (d'), has the following constraints:

- T is continuous (on $[0,1]\times[0,1]$)
- $T(a,b)<T(c,b)$ whenever $0<a<c\leq 1$ and $0<b\leq 1$
- $T(a,b)<T(a,d)$ whenever $0<a\leq 1$ and $0<b<d\leq 1$ (strict monotonicity).

Based on these three conditions, a strict t-conorm was introduced as a dual to a strict t-norm. The order reversing property of the duality was respected; i.e. $\max\leq S\leq S_W$ for any t-conorm S, where max and S_W are the duals of min and T_W, respectively.

Later, the first results characterizing t-norms (t-conorms) were presented. Namely, the characterization of a strict t-norm T (t-conorm S) through an additive generator.

The last substantial step in the foundation of t-norms and t-conorms was given in 1965 by Ling [56]. Among other things, she recognised that continuous t-norms and t-conorms form a topological semigroup on $[0,1]$. She preserved the semigroup theory notation and hence she introduced Archimedean and nilpotent t-norms (and t-conorms).

Recall that a continuous t-norm (t-conorm) is called Archimedean if it fulfils (e'), (e'')

(e') $\quad T(a,a)<a$ for any $a\in(0,1)$

(e'') $\quad S(a,a)>a$ for any $a\in(0,1)$.

A continuous non-strict Archimedean t-norm (t-conorm) is called nilpotent. Note that a continuous Archimedean t-norm (t-conorm) is nilpotent if there is an $a\in(0,1)$ such that $T(a,a)=0$ ($S(a,a)=1$).

Ling gave a complete characterization of continuous t-norms and t-conorms based on the results of Aczél, Schweizer and Sklar, Mosert and Shields and Faucett.

Years later some complementary remarks were added. Also, in 1979 Frank [35] solved the functional equation.

$$T(a,b)+S(a,b)=a+b$$

for continuous t-norms T and t-conorms S. Based on earlier results of Climescu (1946), he introduced the concept of ordinal sums, which play a key role in the investigation of continuous t-norms and t-conorms (and of pseudo-additions as further generalizations).

Now both triangular norms and conorms have become important tools in different contexts. They play a fundamental role in probabilistic metric spaces, multiple-valued logic and especially in fuzzy set theory. In the latter area they are used to generate the fuzzy connectives of the union and intersection of fuzzy sets.

The use of general t-norms and t-conorms for modelling the intersection and union of fuzzy sets most likely goes back to a suggestion by Ulrich Hõhle during the First International Seminar on Fuzzy Set Theory held in Linz (Austria) in 1979. The reason for this was the fact that monotonicity, commutativity, associativity and the boundary conditions were generally treated as indispensable properties of meaningful extensions of the logical ”and” and ”or” in (two-valued) Boolean logic.

1.2.2 Triangular Norms

In order to formulate the triangle inequality property in a *probabilistic metric space,* and following the ideas of K. Menger [60], B. Schweizer and A. Sklar [69] introduced a special class of two-place functions on the unit square, the so-called triangular norms. Together with their duals, the triangular conorms, they have been applied in various mathematical disciplines, such as *probabilistic metric spaces* [71], *fuzzy set theory, multiple-valued logic,* and in the theory of *non-additive measures* [65].

Definition 7. *A triangular norm (t-norm for short) is a function* $T:[0,1]^2\to[0,1]$ *such that for all* $x,y,z\in[0,1]$*, the following four axioms are satisfied:*

(T1)	*Symmetry*	$T(x,y)=T(y,x)$
(T2)	*Associativity*	$T(x,T(y,z))=T(T(x,y),z)$
(T3)	*Monotonicity*	$T(x,y)\le T(x,z)$ *whenever* $y\le z$
(T4)	*Boundary condition*	$T(x,1)=x$

Alternatively, a t-norm has an algebraic definition:

Definition 8. *A t-norm is a commutative lattice ordered semigroup on the unit interval* $[0,1]$*, with unit 1.*

Clearly, the two definitions above are equivalent. (T1) is the commutativity, (T2) means "semigroup", (T3) is the expression "lattice ordered on the unit interval" and (T4) means "with unit 1".

There exist uncountably many t-norms.

Example 9. *The four basic t-norms are:*

(i) *The minimum is given by*

$$T_M(x,y) = \min(x,y)$$

(ii) *The product is given by*

$$T_P(x,y) = xy \tag{1.11}$$

(iii) *The Lukasiewicz is given by*

$$T_L(x,y) = \max(x+y-1,0) \tag{1.12}$$

(iv) *The Weakest t-norm (drastic product) is given by*

$$T_D(x,y) = \begin{cases} \min(x,y) & \text{if } \max(x,y) = 1 \\ 0 & \text{otherwise} \end{cases}$$

Axioms (T1)-(T4) are independent of each other, as can be seen from the following examples of operations on [0,1], where exactly one of the axioms fails to hold:

Example 10. *Consider the following functions:*

(i) *The function* $F : [0,1]^2 \to [0,1]$ *given by*

$$F(x,y) = \begin{cases} 0 & \text{if } (x,y) \in [0,0.5] \times [0,1) \\ \min(x,y) & \text{otherwise} \end{cases}$$

satisfies (T2), (T3) and (T4), but not (T1).

(ii) *The function* $F : [0,1]^2 \to [0,1]$ *given by*

$$F(x,y) = xy\max(x,y)$$

satisfies (T1), (T3) and (T4), but not (T2).

(iii) *The function* $F : [0,1]^2 \to [0,1]$ *given by*

$$F(x,y) = \begin{cases} 0.5 & \text{if } (x,y) \in [0,1]^2 \\ \min(x,y) & \text{otherwise} \end{cases}$$

satisfies (T1), (T2) and (T4), but not (T3).

(iv) *Let* $k \in (0,1)$. *The function* $F : [0,1]^2 \to [0,1]$ *given by*

$$F(x,y) = kxy$$

satisfies (T1), (T2) and (T3), but not (T4).

Remark 11. (i) *From (T1), (T3) and (T4) it is readily seen that for all* $x \in [0,1]$ *each t-norm satisfies the following additional boundary conditions:*

$$T(0,x) = T(x,0) = 0.$$

and

$$T(1,x) = x.$$

Whence, all t-norms coincide on the boundary of the unit square $[0,1]^2$.

(ii) *(T3) and (T1) together imply the following (joint) monotonicity in both components, i.e.,*

$$T(x_1,y_1) \leq T(x_2,y_2) \qquad \textit{whenever} \qquad x_1 \leq x_2 \textit{ and } y_1 \leq y_2.$$

Since t-norms can be regarded as functions mapped from the unit square into the unit interval, a comparison of t-norms is made in the usual way, i.e., point-wise.

Definition 12. *If for two t-norms* T_1 *and* T_2 *the inequality* $T_1(x,y) \leq T_2(x,y)$ *holds for all* $(x,y) \in [0,1]^2$ *then* T_1 *is said to be weaker than* T_2*, and we write in this case* $T_1 \leq T_2$*. We write* $T_1 < T_2$ *whenever* $T_1 \leq T_2$ *and* $T_1 \neq T_2$*.*

Remark 13. *It is not hard to see that* T_D *is the weakest t-norm and* T_M *is the strongest t-norm; that is, for all t-norm* T

$$T_D \leq T \leq T_M.$$

We get the following ordering of the four basic t-norms:

$$T_D < T_L < T_P < T_M.$$

Definition 14. *If* φ *is an automorphism, namely an increasing bijection of the closed unit interval, then the following formula defines the so-called* φ*-transform of* T *(which is also a t-norm):*

$$T_\varphi(x,y) = \varphi^{-1}(T(\varphi(x),\varphi(y))), \qquad x,y \in [0,1].$$

This is clearly an order-isomorphism from an algebraic point of view.

In Definition 7, t-norms were introduced as binary operators. Since they are associative, they can also be viewed as operations with more than two arguments.

Remark 15. *The associativity (T2) property allows one to extend each t-norm* T *to an* n*-ary operation for all* $n \in \mathbb{N}$ *in the usual way by induction, defining on the* n*-tuple* $(x_1,x_2,\ldots,x_n) \in [0,1]^n$

$$T(x_1,x_2,\ldots,x_n) = T(T(x_1,x_2,\ldots,x_{n-1}),x_n).$$

The fact that each t-norm T is weaker than the minimum operator makes it possible to extend it to a (countably) infinitary operation, putting for each $(x_i)_{i\in IN} \in [0,1]^{IN}$:

$$T(x_1,x_2,\ldots,x_k,\ldots) = \lim_{n\to\infty} T(x_1,x_2,\ldots,x_n).$$

The sequence on the right-hand side is clearly non-increasing and bounded from below.

Moreover, the definition can be extended to an arbitrary (not necessarily countable) index set I and $(x_i)_{i\in I}$ as the infimum of all the $T(x_1,x_2,\ldots,x_k,\ldots)$'s, where $x_1,x_2,\ldots,x_k,\ldots$ is a subsequence of $(x_i)_{i\in I}$.

1.2.3 Triangular Conorms

In Schweizer and Sklar's papers [70, 69], triangular conorms were introduced as dual operations of t-norms. Here is the axiomatic definition.

Definition 16. *A triangular conorm (t-conorm for short) is a function $T : [0,1]^2 \to [0,1]$ such that for all $x,y,z \in [0,1]$, the following four axioms are satisfied:*

(S1)	*Symmetry*	$S(x,y) = S(y,x)$
(S2)	*Associativity*	$S(x,S(y,z)) = S(S(x,y),z)$
(S3)	*Monotonicity*	$S(x,y) \leq S(x,z)$ *whenever* $y \leq z$
(S4)	*Boundary condition*	$S(x,0) = x$

Alternatively, a t-conorm has an algebraic meaning:

Definition 17. *A t-conorm is a commutative lattice ordered semigroup on the unit interval $[0,1]$, with unit 0.*

Clearly, the two definitions above are equivalent. (S1) is the commutativity, (S2) means "semigroup", (S3) is the expression "lattice ordered on the unit interval" and (S4) means "with unit 0".

One can see that the axioms of commutativity, associativity and monotonicity are exactly the same as in the case of t-norms. That means that, from an axiomatic point of view, t-norms and t-conorms differ only with respect to the boundary conditions. In fact, the concept of t-norms and t-conorms are dual in some sense.

First, we will give the most important examples.

Example 18. *The four basic t-conorm:*

(i) *Maximum given by*

$$S_M(x,y) = \max(x,y)$$

(ii) *Probabilistic sum given by*

$$S_P(x,y) = x+y-xy$$

(iii) *Lukasiewicz given by*

$$S_L(x,y) = \min(x+y,1)$$

(iv) *Strongest t-conorm given by*

$$S_D(x,y) = \begin{cases} \max(x,y) & \text{if } \max(x,y) = 1 \\ 1 & \text{otherwise} \end{cases}$$

The original definition of t-conorms (Schweizer and Sklar [70] and [69]) is completely equivalent to the axiomatic definition given above. Thus

Proposition 19. *A function $S : [0,1]^2 \to [0,1]$ is a t-conorm if and only if there exists a t-norm T such that for all $(x,y) \in [0,1]^2$*

$$S(x,y) = 1 - T(1-x,1-y) \tag{1.13}$$

Proof. If T is a t-norm, then the operation defined by (1.13) satisfies (S1)-(S4). But if S is a t-conorm, then we can define a function $T : [0,1]^2 \to [0,1]$ by

$$T(x,y) = 1 - S(1-x,1-y).$$

□

It is trivial to check that T is a t-norm and (1.13) holds. This duality allows us to translate many properties of t-conorms. Also, every theorem about t-norms readily holds for t-conorms.

Remark 20. *(T_M,S_M), (T_P,S_P), (T_L,S_L) and (T_D,S_D) are mutually dual to each other.*

1.2.4 Continuous t-norms

Definition 21. *A t-norm is said to be continuous if it is continuous as a two-place function.*

Definition 22. *A continuous t-norm T is called Archimedean if $T(x,x) < x$ is true for all $x \in (0,1)$.*

Definition 23. *A t-norm T has 0-divisors if $T(x,y) = 0$ for some $x,y \in (0,1)$.*

Definition 24. *A t-norm T is strictly increasing if $T(x,y) > T(x,z)$ whenever $x,y,z \in (0,1)$ and $y > z$.*

Definition 25. *A continuous Archimedean t-norm with 0-divisors is called nilpotent. An example is the Lukasiewicz t-norm defined in (1.12).*

Definition 26. *A continuous Archimedean t-norm which is strictly increasing is called strict. An example is the product t-norm defined in (1.11)*

The origin of the following theorem goes back to Aczél [5]. See also Abel [4], Mostert and Shields [62], Miranda [19] and Faucett [33]. The second theorem is due to Mostert and Shields [62] and Miranda [19]. The present form of the theorems is:

Theorem 27. *A t-norm T is strict if and only if T is a* φ*-transformation of the product t-norm.*

Theorem 28. *A t-norm T is nilpotent if and only if T is a* φ*-transformation of the Lukasiewicz t-norm.*

The following representation theorem of continuous Archimedean t-norms in its present form is due to Ling [56]. See Abel [4], Mostert and Shields [62], Miranda [19], Aczél [5] and Faucett [33] as well.

Theorem 29. *A t-norm is T continuous and Archimedean if and only if there exists a strictly decreasing and continuous function* $f : [0,1] \to [0,\infty]$ *with* $f(1) = 0$ *such that*

$$T(x,y) = f^{(-1)}(f(x) + f(y)),$$

where $f^{(-1)}$ *is the pseudoinverse of* f *defined by*

$$f^{(-1)}(x) = \begin{cases} f^{-1}(x) & \text{if } x \leq f(0) \\ 0 & \text{otherwise} \end{cases}$$

- $f(0) = \infty$ *if and only if* T *is strict.*
- $f(0)$ *is finite if and only if* T *is nilpotent.*

Moreover, this representation is unique up to a positive multiplicative constant.

Definition 30. *If a t-norm T has the above representation, then the function f is called an additive generator of T.*

The following method of constructing a new t-norm from a family of given t-norms is based on the results of Climescu [16], Clifford [14], Clifford-Preston [15] concerning ordinal sums of semigroups (see also Ling [56], Frank [35]). Here, we state the form of the theorem which is applied to t-norms.

Theorem 31. *Suppose that* $\{[a_i,b_i]\}_{i\in K}$ *is a countable family of non-over-lapping, closed, proper subintervals of* $[0,1]$*, denoted by* τ*. With each* $[a_i,b_i]\in\tau$ *associate a t-norm* T_i*. Let* T *be a function defined on* $[0,1]^2$ *by*

$$T(x,y)=\begin{cases} a_i+(b_i-a_i)T_i\left(\frac{x-a_i}{b_i-a_i},\frac{y-a_i}{b_i-a_i}\right) & \text{if } (x,y)\in[a_i,b_i]^2 \\ \min(x,y) & \text{otherwise}\end{cases} \tag{1.14}$$

In this case T *is denoted by* $<\{([a_i,b_i],T_i)\}>_{i\in K}$ *and called the ordinal sum of* $\{([a_i,b_i],T_i)\}_{i\in K}$*, and each* T_i *is called a summand. Then* T *is a t-norm.*

Now we will present a characterization of continuous t-norms. The original result of Mostert and Shields [62] corresponded to I-semigroups on $[0,1]$ with 0 as zero and 1 as identity. In the present form, this theorem first appeared in Ling [56]. She used a purely analytical proof.

Theorem 32. *Suppose* T *is a continuous t-norm. Then either* T *is continuous Archimedean or* $T=\min$ *or there exists a family* $\{([a_i,b_i],T_i)\}_{i\in K}$ *with continuous Archimedean t-norms* T_i *such that* T *is the ordinal sum of this family.*

1.3 Aggregative operator

The term *uninorm* was first introduced by Yager and Rybalov [80]. Uninorms are a generalization of t-norms and t-conorms, get by relaxing the constraint on the identity element in the unit interval $\{0,1\}$. Since then many articles have focused on uninorms, both from a theoretical [54, 55, 39, 59, 73, 61] and a practical point of view [79]. The paper of Fodor, Yager and Rybalov [43] is notable since it defined a new subclass of uninorms called representable uninorms. This characterization is similar to the representation theorem of strict t-norms and t-conorms, in the sense that both originate from the solution of the associativity functional equation given by Aczél [6].

The aggregative operators were first introduced in [20] by selecting a set of minimal concepts that must be fulfilled by an evaluation-like operator.

Actually, as mentioned in [43], there is a close relationship between Dombi's aggregative operators and uninorms. In fact, they form a subclass of uninorms.

In 1982, Dombi [20] defined the aggregative operator in the following way:

Definition 33. *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 definition of uninorms, originally given by Yager and Rybalov [80], is the following:

Definition 34. *A uninorm* U *is a mapping* $U : [0,1]^2 \to [0,1]$ *having the following properties:*

- $U(x,y) = U(y,x)$ *(commutativity)*
- $U(x_1,y_1) \geq U(x_2,y_2)$ *if* $x_1 \geq x_2$ *and* $y_1 \geq y_2$ *(monotonicity)*
- $U(x,U(y,z)) = U(U(x,y),z)$ *(associativity)*
- $\exists \nu_* \in [0,1]\ \forall x \in [0,1]\ U(x,\nu_*) = x$ *(neutral element)*

The following representation theorem of strict, continuous on $[0,1] \times [0,1] \setminus (\{0,1\},\{1,0\})$ uninorms (or *representable uninorms*) was given by Fodor et al. [43] (see also Klement et al. [30]).

Theorem 35. *Let* $U : [0,1] \to [0,1]$ *be a function and* $\nu_* \in]0,1[$. *The following are equivalent:*

1. U *is a uninorm with neutral element* ν_* *which is strictly monotone on* $]0,1[^2$ *and continuous on* $[0,1]^2 \setminus \{(0,1),(1,0)\}$.

2. *There exists a strictly increasing bijection* $g_u : [0,1] \to [-\infty,\infty]$ *with* $g_u(\nu_*) = 0$ *such that for all* $(x,y) \in [0,1]^2$, *we have*

$$U(x,y) = g_u^{-1}\left(g_u(x) + g_u(y)\right), \tag{1.15}$$

   where, in the case of a conjunctive uninorm U, *we use the convention* $\infty + (-\infty) = -\infty$, *while, in the disjunctive case, we use* $\infty + (-\infty) = \infty$ *or there exists a strictly increasing*

continuous function $f_a : [0,1] \to [0,\infty]$ *with* $f_a(0) = 0$, $f(\nu) = 1$ *and* $f_a(1) = \infty$. *The binary operator is defined by*

$$U(x,y) = f_a^{-1}(f_a(x)\, f_a(y)) \tag{1.16}$$

for all $(x,y) \in [0,1] \times [0,1]/(0,1),(1,0)$ *and either* $a(0,1) = a(1,0) = 0$
or $a(0,1) = a(1,0) = 1$.

If Eq.(1.15) holds, the function g_u *is uniquely determined by* U *up to a positive multiplicative constant, and it is called an additive generator of the uninorm* U. *Here,* f_a *is called the multiplicative generator function of the operator.*

Such uninorms are called representable uninorms and they were previously introduced as aggregative operators [20].

Definition 36. *A representable uninorm is called an aggregative operator. We will denote it by* $a(x,y)$.

Theorem 1 (Dombi [20])**.** *Let* $a : [0,1]^n \to [0,1]$ *be a function and let a be an aggregative n-valued operator with additive generator g. The neutral value of the aggregative operator is* ν_* *if and only if* $\vec{x} \in [0,1]^n, \forall x$. *It has the following form:*

$$a_{\nu_*}(\boldsymbol{x}) = g^{-1}\left(g(\nu_*) + \sum_{i=1}^{n}(g(x_i) - g(\nu_*))\right).$$

We will use the transformation defined $g(x) = ln(f(x))$ to get the multiplicative operator form

$$a_{\nu_*}(x) = f_a^{-1}\left(f_a(\nu_*)\prod_{i=1}^{n}\frac{f_a(x_i)}{f_a(\nu_*)}\right) = f_a^{-1}\left(f_a^{1-n}(\nu_*)\prod_{i=1}^{n}f_a(x_i)\right), \tag{1.17}$$

where $f_a : [0,1] \to [0,\infty]$. In the following, we will use the multiplication form of the aggregative operator.

1.4 Strict t-norms, t-conorms and aggregative operators

Let

$$c(x,y) = f_c^{-1}(f_c(x) + f_c(y)) \qquad d(x,y) = f_d^{-1}(f_d(x) + f_d(y)),$$

where f_c and f_d are the generator functions of the operators. The shape of these functions can be seen in Figure 1.3.

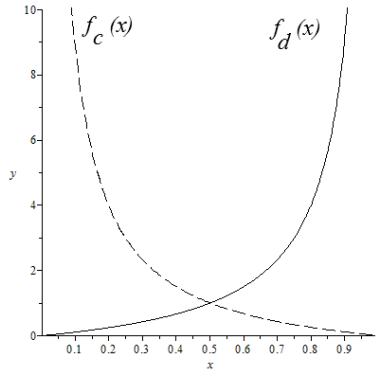

Figure 1.3: The generator function of the conjunctive and disjunctive operators (additive representation)

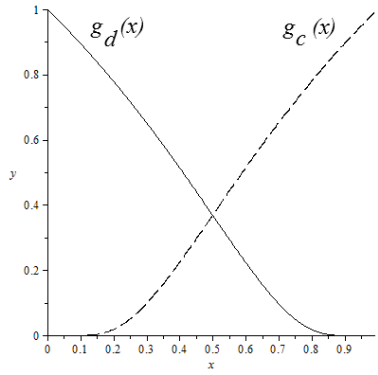

Figure 1.4: The generator function of the conjunctive and disjunctive operators (multiplicative representation)

Let

$$g_c(x) = e^{-f_c(x)} \qquad g_d(x) = e^{-f_d(x)}$$

Then

$$f_c(x) = -\ln(g_c(x)) \qquad f_d(x) = -\ln(g_d(x)).$$

So

$$c(x,y) = f_c^{-1}\left(-\ln(g_c(x)) - \ln(g_c(y))\right) = g_c^{-1}\left(e^{-(-\ln(g_c(x))-\ln(g_c(y)))}\right)$$

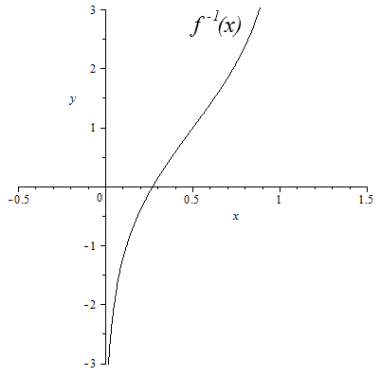

Figure 1.5: The generator function of the aggregative operator in the additive representation case

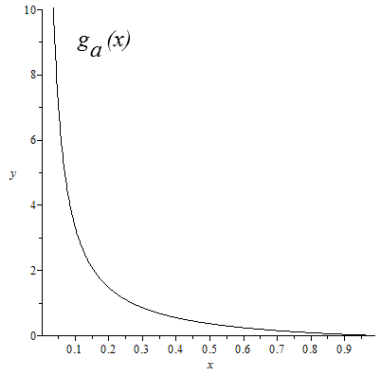

Figure 1.6: The generator function of the aggregative operator in the multiplicative representation case

1.4.1 General form of the aggregative operator

We will use the transformation defined in (1.4) to get the multiplicative operator

$$a_{\nu_*}(x) = f_a^{-1}\left(f_a(\nu_*)\prod_{i=1}^{n}\frac{f_a(x_i)}{f_a(\nu_*)}\right) = f_a^{-1}\left(f_a^{1-n}(\nu_*)\prod_{i=1}^{n}f_a(x_i)\right), \tag{1.18}$$

where $f_a : [0,1] \to [0,\infty]$. In the following, we will use the multiplication form of the aggregative operator.

