DECSAI

Deparment of Computer Science and Artificial Intelligence

Applying Genetic Algorithms in Fuzzy Optimization Problems

F. Herrera, M. Lozano and J.L. Verdegay

January, 1994

Fuzzy Systems & A.I. Reports and Letters 3, 1 (1994), 39-52.

ETS de Ingeniería Informática. Universidad de Granada.
18071 Granada - Spain - Phone +34.58.244019, Fax +34.58.243317

Applying Genetic Algorithms in Fuzzy Optimization Problems*

F. Herrera, M. Lozano, J.L. Verdegay
Dept. of Computer Science and Artificial Inteligence
University of Granada
18071 - Granada, Spain
e-mail: fherrera@ugr.es, lozano@robinson.ugr.es, jverdegay@ugr.es

Abstract

Genetic Algorithms as function optimizers are global optimization techniques based on natural selection principles that can be efficiently used in high dimensional, multimodal and complex problems. Genetic Algorithms are here presented as a tool to solve fuzzy optimization problems either on their associated auxiliary models or by assuming the existence of some fuzzy performance (fitness) function. Finally applications of Genetic Algorithms to find the maximum flow in a network with fuzzy capacities and assignment problems with linguistic labels are studied.

Keywords: Genetic algorithms, fuzzy optimization.

1. Introduction

Methods and techniques of optimization have been successfully used in various fields, and related to technical systems of relatively well-defined structure and behavior, the so-called hard ones. The success has motivated a direct application of the same traditional approaches to the modeling and analysis of what is often called the soft systems in which a key role is played by human judgments, preferences, etc. Unfortunately, the progress in this way have been much less than expected. There are some questions that we can state: Can we get some good solution satisfying a level of performance quickly?. If we work with imprecise (fuzzy) data will we be able to design an efficient algorithm for obtaining an optimal solution?. How to reach an acceptable solution based on human-like reasoning mechanisms?. Is it enough to obtain a good solution?. Here our very start point is that attainment of the optimum is much less important for complex systems that to attain quickly a good solution. Genetic Algorithms (GA) are search methods drawing increasing attention regarding their potential as optimization techniques. They are search algorithms with linear order which do not necessarily find an optimal solution to any problem, but do find good solutions to problems that are resistant to most other known techniques.

GA were formerly introduced and developed by J. Holland and his colleagues at the University of Michigan. He gave the theoretic foundations in his book "Adaptation in Natural and Artificial Systems" (1975) [Hol75]. GA are iterative adaptative search algorithms that use operations found in natural genetic to guide their trek through a search space. They start with a population of randomly generated solutions and advance toward better solutions by applying the genetic operators modeled on the genetic processes occurring

*Work supported by spanish DGICyT under project PB92-0933

in nature. In each generation, relatively good solutions reproduce to give offspring that replace the relatively bad solutions which die. An evaluation or objective function plays the role of the environment to distinguish between good and bad solutions.

The purpose of this paper is to show how GA can be applied to solve some special fuzzy optimization problems. To best achieve this goal, the paper is structured as follows: Section 2 introduces the foundations of GA. Section 3 describes the application of GA to some conventional optimization problems and, finally, Section 4 deals with the use of GA to solve two concrete cases: to find the maximum flow in a network with fuzzy capacities and assignment problems with linguistic labels.

2. Foundations of Genetic Algorithms

GA are different from more normal optimization and search procedures in four ways: a) GAs work with a coding of the parameter set, not the parameters themselves, b) GA search from a population of points, not from a single point, c) GA use objective function information, d) GA use probabilistic transition rules, not deterministic ones. GA require the natural parameter set of the optimization problem to be coded as a finite-length string over some finite alphabet.

Although there are many possible variants of the basic GA, the fundamental underlying mechanism operates on a population of individuals, and consists on three operations: (1) evaluation of individual fitness, (2) formation of a gene pool, and (3) recombination and mutation. The initial population $P(0)$ is chosen randomly and the individuals resulting from these three operations form the next generations's population. The process is iterated until the system ceases to improve. Generally, each individual in the population is represented by a fixed length binary string which encodes values for variables.

During iteration t, the GA maintains a population $P(t)$ of solutions $x_1^t, ..., x_N^t$ (the population size n remains fixed). Each solution, x_i^t , is evaluated by the function $E(\cdot)$, and $E(x_i^t)$ is a measure of fitness of the solution. The fitness value determines the relative ability of an individual to survive and produce offspring in the next generation. In the next iteration $(t+1)$ a new populations is formed on the basis of the operations (2) and (3).

The figure 1 shows the structure of a simple GA, and in the figures 2.1, 2.2, 2.3 are illustrated the basic operations: reproduction, crossover and mutation.

