Constraint Satisfaction Using Soft Quantifiers

Ronald R. Yager
Machine Intelligence Institute, Iona College
New Rochelle, NY 10801
yager@panix.com
(212)249-2047

ABSTRACT

Fuzzy sets and other methods have been used to model a softening of constraints in CP problems. Here we suggest an approach to the softening of the CP problem at the meta level, the process used to aggregate the satisfactions to the individual constraints. We suggest the use of soft quantifiers such as "most" to guide the process of aggregating the satisfactions to the individual constraints. Use is made of the ability to represent these soft quantifiers by fuzzy sets and the ability to implement their authorized aggregation by the OWA operator.

Keyword: fuzzy sets, OWA operators, quantifiers, linguistic, meta-level softening

1. Introduction

Hard constraint satisfaction problems typically involve the determination of a solution to a collection of constraints, C_j for j = 1 to n, where the constraints are binary, either they are or are not not satisfied. It is clearly recognized [1] that in many situations, especially those modeling human reasoning, the constraints imposed are flexible and allow for some notion of degree of satisfaction, gradedness, rather then the all or nothing option of hard constraint satisfaction. One technique for implementing these flexible or soft constraints involves the use of fuzzy sets and fuzzy constraints [2, 3]. However implicit in most of the hard constraint approaches, as well as most approaches to softening, is a another aspect of lack of softness. This additional hardness is manifested at a meta level where the satisfactions to the individual constraints are aggregated to obtain the overall satisfaction. This meta level lack of softness is clearly manifested by the typically used imperative of requiring *all* constraints be satisfied. In some cases a weakening of this strong requirement can be authorized using such aggregation imperatives as "finding a solution that satisfies m out of n of the constraints." Although *m out of n* clearly weakens the *All* aggregation it is still a hard aggregation imperative, "m out of n" is a pure binary concept, it manifests none of the gradedness implicit in softening.

At a semantic level a soft aggregation of constraints is authorized by statements such ***Most*** *constraints should be satisfied* or ***Only about half*** *the constraints need be satisfied*. In these aggregation imperatives the use of the imprecise quantifiers ***most*** and ***only about half*** is an indication of the softness. In this work we describe an approach to implementing the soft aggregation of constraints. This approach described here makes considerable use of OWA operator [4] as a soft aggregation operator.

2. Mathematical Programming with OWA Objectives

In this section we provide a short introduction to the OWA aggregation operator [4-6]. We then introduce and provide a solution to the problem of constrained OWA aggregation.

Definition: An **O**rdered **W**eighted **A**veraging (OWA) operator of dimension n is a mapping $F: R^n \rightarrow R$ having an associated n vector $W = [w_1, w_2, \ldots w_n]$ with $w_j \in [0, 1]$ and $\sum_j w_j = 1$ where

$$F(a_1, \ldots, a_n) = \sum_{j=1}^{n} w_j b_j$$

where b_j is the j^{th} largest of the a_i.

A fundamental aspect of these operators is the re-ordering step. In particular, a weight w_j is not associated with a specific argument but with an ordered position of the argument. This ordering operation provides a nonlinear aspect to this aggregation operation. If we let B indicate $1 \times n$ vector in ith component is b_j, the jth largest of the a_i, we see in vector notation $F(a_1, \ldots, a_n) = W\,B$.

A number of properties can be associated with these operators [4-6] among these are: commutivity, monotonicity and boundedness ($\mathrm{Min}_i[a_i] \leq F(a_1, \ldots a_n) \leq \mathrm{Max}_i[a_i]$). These three conditions implies these operators are mean operators. These operators are also idempotent. By appropriate selection of the weights in W the OWA operator allows for the implementation of a large class of aggregation operators, in [7] Yager describes a large number of these. Some notable cases are the following. **Max**: $w_1 = 1$ and $w_j = 0$ for all $j \neq 1$. **Min**: $w_n = 1$ and $w_j = 0$ for all $j \neq n$. **Average**: $w_j = 1/n$ for all j. **Arrow-Hurwicz:** $w_1 = \alpha$, $w_n = 1 - \alpha$, $w_j = 0$ for all others. **Olympic-Average**: $w_1 = w_n = 0$, $w_j = 1/(n-2)$ for all others. **Mean**: $w_{(n+1)/2} = 1$ and $w_j = 0$ for all others (if n is even then $w_{\frac{n}{2}} = w_{\frac{n}{2}+1} = 0.5$ and $w_j = 0$ for all others).