From an application point of view, the strict monotonously increasing operators are quite useful. They actually have a variety of applications. This is the reason why we will focus on strictly monotonously increasing operators.

1.4.2 Aggregative operator and the self-DeMorgan identity

1.4.3 Weighted aggregative operator

The general form of the weighted operator in the additive representation case is

$$a(\mathbf{w},\mathbf{x}) = g^{-1}\left(\sum_{i=1}^{n} w_i g(x_i)\right). \tag{1.19}$$

We will derive the weighted aggregative operators when ν_* is given.

First, we use the construction

$$g_1(x) = g_a(x) - g_a(\nu_*) \qquad g_1^{-1}(x) = g_a(x + g(\nu_*)),$$

where $\nu_* \in (0,1)$.

$$\begin{aligned} a_{\nu_*}(\mathbf{w},\mathbf{x}) &= g_1^{-1}\left(\sum_{i=1}^{n} w_i g_1(x_i)\right) = \\ &= g_a^{-1}\left(\sum_{i=1}^{n} w_i (g_a(x_i) - g_a(\nu_*)) + g_a(\nu_*)\right) \\ &= g_a^{-1}\left(\sum_{i=1}^{n} w_i g_a(x_i) + g(\nu_*)\left(1 - \sum_{i=1}^{n} w_i\right)\right) \end{aligned} \tag{1.20}$$

1.4.4 General form of the weighted aggregative operator

The multiplicative form of the aggregative operator is

$$a_{\nu_*}(\mathbf{w},\mathbf{x}) = f_a^{-1}\left(f_a(\nu_*)\prod_{i=1}^{n}\left(\frac{f_a(x)}{f_a(\nu_*)}\right)^{w_i}\right) = f_a^{-1}\left(f_a^{1-\sum_{i=1}^{n} w_i}(\nu_*)\prod_{i=1}^{n} f_a^{w_i}(x_i)\right) \tag{1.21}$$

From (1.4.3) if $\sum_{i=1}^{n} w_i = 1$, then $a_{\nu_*}(\mathbf{w},\mathbf{x})$ is independent of ν_* and

$$a(\mathbf{w},\mathbf{x}) = f_a^{-1}\left(\prod_{i=1}^{n} f_a^{w_i}(x_i)\right). \tag{1.22}$$

In the Dombi operator case,

$$a_{\nu_*}(\mathbf{w},\mathbf{x}) = \frac{1}{1+\frac{1-\nu_*}{\nu_*}\prod_{i=1}^{n}\left(\frac{1-x_i}{x_i}\frac{\nu_*}{1-\nu_*}\right)^{w_i}} \tag{1.23}$$

$$a_{\nu_*}(\mathbf{w},\mathbf{x}) = \frac{\nu_*(1-\nu_*)^{\sum_{i=1}^{n} w_i}\prod_{i=1}^{n} x_i^{w_i}}{\nu_*(1-\nu_*)^{\sum_{i=1}^{n} w_i}\prod_{i=1}^{n} x_i^{w_i} + (1-\nu_*)\nu_*^{\prod_{i=1}^{n} w_i}\prod_{i=1}^{n}(1-x_i)^{w_i}} \tag{1.24}$$

If $w_i = 1$, then

$$a(\mathbf{w},\mathbf{x}) = \frac{\prod_{i=1}^{n} x_i^{w_i}}{\prod_{i=1}^{n} x_i^{w_i} + \prod_{i=1}^{n}(1-x_i)^{w_i}}. \tag{1.25}$$

$$a_{\nu_*}(\mathbf{x}) = \frac{(1-\nu_*)^{n-1}\prod_{i=1}^{n} x_i}{(1-\nu_*)^{n-1}\prod_{i=1}^{n} x_i + \nu_*^{n-1}\prod_{i=1}^{n}(1-x_i)} \tag{1.26}$$

If $\nu_* = \frac{1}{2}$, then we get

$$a_{\frac{1}{2}}(\mathbf{x}) = \frac{\prod_{i=1}^{n} x_i}{\prod_{i=1}^{n} x_i + \prod_{i=1}^{n}(1-x_i)}. \tag{1.27}$$

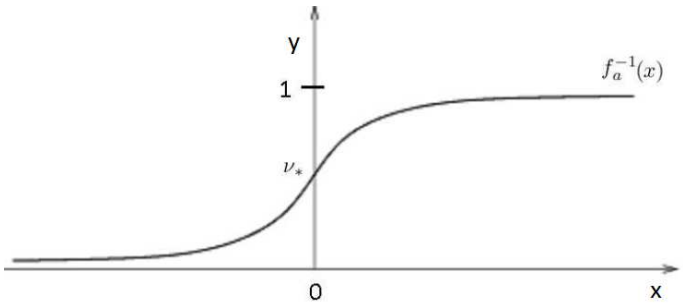

Figure 1.7: ν_* is the neutral element of the aggregative operator

Eq. (1.27) is called the 3 Π operator because it contains of three product operators. This operator was first introduced by Dombi [20].

1.5 Modalities and hedges

In logic theory, modal operators have a variety of applications and even from a theoretical perspective they are interesting to study. Here, we will present different approaches for obtaining

the form of the necessity and possibility operators. These have a simple parametrical form. By changing the parameter value, we get different modalities.

Modal logic has been used in rough sets, where the sets are approximated by elements of a partition induced by an equivalence relation. A natural choice for rough set logic is S5 (Orlowska [63, 64]). Here, the possibility and necessity modalities express outer and inner approximation operators.

To obtain this structure, we equip it with another type of negation operator. In modal logic, it is called an intuitionistic negation operator. In our system, the modalities induced by a suitable composition of the two negation operators generate a modal system with the full distributivity property of the modal operators. The necessity operator is simultaneously distributive over the conjunctive and disjunctive operator and the possibility operator is also simultaneously distributive over the conjunctive and disjunctive operators.

Fuzzy logic is a kind of many-valued logic. If we want to introduce hedges into the theory, we need to provide a proper logical structure. Zadeh introduced the modifier function of fuzzy sets, which plays an important role because fuzzy concepts are related to natural language expressions. The popularity of the fuzzy concept is due to the cognitive aspects of the parameters in the mathematical expressions.

In our system we will use the negation operator with the following properties:

$$\begin{aligned} &dM_1 \qquad n(n(x_1)) = x \\ &dM_2 \qquad n(c(x,y)) = d(n(x), n(y)) \end{aligned} \tag{1.28}$$

The greatest element 1 is interpreted as true, while the negation of false is 0 and 1 is $n(0)$.

$\tau_N[0,1] \to [0,1]$ is unary operator that satisfies the following conditions for the necessity operator.

$$N1. \quad \tau_N(1) = 1 \qquad (N\ principle) \tag{1.29}$$

$$N2. \quad \tau_N(x) \leq x \qquad (T\ principle) \tag{1.30}$$

$$N3. \quad x \leq y \quad implies \quad \tau_N(x) \leq \tau_N(y) \qquad (K\ principle) \tag{1.31}$$

$$N4. \quad \tau_P(x) = n(\tau_N(n(x))) \qquad (DF\diamondsuit\ principle) \tag{1.32}$$

$$[N5. \quad \tau_P(x) = \tau_N(\tau_P(x)) \qquad (\tau_N\ principle)] \tag{1.33}$$

In our system, $N5$ is not required. Only a special parametrical form of τ_P and τ_N satisfies $N5$.

Instead of $N5$, we will demand that the so-called neutrality principle, i.e.

$$N^{'}(5) \qquad \tau_N(\tau_P(x)) = x \qquad (Nprinciple) \tag{1.34}$$

Usually in a modal system the distributivity law is also valid:

$$MD_c(N) \quad c(\tau_N(x), \tau_N(y)) = \tau_N(c(x,y)) \tag{1.35}$$

In a paper by Cattaeno et al, $MD_c(N)$ the distributivity property is not needed for the conjunctive operators, but it is for the disjunctive operators.

$$MD_c(N) \quad d(\tau_N(x), \tau_N(y)) = \tau_N(d(x,y)) \tag{1.36}$$

The consequence of this unusual requirement produces a non-trivial structure [36].

In our system, we will provide the necessary and sufficient conditions for when $MD_c(N)$ and $MD_d(N)$ both hold.

$\tau_P[0,1] \rightarrow [0,1]$ is unary operator that satisfies the following conditions for the possibility operator.

$$P1. \quad \tau_P(0) = 0 \qquad (P\ principle) \tag{1.37}$$

$$P2. \quad x \leq \tau_P(x) \qquad (T\ principle) \tag{1.38}$$

$$P3. \quad x \leq y \quad implies \quad \tau_P(x) \leq \tau_P(y) \qquad (K\ principle) \tag{1.39}$$

$$P4. \quad \tau_N(x) = n(\tau_P(n(x))) \qquad (DF\Box\ principle) \tag{1.40}$$

$$[P5. \quad \tau_N(x) = \tau_P(\tau_N(x)) \qquad (\tau\ principle)] \tag{1.41}$$

In our system, $P5$ is not required. Only a special parametrical form of τ_N and τ_P satisfies $P5$.

Instead of $P5$, we will demand that the so-called neutrality principle hold i.e.

$$P^{'}(5) \qquad \tau_P(\tau_N(x)) = x \qquad (Nprinciple) \tag{1.42}$$

Usually in a modal system the distributivity law is also valid:

$$MD_d(P) \quad d(\tau_P(x), \tau_P(y)) = \tau_P(d(x,y)) \tag{1.43}$$

Similar to the necessity operator, in a paper by Cattaeno et al, $MD_d(P)$ the distributivity property is not needed for the disjunctive operators, but it is for the conjunctive operators.

$$MD_d(P) \quad c(\tau_P(x), \tau_P(y)) = \tau_P(c(x,y)) \tag{1.44}$$

Similar to the necessity operator, in our system we will provide the necessary and sufficient conditions for when $MD_d(P)$ and $MD_c(P)$ both hold.

The linguistic hedge “very” always expresses a tight interval, whereas ”more or less” expresses a looser interval (less tight). In this sense, "very" corresponds to the necessity operator and "‘more or less"’ the possibility operator.
With this starting point, the necessity and possibility operators used in fuzzy logic are based on an extension of modal logic to the continuous case. We begin with the negation operator and we make use of two types of this operator; one that is strict, and one that is less strict. We will show that with these two negation operators we can define the modal hedges.

Modal logic, which is an area of mathematical logic, can be viewed as a logical system obtained by adding logical symbols and inference rules.

This issue is also related in part to linguistic hedges and the corresponding reverse effects ("very", "more or less"), and to the modal operators with mutually reverse modal concepts as well, i.e. one can define the necessity hedge by $\Box x$ and the possibility hedge by $\Diamond x$, which have a mutually reverse effect on x.
We will construct linguistic modal hedges called necessity and possibility hedges. The construction is based on the fact that modal operators can be realized by combining two kinds of negation operators.

In intuitionistic logic, another kind of negation operator also has to be taken into account. Here, $\sim_x$ means the negated value of x. $\sim_1 x$ and $\sim_2 x$ are two negation operator.

In modal logic, $\sim_1 x$ means "x" is impossible. In other words, $\sim_1$ a stronger negation than not "x", i.e. $\sim_2 x$. Because $\sim_1 x$ in modal logic, it means "x is impossible".
We can write

$$impossible\, x = necessity(not\, x) \tag{1.45}$$

$$\begin{aligned} \sim_1 x &:= \textit{impossible}\, x \\ \sim_2 x &:= \textit{not}\, x \end{aligned} \tag{1.46}$$

$$\sim_1 x = \Box \sim_2 x \tag{1.47}$$

As we mentioned above, in modal logic we have two more operators than the classical logic case, namely necessity and possibility; and in modal logic there are two basic identities. These are

$$\sim_1 x = \textit{impossible}(x) = \textit{necessity}(\textit{not}(x)) = \Box \sim_2 x \tag{1.48}$$

$$\Diamond x = \textit{possible}(x) = \textit{not}(\textit{impossible}(x)) = \sim_2 (\sim_1 x) \tag{1.49}$$

If in Eq.(1.48) we replace x by $\sim_2 x$ and using the fact that $\sim_2 x$ is involutive, we get

$$\Box x = \sim_1 (\sim_2 x), \tag{1.50}$$

and with Eq.(1.49), we have

$$\Diamond x = \sim_2 (\sim_1 x). \tag{1.51}$$

Definition 37. *The general form of the modalities is*

$$\tau_{\nu_1,\nu_2}(x) = n_{\nu_1}(n_{\nu_2}(x)), \tag{1.52}$$

where ν_1 *and* ν_2 *are neutral values. If* $\nu_1 < \nu_2$*, then* $\tau_{\nu_1,\nu_2}(x)$ *is a necessity operator and if* $\nu_2 < \nu_1$*, then* $\tau_{\nu_1,\nu_2}(x)$ *is a possibility operator.*

From the above definition, we get

$$\tau_{\nu_1,\nu_2}(x) = f^{-1}\left(f(\nu_1)\frac{f(x)}{f(\nu_2)}\right), \tag{1.53}$$

This can be rewritten as

$$\tau_{\nu,\nu_0}(x) = f^{-1}\left(f(\nu_0)\frac{f(x)}{f(\nu)}\right). \tag{1.54}$$

Definition 38. *We call graded modalities a k composition of the modalities.*

$$\tau_{\Box}(\tau_{\Box}(\ldots \quad \tau_{\Box}(x))) = \Box(\underbrace{\Box(\ldots \quad \Box}_{K}(x))\ldots) = \Box^{K}(x) \tag{1.55}$$

$$\tau_{\Diamond}(\tau_{\Diamond}(\ldots \quad \tau_{\Box}(x))) = \Diamond(\underbrace{\Diamond(\ldots \quad \Diamond}_{K}(x))\ldots) = \Diamond^{K}(x) \tag{1.56}$$

In fuzzy theory, we use two types of negation operator.

By making use of (1.9) and (1.10) we can define the concrete forms of the necessity and possibility operators.

$$\tau(x) = n_{\nu_1}(n_{\nu_2}(x)) = \left(1 - \left(1 - x^{-\frac{\ln 2}{ln\nu_2}}\right)^{\frac{ln\nu_2}{ln\nu_1}}\right)^{-\frac{\ln \nu_1}{ln2}}$$

$$\tau_{\nu_*}(x) = n_{\nu_1}(n_{\nu_2}(x)) = \frac{1}{1 + \left(\frac{1-\nu_*}{\nu_*}\right)^2 \frac{1-x}{x}} \quad \text{or} \quad \tau_{\nu_*}(x) = \frac{1}{1 + \left(\frac{\nu_*}{1-\nu_*}\right)^2 \frac{1-x}{x}}$$

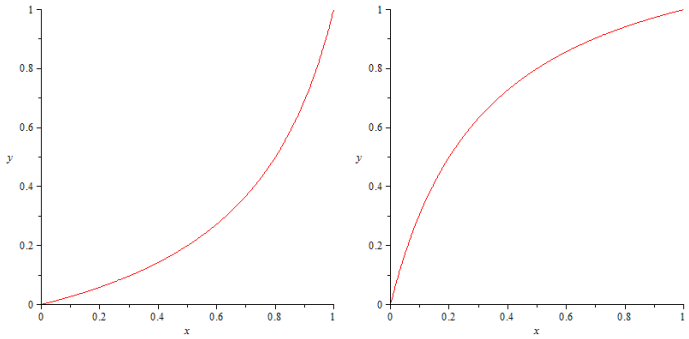

Figure 1.8: Necessity, possibility

This form of negation operator can be found in [22].

1.5.1 Introduction: Hedges in the Zadeh's sense

Zadeh introduced modifier functions of fuzzy sets called linguistic hedges. A number of studies [18, 17, 37] were made which discussed fuzzy logic and fuzzy reasoning with linguistic truth values. However, a systematic view of it has not been presented in the construction of linguistic hedges, which have corresponding reverse effects, such as in the case of "very" and "more or less".

In the early 1970s, Zadeh [86] introduced a class of powering modifiers that defined the concept of linguistic variables and hedges. He proposed computing with words as an extension of fuzzy sets and logic theory (Zadeh [87, 88]). The linguistic hedges (LHs) change the meaning

of primary term values. Many theoretical studies have contributed to the computation with words and to the LH concepts (see De Cock and Kerre [17]; Huynh, Ho, and Nakamori [40]; Rubin [67]; Türksen [75]).

As pointed out by Zadeh [82, 83, 84], linguistic variables and terms are closer to human thinking (which emphasise importance more than certainty) and are used in everyday life. For this reason, words and linguistic terms can be used to model human thinking systems (Liu et al. [57]; Zadeh [81]).

Zadeh [86] said that a proposition such as "The sea is very rough" can be interpreted as "It is very true that the sea is rough." Consequently, the sentences "The sea is very rough," "It is very true that the sea is rough," "(The sea is rough) is very true" can be considered equivalent. In fact, truth function modification permits an algorithmic approach to the calculus of deduction in approximate reasoning [9], by strengthening the liaison connection with classical logic. Since in traditional prepositional logic the validity of a reasoning depends on the simple truth proof of logic propositions [10], in a fuzzy logic we have the truth values that determine the fuzzy set associated with the conclusion of a deduction [78]. Hence, the transformation of a proposition "X is mA" into "(X is A) is mTrue" stresses the dependence of the conclusion on the initial conditions, as is the case in traditional binary logic. For this reason, in a deduction process the analytic representation of expressions such as "very true," "more or less true," "absolutely true" play an important role.
The adverbial locutions "very," "more or less," "absolutely" modify the truth value of the words "true" and "false." The first are called linguistic modifiers, while the second are called linguistic truth values. Different problems arise with them, such as how to build the related characteristic function, for a given combination of modifiers with logic connectives, and how to label the resulting set.

Basic notions of linguistic variables were formalized in different works by Zadeh in the mid 1970s [82, 83, 84]. These papers sought to provide a mathematical model for linguistic variables.

1.5.2 Modalities and hedges

When we apply the Pliant concept the necessity and possibility operators have the same form and the parameters are different. This common form is distributive with both conjunction and

disjunction operators. If we have a logical expression and before the logical expression there are modal operators, we can apply these modal operators directly on the variables. In the fuzzy concept, the variables are membership functions. Now, how should we interpret this action on the membership function? The simple answer is that this is a Hedge. Thus, here there is a simple correspondence between a Hedge and modalities.

In this case, "very true" is the same as "necessarily true", or "who is very young" is the same as "he/she is in our opinion necessarily young". In Figure 1.9, we show the effect of the modal operators on a membership function, where

$$\mu(x) = \frac{1}{1+e^{\frac{4}{15}(x-25)}} \tag{1.57}$$

$$\tau_N(x) = \frac{1}{1+\frac{0.8}{1-0.8}\frac{1-x}{x}} \tag{1.58}$$

$$\tau_P(x) = \frac{1}{1+\frac{0.2}{1-0.2}\frac{1-x}{x}} \tag{1.59}$$

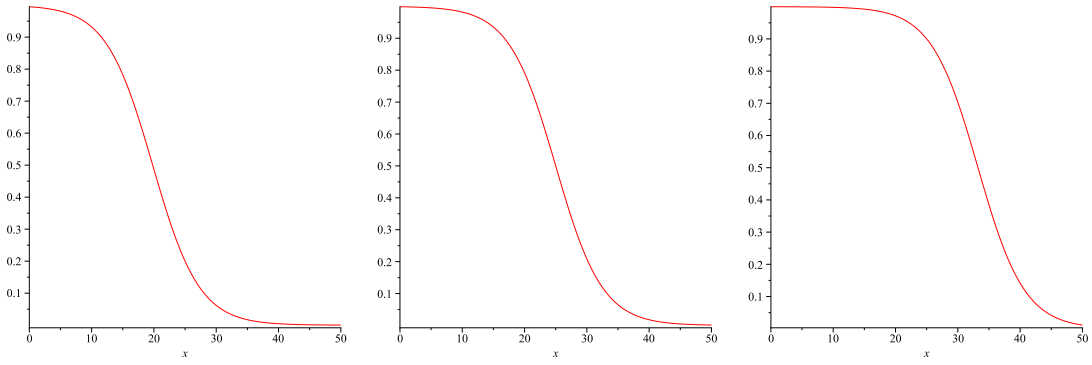

Figure 1.9: Very young (necessarily young), young, more or less young (possible young)

1.5.3 The sharpness operator

As we saw previously, modifiers can be introduced by repeating the arguments of conjunctive and disjunctive operators n-times. In the next step, n will be extended to any real number.

We will introduce the sharpness operator by repeating the arguments of the aggregation operator. Because in the Pliant system we have [25],

$$a(x_1,x_2,\ldots,x_n) = f^{-1}\left(\prod_{i=1}^{n} f(x_i)\right) \qquad \text{and}$$

$$a(\underbrace{x,x,\ldots,x}_{n}) = f^{-1}\left(f^n(x)\right),$$

we can introduce the following definition.

Definition 39. *The sharpness operator is*

$$\chi^{(\lambda)}(x) = f^{-1}\left(f^{\lambda}(x)\right) \qquad \lambda \in R \tag{1.60}$$

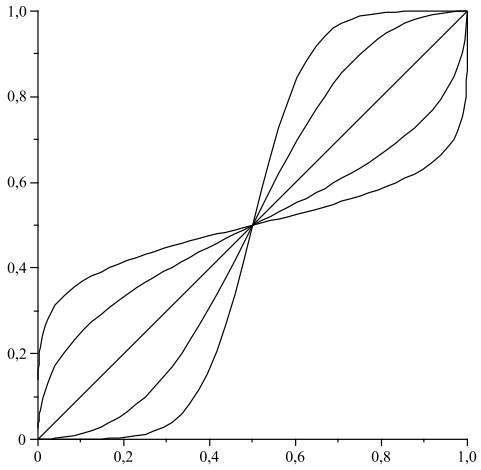

Figure 1.10: The sharpness operator with $\lambda = 1, 2, 4, 1/2, 1/4$ values.