Procedure genetic algorithm
begin (1)
    $t = 0$;
    inicialize $P(t)$;
    evaluate $P(t)$;
    While (**Not** *termination-condition*) **do**
    begin (2)
        $t = t + 1$;
        select $P(t)$ *from* $P(t-1)$;
        recombine $P(t)$;
        evaluate $P(t)$;
    end (2)
end (1)

Figure 1: Structure of a GA

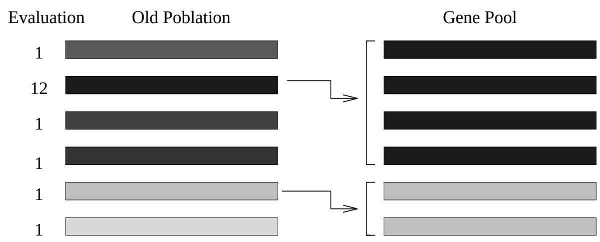

Figure 2.1: Evaluation and contribution to the gene pool

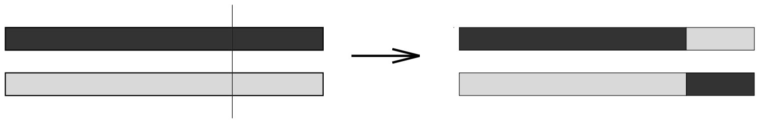

Figure 2.2: Recombination. One-point crossover

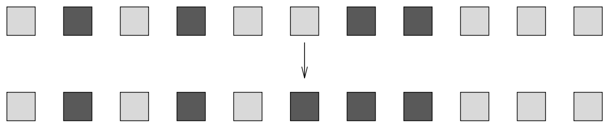

Figure 2.3: Mutation

The crossover operator combines the features of two parent structures to form two similar offspring, this is applied under a random position cross with a probability of performance, the crossover probability, P_c . The mutation operator arbitrarily alters one or more components of a selected structure so as to increase the structural variability of the population. Each position of each solution vector in the population undergoes a random change according to a probability defined by the mutation rate, the mutation probability, P_m.

A schema is a similarity template describing a subset of strings with similarities at certain string positions, [Gol89, Hol75]. Even though a standard GA processes only n structures each generation it also processes approximately n^3 schemata. This is an important propertie of GA that is called *implicit parallelism.*

It is generally accepted that any GA to solve a problem must have five components:

1. A genetic representation of solutions to the problem,
2. a way to create an initial population of solutions,
3. an evaluation function which gives the fitness of each individual,
4. genetic operators that alter the genetic composition of children during reproduction, and
5. values for the parameters that the GA uses (population size, probabilities of applying genetic operators, etc.).

3. Genetic Algorithms and Optimization Problems

GA-based function optimizers have demonstrated their usefulness over a wide range of difficult problems.

According their above description, first a representation of the solutions is needed. GA require the natural parameter set of the optimization problem to be coded as a finite-length string over some finite alphabet. It was noted very early that the choice of representation can itself affect the performance of a GA-based function optimizer.

The binary representation is just one of the ways to represent the space to be searched. Using the $\{0,1\}$ alphabet with a non-combinatorial problem, that is with real variables, we code the variables in the following way: If a single parameter x_i has lower and upper bounds a_i and b_i respectively, first we must apply a discretization over the real parameter and then the standard way of binary coding x_i using n bits is to let real values between $a_i + \frac{k(b_i-a_i)}{(2^n-1)}$ and $a_i + \frac{(k+1)(b_i-a_i)}{(2^n-1)}$ corresponding to the standard binary code for the integer k for $0 \leq k < 2^n - 1$. To avoid talking about intervals, it is referred to the binary code for the integer k above as corresponding to the left end of the interval, namely $a_i + \frac{k(b_i-a_i)}{(2^n-1)}$.

Non-binary string representations are often used, new techniques are directed to use a real valued space where a solution is coded as a floating point vector, that is the real-coded alternative [Gol91, Jan91, Mic92, Esh93].

Initially, GA were applied directly only to unconstrained problems, but many practical problems contain one or more constraints that must also be satisfied. Then it is necessary to incorporate the constraints into GA search. The major dificulty in applicability of GA to various optimization problems is the lack of general methodology for handling constraints.

Considering the next constrained problem,

$$\begin{array}{ll} Max: & f(x) \\ s.t. & \\ & h_i(x) \geq 0 \;\; i = 1, ..., m \\ & x \geq 0 \end{array} \tag{1}$$

diferent approaches to manage the constraints with GA have been proposed:

a) To generate candidate solutions without considering constraints and penalizing them by decreasing the goodness of evaluation function. A constrained problem is transformed into an unconstrained one by associating a penalty with all the constraint violations and including the penalties in the evaluation function as follows:

$$Max: \;\; f(x) + \epsilon r \sum_{i=1}^{m} \phi_i(h_i(x)). \tag{2}$$

where r is a penalty coefficient, ϵ is -1 for maximization and +1 for minimization problems, and ϕ_i is a penalty function related to the i-th constraint (see [Gol89, Ric89, Sie89, Lie91]).

b) The second approach is concentrated on the use of special representation mappings which guarantee (or at least increase the probability of) the generation of a feasible solution and the application of special repair algorithms to "correct" any infeasible solution generated.