In the constrained OWA optimization problem we are interested in finding the maximal value of some objective function which is expressed as an OWA aggregation and where the variables appearing in this objective function are assumed to be constrained by some collection of constraints. While the use of an OWA objective function allows us to represent complex and sophisticated objectives the price we pay for this is an increase in the difficulty of the resulting mathematical programming (MP) problem. In particular, as we shall see, in order to represent the OWA objective function in a way that allows the use of conventional MP problem solving techniques we must introduce a number of additional constraints as additional variables. While the additional constraints are of a linear kind some of the additional variables are of an integer type. In the case in which the initial constraints are linear the constrained OWA optimization problem still retains its linearity although it becomes mixed integer.

Formally the constrained OWA optimization problem is represented as

Max: $F(x_1, \dots x_n)$

Subject to: $C_j(x_1, \dots x_n)$ $\quad j = 1$ to m

In this problem it is assumed that F is a type of OWA operator defined via a particular weighting vector W, C_j are a collection of constraints and the x_i are our variables

The particular difficulty in solving this problem by conventional MP techniques is do to the fact that the OWA operator requires a reordering of the arguments, with $F(x_1, \dots x_n) = W\ B$ the calculation of B an unconventional operation. In the following we shall describe a technique introduced in [8] for converting the operation of W B into a more conventional form. Since the technique for converting the OWA objective into a more conventional form is independent of linearity of the original constraints, the C_j, we shall consider the case in which the C_j are linear as this allows us to more clearly understand the additional complexity introduced by the use of OWA objective functions

The linearly constrained OWA optimization problem is the following MP problem:

Max: $F(x_1, \dots x_n)$

Subject to:

$$
\begin{aligned}
&a_{11}x_1 + a_{12}x_2 + \dots a_{1n}x_n \leq b_1\\
&a_{21}x_2 + a_{22}x_2 + \dots a_{2n}x_n \leq b_2\\
&\qquad\qquad\qquad\qquad\qquad\vdots\\
&a_{m1}x_2 + a_{m2}x_2 + \dots a_{mn}x_n \leq b_m\\
&x_i \in R_i
\end{aligned}
$$

In the above, it is assumed that F is a type of OWA aggregation defined via a particular weighting vector W. The above is an MP problem in which only the objective function is nonlinear. As noted this nonlinearity is of a type involving the ordering of the elements, the x_i's An example of the above formulation would be a situation in which we desire to find a solution to the constraints which has the Maximum Minimum value for the x_i's. In this case our objective would be to find Maximum of $\text{Min}_i[x_i]$, here our OWA aggregation function F would be characterized by the weighing vector in which $w_n = 1$ and all other weights are zero. On the other hand if our objective was to find a solution to the constraints which has the Maximum Maximum value for the x_i's then our OWA aggregation function F would be characterized a vector W in which $w_1 = 1$ and all other weights equal zero.

As we shall subsequently see this type of nonlinearity in the objective function, the ordering of the variables, will yield to a representation that is still linear but one that requires the introduction of some integer, binary, variables. Thus the solution to the above constrained OWA optimization problem will be of the form of mixed integer linear MP problem [9]. However, the current state of the technology provides us with very efficient algorithms for solving these types of problems.

In the following, we shall let A denote the $m \times n$ matrix whose elements are the a_{ij}, let B be the m vector whose elements are b_i, and X be the n vector whose elements are x_i. Using this notation, we can express our basic problem as

Max: $F(X)$ (I)'

Subject to: $AX \leq B$ and $X \in R_i$ (II)

A simple example of the type of problem we have in mind is

Max: $\text{Min}(x_1, x_2, x_3)$

Subject to:

$x_1 + x_2 + x_3 \leq 1$

$0 \leq x_i \leq 1$

Here we desire to find the values for x_1, x_2, and x_3 that are at most one and have the largest smallest value for the three variables.

Another example would be

Max: $\text{Median}(x_1, x_2, x_3)$

Subject to:

$x_1 + x_2 + x_3 \leq 1$

$0 \leq x_i \leq 1$

Here we desire to find values for x_1, x_2, and x_3 that have the largest median and sum to at least 1.