1.6 General form of modifiers

Three types of modifiers were introduced earlier. These are the

1. Negation operator:
$$n_{\nu,\nu_0}(x) = f^{-1}\left(f(\nu_0)\frac{f(\nu)}{f(x)}\right) \tag{1.61}$$
2. Hedge operators, necessity and possibility operators:
$$\tau_{\nu,\nu_0}(x) = f^{-1}\left(f(\nu_0)\frac{f(x)}{f(\nu)}\right) \tag{1.62}$$
3. Sharpness operator:
$$\chi^{(\lambda)}(x) = f^{-1}\left(f^{\lambda}(x)\right) \tag{1.63}$$

These three types of operators can be represented in a common form.

Definition 40. *The general form of the modifier operators is*

$$\kappa^{(\lambda)}_{\nu,\nu_0}(x) = f^{-1}\left(f(\nu_0)\left(\frac{f(x)}{f(\nu)}\right)^{\lambda}\right) \tag{1.64}$$

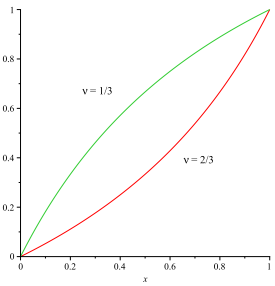

Figure 1.11: κ as a modifier with $\lambda = 1, \nu = 1/3, 2/3$

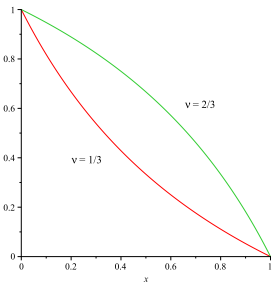

Figure 1.12: κ as a negation with $\lambda = -1, \nu = 1/3, 2/3$

Theorem 41. *Negation (1.61), hedge (1.62) and sharpness (1.63) are special cases of this modifier.*

$$\begin{aligned} \lambda &= -1 && \text{is the negation operator} \\ \lambda &= 1 && \text{is the hedge operator} \\ f(\nu_0) = f(\nu) &= 1 && \text{is the sharpness operator} \end{aligned}$$

Because the generator function is $f(x) = \left(\frac{1-x}{x}\right)^{\alpha}$ in Dombi operator case:

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

1. If $\lambda = -1$, then $\kappa_{\nu}^{(-1)} = n_{\nu}(x)$ is a negation
2. If $\lambda = 1$, then $\kappa_{\nu}^{(1)} = \tau_{\nu}(x)$ is a modifier

   if $\nu < \nu_0$, then $\tau_{\nu}(x)$ is a possibility operator

   if $\nu > \nu_0$, then $\tau_{\nu}(x)$ is a necessity operator
3. If $\lambda = \lambda_0 = \frac{1}{2}$, then $\kappa_{\nu}^{(\lambda)} = \chi^{\lambda}(x)$ is a sharpness operator

In figures [1.11 to 1.16], we plot the different curves for the $\kappa_{\nu}^{(\lambda)}(x)$ function.

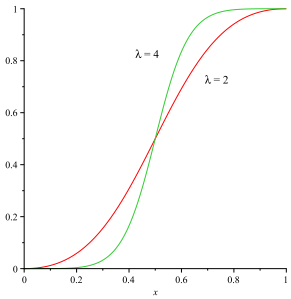

Figure 1.13: κ as a sharpness operator with $\nu = 1/2, \lambda = 2, 4$

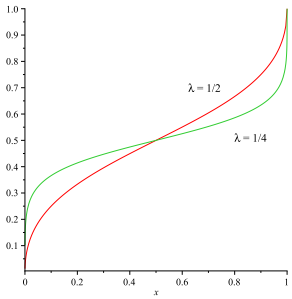

Figure 1.14: κ as a sharpness operator with $\nu = 1/2, \lambda = 1/2, 1/4$

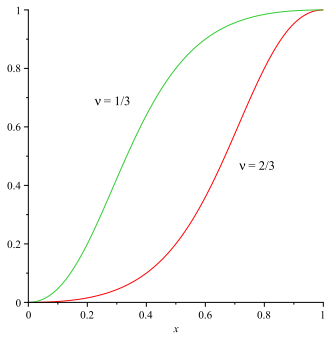

Figure 1.15: The $\kappa_\nu^{(\lambda)}(x)$ function with the parameters $\lambda = 2, \nu = 1/3, 2/3$

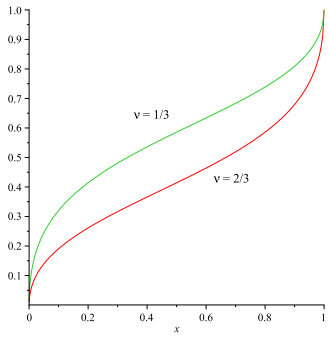

Figure 1.16: The $\kappa_\nu^{(\lambda)}(x)$ function with the parameters $\lambda = 1/2, \nu = 1/3, 2/3$

Chapter 2

Pliant system

2.1 DeMorgan law and general form of negation

We will use the generalized operator based on strict t-norms and strict t-conorms introduced by the authors. Calvo [13] and Yager [77].

Definition 1. *Generalized operators based on strict t-norms and t-conorms which are*

$$c(\boldsymbol{w},\boldsymbol{x}) = c(w_1,x_1;w_2,x_2;\ldots;w_n,x_n) = f_c^{-1}\left(\sum_{i=1}^{n} w_i f_c(x_i)\right), \tag{2.1}$$

$$d(\boldsymbol{w},\boldsymbol{x}) = d(w_1,x_1;w_2,x_2;\ldots w_n,x_n) = f_d^{-1}\left(\sum_{i=1}^{n} w_i f_d(x_i)\right), \tag{2.2}$$

where $w_i \geq 0$.

If $w_i = 1$ we get the t-norm and t-conorm. If $w_i = \frac{1}{n}$, then we get mean operators. If $\sum_{i=1}^{n} w_i = 1$, then we get weighted operators.

2.2 Operators with infinitely many negation operators

Now we will 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.

It is straightforward to see that the min and max operators belong to this class, as does the drastic operator. The next theorem characterizes those strict operator systems that have infinitely many negations and build a DeMorgan system. It is easy to see that $c(x,y) = xy$, $d(x,y) =$

$x+y-xy$ and $n(x)=1-x$ build a DeMorgan system. There are no other negations for building a DeMorgan system, as we will see below.

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*

$$f_c(x)\,f_d(x)=1. \tag{2.3}$$

For proof see [27].

Theorem 2. *The general form of the multiplicative pliant system is*

$$o_\alpha(x,y)=f^{-1}\left(\left(f^\alpha(x)+f^\alpha(y)\right)^{1/\alpha}\right) \tag{2.4}$$

$$n_\nu(x)=f^{-1}\left(f(\nu_0)\frac{f(\nu)}{f(x)}\right) \qquad \textit{or} \tag{2.5}$$

$$n_{\nu_*}(x)=f^{-1}\left(\frac{f^2(\nu_*)}{f(x)}\right), \tag{2.6}$$

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. Depending on the value of α, the operator is

$$\begin{aligned}\alpha>0 \qquad & o\alpha(x,y)=c(x,y)\\ \alpha<0 \qquad & o\alpha(x,y)=d(x,y)\end{aligned} \tag{2.7}$$

$$\begin{aligned}\lim_{\alpha\to\infty} o\alpha(x,y)&=\min(x,y)\\ \lim_{\alpha\to-\infty} o\alpha(x,y)&=\max(x,y)\end{aligned} \tag{2.8}$$

$$\alpha=0^+ \qquad \lim_{\alpha\to0^+} o\alpha(x,y)=\begin{cases}x \textit{ if } y=1\\ y \textit{ if } x=1\\ 0 \textit{ otherwise}\end{cases} \tag{2.9}$$

$$\alpha=0^- \qquad \lim_{\alpha\to0^-} o\alpha(x,y)=\begin{cases}x \textit{ if } y=0\\ y \textit{ if } x=0\\ 1 \textit{ otherwise}\end{cases} \tag{2.10}$$

This operator called the drastic operator.

For proof see [27].

2.3 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. Our starting point is that the fuzzy terms are so-called polar terms. In the table below we summarize some of the most common ones.

Let us choose the often used term "old". The same example exists in Zadeh's seminal paper[85]. We suppose now that the term "old" depends only on age, and we do not care whether most polar terms are always context dependent, i.e. an old professor is defined in another domain than old student. In classical logic we have to fix a dividing line; in our case let it be 63 years old ($a = 63$). If somebody is older than 63 years, then they belongs to the class (set) of old people; otherwise they do not. We can express this as an inequality form, using a characteristic function:

$$\chi_a(x) = \begin{cases} 1 & \text{if } a < x \\ 0 & \text{if } a \geq x \end{cases}$$

The expression $a < x$ is equivalent to the expression $0 < x - a$, so the above could be written as:

$$\chi(x-a) = \begin{cases} 1 & \text{if } 0 < x-a \\ 0 & \text{if } 0 \geq x-a \end{cases}$$

Generally, on the left hand side of the inequality we can have any $g(\mathbf{x})$ function.

$$\chi\big(g(\mathbf{x})\big) = \begin{cases} 1 & \text{if } 0 < g(\mathbf{x}) \\ 0 & \text{if } 0 \geq g(\mathbf{x}) \end{cases}$$

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

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

Instead of a strict relation, we will define a function which provides information on the validity of the relation. Remark: Introducing the distending function in this way allows one to

generalize the concept to R^n (in fuzzy set theory the membership functions is usually defined on R).

Roughly speaking, if the value of $g(\underline{x})$ is large and positive, then $truth(0 < g(\underline{x})) \approx 1$, if $g(\underline{x})$ is large and negative, then the $truth(0 < g(\underline{x})) = 0$ (i.e. false) and if $g(\underline{x}) = 0$, then $truth(0 < g(\underline{x}) = 1/2$, i.e. we are uncertain. $\delta(\underline{x})$ can be interpreted as a distending of the inequality relation.

The distending function is an approximation of the characteristic function. In classical mathematics, we speak of an open or closed interval and, according to this, the characteristic function takes the value 0 or 1 on the border. In our case, we will ignore this definition and define the characteristic function such that on the borders it takes the value of $\frac{1}{2}$. We should mention that a border with 50% probability belongs to the set and 50% to the complementer set.

If we define the set $0 < x$, then our approach is

$$\chi(x) = \begin{cases} 1 & \text{if} \quad x > 0 \\ \frac{1}{2} & \text{if} \quad x = 0 \\ 0 & \text{if} \quad x < 0 \end{cases} \tag{2.11}$$

If the set is described by $0 < g(x)$, then the corresponding characteristic function is $\chi(g(x))$. The distending function is an approximation of $\chi(x)$ defined by (2.11) in the following sense.

$$\delta_{\nu_0}^{(\lambda)}(x) = \begin{cases} > \nu_0 & \text{if} \quad x > 0 \\ = \nu_0 & \text{if} \quad x = 0 \\ < \nu_0 & \text{if} \quad x < 0 \end{cases} \tag{2.12}$$

The sigmoid function has the following properties:

$$\sigma(x) = \frac{1}{1+e^{-x}} = \begin{cases} \sigma(x) > \frac{1}{2} & \text{if } x > 0 \\ \sigma(x) = \frac{1}{2} & \text{if } x = 0 \\ \sigma(x) < \frac{1}{2} & \text{if } x < 0 \end{cases}$$

The sigmoid function is able to model an inequality. If we substitute x with a given $g(x)$ func-

tion, then

$$\sigma(g(x)) = \frac{1}{1+e^{-g(x)}} = \begin{cases} \sigma(g(x)) > \frac{1}{2} & \text{if } g(x) > 0 \\ \sigma(g(x)) = \frac{1}{2} & \text{if } g(x) = 0 \\ \sigma(g(x)) < \frac{1}{2} & \text{if } g(x) < 0 \end{cases}$$

2.4 Pliant operators

In multiplicative pliant systems the corresponding aggregative operators of the strict t-norm and strict t-conorm are equivalent, and DeMorgan's law is obeyed with the (common) corresponding strong negation of the strict t-norm or t-conorm.

We can summarize the properties of the multiplicative pliant system like so:

$$c(\mathbf{x}) = f^{-1}\left(\sum_{i=1}^{n} f(x_i)\right) \qquad c(\underline{\mathbf{w}},\underline{\mathbf{x}}) = f^{-1}\left(\sum_{i=1}^{n} w_i f(x_i)\right) \tag{2.13}$$

$$d(\mathbf{x}) = f^{-1}\left(\frac{1}{\sum_{i=1}^{n}\frac{1}{f(x_i)}}\right) \qquad d(\underline{\mathbf{w}},\underline{\mathbf{x}}) = f^{-1}\left(\frac{1}{\sum_{i=1}^{n}\frac{w_i}{f(x_i)}}\right) \tag{2.14}$$

$$a_{\nu_*}(\mathbf{x}) = f^{-1}\left(f(\nu_*)\prod_{i=1}^{n}\frac{f(x_i)}{f(\nu_*)}\right) \qquad a_{\nu_*}(\underline{\mathbf{w}},\underline{\mathbf{x}}) = f^{-1}\left(f(\nu_*)\prod_{i=1}^{n}\left(\frac{f(x_i)}{f(\nu_*)}\right)^{w_i}\right) \tag{2.15}$$

$$a(\mathbf{x}) = f^{-1}\left(\prod_{i=1}^{n} f(x_i)\right) \qquad a(\underline{\mathbf{w}},\underline{\mathbf{x}}) = f^{-1}\left(\prod_{i=1}^{n} f^{w_i}(x_i)\right) \tag{2.16}$$

$$n(\mathbf{x}) = f^{-1}\left(\frac{f^2(\nu_*)}{f(x)}\right), \tag{2.17}$$

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

where $f(x)$ is the generator function of the strict t-norm.

It was shown in [23] that the multiplicative pliant system fulfils the DeMorgan identity and the correct strong negation is defined by Eq.(2.17).

For example, let $f_c(x) = -\ln x$, the additive generator of the product operator. Assuming we have a pliant system, $f_d(x) = (-\ln x)^{-1}$ is a valid generator of a strict t-conorm. Their corresponding strong negation operators are the same as $n_c(x) = n_d(x) = n(x) = \exp[\frac{(\ln(\nu_*))^2}{\ln x}]$,

so that $n(1) = \lim_{x\to 1} n(x)$, for which $c(x,y) = xy$ and

$$d(x,y) = \exp\left[\frac{\ln x \ln y}{\ln xy}\right] \tag{2.18}$$

form a DeMorgan triplet.

2.4.1 The Dombi operator system

In another example, the Dombi operators form a pliant system. The operators are

$$c(\mathbf{x}) = \frac{1}{1+\left(\sum_{i=1}^{n}\left(\frac{1-x_i}{x_i}\right)^{\alpha}\right)^{1/\alpha}} \qquad c(\mathbf{x}) = \frac{1}{1+\left(\sum_{i=1}^{n} w_i\left(\frac{1-x_i}{x_i}\right)^{\alpha}\right)^{1/\alpha}} \tag{2.19}$$

$$d(\mathbf{x}) = \frac{1}{1+\left(\sum_{i=1}^{n}\left(\frac{1-x_i}{x_i}\right)^{-\alpha}\right)^{-1/\alpha}} \qquad d(\mathbf{x}) = \frac{1}{1+\left(\sum_{i=1}^{n} w_i\left(\frac{1-x_i}{x_i}\right)^{-\alpha}\right)^{-1/\alpha}} \tag{2.20}$$

$$a_{\nu_*}(\mathbf{x}) = \frac{1}{1+\frac{1-\nu_*}{\nu_*}\prod_{i=1}^{n}\left(\frac{1-x_i}{x_i}\frac{\nu_*}{1-\nu_*}\right)} \qquad a_{\nu_*}(\mathbf{x}) = \frac{1}{1+\left(\frac{1-\nu_*}{\nu_*}\right)\prod_{i=1}^{n}\left(\frac{1-x_i}{x_i}\frac{1-\nu_*}{\nu_*}\right)^{w_i}} \tag{2.21}$$

$$n(x) = \frac{1}{1+\left(\frac{1-\nu_*}{\nu_*}\right)^2\frac{x}{1-x}}, \tag{2.22}$$

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

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

$$f_c(x) = \left(\frac{1-x}{x}\right)^{\alpha} \qquad f_d(x) = \left(\frac{1-x}{x}\right)^{-\alpha}, \tag{2.23}$$

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.

Eqs.(2.19), (2.20), (2.21), (2.22) can be found in various articles of Dombi. Eqs.(2.19) and (2.20) can be found in [21], Eq.(2.21) in [20] and Eq.(2.22) can be found in [23].

Eq. (2.21) is called the 3Π operator because it can be written in the following form:

$$a(\mathbf{x}) = \frac{\prod_{i=1}^{n} x_i}{\prod_{i=1}^{n} x_i + \prod_{i=1}^{n}(1-x_i)} \tag{2.24}$$

Chapter 3

Decision making

3.1 Introduction

A decade ago a new computing infrastructure called the Grid was born. Ian Foster et. al. made this technology immortal by publishing the bible of the Grid [48] in 1998. Grid Computing has become a separate research area since then: currently grids are targeted by many world-wide projects. A decade is a long time. Although the initial goal of grids to serve various scientific communities by providing a robust hardware and software environment is still unchanged, different middleware solutions have been developed (Globus Toolkit [34], EGEE [1], UNICORE [32], etc.). The realizations of these grid middleware systems formed production grids that are mature enough to serve scientists having computation and data intensive applications. Nowadays research directions are focusing on user needs, where more efficient utilization and interoperability play key roles. To solve these problems, grid researchers have two options: as a member of a middleware developer group they can come up with new ideas or newly identified requirements and go through the long process of designing, standardizing and implementing the new feature, then wait for the next release containing the solution. Researchers sitting on the other side or unwilling to wait for years for the new release, need to rely on the currently available interfaces of the middleware components and have to use advanced techniques of other related research domains (peer-to-peer, Web computing, artificial intelligence, etc.). Here, we went for the second option to improve grid resource utilization with an interoperable resource management service.

Since the management and beneficial utilization of highly dynamic grid resources cannot be handled by the users themselves, various grid resource management tools must be developed and must support different grids. User requirements create certain properties that resource

managers must learn to support. This development is still continuing, and users need to distinguish between brokers and to migrate their applications when they move to a different grid. Interoperability problems and multi-broker utilization have led to the need for higher level brokering solutions. The meta-brokering approach requires a higher level resource management by allowing the automatic and simultaneous utilization of grid brokers. Scheduling at this level requires sophisticated approaches because high uncertainty exists at each stage of grid resource management. Despite these difficulties, this work addresses the resource management layer of middleware systems and offers an enhanced scheduling technique for improving grid utilization in a high-level brokering service. The main contribution of here lies in an enhanced scheduling solution based on the *Pliant System*, which is applied to the resource management layer of grid middleware systems.

3.2 Meta-Brokering in Grid Systems

Meta-brokering refers to a higher level of resource management, which utilizes an existing resource or service brokers to access various resources. In some generalized way, it acts as a mediator between users or higher level tools and environment-specific resource managers. The main tasks of this component are to *gather* static and dynamic broker properties, and to *schedule* user requests to lower level brokers; that is, to match job descriptions with broker properties. Afterwards, the job needs to be *forwarded* to the selected broker.

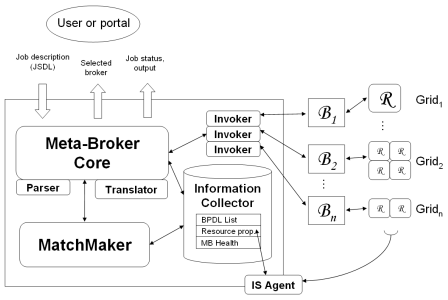

Figure 3.1: Components of the Meta-Broker.

Figure 3.1 provides a schematic diagram of the Meta-Broker (MB) architecture [46], including the components needed to fulfil the above-mentioned tasks. Different brokers use different service or resource specification descriptions to interpret the user request. These documents need to be written by the users to specify the different kinds of service-related requirements. For the resource utilization in Grids, OGF [2] developed a resource specification language standard called JSDL [7]. As JSDL is sufficiently general to describe the jobs and services of differ-

ent grids and brokers, this is the default description format of MB. The *Translator* component of the Meta-Broker is responsible for translating the resource specification defined by the user to the language of the appropriate resource broker that MB selects for a given request. These brokers have various features for supporting different user needs, hence an extendable Broker Property Description Language (BPDL) [46] is required to express metadata about brokers and the services they provide. The *Information Collector* (IC) component of MB stores data about the accessible brokers and historical data about previous submissions. This information tells us whether the chosen broker is available, and/or how reliable its services are. During broker utilization the successful submissions and failures are tracked, and for these events a ranking is updated for each special attribute in the BPDL of the appropriate broker (these attributes are listed above). In this way, the BPDL documents represent and store the dynamic states of the brokers. In order to support load balancing, there is an *IS Agent* (IS stands for Information System) reporting to the IC, which regularly checks the load of the underlying resources of each linked broker, and stores this data. The matchmaking process consists of the following steps: The *MatchMaker* (MM) compares the received descriptions with the BPDL of the registered brokers. This selection determines a group of brokers that can provide the required service. Otherwise, the request is rejected. In the second phase the MM counts a rank for each of the remaining brokers. This rank is calculated from the broker properties that the IS Agent updates regularly, and from the service completion rate that is updated in the BPDL for each broker. When all the ranks have been counted, the list of the brokers is ordered by these ranks. Lastly, the first broker of the priority list is selected, and the *Invoker* component forwards the request to the broker.

As regards related works, other approaches usually try to define common protocols and interfaces among scheduler instances enabling inter-grid usage. The meta-scheduling project in LA Grid [66] seeks to support grid applications with resources located and managed in different domains. They define broker instances with a set of functional modules. Each broker instance collects resource information from its neighbours and saves the information in its resource repository. The resource information is distributed over the different grid domains and each instance will have a view of the available all resources. The Koala grid scheduler [41] was designed to work on DAS-2 interacting with Globus middleware services with the main features of data and processor co-allocation; later it was extended to support DAS-3 and Grid'5000. Their policy is to use a remote grid only if the local one is saturated. They use a so-called delegated matchmaking (DMM), where Koala instances delegate resource information in a peer-to-peer manner. Gridway introduces a Scheduling Architectures Taxonomy [53]. Its Multiple Meta-

Scheduler Layers use Gridway instances to communicate and interact through grid gateways. These instances can access resources belonging to different administrative domains. They also pass user requests to another domain in cases where the current one is overloaded. Comparing these related approaches, we can state that all of them use a new method to expand current grid resource management boundaries. Meta-brokering appears in the sense that different domains are being examined as a whole, but they rather delegate resource information among domains, broker instances or gateways through their own, implementation-dependent interfaces. Their scheduling policies focus on resource selection by using aggregated resource information sharing, but our approach targets broker selection based on broker properties and performances.