These two approaches involve transforming potential solutions of the problem into a form suitable for a GA using penalty functions or intelligent decoding schemes and repair mechanisms.

c) Another different approach is to introduce richer data structures together with an appropriate family of applicable genetic operators which can hide the constraints presented in the problem (see [Vig91, Mic92, Nak91]).

d) A final approach provides a way of handling constraints that is both general and problem independent. It combines some of the ideas of the previous approaches. The main idea behind this approach lies in the elimination of the equalities present in the set of constraints, and careful design of special genetic operators, which guarantee to keep all chromosomes within the constrained solution space. The GENOCOP system (GEnetic algorithm for Numerical Optimization for COnstrained Problems) is behind of this approach (see [Mic91, Mic92]).

Therefore, we must note that for optimization problems it is possible to transform the problem into a form suitable for a GA (penalty functions, ...), and to design specific GA for solving them, with special representations, incorporating problem specific knowledge into GA, with modified operator, etc. These different techniques have allowed to apply GA to a wide class of optimization problems: Linear programming problems, Transportation problems, Quadratic assignment problems, Traveling salesman problems, Vehicle routing, Job-shop problems,Bin-packing problems, Graph partitioning, Set covering problems, Graph Coloring, Scheduling, Facility layout, and others NP-complete problems (see [Vol93] List of References of Evolutionary Algorithms in Management Science).

4. Genetic Algorithms Applications to Fuzzy Optimization

Let $S = (s_1, s_2, ..., s_N)$ be a population of N individuals, and denote $E(\cdot)$ the evaluation function giving a fitness for every s_i. The fitness $E(s_i)$ for a solution candidate s_i serves to determine a selection probability P_{S_i}.

The proportional selection [Gol89, Hol75] is the main selection scheme used in GA,

$$P_{S_i} = \frac{E(s_i)}{\sum_{j=1}^{N} E(s_j)}, \ i = 1, ..., N, \tag{3}$$

(an overview about the selection probabilities proposed by different authors can be found in [Bac91]).

When we work in a fuzzy environment, the feasible solutions of an optimization problem can hold associated fuzzy objectives [Fed91], then the fitness associated to an individual is defined by a fuzzy number. Then, would be necessary to define a method to obtain the selection probabilities of the individuals. Next, we propose a method to do it, and we present some applications of GA to fuzzy optimization problems.

4.1. Selection probability from fuzzy fitness

Consider the fuzzy fitness associated to each individual of the population, obtained by means of the evaluation function, $E(\cdot)$, and represented as a fuzzy number, $E : S \longrightarrow F(R)$, $s_i \in S$, $E(s_i) \in F(R)$, where $F(R)$ is the set of real fuzzy numbers.

From these fuzzy numbers it is necessary to define the selection probability associated to each individual to obtain a gene pool. We can consider a function $f : F(R) \longrightarrow R$, which according to (3) allows to define the selection probability from the fuzzy fitness.

In fact, for each individual s_i of the population S, P_{S_i} can be defined as

$$P_{S_i} = \frac{f(E(s_i))}{f(\sum_{j=1}^{N} E(s_j))}, \tag{4}$$

where the function $f : F(R) \longrightarrow R$ must be such that the set $\{P_{S_i}\}_{i=1,...,N}$ verifies to be both a distribution of probability and a selection probability which allows to reproduce the best strings, that is, the strongest individuals. In order to the function f verifies these desired properties, we introduce the next conditions for the function f:

1) Preserve the addition operation

$$f(\sum_{i=1}^{N} E(s_j)) = \sum_{i=1}^{N} f(E(s_j)), \tag{5}$$

necessary condition in order that $\sum_{i=1}^{N} P_{S_i} = 1$.

2) Preserve the order

$$E(s_k) \leq E(s_j) \Longleftrightarrow f(E(s_k)) \leq f(E(s_j)), \tag{6}$$

necessary condition for reproducing the strongest strings by means of the selection probability. Because of this condiction, the strongest individuals have larger selection probability,

$$E(s_k) \leq E(s_j) \Longleftrightarrow P_{S_k} \leq P_{S_j}.$$

An important class of functions which verify these conditions, (5) and (6), are the linear ranking functions [Bor85, Cam92, Zhu92].