In order to obtain the solution to this constrained OWA optimization problem, we need to provide some mechanism for introducing the reordering operation required in the OWA operator in a manner consistent with MP techniques. The modeling of the ordering type nonlinearity requires the introduction of an additional number of constraints, which are all linear, as well as the introduction of some additional variables some which are binary 0/1 integer variables.

We now describe the procedure used to convert this problem into a more conventional programming problem [8]. Our first step is to introduce an additional collection of n variables, the ordered variables, which we denote as $y_1, y_2, \ldots y_n$. Thus y_i is the ith largest of the x_j. We shall find it convenient at times to express this as a column vector Y in which y_i is the i^{th} element. Using these ordered variables we can express our objective function (I)' in a purely linear form as

Max: $\sum_i w_i y_i$ (I)

We can very succinctly express this objective function as **Max:** $W^T Y$.

Having introduced the vector Y into the objective function, we now proceed to introduce additional constraints in the problem to assure ourselves that y_i is the i^{th} largest of the x_j.

The first step in this process is to inform the algorithm of the required ordering of the y_i, that is that $y_1 \geq y_2 \geq y_3 \geq \ldots y_n$. We can express this information in a collection of $n - 1$ constraints

$y_{i+1} - y_i \leq 0$ for $i = 1$ to $n\text{-}1$ (III)

Letting G be the $n - 1 \times n$ vector whose elements are shown below

$$\begin{bmatrix} -1 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & -1 & 1 \end{bmatrix}$$

we can express (III) as $GY \leq 0$.

With the introduction of (III), our optimization problem becomes

Max: W^TY (I)

Subject to: $AX \leq B$ (II)

$GY \leq 0$ (III)

We must now relate the y_i to the x_j in the appropriate manner. The first step in this direction is to impose the condition that y_n is equal to the smallest (n^{th} largest) of the x_i. To assure ourselves of the satisfaction of this condition, we impose the following collection of constraints:

$$y_n - x_i \leq 0 \qquad \text{for } i = 1, \ldots n$$

We can express these constraints in a vector notation as $y_nI - X \leq 0$, here I an n column vector with all elements equal one.

Next, we desire to impose the condition that y_{n-1} is the second smallest of the x_i. To do this, we impose the following constraints:

$$y_{n-1} - x_j - K z_{n-1,\, j} \leq 0 \qquad j = 1 \text{ to } n$$

$$\sum_{j=1}^{n} z_{n-1,j} \leq 1 \qquad \text{(IV)}_{n-1}$$

$$z_{n-1,\, j} \in \{0, 1\} \qquad j = 1, \ldots n$$

We emphasize that $z_{n-1,\, j}$ is restricted to be a binary integer variable, takes either 1 or 0 as a value. The need for introducing $z_{n-1,\, j}$ forces us to have an integer programming problem. In the above, K is some constant much larger than any value that any of the x_j or y_i can assume. To see how this works, we note that if $z_{n-1,j} = 0$, then we effectively have $y_{n-1} - x_j \leq 0$, and thus y_{n-1} is restricted to be not greater than x_j. On the other hand, in the case when $z_{n-1,j} = 1$ we get $y_{n-1} - x_j \leq K$. Since K is some large number compared with the values for y_{n-1} and x_j, this effectively poses no constraint. Thus, the condition of $z_{n-1,\, i} = 1$ means no restriction is imposed. In addition, since $\sum_j z_{n-1,j} \leq 1$, this means that at most one of the $z_{n-1,j}$ can be one, and in turn only one of the restrictions, $y_{n-1} \leq x_j$ is withdrawn. In order to make y_{n-1} as large as possible, a condition implicitly imposed because of the non-negative nature of w_{n-1}, the algorithm will choose to relax the most severe restriction which requires y_{n-1} to be less than the smallest of the x_i. The relaxation of this restriction allows y_{n-1} to become equal to the second smallest of x_i.

We can express the constraints just introduced in vector notation

$$y_{n-1}I - X - K Z_{n-1} \leq 0$$

$$I^T Z_{n-1} \leq 1 \qquad \text{(IV)}_{n-1}$$

$$Z_{n-1} \in \{0, 1\}$$

In this notation, Z_{n-1} is an n column vector whose components are $z_{n-1,i}$.