3.3 Scheduling Algorithms

Earlier on we introduced the Pliant System and Grid Meta-Broker, and showed how the default matchmaking process is carried out. The main contribution of this part is to *see how* the scheduling part of this matchmaking process can be enhanced. To achieve this, we created a Decision Maker component based on functions of the *Pliant system*, and inserted it into the MatchMaker component of the Meta-Broker. The first part of the matchmaking is unchanged: the list of the available brokers is filtered according to the requirements of the actual job read from its JSDL. Then a list of the remaining brokers along with their performance data and the background grid load are sent to the Decision Maker in order to determine the most suitable broker for the actual job. The scheduling techniques and the scheduling process are described below.

Decision Maker uses a random number generator, and we chose a JAVA solution that generates pseudorandom numbers. The JAVA random generator class uses a uniform distribution and 48-bit seed, and the latter is modified by a linear congruential formula [49]. We also developed a unique random number generator that generates random numbers with a given distribution. We call this algorithm the generator function. In our case we defined a score value for each broker, and we created a distribution based on the score value. For example, the broker which has the highest score has the biggest chance of being chosen.

In this algorithm, the inputs are the broker id and the broker score, which are integer valued (see Table 3.1).

The next step is to choose a broker and put it into a temporary array: the cardinality is determined by the score value (see Table 3.2).

After the temporary array is filled, we shuffle this array and choose an array element using

Table 3.1: Inputs of the algorithm

BrokerID	Score
3	2
4	3
5	1
6	2

Table 3.2: Elements in the temporary array

Broker ID	3	3	4	4	4	5	6	6
Array ID	1	2	3	4	5	6	7	8

the JAVA random generator. In the example shown in Table 3.3, the generator function chose the broker with id 4.

Table 3.3: Shuffled temporary array

Broker ID	4	3	6	3	4	4	5	6
Array ID	1	2	3	4	5	6	7	8

Java Random generator: **5**

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.

- The success counter gives the number of jobs that finished without any errors.
- The failure counter shows the number of failed jobs.
- The load counter tells us the actual load of the grid behind the broker (in percentage terms).
- The running jobs counter shows the number of jobs sent to the broker which have not yet finished.

We developed five different [47, 29] kinds of decision algorithms. The trivial algorithm uses only a random generator to select a broker. The first three algorithms take into account the first three broker properties. These algorithms define a score number for each broker and use the generator function to select one. To calculate the score value, we build a weighted sum of the evaluated properties. This number is always an integer number. Furthermore, the second and third decision algorithms take into account the maximum value of the failure and load counter. This means that we extract the maximum value of the properties before multiplying them by the weight. The generator function of the third algorithm chooses a broker whose score number is not smaller than the half of the highest score value.

After testing different kinds of weighted systems, we found that the most useful weights (see in Table 3.4) that represent the weights of the decision algorithms applied here [47]:

Table 3.4: The weights of the decision makers

Decision Maker	Success_weight	Failed_weight	Load_weight
Decision I.	3	0.5	1
Decision II.	4	4	4
Decision III.	4	4	4

We developed *two* other types of decision algorithms [29] that took into account all the broker properties. These algorithms define a score number for each broker and use the generator function to select a broker. Algorithms that are related to the Pliant system use the kappa function to determine the broker's score number.

Because the Pliant system is defined in the $[0, 1]$ interval, we need to *normalize* the input value. These two algorithms differ only in this step:

1. The first algorithm uses a linear transformation called Decision4.

2. The second algorithm uses the sigmoid function to normalise the input values, which is called Decision5.

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 3.2). The opposite of this case is true for the success counter (see Figure 3.3).

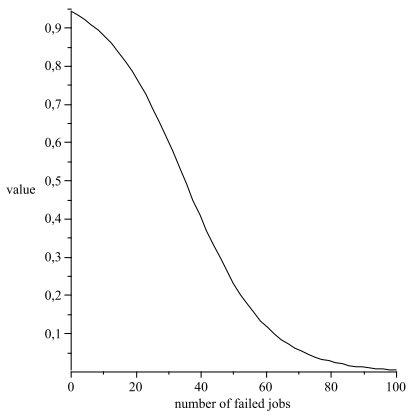

Figure 3.2: Normalising the failed jobs counter using Sigmoid function

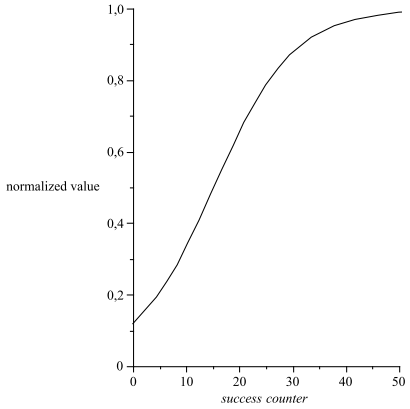

Figure 3.3: Normalising the success counter using the Sigmoid function

In the next step we can modify the normalised property value by using the same Kappa function (see Figure 3.4). We can also define the expected value of the normalisation via the ν and λ parameters.

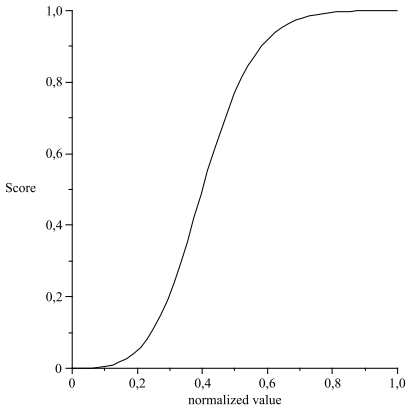

Figure 3.4: Normalized parameter values using the Kappa function

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.

When the Meta-Broker is running, the first two broker properties (the success and failure counters) are incremented via a feedback method that the simulator (or a user or portal in real-world cases) calls after the job has finished. The third and fourth properties, the load value and the running jobs, are handled by the IS Agent of the Meta-Broker, queried from an information provider (Information System) of a Grid. During a simulation this data is saved to a database by the Broker entities of the simulator. This means that by the time we start the evaluation and before we receive feedback from finished jobs, the algorithms can only rely on the background load and running processes of the grids. 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. With this additional training method, we can expect shorter execution times because we will select more reliable brokers.

3.4 Evaluation

In order to evaluate our proposed scheduling solution, we created a general simulation environment, where all the related grid resource management entities could be simulated and coordinated. The GridSim toolkit [12] is a fully extendable, widely used and accepted grid simulation tool; and these are the main reasons why we chose this toolkit for our simulations. It can be used for evaluating VO-based resource allocation, workflow scheduling, and dynamic resource provisioning techniques in global grids. It supports modeling and the simulation of heterogeneous grid resources, users, applications, brokers and schedulers in a grid computing environment. It provides primitives for the creation of jobs (called gridlets), mapping of these jobs to resources, and their management, hence resource schedulers can be simulated to study scheduling algorithms. GridSim provides a multilayered design architecture based on SimJava [38], a general purpose discrete-event simulation package implemented in Java. It is used for handling the interactions or events among GridSim components. All components in GridSim communicate with each other through message passing operations defined by SimJava.

Our general simulation architecture can be seen in Figure 3.5. In the right hand corner we can see that the GridSim components were used for the simulated grid systems. Resources can be defined with different grid-types. Resources consist of more machines, to which workloads

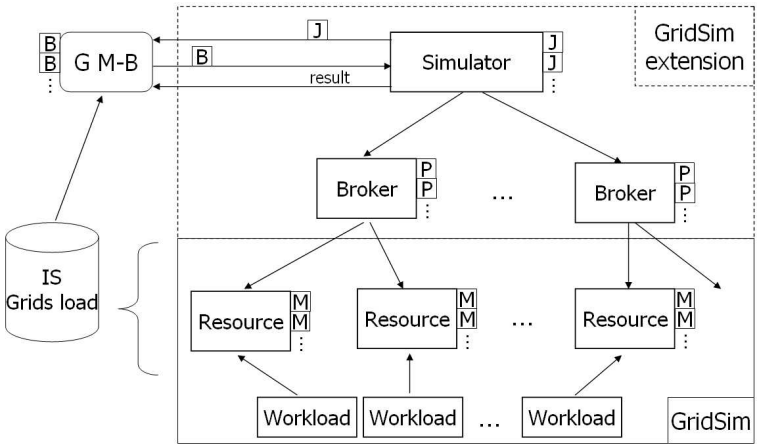

Figure 3.5: Meta-Brokering simulation environment based on GridSim

can be set. On top of this simulated grid infrastructure we can set up brokers. The Broker and Simulator entities were developed by us to enable the simulation of meta-brokering. Brokers are extended GridUser entities:

- They can be connected to one or more resources;
- Different properties can be set to these brokers (agreement handling, co-allocation, advance reservation, etc.);
- Some properties can be marked as unreliable;
- Various scheduling policies can be defined (pre-defined ones: rnd – random resource selection, fcpu – resources having more free cpus or fewer waiting jobs are selected, nfailed – resources having fewer machine failures are selected);
- Generally resubmission is used when a job fails due to resource failure;
- Next, they report to the IS Grid load database by calling the feedback method of the Meta-Broker with the results of the job submissions (this database has a similar purpose to that of a grid Information System).

The Simulator is an extended GridSim entity:

- It can generate a requested number of gridlets (jobs) with different properties, start and run times (lengths);
- It is related to the brokers and is able to submit jobs to them;

- The default job distribution is the random broker selection (but the middleware types are taken into account at least);
- In the case of job failures, a different broker is selected for the given job;
- It is also related to the Grid Meta-Broker through its Web service interface and is able to call its matchmaking service for broker selection.

3.4.1 Preliminary testing phase

Table 3.5: Preliminary evaluation setup.

Broker	Scheduling	Properties	Resources	Workload
1.	fcpu	A	8	20*8
2.	fcpu	B	8	20*8
3.	fcpu	C	8	20*8
4.	fcpu	A_F	8	20*8
5.	fcpu	B_F	8	20*8
6.	fcpu	C_F	8	20*8
7.	nfail	$A_F B$	10	20*10
8.	nfail	AC_F	10	20*10
9.	nfail	$B_F C$	10	20*10
10.	rnd	-	16	20*16

Table 3.5 shows the details of the preliminary evaluation environment. 10 brokers can be used in this simulation environment. The second column denotes the scheduling policies used by the brokers: fcpu means the jobs are scheduled to the resource with the most free cpus, nfail means those resources are selected that have fewer machine failures, and rnd means randomized resource selection. The third column shows the capabilities/properties (e.g.: coallocation, checkpointing, ...) of the brokers: three properties are used in this environment. Here, subscript F means unreliability; a broker having such a property may fail to execute a job with the requested service with a probablity of 0.5. The fourth column contains the number of resources utilized by a broker, while the fifth column contains the number of background jobs submitted to the broker (SDSC BLUE workload logs taken from the Parallel Workloads Archive [3]) during the evaluation timeframe.

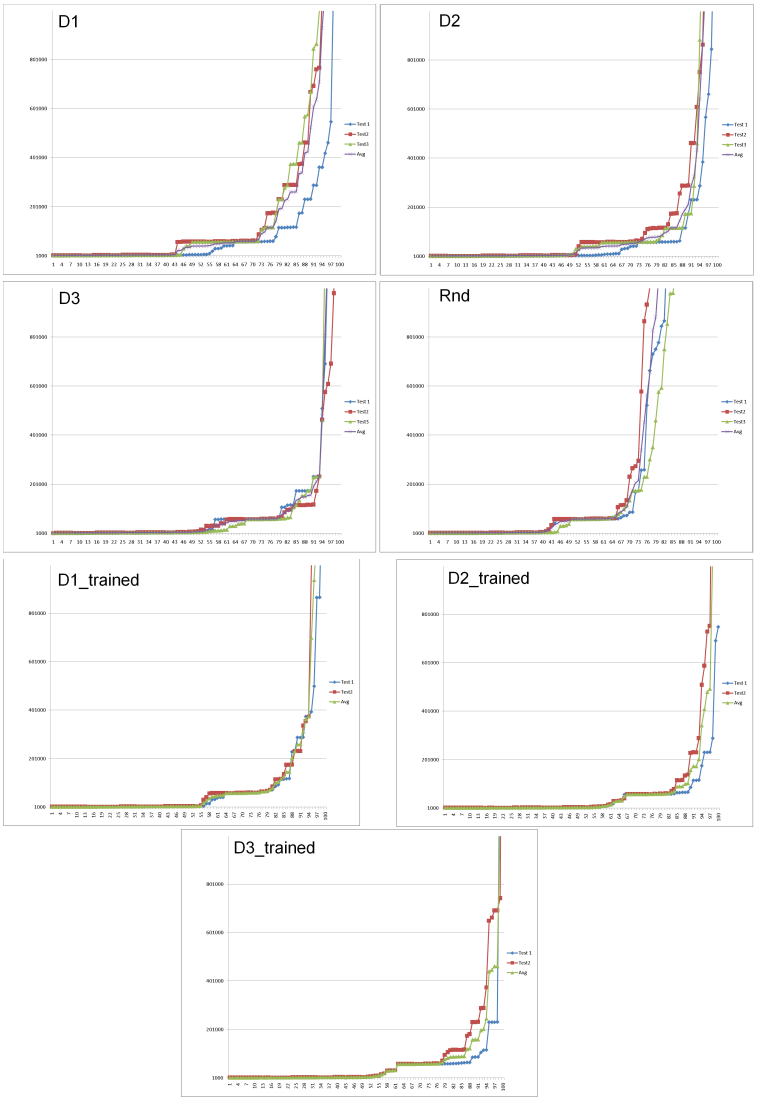

Figure 3.6: Diagrams of the preliminary evaluation for each algorithm

As shown in the table, we utilised 10 brokers to perform our first experiment. In this case we submitted 100 jobs to the system, and measured the makespan of all the jobs (time past from submission up to a successful completion, including the waiting time in the queue of the resources and resubmissions on failures). Out of the 100 jobs, 40 had no special property (this means all the brokers could successfully execute them), while for the rest of the jobs the three properties were distributed equally: 20 jobs had property A, 20 had B and 20 had C. Each resource of the simulated grids was utilised by 20 background jobs (workload) with different submission times based on the distribution defined by the SDSC BLUE workload logs.

Figure 3.6 shows the detailed evaluation runs with the scheduling algorithms Decision 1 (D1), 2 (D2), 3 (D3) and without the use of the Meta-Broker (randomized broker selection – Rnd), respectively. In Figure 3.7 we can see the averages of the tests with the different algorithms. This illustrates best the differences between the simulations with and without the

use of the Meta-Broker.

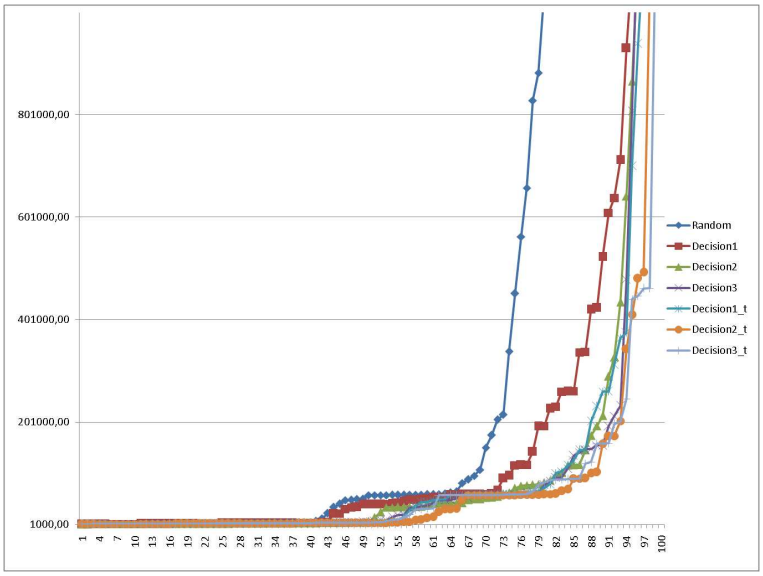

Figure 3.7: Summary diagram of the preliminary evaluation

After reviewing the diagrams of the preliminary evaluations, we can state that all the proposed scheduling algorithms (D1, D2 and D3) provide shorter execution times than the random broker selection. In the main evaluation phases, our goal will be to set up a more realistic environment and test it using a bigger number of jobs.

3.4.2 Main testing phase

We created two different kinds of evaluation environment. Based on the findings of [47] we tested the first three Decision algorithm, and the best algorithm was tested with the Pliant algorithm [29].

Test environment I.

Table 3.6 shows the evaluation environment used in the main evaluation. The simulation setup was derived from real-life production grids: current grids and brokers support ony a few special properties: we used four. To determine the (proportional) number of resources in our simulated grids, we compared the sizes of current production grids (EGEE VOs, DAS3, NGS, Grid5000, OSG, ...). We employed the same notations in this table as before.

In the main evaluation we utilised 14 brokers. In this case, we submitted 1000 jobs to the system, and again measured the makespan of all the jobs. Out of the 1000 jobs, 100 had no special property, while for the rest of the jobs, the four properties were distributed in the following way: 30 jobs had property A, 30 had B, 20 had C and 10 had D. The workload logs

Table 3.6: Main evaluation setup.

Broker	Scheduling	Properties	Resources	Workload
1.	fcpu	A	6	50*6
2.	fcpu	A_F	8	50*8
3.	fcpu	A	12	50*12
4.	fcpu	B	10	50*10
5.	fcpu	B_F	10	50*10
6.	fcpu	B	12	50*12
7.	fcpu	B_F	12	50*12
8.	fcpu	C	4	50*4
9.	fcpu	C	4	50*4
10.	fcpu	$A_F D$	8	50*8
11.	fcpu	AD	10	50*10
12.	fcpu	AD_F	8	50*8
13.	fcpu	AB_F	6	50*6
14.	fcpu	ABC_F	10	50*10

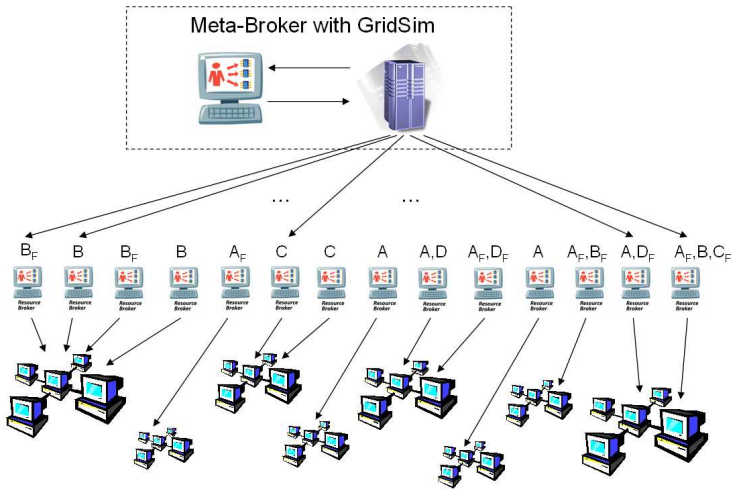

Figure 3.8: Simulation in the main evaluation environment

contain 50 jobs for each resource. Figure 3.8 gives a graphical representation of the simulation environment.

In the first phase of the main evaluation the simulator submitted all the jobs at once, just like in the preliminary evaluation. The results for the first three algorithms of this phase can be seen

in Figure 3.9.

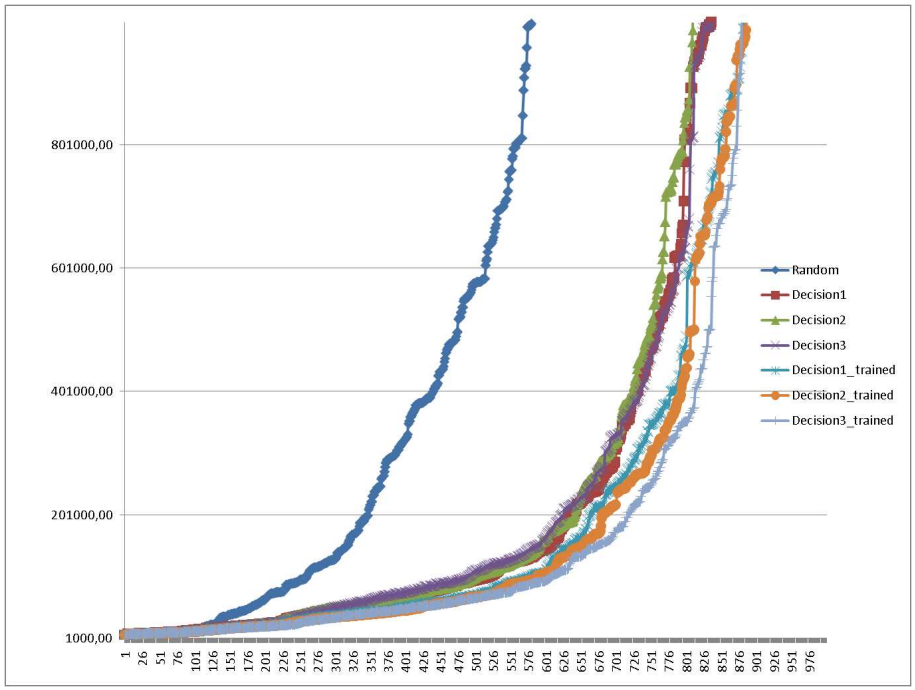

Figure 3.9: Diagram of the first phase of the main evaluation

In the first phase, we could not exploit all the features of the algorithms because we submitted all the jobs at once and the performance data of the brokers were not updated early enough for the matchmaking process. To avoid this, in the last phase of the main evaluation we submitted the jobs periodically: 1/3 of the jobs were submitted at the beginning then the simulator waited for 200 jobs to finish and update the performances of the brokers. After this, the simulator again submitted 1/3 of all the jobs and waited for 300 more to finish. Then, the rest of the jobs (1/3 again) were submitted. In this way, the broker performance results could be used by the scheduling algorithms. Figure 3.10 shows the results of the last evaluation phase. Here, we can see that the runs with training did a lot with trained values because the feedback of the first submission period compensateed for the lack of training.

Figure 3.11 provides a visual summary of the different evaluation phases. The above columns show the average values of each evaluation run with the same parameters. The results clearly show that with more intelligence (more sophisticated methods) in the system, the performance increases. The most advanced version of the first three proposed meta-brokering solution is the Decision Maker with the algorithm called Decision3 with training. Once the number of brokers and job properties are big enough to set up this Grid Meta-Broker Service for inter-connecting several Grids, with the above scheduling algorithms our service will be ready to serve thousands of users even under hcondition of high uncertainty.