4.2. Linear ranking functions

Fuzzy numbers [Dub80] are fuzzy subsets of R whose α-cuts are closed and bounded intervals on R when $\alpha > 0$. If $\alpha = 0$, A^0 will denote the closure of $supp(A)$. Hence, $\forall \alpha \in [0,1]$ the α-cuts of A will be represented by

$$A_\alpha = [a_1(\alpha), a_2(\alpha)] \tag{7}$$

Extended sum and product by positive real numbers are considered to be defined in $F(R)$ by means of the Zadeh's Extension Principle. Hence, given any two fuzzy numbers $A, B \in F(R)$, $\forall \alpha \in [0,1]$, the following result concerned on their α-cuts will be used,

$$(A + B)^\alpha = [a_1(\alpha) + b_1(\alpha), a_2(\alpha) + b_2(\alpha)] \tag{8}$$

$$(rA)^\alpha = [ra_1(\alpha), ra_2(\alpha)] \quad \forall r \in R \ r > 0. \tag{9}$$

The problem of comparison of fuzzy numbers has been widely investigated in the literature. Many fuzzy numbers ranking procedures (FNRP) can be found for instance in [Bon85, Zhu92]. Here we will focus on those FNRP which verify the conditions (8) and (9), linear ranking functions, what is no too restrictive because many well known FNRP may be formulated by using linear ranking functions in some way.

Consider $A, B \in F(R)$, a simple method of comparison between them consists on the definition of a certain function $f : F(R) \longrightarrow R$. If this function $f(\cdot)$ is known, then $f(A) > f(B)$, $f(A) = f(B)$, $f(A) < f(B)$ are equivalent to $A > B$, $A = B$, $A < B$ respectively. Usually, f is called a Linear Ranking Function (LRF) if

$$\forall A, B \in F(R);\ \forall r \in R\ r > 0;\ f(A + B) = f(A) + f(B) \text{ and } f(rA) = rf(A). \tag{10}$$

As it is well known, from this definition several FNRP may be considered. In particular, properties (8) and (9) suggest that FNRP using linear functions of the α-cuts could be expressed by LRF. A good study of these LRF can be found in [Cam92]. Here, some of them are shown.

4.2.1. Linear functions based on only one α-cut

Consider α, $\lambda \in [0,1]$ and $A \in F(R)$. One defines

$$f(A) = \lambda a_1(\alpha) + (1-\lambda)a_2(\alpha). \tag{11}$$

The definition contains two parameters which depend on the decision-maker: the first, α, is an accomplishment degree of the comparison, and the second, λ, is and optimism-pessimism level. Particular cases of this definition are:

(a) If $\alpha = 1$ and $A \in F(R)$ is any unimodal fuzzy number, then the first index of Yager, [Yag78], is obtained.

(b) If $\alpha \in [0,1]$ is any fixed value and $\lambda = 0$, then the index of Adamo [Ada80] is deduced.

This definition is a particular case of the discrete method proposed in [Gon91]. From the properties (8) and (9) is very easy to show as $f(\cdot)$ is linear in the sense of the above definition.

4.2.2. Linear functions based on all the α-cuts

Let $A \in F(R)$, $\lambda \in [0,1]$ and $P(\cdot)$ be any additive measure on $[0,1]$. One defines

$$f(A) = \int_0^1 (\lambda a_1(\alpha) + (1-\lambda)a_2(\alpha))\, dP(\alpha). \tag{12}$$

As it can be seen, now one has only one parameter, λ, which acts as an optimism-pessimism degree. Particular cases of this definition are:

(a) If $P(\cdot)$ is the Lebesgue's measure and $\lambda = \frac{1}{2}$, then the fourth index of Yager, [Yag81], is obtained.

(b) If $P(\cdot)$ is given by $P([a,b]) = b^2 - a^2$ and $\lambda = \frac{1}{2}$, then the index of Tsumura, [Tsu81], is deduced.

This general definition of $f(A)$ was proposed in [Gon90], where also was shown as this ranking function is linear. In particular for triangular fuzzy number $f(A)$ takes the following expression:

$$f(A) = a_1(1) - \frac{(a_1(1) - a_1(0))}{(r+1)} + \lambda \frac{(a_2(0) - a_1(0))}{(r+1)}$$

where r is a parameter according to which $f(A)$ can take values either close to the modal values ($r > 1$) or close to the values on the support ($r < 1$).

4.3. Applications

A lot of real problems consist on the optimization of a function such that all the characteristics of the problem can be completely represented by it. But the solution of an optimization problems may lead to a polynomial problem of high order or a NP problem, as it is the case of many optimization problems in network and graphs, decision problems, satisfiability problems, etc. Many of them have served to show the power of the GA.