We now turn to the situation of obtaining y_{n-2}, the third smallest of the x_i. In order to obtain this, we impose the constraints

$$\begin{aligned} & y_{n-2}I - X - K\,Z_{n-2} \le 0 \\ & I^T Z_{n-2} \le 2 & \qquad (IV)_{n-2} \\ & Z_{n-2} \in (0, 1) \end{aligned}$$

here Z_{n-2} is an n column vector whose components are $z_{n-2,j}$ whose components are restricted to be 0 or 1. Here that two of the constraints $y_{n-2} - x_j - K\,z_{n-2,j} \le 0$ are allowed to be relaxed, and hence y_{n-2} can be evaluated to the third smallest value.

Generalizing the results, we see that for y_j we get the following:

$$\begin{aligned} & y_j\,I - X - K\,Z_j \le 0 \\ & I^T Z_j \le n - j \\ & Z_j \in \{0, 1\} \end{aligned}$$

We now see the final structure of the mixed integer programming problem we must solve in order to solve the constrained OWA optimization problem.

Max: W^TY (I)

Subject to:

$$\begin{aligned} & AX \le B & & (II) \\ & GY \le 0 & & (III) \\ & y_j\,I - X - KZ_j \le 0 & & \text{for } j = 1, n \\ & I^TZ_j \le n - j & & \text{for } j = 1, n \\ & Z_j \in \{0, 1\} & & \text{for } j = 1, n \end{aligned}$$

In the above, we recall that Z_j is an n vector; thus, we need to introduce $n \times n$ binary integer variables to accomplish our task. Actually, all we need are $n \times (n - 1)$ binary integer variables since $Z_n = 0$ because

$$I^TZ_n \le n - n \le 0$$

and hence $z_{n,i} = 0$

3. Quantifier Guided Constraint Aggregation

It early was established that fuzzy subsets can be used for modeling soft constraint satisfaction problems [10]. Assume X is a space of alternatives and we have a collection of constraints which are of concern in a given problem. We can represent a soft constraint, as well as a hard constraint, as a fuzzy subset A_i where for each $x \in X$, $A_i(x)$ indicates the degree to which constraint i is satisfied by x. Furthermore, it was suggested in [10] that the imperative of satisfying all the constraints can be formulated as a desire to satisfy A_1 *and* A_2 *and* .... *and* A_n. This situation can modeled via the use of the intersection of fuzzy sets, $D = A_1 \cap A_2 \ldots \cap A_n$. Thus the degree to which x satisfies *all* the constraints can be obtained as D(x), where $D(x) = Min[A_1(x), . . . A_n(x)]$. Although the use of fuzzy sets to represent the constraints allows for a softening of the satisfactions to the individual constraints when combining the satisfactions to the individual constraints we are imposing a very hard imperative in that we require *all* constraints be considered. To emphasize this we can express this formulation as $D = All[A_1, A_2, ..., A_n]$.

Yager [11] suggested a softening of the imperative used to aggregate the individual constraints by replacing the quantifier *all* guiding the aggregation by other softer quantifiers. Central to the approach suggested by Yager is the use of the OWA operator to implement constraint satisfaction aggregations authorized by quantifiers. This implementation in turn makes considerable use ability to represent quantifiers as fuzzy subsets.

Classical binary logic has at its disposal two quantifiers, the universal quantifier, *for all*, and the existential quantifier *there exists*. Human discourse, rather than being restricted to only these two quantifiers, uses a large number of linguistic quantifiers. Examples of these objects are words such as *most*, *few*, *at least about half*, *about five*, *some*, *many*, etc. A softening of the process of constraint satisfaction aggregation can be had if we replace the use of the hard quantifier *all* by using a softer quantifier such as, for example, *most*. In this case the fuzzy subset D representing the overall satisfaction would formulated as $D = Most[A_1, A_2, ..., A_n]$. In the following we introduce the machinery necessary needed to evaluate D.

As noted by Zadeh [12], quantifiers fall into two categories - absolute and proportional; our concern here is with proportional type quantifiers. He suggested that a proportional type quantifier, such as *about half* or *most*, can be represented as a fuzzy subset Q of the unit interval. In this representation, Q is defined so that for each $r \in [0, 1]$, the membership grade, Q(r), indicates the degree to which the proportion r satisfies the concept which Q is representing. In this representation the quantifier *all* would be represented by a Q such that Q(1) = 1 and Q(r) = 0 for $r \neq 1$. On the other hand a quantifier such as *most* could be represented for example by a fuzzy subset Q such that Q(r) = 0 for $r \leq 0.5$, Q(r) = 2.5 r - 1.25 for $0.5 < r \leq 0.9$ and Q(r) = 1 for $r \geq 0.9$.