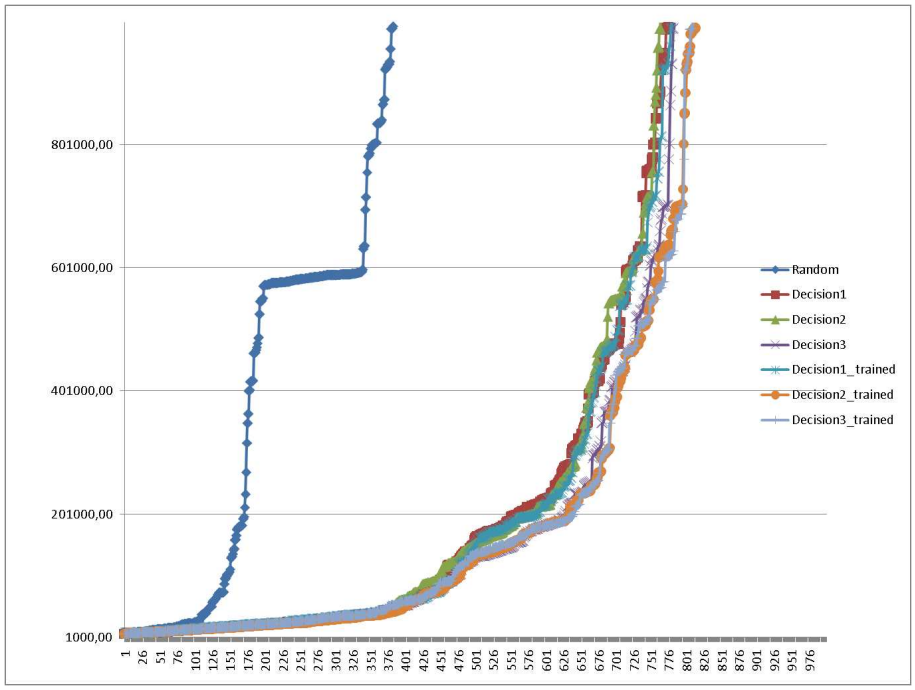

Figure 3.10: Diagram of the second phase of the main evaluation

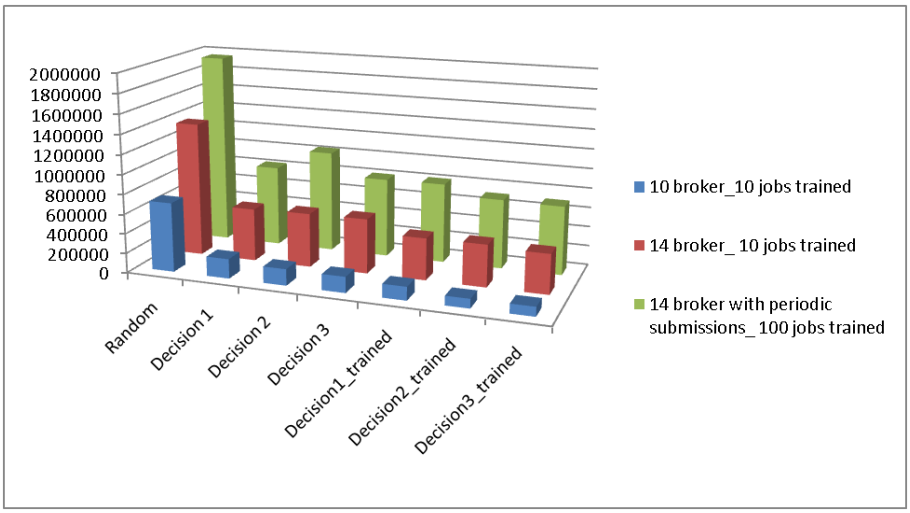

Figure 3.11: Summary of the evaluation results

Test environment II.

Table 3.7 shows the *evaluation environment* used in our evaluation. The simulation setup was derived from real-life production grids: current grids and brokers support only a few special properties: here we used four. To determine the number of resources in our simulated grids we compared the sizes of current production grids (EGEE VOs, DAS3, NGS, Grid5000, OSG, etc.). In the evaluation we utilised 14 brokers. We submitted 1000 jobs to the system, and measured the makespan of all the jobs. Out of the 1000 jobs 100 had no special properties, while for the rest of the jobs four key properties were distributed in the following way: 300 jobs

Table 3.7: Evaluation environment setup.

Broker	Scheduling	Properties	Resources
1.	fcpu	A	6
2.	fcpu	A_F	8
3.	fcpu	A	12
4.	fcpu	B	10
5.	fcpu	B_F	10
6.	fcpu	B	12
7.	fcpu	B_F	12
8.	fcpu	C	4
9.	fcpu	C	4
10.	fcpu	$A_F D$	8
11.	fcpu	AD	10
12.	fcpu	AD_F	8
13.	fcpu	AB_F	6
14.	fcpu	ABC_F	10

had property A, 300 had B, 200 had C and 100 had D. The second column above denotes the scheduling policies used by the brokers: fcpu means the jobs are scheduled to the resource with the highest free cpu time. The third column shows the capabilities/properties (like coallocation, checkpointing) of the brokers: here we used A, B, C and D in the simulations. The F subscript means unreliability, a broker having the kind of property that may fail to execute a job with the requested service with a probablity of 0.5. The fourth column contains the number of resources utilized by a broker. As a background workload, 50 jobs were submitted to each resource by the simulation workload entities during the evaluation timeframe. The SDSC BLUE workload logs were used for this purpose, taken from the Parallel Workloads Archive [3].

In order to test all the features of the algorithms, we submitted the jobs periodically: 1/3 of the jobs were submitted at the beginning then the simulator waited for 200 jobs to finish and update the performances of the brokers. After this phase the simulator again submitted 1/3 of all the jobs and waited for 200 more to finish. Lastly the remaining jobs (1/3 again) were submitted. In this way, the broker performance results could be updated and monitored by the scheduling algorithms.

In the previous section we explained how the two algorithms called Decision4 and Decision5

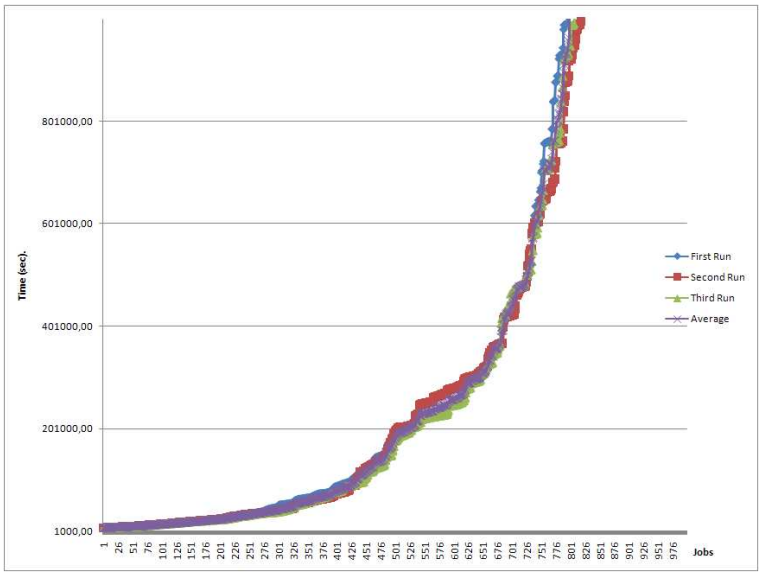

Figure 3.12: Results of the Decision 4 algorithm

(both based on the Pliant system) work. For the evaluation part we repeated each experiment *three times*. The measured simulation results of the Decision4 algorithm can be seen in Figure 3.12. We noticed that the measured runtimes for the jobs were very close to each other. When comparing the various simulation types we always used the median: we counted the average runtime of the jobs in each of the three series and discarded the best and the worst simulations.

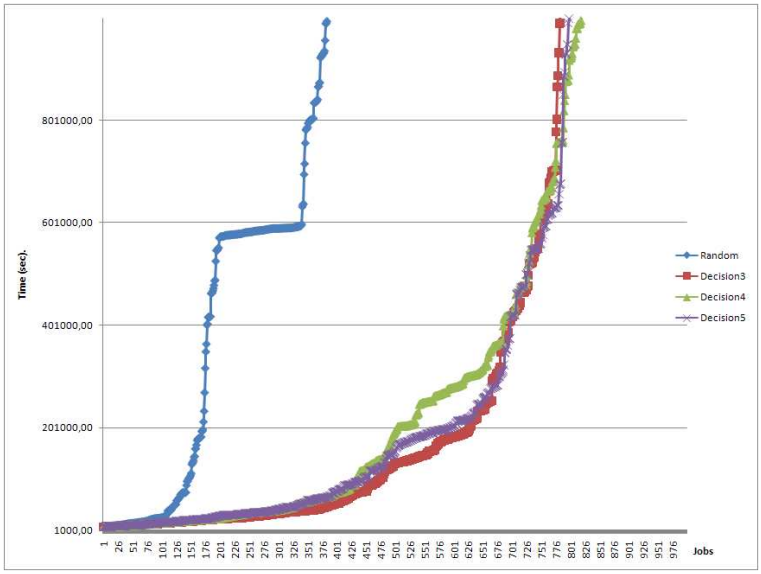

Figure 3.13: Simulation results for the three decision algorithms compared with the random decision maker

A comparison of the simulation results can be seen in Figure 3.13. We found that in our previous work [47] we used only random number generators to boost the Decision Maker, and proposed three algorithms called Decision1, Decision2 and Decision3. Because Decision3 gave the best results, we will compare our new measurements with the results of this algorithm. We

can see that for around 1/3 of the simulations, Decision3 provides better results, but the overall makespans are better for the new algorithms.

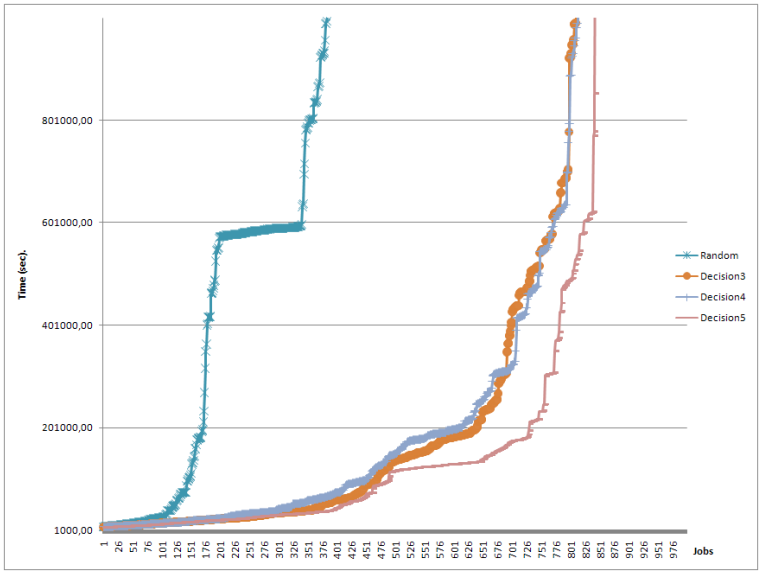

Figure 3.14: Simulation results for three decision algorithms with training compared with the random decision maker

The simulation results for the algorithms with training can be seen in Figure 3.14. As we mentioned earlier, we used a training process to initiate the performance values of the brokers before job submissions. In this way, the decisions for the first round of jobs can be made better. Upon examining the results, Decision4 still performs about the same as Decision3, but Decision5 clearly *outperforms* the other two.

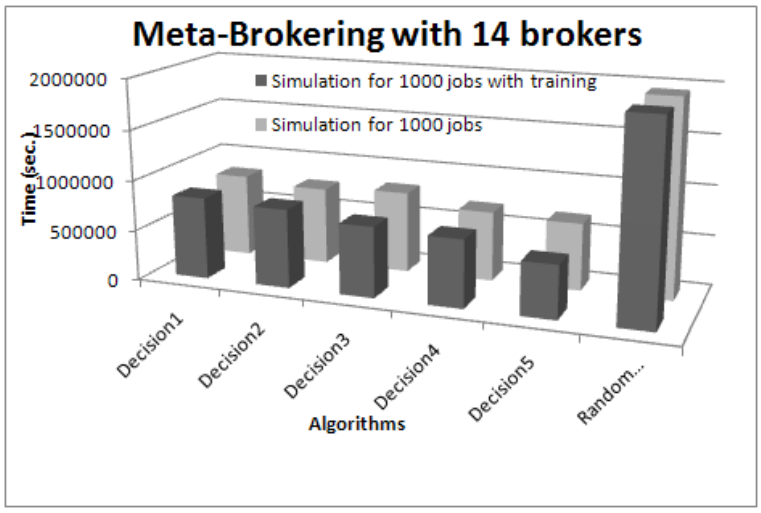

Figure 3.15: Simulation in the main evaluation environment

In Figure 3.15, 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 *most advanced* version of our proposed meta-brokering solution is called the Decision Maker using the algorithm called *Decision5 with training*. Once the number of brokers and job properties are sufficiently high to set up this Grid Meta-Broker Service for inter-connecting several Grids, the new scheduling algorithms will be ready to serve thousands of users even under conditions of *high uncertainty*.

3.5 Summary

In this chapter we discussed decision-related problems in the Grid environment. The Grid Meta-Broker itself is a standalone Web-Service that can serve both users and grid portals. This novel enhanced scheduling solution permits a higher level, interoperable brokering by utilising existing resource brokers of different grid middleware. It gathers and utilises meta-data about brokers taken from various grid systems to establish an adaptive meta-brokering service. We developed several new scheduling components for this Meta-Broker. 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. We evaluated our algorithms in a grid simulation environment based on GridSim, and performed simulations with real workload samples. The evaluation results accord with our expected utilisation gains; namely, the enhanced scheduling provided by the modified Decision Maker results in a *more efficient* job execution.

Chapter 4

The approximation of functions and function decomposition

4.1 Introduction

Functions have a very important role in science and technology and in our everyday lives. They can be represented in terms of their coordinates or by using some mathematical expression. Usually, if the coordinates are given, then it is important to know what kind of expression approximately describes it, because sometimes interpolation or extrapolation questions have to be addressed. The function can also be used to calculate values at any given point. In this way, we can construct a function and define its parameters. In other words, we can compress this information using a function, which will involve some learning process. In science and technology in most cases we can get samples to determine the relationship between the input and output values, which is called curve-fitting, because usually we do not require an exact fit, but only an approximation. One way to approximate a function with coordinates is via an interpolation process. Interpolation is a method where we determine a function (which may be a polynomial) that best fits the given data points and using this result we can determine the function value if new data points are given. We can regard interpolation as a specific kind of curve-fitting, where the function must go through the data points. There is a range of interpolation methods available for this problem such as linear, polynomial, spline and trigonometric methods. It is also possible to use neural networks for approximation purposes.

The polynomial interpolation has the following general form:

$$y(\mathbf{w}, x) = \sum_{i=0}^{n} w_i x^i, \tag{4.1}$$

where n is the order of the polynomial. The polynomial coefficients $w_0, \ldots w_n$ will be collec-

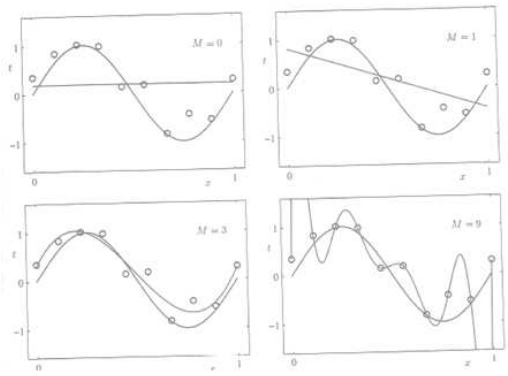

Figure 4.1: Plots of polynomials with different orders of *n*

tively denoted by the vector **w**. Note that although the polynomial function y(**w**,x) is a nonlinear function in *x*, it is a linear function in **w**. Here, the values of the coefficients will be determined by fitting the polynomial to the training points.

Using an interpolation method we have to determine *n* parameters in Equation (4.1), and we have just *n* coordinates, so we cannot verify any compression. Interpolation nowadays is of less interest than it once was a few years ago. Curve-fitting can be done by minimizing the error function that measures the misfit between the function for any given value of **w** and the data points. One simple and widely used error function is the sum of the squares of the errors between the predicted $y(\mathbf{w}, x_n)$ for each data point x_n and the corresponding target values y_n, so in effect we have to minimise the 'energy function':

$$E(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{n}(y(x_i, \mathbf{w}) - y_i)^2 \tag{4.2}$$

Clearly, it is a nonnegative quantity that would be zero if and only if the function $y(\mathbf{w}, x)$ were to pass exactly through each training data point; that is, if it were a perfect fit.

We can solve the curve-fitting problem by choosing a **w** for which $E(\mathbf{w})$ is as small as possible. However, every type of method has its drawbacks and this one is no different.

1. The main problem here is how to choose the order *n* of the polynomial and, as we shall see, this will turn out to be the problem of model comparison or model selection. In Figure (4.1) we give four examples of fitting polynomials of orders M = 0, 1, 3, and 9 to a data set, where we use sample data taken from a sine function with noise.

2. They are not accurate enough.

3. The parameters that we get after optimization provide no direct information about the behaviour of the function; i.e. varying a parameter does not affect this function.

4. It would be nice if we could modify a certain part of the function by varying the parameter. For example, if we would like to increase the maximum value at a certain point, we could do this by varying a parameter using an approximation or interpolation (Fourier series, Taylor series, etc.). It is not possible to vary a parameter so as to modify just a part of the function as the parameter and the rest of the function describe the whole of it.

5. Very often it would be useful to describe a function by its variations, e.g. "it slowly increases, then suddenly changes its behaviour and speeds up and after reaching its maximum value, it suddenly goes down". Using classical function construction procedures, it is not so easy to find a parametrical mathematical expression which corresponds to the natural language description of the function, but it would be useful in fields like economics and marketing.

Here, we will present a solution that solves some of these problems [28, 45]. Our aim is to approximate the function with the help of membership-like functions. We need a kind of membership function which approximates the characteristic function. We get it by introducing the distending function which describes inequalities. Using the conjunction operator with the distending function, we get the desired function class. We will also call this positive and negative effects, whose mathematical description can be realised by using continuous-valued logic. Here we will use a special one called the Pliant concept with Dombi operators included. After an aggregative procedure we get the derived function. Aggregation was first introduced by Dombi [26], but later the fuzzy community rediscovered and generalized the concept and called it the uninorm. Instead of the membership function we shall use soft inequalities and soft intervals which are called distending functions. All of the parameters introduced have a definite meaning. Also, it can be proved that certain function classes may be uniformly approximated.

4.2 Distending function

In Chapter 2 we show how important is the Distending function. In this section we provide additional information that is used for the approximation technique. In fuzzy logic theory, the membership function plays an important role. In Pliant logic we use a soft inequality and we call it the distending function.

Here f is the generator function of the logical connectives, λ is responsible for the sharpness and a is the threshold value.

The approximation process is developed within the framework of the Pliant system.

Definition 2. *The Pliant system is a strict, monotonously increasing t-norm and t-conorm. The following expression is valid for the generator function:*

$$f_c(x) \cdot f_d(x) = 1$$

Definition 3. *The general form of distending function is*

$$\delta_a^{(\lambda)}(x) = f^{-1}\left(e^{-\lambda(x-a)}\right) \qquad \lambda \in R, a \in R$$

The semantic meaning of $\delta_a^{(\lambda)}$ is

$$truth(a <_\lambda x) = \delta_a^{(\lambda)}(x)$$

Remark 1:

1. In the Pliant system f could be the generator function of the conjunctive operator or the disjunctive operator. The form of $\delta_a^{(\lambda)}(x)$ is the same in both cases.
2. In the Pliant concept, the operators and membership are closely related.
3. Using the soft inequality with the distending function, we cannot describe a membership like "middle age".

The distending function in the Dombi operator case is the sigmoid (logistic) function:

$$\sigma_a^{(\lambda)}(x) = \frac{1}{1+e^{-\lambda(x-a)}}$$

4.2.1 Distending interval

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

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

Using the Pliant concept, we translate it into two inequalities corresponding to an "and" (conjunctive) operator.

$$truth(a <_{\lambda_1} x) \; and \; truth(x <_{\lambda_2} b)$$

Theorem 2. *In a pliant system if the initial conditions are*

$$\delta_{a,b}^{\lambda_1,\lambda_2}(a) = \nu_0 \qquad \delta_{a,b}^{\lambda_1,\lambda_2}(b) = \nu_0, \tag{4.3}$$

then the distending interval is

$$\delta_{a,b}^{\lambda_1,\lambda_2}(x) = f^{-1}\left(\frac{1}{A}\left(A_1 e^{-\lambda_1(x-a)} + A_2 e^{-\lambda_2(b-x)}\right)\right), \tag{4.4}$$

where

$$\begin{aligned} A &= \frac{1}{f(\nu_0)}\left(1 - e^{-(\lambda_1+\lambda_2)(b-a)}\right) \\ A_1 &= 1 - e^{-\lambda_2(b-a)} \\ A_2 &= 1 - e^{-\lambda_1(b-a)} \end{aligned} \tag{4.5}$$

Proof 1. *It is a straightforward calculation.*

In the Dombi operator case, the distending function has the following form:

$$\sigma_{a,b}^{\lambda_1,\lambda_2}(x) = \frac{1}{1 + \frac{1-\nu_0}{\nu_0}\frac{1}{A}\left(A_1 e^{-\lambda_1(x-a)} + A_2 e^{-\lambda_2(b-x)}\right)}, \tag{4.6}$$

where

$$\begin{aligned} A &= 1 - e^{-(\lambda_1+\lambda_2)(b-a)} \\ A_1 &= 1 - e^{-\lambda_2(b-a)} \\ A_2 &= 1 - e^{-\lambda_1(b-a)} \end{aligned}$$

In Figure (4.2), we have plotted $\sigma_{a,b}^{\lambda_1,\lambda_2}(x)$ using different parameter values.

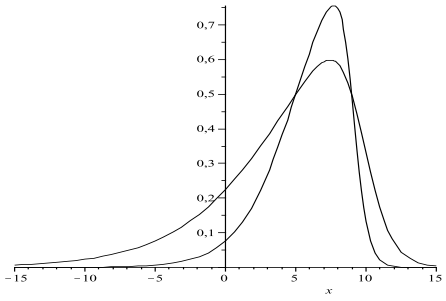

Figure 4.2: $\sigma_{a,b}^{\lambda_1,\lambda_2}(x)$ if $a = 5, b = 9, \nu_0 = \frac{1}{2}, \lambda_1 = \frac{1}{2}, \lambda_2 = 2$ and $\lambda_1 = \frac{1}{4}, \lambda_2 = 1$

The following properties hold for the distending interval:

Theorem 3.

$$\delta_{a,b}(x) = \lim_{\lambda_1\to\infty,\lambda_2\to\infty} \delta_{a,b}^{\lambda_1,\lambda_2}(x) = \begin{cases} 0 & if \quad x < a \\ \nu_0 & if \quad x = a \\ 1 & if \quad a < x < b \\ \nu_0 & if \quad x = b \\ 0 & if \quad b < x \end{cases} \tag{4.7}$$

See Figure (4.3)

Proof 2. *Because* $\delta_{a,b}^{\lambda_1,\lambda_2}(a) = \delta_{a,b}^{\lambda_1,\lambda_2}(b) = \nu_0$, *we have to prove just 3 cases:*

$$\begin{aligned} if\, x < a \quad then \lim_{\lambda_1,\lambda_2} \left(A_1 e^{-\lambda_1(x-a)} + A_2 e^{-\lambda_2(b-x)}\right) \to \infty \\ if\, x > b \quad then \lim_{\lambda_1,\lambda_2} \left(A_1 e^{-\lambda_1(x-a)} + A_2 e^{-\lambda_2(b-x)}\right) \to \infty \\ if\, a < x < b \quad then \lim_{\lambda_1,\lambda_2} \left(A_1 e^{-\lambda_1(x-a)} + A_2 e^{-\lambda_2(b-x)}\right) \to 0 \end{aligned}$$

This is obvious because of the properties of the exponential function.