Fuzzy logic based optimization approaches are proposed for representing uncertainty and approximation in relationships among system variables. Among them, we have selected two fuzzy optimization problems whose associated model may be solved by GA, as in fuzzy environment they can be used as a flexible tool for optimization and search.

a) Maximum flow in a network with fuzzy capacities, [Baz77, Cha82 ,Cha84 ,Cha87]

Let $S = (N, A)$ be a network with m vertices and n arcs where a liquid or a gas go through it from a source node n_s until a sink node n_t with fuzzy arc capacities. N denotes the set of nodes and $A \subset NxN$ the set of arcs. The problem consists on finding the maximum flow from n_s to n_t where associated to each arc (i, j) there is a capacity $l_{ij} = 0$ giving a lower bound for the flow, and a fuzzy capacity $\underset{\sim}{u_{ij}} \in F(R)$ given an upper bound for the flow on the arc flows. Next the concept of cut in a network (basic in the development of the problem) as well as the concept of fuzzy capacity of a cut are introduced.

Definition 1. (Separation of the nodes n_s and n_t) [Baz77]

Let N_1 be any set of nodes in the network such that n_s belongs to N_1 and n_t do not belong to it, and let N_2 be $N_2 = N - N_1$. Then $(N_1, N_2) \equiv \{(i, j) : n_i \in N_1, n_j \in N_2\}$ is called a cut which separates the nodes n_s and n_t.

Definition 2. [Cha84]

The fuzzy capacity of the cut (N_1, N_2) is the fuzzy number defined as

$$\underset{\sim}{C}(N_1, N_2) = \sum_{(N_1,N_2)} \underset{\sim}{u_{ij}} = \sum_{\substack{n_i \in N_1 \\ n_j \in N_2}} \underset{\sim}{u_{ij}} \tag{13}$$

extending the addition operation on fuzzy numbers.

Theorem 1. (Maximum flow-minimum cut theorem [For56])

The value of the maximum flow from n_s to n_t is equal to the value of the minimum cut-set (N_1, N_2) separating n_s from n_t.

The value of such a cut-set is the sum of the capacities of all arcs of S whose initial vertices are in N_1 and the final vertices in N_2. Then the minimum cut-set is the cut-set with the smallest value $\underset{\sim}{C}(N_1, N_2)$.

Thus, the problem is reduced to obtain a partition of the set N into N_1, N_2, which separates the two vertices n_s, n_t, and with minimun $\underset{\sim}{C}(N_1, N_2)$. It is possible to formulate the minimum cut problem as a 0-1 programming problem as follows: Let x_i be a 0-1 variable associated with the node $n_i \in N$ defined as

$$x_i = \begin{cases} 1 & \text{if } n_i \in N_1 \\ 0 & \text{if } n_i \in N_2 \end{cases}$$

then the minimum cut problem can be formulated as to minimize

$$g(x) = \sum_{i=1}^{m} \sum_{j=i+1}^{m} \underset{\sim}{u_{ij}} \,[x_i(1 - x_j) + (1 - x_i)x_j] \tag{14}$$

and a boolean function without constraints is obtained. In order to solve this problem by using GA the following evaluation function $E(\cdot)$ is introduced,

$$E(x) = \sum_{i=1}^{m} \sum_{j=i+1}^{m} \underset{\sim}{u_{ij}} - g(x) \tag{15}$$

and therefore to maximize $E(x)$, which obviously is equivalent to minimize $g(x)$, is to be done. Then with an adecuate genetic representation of the solutions, the selection probability for each individual of the population will be obtained using a LRF, $f(\cdot)$, as it was described in the expression (4).

Remark: As it is known, [Chr75], if we consider a network with various source nodes and sink nodes, and assume that flow can go from any source to any sink. The problem of finding the maximum total flow from all the sources to all the sinks can be converted to the simple (n_s to n_t) maximum flow problem by adding a new artificial source node n_s and a new artificial sink node n_t with added arcs leading from n_s to each of the real source nodes and from every real sink to n_t. The figure 3 shows this.

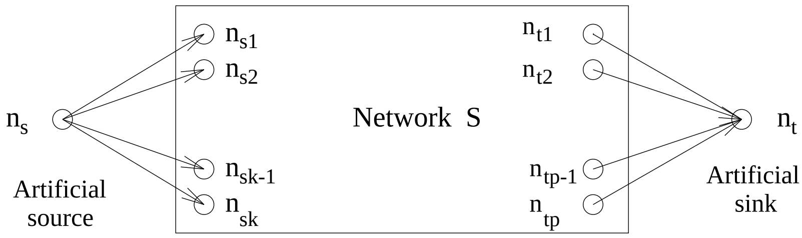

Figure 3: Network with source nodes and sink nodes

b) Assignment Problems with linguistic labels

The linguistic approach assumes that the variables which participate in the problem are assessed by means of linguistic terms instead of numerical values, [Zad75]. This approach is adequate for a lot of problems, because it allows a representation of the experts information in a more direct and adequate form when they are unable of expressing this with precision.