While we shall not here involve ourselves with the details of representing quantifiers as fuzzy sets we shall indicate class of quantifiers that are useful in aggregating constraint satisfactions. These quantifiers called **RE**gular **M**onotonic (REM) quantifiers are useful in characterizing aggregations imperatives in which essentially the more objects included the better Examples of these quantifiers are: *all*, *there exists*, *at least about* α, *most*. Formally REM quantifiers satisfy the conditions Q(0) = 0, Q(1) = 1 and $Q(r_1) \geq Q(r_2)$ if $r_1 > r_2$.

Using a fuzzy set representation of quantifiers Yager [4, 11] suggested a method for evaluating $D = Q[A_1, A_2, ..., A_n]$ which makes use of an OWA operator. Specifically any REM quantifier can be associated with an n dimensional OWA operator F_Q such that its weighting vector W is defined by

$$w_j = Q(\frac{j}{n}) - Q(\frac{j - 1}{n})$$

Once having this corresponding OWA operator we can obtain

$$D(x) = F_Q(A_1(x), A_2(x), ..., A_n(x))$$

4. Fuzzy Softening of Linear Constraint Satisfaction Problems

A typical linear constraint satisfaction problem can be formulated as finding a solution vector x that satisfies:

$$B_i\,x \le d_i \qquad i = 1 \text{ to } m \qquad (I)$$
$$x \ge 0$$

In the above, x is an $n \times 1$ vector and B_i is an $1 \times n$ vector. A softening of this problem is one in which we desire to find a solution vector x that satisfies:

$$B_i\,x \underset{\sim}{\le} d_i \qquad i = 1 \text{ to } m$$
$$x \ge 0$$

The symbol $\underset{\sim}{\le}$ denotes a softening or fuzzification of the crisp less than or equal to operator ≤. Here some degree of latitude is allowed in satisfying the constraint $B_i\,x \le d_i$. Rather then a binary situation of either satisfying the constraint or not we allow some partiality (gradedness) of satisfaction. Essentially we are replacing the binary relationship ≤ by a fuzzy relationship $\underset{\sim}{\le}$. In order to implement this softening each row $B_ix \underset{\sim}{\le} d_i$ can be represented as a fuzzy subset $G_i(x)$ indicating the degree to which the condition $B_ix \underset{\sim}{\le} d_i$ is satisfied. The softened constraint satisfaction problem becomes one of finding the vector x that best satisfies *all* the G_i . In this case, with D = *All*$[G_1, G_2, ...,G_n]$, $D(x) = \text{Min}_i[G_i(x)]$ indicates the degree to which x satisfies all the constraints. We are faced with the problem of finding the solution vector x that Maximizes $\text{Min}_i[G_i(x)]$.

In order to represent the fuzzy subsets G_i we draw upon an idea suggested by Zimmermann [13, 14] which provides a very natural and useful way of softening the constraints He suggests introducing a tolerance level p_i for each constraint and defining

$$G_i(x) = \begin{cases} 1 & B_i\,x \le d_i \\ 1 - \dfrac{B_i\,x - d_i}{p_i} & \text{if } d_i < B_i\,x \le d_i + p_i \\ 0 & \text{if } B_i\,x > d_i + p_i \end{cases}$$

The function is linear inside the tolerance range $[d_i, d_i + p_i]$. A more useful formulation for the fuzzy subset G_i can be obtained [13] if we introduce an additional variable t_i measuring the degree of violation of the ith goal, $G_i(x) = 1 - \frac{t_i}{p_i}$. Using this variable, we can express the softened linear constraint satisfaction problem as an MP problem:

Maximize: $\text{Min}_i[g_i]$

Such that: $g_i = 1 - \frac{t_i}{p_i}$

$B_i x - t_i \leq d_i$

$t_i \leq p_i$

$x_i \geq 0$ and $t_i \geq 0$

Here we note the g_i indicates the degree of satisfaction of ith constraint. Implicit in the formulation of the objective function $\text{Min}_i[g_i]$ is the imperative that we try to satisfy all the constraints. A further softening of this linear constraint satisfaction problem can be had if we don't require satisfaction of all the constraints but allow the satisfaction of some quantifier guided aggregation of the individual constraints, that is try to solve $Q[G_1, G_2, \ldots, G_n]$ instead of $All[G_1, G_2, \ldots, G_n]$. In order to accomplish this we can use the ideas on quantifier guided aggregation presented earlier. Letting Q be the fuzzy subset representing the quantifier we are using, we calculate the weights defining its associated OWA operator, $w_i = Q(\frac{i}{m}) - Q(\frac{i-1}{m})$ where m is the number of constraints in the problem. We then express our constraint satisfaction problem as a

Maximize: $F_W(g_1, \ldots g_n)$

Such that: $g_i = 1 - \frac{t_i}{p_i}$

$B_i x - t_i \leq d_i$

$t_i \leq p_i$ (II)

$x_i \geq 0$ and $t_i \geq 0$,

where F_W is the OWA aggregation operator obtained using the weights w_i derived above.

The preceding softened constraint satisfaction problem can be seen to be an example of the type of linearly constraint OWA optimization which we discussed earlier. In order to operationalize the ordering process required by the OWA aggregation in the objective function, we must introduce some other variables and additional constraints just as was done in the earlier section.

We introduce the variables $y_1, \ldots y_m$ where y_j indicates the j^{th} largest of the g_i. Using this we can express the objective function in a simple linear form, $\sum_j w_j y_j$. We now must introduce the constraints to obtain the y_j. To do this we need a collection of constraints $y_{j+1} - y_j \leq 0$ for j = 1, m – 1. Furthermore, we introduce a collection of constraints for each y_j to assure us it is the j^{th} largest. To accomplish this we use the following collection for each i = 0, . . . m – 1:

$y_{n-i}\, I - g - K\, Z_{m-i} \leq 0$

$I^T Z_{m-1} \leq 1$

$Z_{m-i,j} \in \{0, 1\}$

In the above, I, g, and Z_{m-i} are

$$I = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}, \; g = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_m \end{bmatrix}, \; Z_{n-i} = \begin{bmatrix} z_{m-i,1} \\ z_{m-i,2} \\ \vdots \\ z_{m-i,2} \end{bmatrix}$$

5. References

[1]. Freuder, E., "Partial constraint satisfaction," Proceedings of the International Joint Conference on Artificial Intelligence, 278-283, 1989.

[2]. Yager, R. R., "Some extensions of constraint propagation of label sets," Int. J. of Approximate Reasoning 3, 417-436, 1989.

[3]. Dubois, D., Fargier, H. and Prade, H., "Propagation and satisfaction of flexible constraints," in Fuzzy Sets, Neural Networks and Soft Computing, edited by Yager, R. R. and Zadeh, L. A., New York: Van Nostrand Reinhold, 166-187, 1994.

[4]. Yager, R. R., "On ordered weighted averaging aggregation operators in multi-criteria decision making," IEEE Transactions on Systems, Man and Cybernetics 18, 183-190, 1988.

[5]. Yager, R. R. and Filev, D. P., Essentials of Fuzzy Modeling and Control, John Wiley: New York 1994.

[6]. Yager, R. R. and Kacprzyk, J., The Ordered Weighted Averaging Operators: Theory and Applications, Kluwer: Norwell, MA, 1997.

[7]. Yager, R. R., "Families of OWA operators," Fuzzy Sets and Systems 59, 125-148, 1993.

[8]. Yager, R. R., "Constrained OWA aggregation," Fuzzy Sets and Systems 81, 89-101, 1996.

[9]. Plane, D. R. and McMillan, C., Discrete Optimization, Prentice-Hall, 1971.

[10]. Bellman, R. E. and Zadeh, L. A., "Decision-making in a fuzzy environment," Management Science 17:4, 141-164, 1970.

[11]. Yager, R. R., "Quantifier guided aggregation using OWA operators," International Journal of Intelligent Systems 11, 49-73, 1996.

[12]. Zadeh, L. A., "A computational approach to fuzzy quantifiers in natural languages," Computing and Mathematics with Applications 9, 149-184, 1983.

[13]. Zimmermann, H. J., "Description and optimization of fuzzy systems," International Journal of General Systems 2, 209-215, 1976.

[14]. Zimmermann, H. J., Fuzzy Set Theory--and Its Applications, Kluwer-Nijhoff: Dordrecht, 1985.