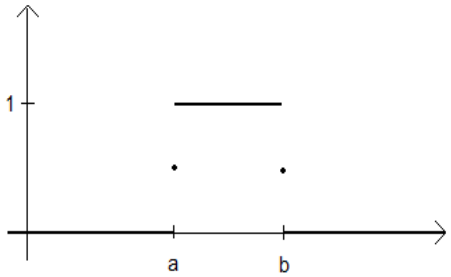

Figure 4.3: $\delta_{a,b}(x)$ function

From (4.6) we can derive another type of function where *a* and *b* are equal, which will call an impulse function. This means that the intervals are not given, but just the value where the function is a maximum.

Theorem 4. *The following limit property holds:*

$$\delta_a^{\lambda_1,\lambda_2}(x) = \lim_{a\to b} \delta_{a,b}^{\lambda_1,\lambda_2}(x) = \tag{4.8}$$

$$f^{-1}\left(f(\nu_0)\left(\frac{\lambda_2}{\lambda_1+\lambda_2}e^{-\lambda_1(x-a)} + \frac{\lambda_1}{\lambda_1+\lambda_2}e^{-\lambda_2(a-x)}\right)\right)$$

In the Dombi operator case

$$\sigma_a^{\lambda_1,\lambda_2}(x) = \lim_{a\to b} \sigma_{a,b}^{\lambda_1,\lambda_2}(x) = \tag{4.9}$$

$$\frac{1}{1+\frac{1-\nu_0}{\nu_0}\left(\frac{\lambda_2}{\lambda_1+\lambda_2}e^{-\lambda_1(x-a)}+\frac{\lambda_1}{\lambda_1+\lambda_2}e^{-\lambda_2(a-x)}\right)}$$

Proof 3. *The proof is based on a limes property and we use the L'Hospital rule.*

In Figure (4.4), $\sigma_a^{\lambda_1,\lambda_2}(x)$ is shown with typical values.

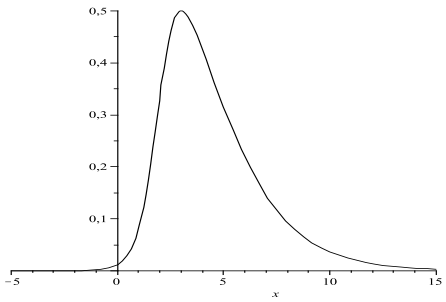

Figure 4.4: $\sigma_a^{\lambda_1,\lambda_2}(x)$ if $a = 3, \nu_0 = \frac{1}{2}, \lambda_1 = 2, \lambda_2 = \frac{1}{2}$

We can get an impulse function from $\delta_a^{\lambda_1,\lambda_2}(x)$.

Theorem 5.

$$\delta_a(x) = \lim_{\lambda_1,\lambda_2\to\infty} \delta_a^{\lambda_1,\lambda_2}(x) = \begin{cases} \nu_0 & \text{if} \quad x = a \\ 0 & \text{if} \quad x \neq a \end{cases} \tag{4.10}$$

Proof 4. *It is similar to the proof of Theorem 2.*

Now we can use Equation (4.6) if a and b are given and Equation (4.8) if the maximum point a is given.

4.3 Construction of the function

Because the aggregation has a neutral value, we have to transform the interval into $[0,\nu]$ or $[\nu,1]$. We will define positive and negative effects using the distending interval. That is,

$$P_{a_1,a_2}^{\lambda_1,\lambda_2}(x) = \frac{1}{2}\left(1+\gamma\sigma_{a,b}^{\lambda_1,\lambda_2}(x)\right) \tag{4.11}$$

$$N_{a_1,a_2}^{\lambda_1,\lambda_2}(x) = \frac{1}{2}\left(1-\gamma\sigma_{a,b}^{\lambda_1,\lambda_2}(x)\right), \tag{4.12}$$

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

Equations (4.11) and (4.12) have a common form if $\gamma \in [-1,1]$; namely,

$$E_{a_1,a_2}^{\lambda_1,\lambda_2}(\gamma,x) = \frac{1}{2}\left(1+\gamma\sigma_{a,b}^{\lambda_1,\lambda_2}(x)\right) \tag{4.13}$$

Here, if $\gamma > 0$ then we have a positive effects and if $\gamma < 0$ we have negative effects.

If the functions belong to the integrable function in the Riemannian sense, then there exist upper or lower approximations of rectangles.
We will use this fact in the next theorem.

Theorem 6. *Let $f(x)$ be an integrable function in the Riemannian sense, and let $a_1 < a_2 \ldots < a_n$ be a discretisation of the interval of the domain of the approximated function and let*

$$G(x) = \sum_{i=1}^{m} y_i \delta_{a_i,a_{i+1}}^{\lambda_i,\lambda_{i+1}}(x). \tag{4.14}$$

Then $\int \| f(x) - G(x) \| \to 0$ if $max \|a_{i+1} - a_i\| \to 0$ and $\lambda_i \to \infty$.

See Figure (4.5).

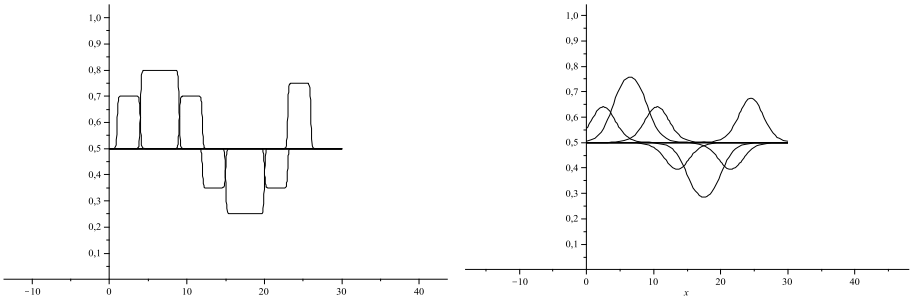

Figure 4.5: Rectangles without constructing an approximation, $\lambda = 16$ and $\lambda = 1$

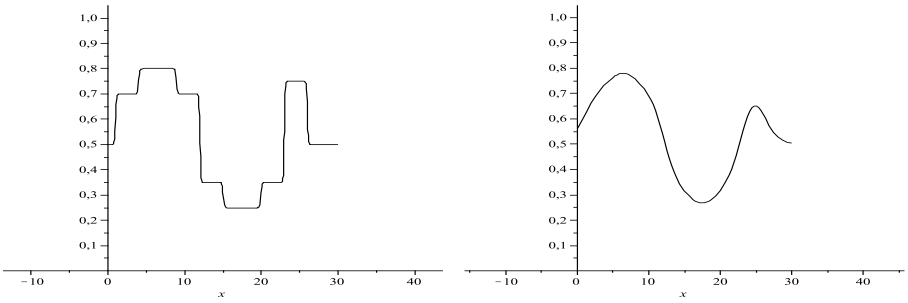

Figure 4.6: Rectangles after constructing an approximation if $\lambda = 16$ and $\lambda = 1$

Because $\delta_{a,b}(x)$ is a rectangle and the aggregation of the rectangles are rectangles, we can define an interval where $0 < a_1 < a_2 \ldots < a_n < 1$. The discretisation of an interval rectangle

approximation will be sufficiently good if the a_i, a_j intervals are small enough. So our method can use any function that is integrable in the Riemannian sense.

We can use the impulse function to interpolate the function.

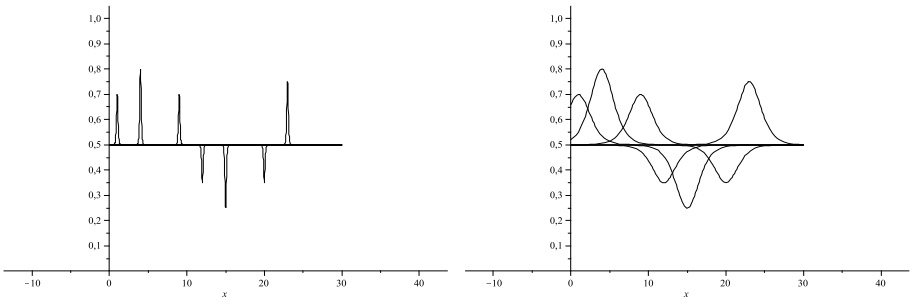

Figure 4.7: Interpolative approximation without aggregation if $\lambda = 16$ and $\lambda = 1$

If λ_1, λ_2 are not too large, then we can get a smooth approximation.

Similarly, Theorem 5 is valid if we use the impulse function

$$H(x) = \sum_{i=1}^{m} y_i \delta_{a_i}^{\lambda_1,\lambda_2}(x) \tag{4.15}$$

In Figure (4.6) we show the rectangle approximation of a function, where $\lambda = 16$ and $\lambda = 1$, while in Figure (4.7) we show the interpolation when $\lambda = 16$ and $\lambda = 1$.

4.4 Function decomposition

In the previous section we saw that we could construct a desired function using the aggregation operator and functions that model the effects. When applying it, the reverse case may sometimes be helpful too. That is, if the function is given, we need to decompose it into positive and negative effects. We will show that we can do this using an optimization method. We can find a wide variety of optimization techniques. If the initial values are properly chosen, it is not hard to get the global minimum by using a local search algorithm. Here, we will apply the well-known BFGS method [68]. This is one of the hill-climbing optimization techniques that look for the stationary point of a function where the gradient is zero. Because we can define initial points that are not far away from the optimum, the BFGS method should be able to find the optimal solution within a couple of iterations.

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.

The global procedure seeks to find all the effects simultaneously.

Now let we choose a function $F : R \to [0,1]$ to be approximated. Our task is to decompose it into effects. This can be done by our distending function (approximation) or impulse function (interpolation) procedures. First, the usual step is to smooth the function $F(x)$.

4.4.1 Algorithm for using the distending function

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

$$a_1 = c_1 - \frac{c_1 + c_2}{2}, \quad b_1 = \frac{c_1 + c_2}{2},$$
$$a_2 = \frac{c_1 + c_2}{2}, \quad b_2 = \frac{c_2 + c_3}{2} \ldots$$
$$a_n = \frac{c_{n-1} + c_n}{2}, \quad b_n = c_n + \frac{c_{n-1} + c_n}{2},$$

where

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

We will suppose that there is a maximum or minimum value like that shown in Figure (4.8) below.

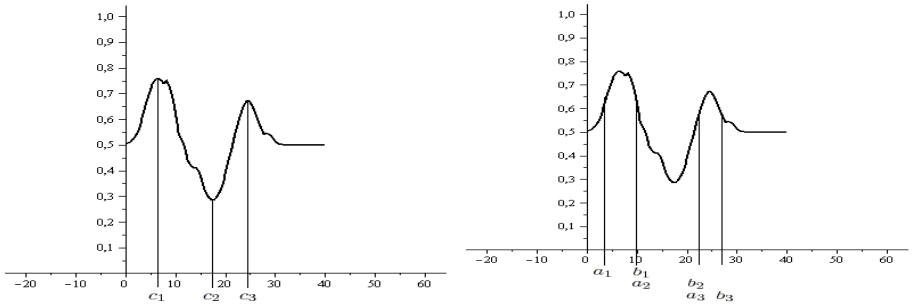

Figure 4.8: Extreme values and intervals for the sample function

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 use equations 4.14 and 4.15 to 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 (4.9).

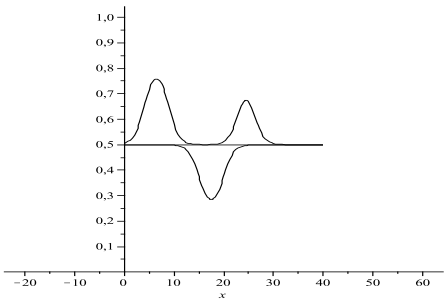

Figure 4.9: Optimal components

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 (4.10).

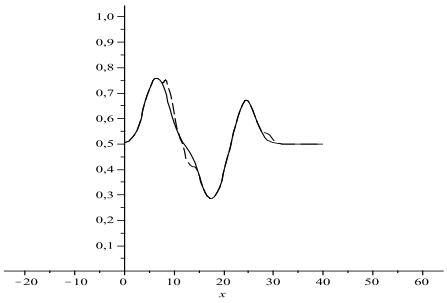

Figure 4.10: The function and its approximation

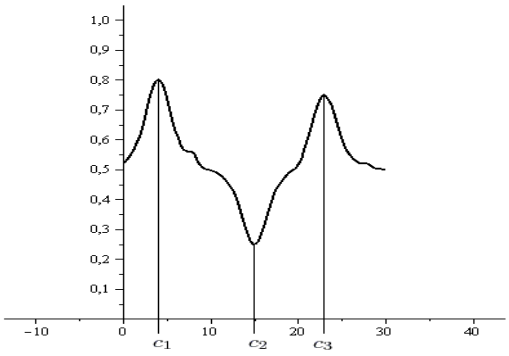

Figure 4.11: Extreme values of the function

4.4.2 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. (See Figure (4.11)).

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 (4.12).

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

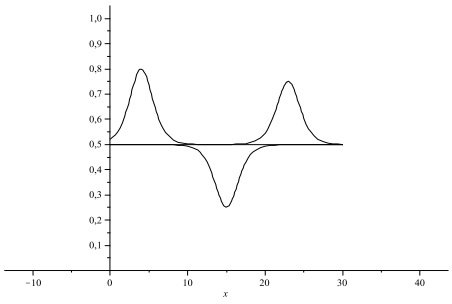

Figure 4.12: Main and optimal effects for the interpolative case

In Figure (4.13) we plotted the results of applying this procedure.

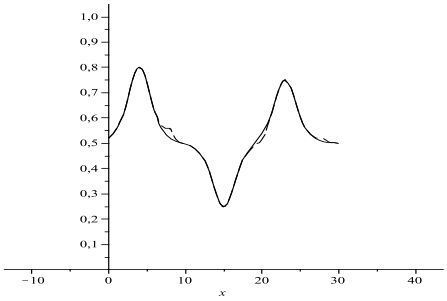

Figure 4.13: The function and its interpolative approximation

4.5 Summary

Here, we developed a new type of non-linear regression method that is based on the distending function and provides a natural description of the function. Our algorithm used the BFGS method to get accurate effects. Then we showed that this procedure is effective if all the data points given are based on the distending function. We found that this method is fast (only a few iteration steps are required for the optimisation method) and easy to use. With this technique, it is possible to change only a part of the function, instead of the usual case where we cannot.

Chapter 5

Cognitive systems

5.1 Introduction

When we have to deal with a sophisticated system, we are confronted by certain difficulties as we have to represent it as a dynamic system. Using a dynamic system model can be hard computationally. In addition, representing a system with a mathematical model may be difficult, or even impossible. Developing a model requires effort and specialized knowledge. Usually a system involves complicated causal chains, which might be non-linear. It should also be mentioned that numerical data may be hard to obtain, or it may contain certain errors, noise and incomplete values. Our approach seeks to overcome the above-mentioned difficulties. It is a qualitative approach where it is sufficient to have a rough description of the system and deep expert knowledge is not necessary. A similar approach was proposed by Kosko [50, 52, 51], and it is called the Fuzzy Cognitive Map (FCM). 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 5.1 we can see a typical FCM model, which is a directed graph.

FCM allows us to perform qualitative simulations and experiment with a dynamic model. It has better properties than expert systems or neural networks since it is relatively easy to use, it represents structured knowledge and inferences can be computed by numeric matrix operations instead of applying rules. Here we will use another method, (which is a modification of the FCM concept) which better matches real world modeling and it is called Pliant Cognitive Maps [44, 42]. We use cognitive maps to represent knowledge and to model decision making, which was first introduced by Axelrod [8]. Kosko used fuzzy values and matrix multiplication to calculate the next state of a system. Here instead of values, we use time dependent functions

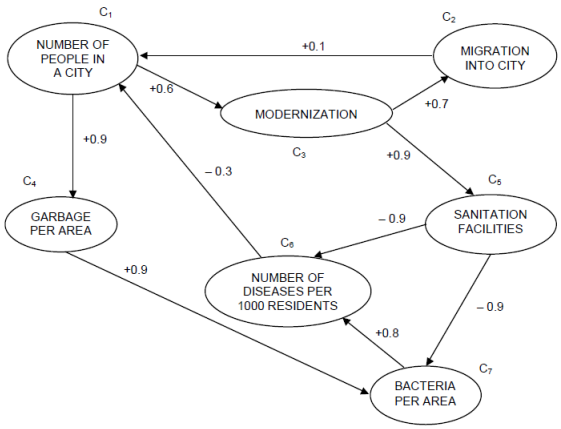

Figure 5.1: The FCM model

that are similar to impulse functions that represent positive and negative influences. Another improvement is that we can drop the concept of matrix multiplication. On the one hand, matrix multiplication is not well-suited in continuous logic (or fuzzy logic), where the truth value is one and the false value is zero. On the other hand, general operators are more efficient for calculating the next step of a simulation. Logic and the cognitive map model correspond to each other in the PCM case. It is easier to construct a PCM and after we have run PCM simulations and compared them with the real world, extracting knowledge is much easier. Combining cognitive maps with logic helps us to extract knowledge more efficiently in contrast to those that use rule-based systems. The standard knowledge representation in expert systems is achieved through a decision tree. This form of knowledge representation in most cases cannot model the dynamic behaviour of the real world. The cognitive map describes the whole system by a graph showing the cause-effects that connect concepts. It is a directed graph with feedback that describes the real-world concepts and the casual influences between them. From a logic point of view, causal concepts are unary operators of a continuous-valued logic containing negation operators in the case of inhibition effects. The value of the node reflects the degree of system activity at any given time. Concept values are expressed on a normal [0,1] range. Values do not denote exact quantities, but the degree of activation. The inverse of the normalization might express the values coming from the real world; i.e. using a sigmoid function. Unlike Fuzzy Cognitive Map, we do not use thresholds to force it to take values between zero and one. The mapping is a variation of the "fuzzification" process in fuzzy logic, and it always hinders our desire to get quantitative results. In Pliant logic we map the real world into the logical model. These maps are continuous, strictly monotonous increasing functions, and so the inverse of these functions yields data about the real world.

5.2 Pliant Cognitive Maps

In the FCM, the causal relationship is expressed by either positive or negative functions having different weights. As we mentioned earlier, this will be replaced by unary operators in the PCM. Let $\{C_1, \ldots, C_m\}$ be a set of concepts. Define a directed graph over the concepts. A directed edge has a weight w_{ij} from concept C_i to concept C_j. The weight measures the influence of C_i on C_j, where

- 0.5 is the neutral value,
- 0 is maximum negative and
- 1 is maximal positive influence or causality.

In the FCM, the weight value $w_{ij} \in [-1, 0, 1]$. In our case,

- $w_{ij} > 0.5$ means there is a direct (positive) causal relationship between concepts C_i and C_j. That is, the increase (decrease) in the value of C_i leads to an increase (decrease) in the value of C_j.
- $w_{ij} < 0.5$ means there is an inverse (negative) causal relationship between concepts C_i and C_j. That is, the increase (decrease) in the value of C_i leads to a decrease (increase) in the value of C_j.
- $w_{ij} = 0.5$ means there is no causal relationship between C_i and C_j.

During the simulation, the activation level a_i of concept C_i is calculated in an iterative way. In the FCM, the calculation rule was introduced to calculate the value of each concept based only on the influence of the interconnected concepts

$$A_i^t = f\left(\sum_{i \neq j} A_j^{t-1} \cdot W_{ji}\right),$$

where A_i^t is the value of concept C_i at time step t, A_j^{t-1} is the value of concept C_j at time step $t-1$, W_{ji} is the weight of the causal interconnection from concept jth toward concept ith and f is a threshold function. One of the most popular threshold functions is the sigmoid function, where $\lambda > 0$ determines the steepness of the continuous function f and squashes the content of the function in the interval [0,1]:

$$f(x) = \frac{1}{1 + e^{-\lambda x}}$$

. A more general FCM formula was proposed By Stylios et al. [72] to calculate the values of concepts at each time step. Namely,

$$A_i^t = f\left(k_1^i \sum_{i \neq j} A_j^{t-1} \cdot W_{ji} + k_2^i A_i^{t-1}\right)$$

The coefficients k_1^i and k_2^i must satisfy the conditions $0 \leq k_1^i \leq 1$ and $0 \leq k_2^i \leq 1$. The coefficient k_1^i expresses the influence of the interconnected concepts in the configuration of the new value of concept A_i. The coefficient k_2^i represents the proportion of the contribution of the previous value of the concept in the computation of the new value. The FCM has the advantage that we get a new state vector by multiplying the previous state vector a by the edge matrix W, which shows the effect of the change in the activation level of one concept on another. In the Pliant concept we aggregate the influences instead of summing up the values. The result always remains between 0 and 1, so we do not need normalization as an additional step. The aggregation in Pliant logic is a general operation that contains conjunctive operators and disjunctive operators as well. Depending on the parameter - called the neutral value - of the aggregation operator, we can build logical operators like Dombi operators. Using PCMs (Pliant Cognitive Maps) we can answer "what if" questions based on some initial scenario. For example, let A_i be the initial state vector. The new state is calculated repeatedly with the aggregation operator until the system converges to $\left|A_i^t - A_i^{t-1}\right| < \varepsilon$. We will get the resulting equilibrium vector, and this will provide a set of answers to our "what-if" questions. Our PCM can be used in any area covered by the FCM.

5.3 Components of the PCM

Now we will introduce the components of the Pliant Cognitive Maps.

5.3.1 Aggregator operator

Besides the logical operators constructed in fuzzy theory, a non-logical operator also appears. The reason for this is the insufficiency of using either conjunctive or disjunction operators for real-world situations [89]. The rational form of an aggregation operator is [20]:

$$a(x_1, \ldots, x_n) = \frac{1}{1 + \frac{1-\nu_0}{\nu_0} \cdot \left(\frac{\nu}{1-\nu}\right)^{\sum w_i - 1} \cdot \prod_{i=1}^{n} \left(\frac{1-x_i}{x_i}\right)^{w_i}}$$

We can model conjunctive and disjunctive operators with the aggregation operator. If v is close to 0, then the operation has a disjunctive characteristic; and if v is close to 1, then the

operation has a conjunctive characteristic. From this property it can be seen that by using the aggregation we have more possibilities than that got simply by using the sum function in the FCM. By altering the neutral values at the nodes, different operations can be performed.