Among them, we will study here the assignment problem with linguistic labels, which is addressed as follows: Consider m workers and m jobs, and a linguistic valuation about the assignment of the worker i to the job j. This valuation would be stablished in terms of $L = \{$impossible, extremely-unlikely, very-low-chance,...$\}$, it shows the fitting between the worker i and the job j, and will be denoted by e_{ij}. Thus, we need a term set which will define the uncertainty granularity, i.e. the finest level of distinction among different quantifications of uncertainty. The elements of the term set determine the granularity of the uncertainty and this granularity will limit the ability to differentiate between two similar operators.

In [Bon85] it was studied the use of term set with odd cardinal, where the middle term represents a probability of "aproximately 0.5" and the rest terms are placed simetrically around it, and the limit of granularity is 11 or no more than 13. We choose a set of nine labels studied in [Bon85], E_i $i = 1, \ldots, 9$, denoted by L and with the following terms

$$\begin{aligned} L = \{ & impossible,\ extremely-unlikely,\ very-low-chance,\ small-chance, \\ & it\_may,\ meaningful\_chance,\ most\_likely,\ extremely\_likely,\ certain\} \end{aligned} \tag{16}$$

where the terms can be modified according to the criteria of the decision-maker.

The semantic of each element of the term set is given by a fuzzy number defined on the [0,1] interval, N_i, which can be described by its membership function μ_{N_i}.

Provided that the linguistic assessments are just approximate ones, trapezoidal membership functions are good enough to capture their vagueness. The parametric representation of the trapezoidal membership functions is achieved by the 4-uple $(a_i, b_i, \alpha_i, \beta_i)$. The figure 4 shows the membership function of N_i.

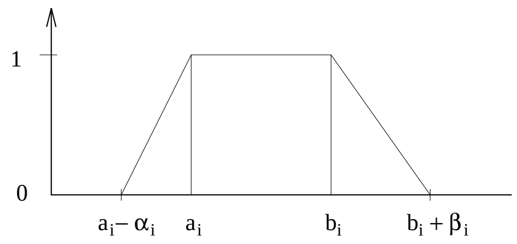

Figure 4: Membership Functions of N_i

The fuzzy numbers associated with each element were derived from an adaptation of the results of psychological experiments on the use of linguistic probabilities [Bey82, Bon85]. The table 1 indicates the semantic of the proposed term set L, [Bon85],

impossible	(0 0 0 0)
extremely_unlikely	(0 .02 0 .05)
very_low_chance	(.1 .18 .06 .05)
small_chance	(.22 .36 .05 .06)
it_may	(.41 .58 .09 .07)
meaningful_chance	(.63 .80 .05 .06)
most_likely	(.78 .92 .06 .05)
extremely_likely	(.98 1 .05 0)
certain	(1 1 0 0)

Table 1: Semantic of the term set elements L

and the figure 5 illustrates the membership distribution of the term set elements.

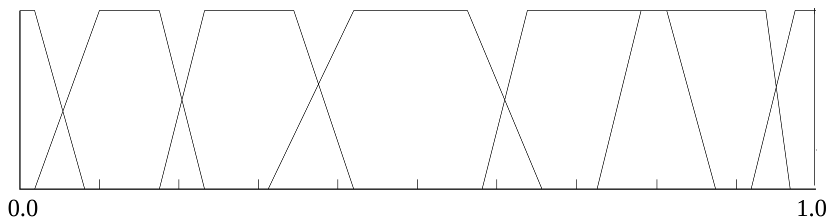

Figure 5: Membership Distribution of Elements in L

To solve the problem, a mathematical model for the assignment problem with linguistic labels may be written as

$$\begin{array}{ll} Max & \sum_{i=1}^{m}\sum_{j=1}^{m} n_{ij}x_{ij} \\ s.t. & \sum_{j=1}^{m} x_{ij} = 1 \;\; i = 1,...,m, \\ & \sum_{i=1}^{m} x_{ij} = 1 \;\; j = 1,...,m, \\ & x_{ij} \in \{0,1\} \;\; i,j = 1,...,m, \end{array} \tag{17}$$

where n_{ij} are the fuzzy numbers defined on the [0,1] interval and associated to the labels $e_{ij} \in L$.