5.3.2 Creating influences

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 5.2, we observe a basic influence like that mentioned in [44]. 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 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:

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

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

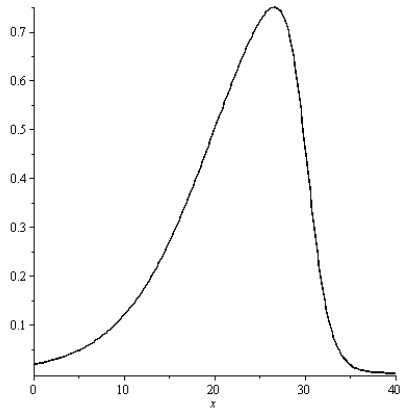

Figure 5.2: An average influence.

In Figure 5.3 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 netural value is 1/2. To

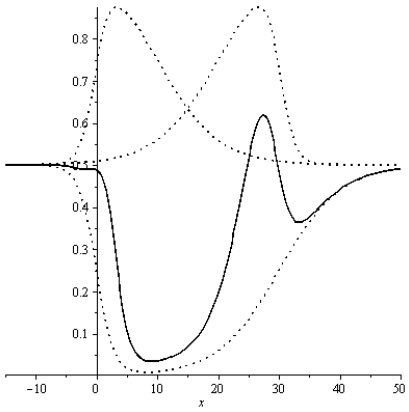

Figure 5.3: The aggregation of influences.

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 negative impulse. To create an effect we will utilize the following transformation functions:

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

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

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 slightes larger to 0. Here, we will use the $[0.15, 0.95]$ domain. In Figure 5.4, we can see main effects by just using sigmoid functions.

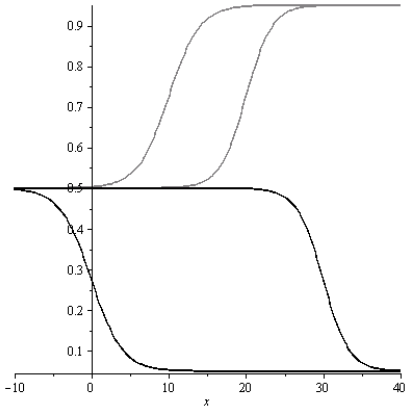

Figure 5.4: An average influence got by using sigmoids.

In Figure 5.5, we see the aggregation of positive and negative effects by just using sigmoid functions.

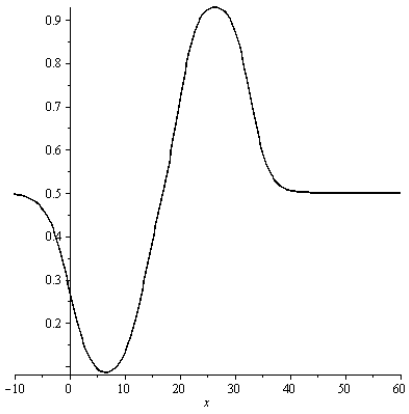

Figure 5.5: The aggregation of sigmoid influences.

5.4 Construction of the PCM

To simulate the system, all we have to do is to aggregate the influences. The aggregation operator is a guarantee that we will use influences in the right way. This means the requirements of the simulation is 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.

5.5 PCM Framework

Now we are ready to make a simulation test. For this, we developed a program to test the system [44]. First we will study fixed, predefined situations. These situations tell us that the system is very flexible and is easy to adapt to different situations. The simulation is based on directed graphs. The nodes are illustrated with squares. Between the nodes there are edges. Instead of

using arrows, we represent the direction of the edge by a filled circle. If the edge leads from the vertex v to vertex u, then we place the filled circle closer to u. In Figure 5.6, an example is given with two nodes and the direction between the nodes is from 2 to 1.

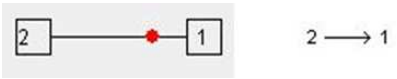

Figure 5.6: Graph representation.

There are two kinds of nodes:

- Input nodes (i-nodes): Here, no edges lead to the node. The index of the input node is not in the centre of the square.
- Inner nodes (inner nodes): Here all other types of nodes are inner nodes. The index of the node is in centre of the square.

In Figure 5.6, index 2 is an i-node and 1 is an inner node. There are two ways to add a new node. If the node is an inner node, then we set:

- The name of the node.
- The initial value, i.e. the expectation value (in our model it is the neutral value 0.5).
- The 2D coordinates (for visualization).

We can also provide a brief description of the node. See figures 5.7 and 5.8.

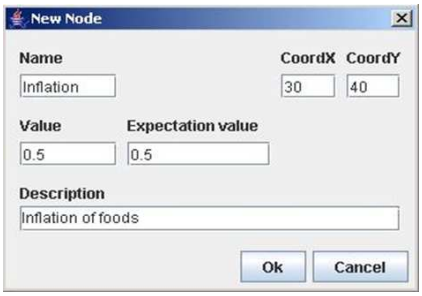

Figure 5.7: New Node dialogue box.

If the node is an i-node, then instead of giving the initial value we can provide input data. There are three ways of doing this. These are via

- a table, we can set input data by our self in every time period.

Figure 5.8: The result of adding new node.

- algebraic functions, (sin, cos, exp, sigmoid, etc.), calculate the selected function values.
- generating noise. We generate random values by a normal distribution between $[0.5-\varepsilon, 0.5+\varepsilon]$ (default: $\varepsilon = 0.1$s), which is simulated noise.

In figures 5.9 and 5.10 we see how the data can be entered.

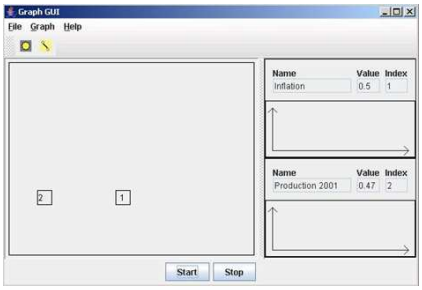

Figure 5.9: Input and Inner Node in the GUI.

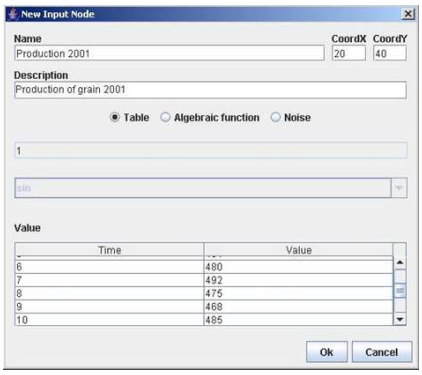

Figure 5.10: Input node dialogue box.

The input values are transformed into [0,1] with the sigmoid function where a is the basic value (the expectation level), and λ is the sharpness of the function. It is reasonable to set $\lambda = \frac{4}{x_{max}-x_{min}}$, because λ is the slope of the sigmoid function, where x_{max}/x_{min} is the largest / smallest value. The sigmoid transformation is necessary because our system is always works between zero and one.

The next step is to connect the nodes. To add a new edge, we define

- the index of the source node,
- the index of the destination node,
- the influence (positive or negative),
- the expectation value(ν).

See figures 5.11 and 5.12.

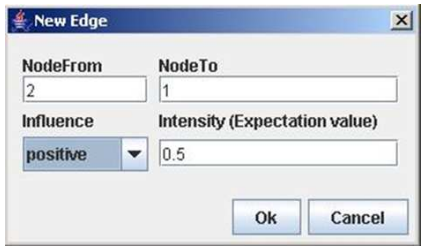

Figure 5.11: New edge dialogue box.

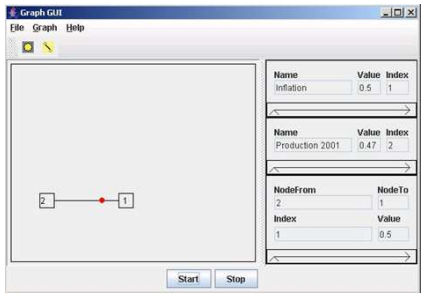

Figure 5.12: Input and inner node related with positive influences

Now we have defined the system. Next we can describe the simulation. In one cycle, the following calculations are performed:

1.
   - We find the source of the edge.
   - We transform the source value by the intensity:

$$f_{edge} = \frac{1}{1 + \frac{\nu}{1-\nu}\frac{1-node(value)}{node(value)}}$$

   - We calculate the edge influence using a sigmoid function:

$$f_{newedge} = \frac{1}{1 + e^{-4*\lambda(x-0.5)}},$$

   where ν is the edge expectation value.

If $\lambda = 1$, the influence is positive. If $\lambda = -1$, the influence is negative. We use the sigmoid function, because in the real world the influences never reach extreme values, i.e. they always lie between zero and one.

2. Calculate the new value of the nodes.

   - We collect the influences that lead to the node.
   - We transform the influences and then multiply them by $f_{inftr} = \prod \frac{1-f_{inf}(value)}{f_{inf}(value)}$
   - We use the following function to get the actual value: $f_{nodenew}(value) = \frac{1}{1+f_{inftr}\frac{1-node(value)}{node(value)}}$, which is an aggregation.

3. Set the actual value of i-node.

5.5.1 Validation of the PCM concept and the Framework

First we will check whether our PCM concept fulfils the basic properties.

1. We have two input nodes with the same ν value. These two input nodes have one common inner node. The input nodes always have the same value, and on their edge they have an opposite influence; i.e. one is positive and the other is negative. The result of the aggregation should be the neutral value (0.5). See Figures 5.13 - 5.16.

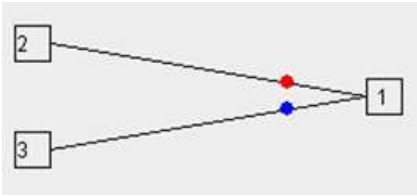

Figure 5.13: Graph representation of the simulation.

Figure 5.14: Input node values.

2. 2. The configuration here is the same as in the first experiment. Here, we change only the ν values. We can see the effect of the new value. See Figures 5.17 - 5.20.

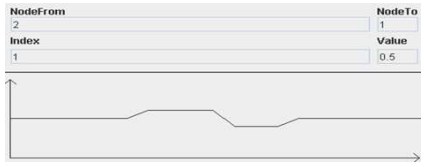

Figure 5.15: Positive influence.

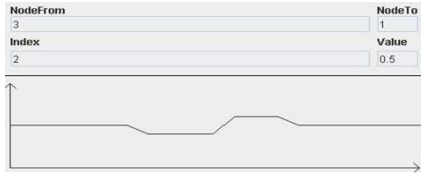

Figure 5.16: Negative influence.

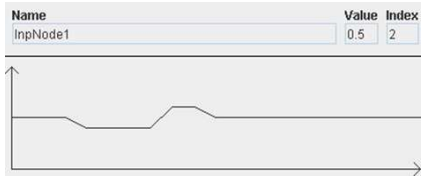

Figure 5.17: The input values (the minimum value is 0.4, and the maximum value is 0.6).

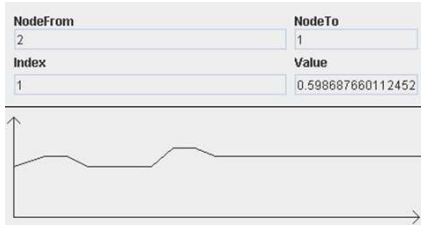

Figure 5.18: Positive influences when $\nu = 0.4$

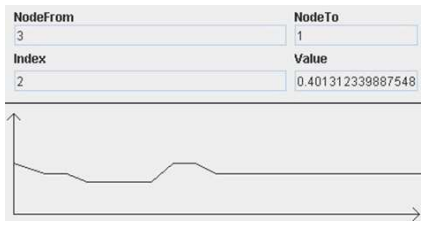

Figure 5.19: Positive influences when $\nu = 0.6$

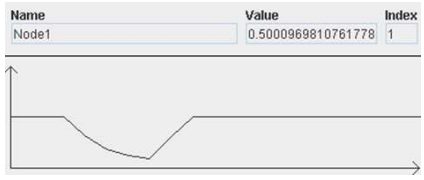

Figure 5.20: The result of aggregating the influences.

3. Now we have two positive influences with the same neutral values, but the input values are just the opposite (input2(value) = 1-input1(value)) of each other. The result should also be a neutral value. See Figures 5.21 - 5.23. We also repeat this test using the functions $sin(x)$ and $cos(x+\frac{\pi}{2})$. See Figure 5.24 and 5.25.

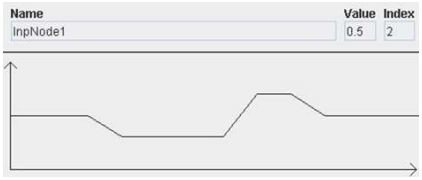

Figure 5.21: Input node1 values (the minimum value is 0.4 and the maximum value is 0.6)

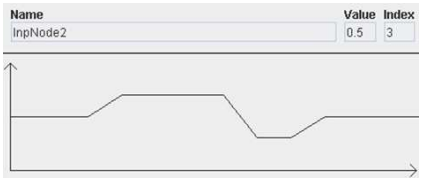

Figure 5.22: Input node2 values (the minimum value is 0.4 and the maximum value is 0.6)

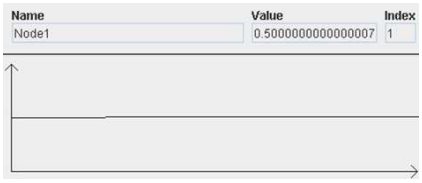

Figure 5.23: The result of the aggregating two positive influences.

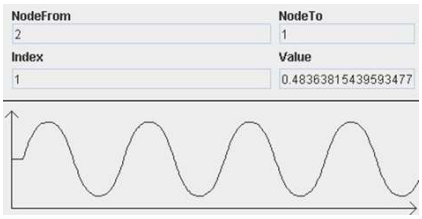

Figure 5.24: Positive influences of $sin(x)$ function.

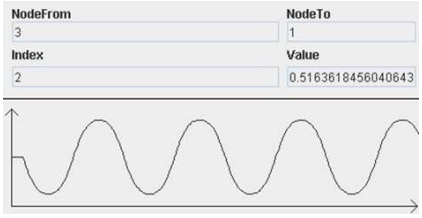

Figure 5.25: Positive influences of the $cos(x+\frac{\pi}{2})$ function.

4. In this test we have a small complex system with three inputs and two inner nodes. Two input nodes generate noise of different intensity, while the third one is a periodic function. See Figure 5.26.

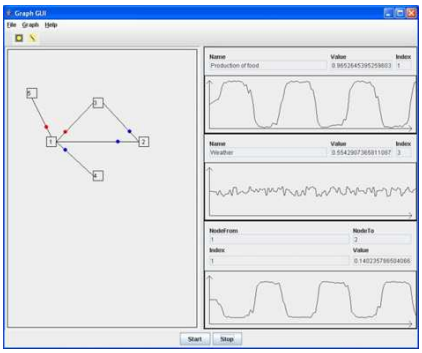

Figure 5.26: Small complex system.

5.6 Heat exchanger applications

A heat exchanger is a standard device in the chemical and process industry [58]. 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 non-linear behaviour, strongly depends on the flow rates and on the temperature of the medium. A cross-flow water/air heat exchanger is considered, which is subject to immeasurable or non-modeled disturbances that require the use of knowledge-based techniques. In this problem our task is to construct behavioural model for the heat exchanger system, which will control the water outlet temperature by modifying the flow rate of the air.

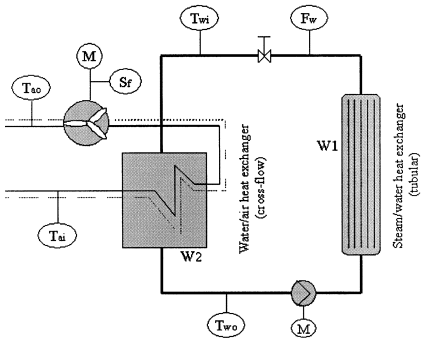

Figure 5.27: Typical heat exchanger system.

In Figure 5.27, we have a typical system setup. It is well known that the FCM can be used to model and control the heat exchanger process [72]. In most process industries, the thermal plant comprises two heat exchangers, but in our example (see in Figure 5.2) we just

have the secondary circuit. The system contains two circuits W_1 nad W_2. Here W_1 is a circuit that is a tubular steam/water heat exchanger, while W_2 is the cross-flow water/air exchanger. The water in the given circuit is heated by means of W_1. On the left hand side of the circuit, the water is cooled in the cross-flow water/air heat exchanger W_2. A fan sucks in cold air from the environment (temperature T_{ai}). After passing the heat exchanger and the fan, the air is blown out back into the environment. The water temperature T_{wo} is controlled by varying the fan speed S_f. The control variable T_{wo} depends on the manipulated variable S_f and the measurable disturbances: inlet water temperature T_{wi}, air temperature T_{ai} and water flow rate F_w. In most systems, the water flow rate is usually regulated by a PI-controlled pneumatic valve, which strongly influences the behaviour of the heat exchanger W_2 and it is a major challenge to design a temperature controller for T_{wo} when the flow rates vary over a wide range [11, 31]. The operators of the heat exchanger gather oprating data that can be used to build a model. To develop a Cognitive system we have to determine the concepts. Here, concepts represent the physical input and output variables of the process[72]. Experts define five concepts for this situation:

- Concept1: The fan speed S_f, which is the manipulated variable.
- Concept2: The water flow rate F_w.
- Concept3: The water inlet temperature T_{wi}.
- Concept4: The air inlet temperature T_{ai}. The environmental temperature cannot be altered as it depends on the weather and season.
- Concept5: The water outlet temperature T_{wo}, which is the output of the model.

In the next step the causal interconnections between any two concepts have to be determined. Experts can describe the relation between concepts according to the system. The connections between concepts are

- Linkage1: It connects concept1 (fan speed S_f) with concept5 (water outlet temperature T_{wo}). When the value of S_f increases, the value of T_{wo} decreases.
- Linkage2: It connects concept2 (flow rate F_w) with concept5 (water outlet temperature T_{wo}). When the value of F_w increases, the value of T_{wo} increases.
- Linkage3: It connects concept2 (flow rate F_w) with concept1 (fan speed S_f). When the value of F_w increases, the value of S_f increases.

- Linkage4: It connects concept3 (water inlet temperature T_{wi}) with concept5 (water outlet temperature T_{wo}). When the value of T_{wi} increases, the value of T_{wo} increases.
- Linkage5: It connects concept3 (water inlet temperature T_{wi}) with concept1 (fan speed S_f). When the value of T_{wi} increases, the value of S_f increases.
- Linkage6: It connects concept3 (water inlet temperature T_{wi}) with concept2 (flow rate F_w). When the value of T_{wi} increases, the value of F_w decreases.
- Linkage7: It connects concept4 (air inlet temperature T_{ai}) with concept5 (water outlet temperature T_{wo}). When the value of T_{ai} increases, the value of T_{wo} increases.
- Linkage8: It connects concept4 (air inlet temperature T_{ai}) with concept1 (fan speed S_f). When the value of T_{ai} increases, the value of S_f decreases.
- Linkage9: It connects concept5 (water outlet temperature T_{wo}) with concept2 (flow rate F_w). When the value of T_{wo} increases, the value of F_w decreases.
- Linkage10: It connects concept5 (water outlet temperature T_{wo}) with concept1 (fan speed S_f). When the value of T_{wo} increases, the value of S_f increases.

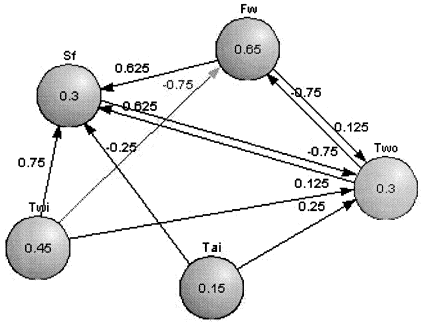

Figure 5.28: FCM model for heat exhanger system.

Figure 5.28 shows the system that describes models and controls the heat exchanger system. The FCM model for the heat exchanger is in accordance with the models and experiments described in [58, 31]. It is also possible to create an influence matrix of the system like that given in Table 5.1.

To simulate the real environment in the FCM, the values of concepts correspond to real measurements that have been transformed to the interval [0,1]. The corresponding mechanism is needed that will transform the measures of the system to their representative values of concepts in the FCM model and vice versa. The initial measurements of the heat exchanger system have been transformed to concept values and the initial vector of the FCM

Table 5.1: The weighted matrix of the model.

S_f	F_w	T_{wi}	T_{ai}	T_{wo}
0	0.625	0.75	-0.25	0.625
0	0	-0.75	0	-0.75
0	0	0	0	0
0	0	0	0	0
-0.75	0.125	0.25	-0.75	0

is the $A_0 = [0.3\ 0.65\ 0.45\ 0.15\ 0.3]$. In Figure 5.28, the initial value of each concept and the interconnections with their weights have also been included. For these initial values of concepts, the FCM starts to simulate the behaviour of the process. In the FCM domain, a running step is defined as the time step during which the values of the concepts are calculated. The value of each concept is defined by taking all the causal linkage weights pointing to this concept and multiplying each weight by the value of the concept that causes the linkage, and adding the last value of each concept. Afterwards, the sigmoid function with $\lambda = 1$ is applied and hence the result falls in the range [0,1]. After performing the simulation we got the following results:

Table 5.2: Simulation results using the FCM

Iteration	S_f	F_w	T_{wo}	Improvement:
1	0.77	0.52	0.56	0.8577
2	0.85	0.44	0.54	0.1906
3	0.85	0.43	0.51	0.0419
4	0.85	0.43	0.5	0.0107
5	0.85	0.43	0.5	0.0036

This table doesn't contain input node values where the values are same all the time. Evaluating the results we see that the fan speed S_f has increased, the value of flow rate F_w has decreased and after the third step, the water outlet temperature T_{wo} falls below 0.50. We also observe that the values between the two simulation steps are decreasing, but this decrease is not uniform, which is not as good as we initially expected. These concepts control physical devices, so ideally we should change values in a smooth way.

5.6.1 Evaluate with the PCM

Our method works on real measurements, which means that we do not need to transform it to real value between [0,1]. In our model we use the same concepts and relations, and the initial values of the concepts are the same as before. So first of all to evaluate the our method, we need to identify the range of each concept parameters. Because previous articles do not mention these values, we decided to use the following values:

Table 5.3: The range values of the concept.

Concept	Minimum	Maximum	Default
S_f	100	500	250
F_w	2	20	6
T_{wi}	20	50	30
T_{ai}	18	35	24
T_{wo}	20	40	30

The default value is used to specify the real values of the first step. With these range values we can define a sigmoid function that can be used for the calculation. For example the initial value of the S_f is 0.3, and we will use the following sigmoid function:

$$S_f(x) = \frac{1}{1+e^{-\frac{4}{500}(x-250)}}$$

In this case, the real value of S_f should be 144.08. With these calculations, we can compute the initial values of the concepts: This method also shows that with each simulation step it is

Table 5.4: The initial values of the concepts.