Basically, we have two different approaches for handling with the objective function. The first, considering the objective as the sum of the fuzzy numbers associated with the labels e_{ij}, such that we would have a fuzzy objective. The second combining the labels in order to obtain another label which values the combination workers-jobs in each allocation of the variables. For this second, there exist basically two different approaches for aggregation and comparison of linguistic values: The first one acts by direct computations on labels whereas the second uses the associated membership functions. Most available techniques belong to the last kind, however, the final results of those methods are fuzzy sets which do not correspond to any label in the original term set. If one want to have finally a label, a "linguistic approximation" is needed [Zad75, Zad79]. There are neither general criteria to evaluate the goodness of an approximation nor a general method for associating a label to a fuzzy set, so that specific problems may require specifically developed methods. A review of some methods for linguistic approximation may be found in [Zad75]. It may be used the criterion of minimizing the euclidean distance between the $\mathbf{R}^2$-points (gravity center, area) associated to the mean fuzzy number obtained as ,$(\sum_{i=1}^{m}\sum_{j=1}^{m} n_{ij}x_{ij})/m$ [Bon85, Del90], obtained in this way the linguistic valuations of the assignments.

Following the ideas presented in the section 4.1 in order to obtain the selection probabilities, these are obtained either from the fuzzy objective as sum of fuzzy numbers associated to labels, with evaluation function

$$E(x) = \sum_{i=1}^{m}\sum_{j=1}^{m} n_{ij}x_{ij}$$

or from the fuzzy numbers associated to the linguistic valuations of the assignments, with evalutation function $E(\cdot), E : \mathbf{S} \to \mathbf{F}(L)$, where $\mathbf{F}(L)$ denote the set of fuzzy numbers associated with the term set.

In [Mic92, Vig91] it is described the algorithm Genetic-2, a GA for linear transportation problems. It is possible to design a genetic representation of the assignment problem with linguistic labels according this GA with the corresponding modification for obtaining the selection probabilities form the fuzzy evaluation function.

5. Conclusions

In this paper have been presented GA, which have received a great deal of attention regarding their potential as optimization techniques. The level of interest and success in this area has led to a number of improvements and progress in the use of GA. We have proposed a method in order to obtain the selection probabilities from fuzzy fitness associated to the individuals of the populations.

We have reviewed in summary form the application of GA to optimization problems, and a description of some applications of GA to fuzzy optimization problems has been presented.

The use of GA may give a great potential to fuzzy logic based optimization approach for representing uncertainty and approximation in relationnships among system variables, because of the potential of GA in fuzzy environment as a flexibility tool for optimization and search.

References

[Ada80] J.M. Adamo, Fuzzy Decision Trees. Fuzzy Sets and Systems 4, 1980, 207-219.

[Bac91] T. Back, F. Hoffmeister, Extended Selection Mechanisms in Genetic Algorithms. In: R.K. Belew, L.B. Booker, Eds., Proc. of the Fourth Int. Conf. on Genetic Algorithms. Morgan Kaufmann Publishers, San Mateo, 1991, 92-99.

[Baz77] M. S. Bazaraa, J.J. Jarvis, Linear Programming and Network Flows. John Wiley & Sons, 1977.

[Bey82] R. Beyth-Marom, How Probable is Probable? A Numerical Taxonomy Translation of Verbal Probability Expressions. Journal of Foecasting 1 (1982) 257-269.

[Bon85] P.P. Bonissone, K.S. Decker, Selecting Uncertainty Calculi and Granularity: An Experiment in Trading-off Precision and Complexity. KBS Working Paper, General Electric Corporate Research and Development Center. Schenectady, New York (1985).

[Bor85] G. Bortolan, R. Degani, A Review of some Methods for Ranking Fuzzy Subsets. Fuzzy Sets and Systems 15 (1985) 1-20.

[Cam92] L. Campos, A. Gonzalez, M.A. Vila, On the use of the Ranking Function approach to solve Fuzzy Matrix Games in a direct way. Fuzzy Sets and Systems 49 (1992) 193-203.

[Cha82] S. Chanas, W. Kolodziejczyk, Maximum Flow in a Network with Fuzzy Arc Capacities. Fuzzy Sets and Systems 8 (1982) 165-173.

[Cha84] S. Chanas, W. Kolodziejczyk, Real-valued Flows in a Network with Fuzzy Arc Capacities. Fuzzy Sets and Systems 13 (1984) 139-151.

[Cha87] S. Chanas, Fuzzy Optimization in Networks. In: J. Kacprzyk and S: Orlovski, Eds., Optimization Models using Fuzzy Sets and Possibility Theory. Reidel P. Co., Boston, 1987, 303-327.

[Chr75] N. Christofides, Graph Theory. An Algorithmic Approach. Academic Press, 1975.

[Dav91] L. Davis, Handbook of Genetic Algorithms.Van Nostrand Reinhold, New York, 1991.

[Dub80] D. Dubois, H. Prade, Fuzzy Sets and Systems. Theory and Applications. Academic Press. New York, 1980.