Sf	Fw	Twi	Tai	Two
144.08	3.7	27.5	8.8	21.52

easy to obtain the real value. In the classical FCM method, the influence does not change during the simulation. In order to compare it with our method, we will also define a constant influence. Hence in a simulation step we will calculate the new concept value in the following way. For each node we will create a set that contains all incoming nodes. For each node in the set we will apply the following expression to calculate the strength of the incoming node:

$$\left(\frac{1-x_i}{x_i}\right)^{w_{ij}},$$

where x_i is the actual value of the node and w_{ij} is the value of the influence between concept C_i and C_j, for a given i and j. After, we can calculate each node in the set then we will use the

aggregation operator to calculate the new value of the concept. For example, the new value of the F_w is calculated by using the following expression:

$$\frac{1}{1+\left(\frac{1-0.45}{0.45}\right)^{-0.75}\left(\frac{1-0.3}{0.3}\right)^{-0.75}}$$

Now we can run the simulation until it exceeds a given limit. The following table shows the results of our simulation.

Evaluating the results, we can see that the results are different from those got by applying the FCM method. In Figure 5.29 we can see how the parameter value varies.

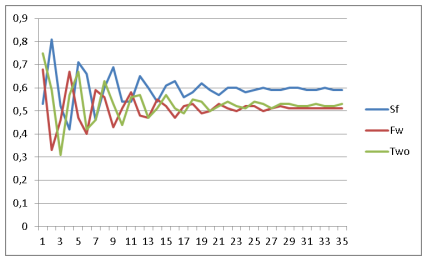

Figure 5.29: Results of the PCM model.

The fan speed S_f has decreased, the value of flow rate F_w has decreased, the water outlet temperature T_wo now has a value below 0.50. We can also see that the value changes between two simulation steps is decreasing (see Figure 5.30), but the decrease is smooth, and this is why it requires more simulation steps to model it.

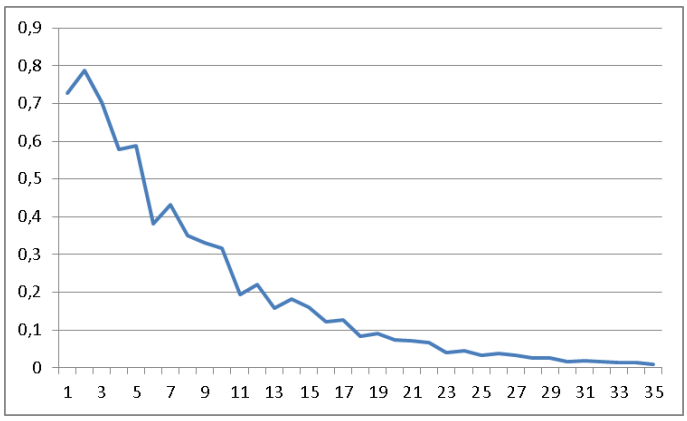

Figure 5.30: Sum of value change in each step.

5.7 Summary

In this chapter we used numerical methods to model complex systems based on positive and negative influences. This concept is similar to the FCM, but the functions and the aggregation

Table 5.5: Simulation results of the PCM.

Iteration	S_f	F_w	T_{wo}	Improvement:
1	0.53	0.68	0.75	0.7262
2	0.81	0.33	0.59	0.7870
3	0.52	0.46	0.31	0.7024
4	0.42	0.67	0.57	0.5787
5	0.71	0.47	0.67	0.5881
6	0.66	0.40	0.42	0.3810
7	0.46	0.59	0.46	0.4327
8	0.60	0.56	0.63	0.3492
9	0.69	0.43	0.53	0.3310
10	0.54	0.51	0.44	0.3176
11	0.54	0.58	0.56	0.1951
12	0.65	0.48	0.57	0.2203
13	0.60	0.47	0.47	0.1590
14	0.54	0.55	0.51	0.1829
15	0.61	0.52	0.57	0.1602
16	0.63	0.47	0.51	0.1224
17	0.56	0.52	0.49	0.1266
18	0.58	0.53	0.55	0.0833
19	0.62	0.49	0.54	0.0908
20	0.59	0.50	0.50	0.0734
21	0.57	0.53	0.52	0.0712
22	0.60	0.51	0.54	0.0675
23	0.60	0.50	0.52	0.0405
24	0.58	0.52	0.51	0.0462
25	0.59	0.52	0.54	0.0337
26	0.60	0.50	0.53	0.0380
27	0.59	0.51	0.51	0.0336
28	0.59	0.52	0.53	0.0256
29	0.60	0.51	0.53	0.0264
30	0.60	0.51	0.52	0.0174
31	0.59	0.51	0.52	0.0191
32	0.59	0.51	0.53	0.0155
33	0.60	0.51	0.52	0.0148
34	0.59	0.51	0.52	0.0141
35	0.59	0.51	0.53	0.0084

procedures are quite different. It is based on a continuous-valued logic and all the parameters have a semantic meaning. We developed two different kinds of method to create effects. We

also described a framework that was developed by us. With this framework, we gave some basic examples that explained how Pliant Cognitive Maps work. Here, we showed that we can use this method in a real-world environment. The values between the simulation steps smoothly decrease, but it requires more simulation steps. In our example we used the same influence for each concept all the time, but it is possible to change the strength of the influence and mathematically model real-world situations better.

Bibliography

[1] Egee middleware technical website. Technical report. http://egee-technical.web.cern.ch/egee-technical.

[2] Open grid forum website. http://www.ogf.org, 1999.

[3] Parallel workloads archive website. http://www.cs.huji.ac.il/labs/parallel/workload, 2009.

[4] N.H. Abel. Untersuchungen der funktionen zweier unabhangigen veranderlichen grösen x und y wie $f(x,y)$, welche die eigenschaft haben, das $f(z, f(x,y))$ eine symmetrische funktion von x,y und z ist. *J. Reine Angew. Math.*, 1:11–15, 1826.

[5] J. Aczél. Sur les opérations definies pour des nombres réels. *Bull. Soc. Math. France*, 76:59–64, 1949.

[6] J. Aczél. *Lectures on functional equations and their applications*. Academic Press, New York, 1966.

[7] A. Anjomshoaa, F. Brisard, M. Drescher, D. Fellows, A. Ly, S. McGough, D. Pulsipher, and A. Savva. Job submission description language (jsdl) specification, version 1.0. Technical report, 2005. http://www.gridforum.org/documents/GFD.56.pdf.

[8] Robert Axelrod. *Structured of Decision: the cognitive maps of political elites*. Princeton University press, New Jersey, 1976.

[9] J.F. Baldwin. A new approach to approximate reasoning using a fuzzy logic. *Fuzzy Sets and Systems*, 2:309–325, 1979.

[10] M. Bergmann, J. Moor, and J. Nelson. *Logic book*. McGraw-Hill, New York, 1990.

[11] S. Bittanti and L. Piroddi. Nonlinear identification and control of a heat exchanger: a neural network approach. *J. Franklin Inst.*, pages 135–153, 1997.

[12] Rajkumar Buyya, Manzur Murshed, and David Abramson. Gridsim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing. In *Journal of Concurrency and Computation: Practice and Experience (CCPE*, pages 1175–1220. Wiley Press, 2002.

[13] Tomasa Calvo and Radko Mesiar. Weighted triangular norms-based aggregation operators. *Fuzzy Sets Syst.*, 137(1):3–10, July 2003.

[14] A.H. Clifford. Naturally totally ordered commutative semigroups. *Amer. J. Math.*, 76:631–646, 1954.

[15] A.H. Clifford and G.B. Preston. The algebraic theory of semigroups. *Amer. Math. Soc.*, 1, 1961.

[16] A.C. Climescu. Sur l'équation fonctionelle de l'associativité. *Bull.école Polytechnique Iassy*, 1:1–16, 1946.

[17] M. De Cock and E. E. Kerre. Fuzzy modifiers based on fuzzy relations. *Information Sciences*, 160:173–199, 2004.

[18] E. Cox. *The fuzzy systems handbook: a practitioner's guide to building, using, and maintaining fuzzy systems.* Academic Press Professional, 1994.

[19] A.B. Paalman de Miranda. Topological semigroups. *Mathematical Centre Tracts*, 11, 1964.

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

[21] 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.

[22] J. Dombi. Towards a general class of operators for fuzzy systems. *IEEE Transaction on Fuzzy Systems*, 16:477–484, 2008.

[23] J. Dombi. Demorgan systems with infinite number of negation. 2010.

[24] J. Dombi. De morgan systems with an infinitely many negations in the strict monotone operator case. *Information Sciences*, 181:1440–1453, 2011.

[25] J. Dombi. On a certain class of aggregation operators. *Information Sciences*, pages 2038–2044, 2011.

[26] Jozsef Dombi. Basic concept for a theory of evaluation: the aggregation operator. *Europian Journal of Operations Research*, 10, 1982.

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

[28] József Dániel Dombi and József Dombi. Creating and decomposing functions using fuzzy functions. In Joaquim Filipe and José Cordeiro, editors, *ICEIS (2)*, pages 400–403. SciTePress, 2010.

[29] József Dániel Dombi and Attila Kertész-Farkas. Scheduling solution for grid meta-brokering using the pliant system. In Joaquim Filipe, Ana L. N. Fred, and Bernadette Sharp, editors, *ICAART (2)*, pages 46–53. INSTICC Press, 2010.

[30] R. Mesiar E.P. Klement and E. Pap. *Triangular norms*. Kluwer, 2000.

[31] E.-J. Ernst and O. Hecker. Predictive control of a heat exchanger. *Proc. Seminar Theory Applications Model Based Predictive Control*, pages 1–18, 1996.

[32] D. W. Erwin and D. F. Snelling. *UNICORE: A Grid Computing Environment*. Springer, 2001.

[33] W.M. Faucett. Compact semigroups irreducibly connected between two idempotents. *Proc. Amer. Math. Soc.*, 6:741–747, 1955.

[34] I. Foster and C. Kesselman. The globus project: A status report. *Heterogeneous Computing Workshop*, pages 4–18, 1998.

[35] H.J. Frank. On the simultaneous associativity of $f(x,y)$ and $x+y-f(x,y)$. *Aequat.Math.*, 19:192–226, 1979.

[36] D. Ciucci G. Cattaneo and D. Dubois. Algebraic models of deviant modal operators based on demorgan and kleene lattices. *Information Sciences*, 181:4075–4100, 2011.

[37] S. Horikawa, T. Furuhashi, and Y. Uchikawa. A new type of fuzzy neural network based on a truth space approach for automatic acquisition of fuzzy rules with linguistic hedges. *International Journal of Approximate Reasoning*, 13:249–268, 1995.

[38] Fred Howell and Ross Mcnab. Simjava: A discrete event simulation library for java. In *Proc. of the International Conference on Web-Based Modeling and Simulation*, pages 51–56, 1998.

[39] S.-K. Hu and Z.-F. Li. The structure of continuous uni-norms. *Fuzzy Sets and Systems*, 124:43–52, 2001.

[40] V. N. Huynh, T.B. Ho, and Y. Nakamori. A parametric representation of linguistic hedges in zadeh's fuzzy logic. *International Journal of Approximate Reasoning*, 30:203–223, 2002.

[41] Epema D.H.J. Tannenbaum T. Farrellee M. Livny M. Iosup, A. Inter-operating grids through delegated matchmaking. *International Conference for High Performance Computing, Networking, Storage and Analysis (SC07)*, 2007.

[42] J. D. Dombi J. Dombi. Pliant cognitive map using conjunctive operator. *Proceedings of IEEE International Workshop on Soft Computing Applications*, pages 11–16, 2005.

[43] 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.

[44] Jozsef Daniel Dombi Jozsef Dombi. Cognitive maps based on pliant logic. *16TH EUROPEAN SIMULATION Symposium*, pages 63–69, 2005.

[45] József Dániel Dombi József Dombi. Function decomposition using distending function. pages 191–196, 2009.

[46] A. Kertesz and P. Kacsuk. Meta-broker for future generation grids: A new approach for a high-level interoperable resource management. In *Grid Middleware and Services Challenges and Solutions*, pages 53–63. Springer US, 2008.

[47] Attila Kertesz, Jozsef Daniel Dombi, and Jozsef Dombi. Adaptive scheduling solution for grid meta-brokering. *Acta Cybernetica*, 19(1):105–123, 2009.

[48] Carl Kesselman and Ian Foster. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers, November 1998.

[49] Donald E. Knuth. *The art of computer programming, Volume 2 (3rd ed.): seminumerical algorithms*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1997.

[50] Bart Kosko. Fuzzy entropy and conditioning. *Information Sciences*, 40:165–174, 1986.

[51] Bart Kosko. *Neural Networks and Fuzzy Systems*. Prentice Hall, Englewood Cliffs, 1992.

[52] Bart Kosko. *Fuzzy Thinking*. Flamingo, 1994. hardcopy.

[53] Huedo E. Llorente I.M. Leal, K. *A decentralized model for scheduling independent tasks in Federated Grids.*, volume 25. Future Generation Computer Systems, 2009.

[54] Y.-M. Li and Z.-K. Shi. Weak uninorm aggregation operators. *Information Sciences*, 124:317–323, 2000.

[55] Yong Ming Li and Zong Ke Shi. Remarks on uninorms aggregation operators. *Fuzzy Sets and Systems*, 114:377–380, 2000.

[56] C.H. Ling. Representation of associative functions and statistical triangle inequalities. *Publ. Math. Debrecen*, 12:189–212, 1965.

[57] B. D. Liu, C. Y. Chen, and J. Y. Tsao. Design of adaptive fuzzy logic controller based on linguistic-hedge concepts and genetic algorithms. *IEEE Transactions on Systems Man and Cybernetics, Part B*, 31(1):32–53, 2001.

[58] 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.

[59] G. Mayor M. Mas and J. Torrens. The distributivity condition for uninorms and t-operators. *Fuzzy Sets and Systems*, 128:209–225, 2002.

[60] K. Menger. Statistical metrics. *Proc. Nat. Acad. Sci.*, 28:535–537, 1942.

[61] M. Monserrat and J.Torrens. On the reversibility of uninorms and t-operators. *Fuzzy Sets and Systems*, 131:303–314, 2002.

[62] P.S. Mostert and A.L.Shields. On the structure of semigroups on a compact manifold with boundary. *Ann. Math.*, 65:117–143, 1957.

[63] E. Orlowska. A logic of indiscernibility relations. *LNCS, Springer-Verlag, Berlin*, 208:177–186, 1985.

[64] E. Orlowska. Kripke semantics for knowledge representation logics. *Studia Logica*, 49:255–272, 1990.

[65] E. Pap. *Null-Additive Set Functions*. Kluwer and Ister Science, 1955.

[66] Guim F. Corbalan J. Fong L.L. Liu Y.G. Sadjadi S.M. Rodero, I. Looking for an evolution of grid scheduling: Meta-brokering. *Coregrid Workshop in Grid Middleware'07*, 2008.

[67] S.H. Rubin. Computing with words. *IEEE Transactions on Systems Man and Cybernetics, Part B*, 29(4):518–524, 1999.

[68] Andrzej Ruszczyński. *Nonlinear Optimization*. Princeton University Press, Princeton, New Jersey, 2006.

[69] B. Schweizer and A. Sklar. Espaces métriques aléatories. *Acad. Sci. Paris*, 247:2092–2094, 1958.

[70] B. Schweizer and A. Sklar. Statistical metric spaces. *Pacific J. Math*, 10:313–333, 1960.

[71] B. Schweizer and A. Sklar. *Probabilistic Metric Spaces*. North Holland, 1983.

[72] 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.

[73] B.D. Baets T. Calvo and J. Fodor. The functional equations of frank and alsina for uninorms and nullnorms. *Fuzzy Sets and Systems*, 120:385–394, 2001.

[74] E. Trillas. Sobre functiones de negacion en la teoria de conjuntas difusos. *Stochastica*, 3:47–60, 1979.

[75] I. B. Türksen. A foundation for cww: Meta-linguistic axioms. *IEEE Fuzzy Information*, Processing NAFIPS'04:395–400, 2004.

[76] A. Wald. On statistical generalization of metric spaces. *Proc. Nat. Acad. Sci. U.S.A.*, 29:196–197, 1943.

[77] Ronald R. Yager. Weighted triangular norms using generating functions. *International Journal of Intelligent Systems*, 19:217–231, 2004.

[78] R.R. Yager. Approximate reasoning as a basis for rule-based expert-systems. *IEEE Trans. Systems Man Cybernet*, SMC-14:636–643, 1984.

[79] R.R. Yager. Uninorms in fuzzy systems modeling. *Fuzzy Sets and Systems*, 122:167–175, 2001.

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

[81] L. A. Zadeh. Quantitative fuzzy semantics. *Information Sciences*, 3:159–176, 1971.

[82] L. A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning, parts 1. *Information Sciences*, 8:199–249, 1975.

[83] L. A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning, parts 2. *Information Sciences*, 8:301–357, 1975.

[84] L. A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning, parts 3. *Information Sciences*, 9:43–80, 1975.

[85] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[86] L.A. Zadeh. A fuzzy-set - theoretic interpretation of linguistic hedges. *Journal of Cybernetics*, 2(3):4–34, 1972.

[87] L.A. Zadeh. Fuzzy logic = computing with words. *IEEE Trans. Fuzzy Systems*, 4:103–111, 1996.

[88] L.A. Zadeh. From computing with numbers to computing with words-from manipulation of measurements to manipulation of perceptions. *IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications*, 45(1):105–119, 1999.

[89] H. J. Zimmermann and P. Zysno. Latent connectives in human decision making. *Fuzzy Sets and Systems*, 4(1):37–51, 1980.

SUMMARY OF THE RESULTS OF THE THESIS

In Chapter 1 I reviewed fuzzy sets and operators that are related to this thesis. I introduced and described the most important properties and theorems of negation operator, the t-norm operator, t-conorm operator and aggregative operator. After, I explained the connection between the modalities and hedges, and the general form of the modalities.

In Chapter 2 I presented the Plaint system, which is a subset of Fuzzy system. I gave a definition of the Pliant system, and then I explained the operators of Pliant system. I also introduced the distending function that is used in making function approximations.

In Chapter 3, I applied the Plaint system in the Grid environment as a decision support algorithm. In the first part 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. 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 in order to select a good performing broker for user jobs even under conditions of high uncertainty. After, I presented our results. We evaluated our algorithms in a grid-simulation environment based on GridSim, and performed simulations with real-world workload samples. 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.

The main results presented in Chapter 4 are as follows. First, I described the basic 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 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 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.

In the final chapter I presented the Cognitive Map. Here, our 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. 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. 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 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.

MAGYAR NYELVŰ ÖSSZEFOGLALÓ

Az első fejezetben tézishez köthető fogalmak kerültek bevezetésre, így a fuzzy halmazok és operátorok. Továbbá ismertettem a negációs operátort, t-normát, t-conormát és aggregative operátort. Ezekre vonatkozó alapvető tételek bemutatásra kerültek. Végezetül ismertettem a módosító szavak és a modál operátorok közötti kapcsolatot megadva ezek általános formuláját.
A második fejezetben bemutattam a Pliant rendszer, amely tekinthető a Fuzzy rendszerek speciális alrendszerének. Ebben a fejezetben kerül sor a Pliant rendszer definíciójára, speciális operátoraira. A Fuzzy elmélet halmazhoztartozási függvényének szerepét itt a distending függvény veszi át, amelynek megadása is ebben a fejezetben történt meg.
A harmadik fejezettől kezdve kerülnek ismertetésre a saját eredményeim. Így a Pliant rendszert döntéstámogató algoritmusként alkalmaztam Grid rendszereknél. A harmadik fejezet elején bemutattam a Grid rendszereket és azok alapelemeit. A Grid Metabróker egy olyan webszolgáltatás alapú elem, amely ki tudja szolgálni a felhasználókat, a grid portálokat és a brókerekkel közvetlen kapcsolatban áll. Egy újszerű ütemező algoritmus felhasználása lehetővé tesz egy magasabb szintű, együttműködő bróker használatot úgy, hogy felhasználunk már meglévő más grid rendszerben lévő brókereket. A Grid Metabróker azért gyűjti össze és hasznosítja a különböző rendszerben lévő brókerek metaadatait, hogy létrehozzon egy adaptív metabróker szolgáltatást. A fejezetben bemutattok néhány döntéstámogató ütemező algoritmust, amely a Metabróker komponenshez készült. Ezen algoritmusok a döntés meghozatalához a brókerek jellemzőit használja fel. A legjobb megoldást a "Pliant function with random generation" nevű algoritmus adta. Ez a felhasználó által beküldött feladathoz - a rendszer nagy bizonytalansága mellett is - a legjobban teljesítő brókert választja. A fejezet végén kerül sor az eredmények bemutatására. A tesztek valós terhelési adatokon a GridSim szimulációs környezetben kerültek kiértékelésre. Az eredmények nagy nyereséget mutattak, azaz a javított ütemező algoritmus sokkal hatékonyabb feladat futási eredményt adott.
A negyedik fejezetben általános közelítő eljárásokat ismertetem. Rámutatok ezen eljárások néhány hiányosságára. Ezek alapján kifejlesztettem egy új típusú nem lineáris regressziós

eljárást, amely a felfújó függvényen alapul és a függvény természetes leírását adja. A közelítés során a felfújó függvényt két féle módon is lehet alkalmazni. A felfújó függvény létrehozása során az egyik esetben a függvény csúcsát, a másikban egy intervallumot használtam fel. Az algoritmusom a BFGS eljárást is használja, így pontosabb közelítést érhető el. Megmutattam, hogy az eljárás gyors, hatékony (csak néhány lépés szükséges az optimalizáló eljárásnak) és egyszerűen használható. Ezzel a módszerrel lehetséges a függvény egy részét direkt módon megváltoztatni szemben a szokásos közelítő eljárásokkal.

Az utolsó fejezet a "Cognitive Map"-el foglalkozik. Célom a komplex, dinamikusan változó rendszerek modellezése volt a Pliant rendszer segítségével. A kidolgozott koncepció hasonló az irodalomban már ismert "Fuzzy Cognitive Map"-hez. Az általam kidolgozott esetben mind az alkalmazott függvények és az összegzések is mások. A módszer a folytonos logikán alapul és a paramétereknek szemantikus jelentése is van. Két módszert határoztam meg a hatások leírására és azt is bemutatom, hogy miként kell felépíteni a "Pliant Coginitve Map"-et. Továbbá bemutattam az általam kifejlesztett keretrendszert. Példákon keresztül mutatom be, hogy hogyan működik a "Pliant Cognitive Map". Végezetül megmutatom azt is, hogy a rendszer valós példán is megfelelően működik. A szimuláció kiértékelése során megállapítottam, hogy a szimulációs lépések közötti értékváltozások egyenletesen csökkennek, azonban több lépés szükséges a kiegyensúlyozott állapot eléréséhez. A valós példán a hatások értékeit nem változtattam a szimuláció során. A modell segítségével azonban lehetséges a hatások paramétereinek a változtatása és véleményem szerint a valós folyamatok így jobban modellezhetőek.