[Del90] M. Delgado, J.L. Verdegay, M.A. Vila, Linguistic Assessments in Decision Making Problems. Technical Report N. 90-1-2, December 5, 1990.

[Esh93] L.J. Eshelman, J.D. Schaffer, Real-Coded Genetic Algorithms and Interval-Schemata. En: Foundations of Genetic Algorithms-2, L.Darrell Whitley (Ed.), Morgan Kaufmann Publishers, San Mateo, 1993, 187-202.

[Fed91] M. Fedrizzi, J. Kacprzyk, J.L. Verdegay, A Survey of Fuzzy Optimization and Fuzzy Mathematical Programming. In: M. Fedrizzi, J. Kacprzyk, M. Roubens, Eds., Interactive Fuzzy Optimization. Springer Verlag, Berlin, 1991, 15-28.

[For56] L.R. Ford, D.R. Fulkerson, Maximal flow through a network. Canad. J. Math 8 (1956) 399-404.

[Gol89] D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, New York, 1989.

[Gon90] A. Gonzalez, A study of the Ranking Function approach through Mean Values. Fuzzy Sets and Systems 35 (1990) 29-41.

[Gon91] A. Gonzalez, A. Vila, A discrete Method to study Indifference and Order Relation between Fuzzy Numbers. Information Sciences 56 (1991) 245-288.

[Hol75] J.H. Holland, Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor, 1975.

[Jan91] C.Z. Janikow and Z. Michalewicz, An Experimental Comparison for Binary Floating Point Representations in Genetic Algorithms. In: R.K. Belew, L.B. Booker, Eds., Proc. of the Fourth Int. Conf. on Genetic Algorithms. Morgan Kaufmann Publishers, San Mateo, 1991, 31-36.

[Lie91] G.E. Lieppins, W.D. Potter, A Genetic Algorithm Approach to Multiple-Fault Diagnosis. In: L. Davis, Ed., Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York, 1991.

[Mic91] Z. Michalewicz, C.Z. Janikow, Handling Constraints in Genetic Algorithms. In: R.K. Belew, L.B. Booker, Eds., Proc. of the Fourth Int. Conf. on Genetic Algorithms. Morgan Kaufmann Publishers, San Mateo, 1991, 151-157.

[Mic92] Z. Michalewicz, Genetic Algorithms + Data Structures = Evoluton Programs. Springer Verlag, Berlin, 1992.

[Nak91] R. Nakano and T. Yamada, Conventional Genetic Algorithms for Job Shop Problems. In: R.K. Belew, L.B. Booker, Eds., Proc. of the Fourth Int. Conf. on Genetic Algorithms. Morgan Kaufmann Publishers, San Mateo, 1991, 474-479.

[Ric89] J.T. Richardson, M.R. Palmer, G. Liepins, and M. Hilliard, Some Gidelines for Genetic Algorithms with Penalty Functions. In: J. Schaffer, Ed., Proc. of the Third Int. Conf. on Genetic Algorithms. Morgan Kaufmann Publishers, 1989, 191-197.

[Sie89] W. Siedlecki, J. Skalanski, constrained Genetic Optimization via Dynamic Reward-Penalty Balancig and Its Use in Pattern Recognition. In: J. Schaffer, Ed., Proc. of the Third Int. Conf. on Genetic Algorithms. Morgan Kaufmann Publishers, 1989, 141-150.

[Tsu81] Y. Tsumura, T. Terano, M. Sugeno, Fuzzy fault Tree Analysis. Summary of papers on general fuzzy problems. Report n.7, (1981) 21-25.

[Vig91] G.A. Vignaux, Z. Michalewicz, A Genetic Algorithm for the Linear Transportation Problem. IEEE Transactions on Systems, Man, and Cybernetics 21 2 (1991) 445-452.

[Vol93] Volker Nissen, Evolutionary Algorithms in Management Science: An Overview and List of References. Papers on Economics and Evolution #9303, Universitaet Goettingen, Germany, July 1993.

[Yag78] R.R. Yager, Ranking Fuzzy Subset over the Unit Interval. Proc. CDC, (1978) 1735-1737.

[Yag81] R.R. Yager, A Procedure for Ordering Fuzzy Subsets of the Unit Interval. Information Sciences 24 (1981) 143-161.

[Zad75] L. Zadeh, The Concept of a Linguistic Variable and its Applications to Approximate Reasoning. Part I, Information Sciences 8, (1975), 199-249, Part II, Information Sciences 8, (1975), 301-357, Part III, Information Sciences 9 (1975) 43-80.

[Zad79] L. Zadeh, Fuzzy Sets and Information Granularity. In: M.M. Gupta et al. Eds., Advances in Fuzzy Sets Theory and Applications. North Holland Publishing Company, New York, 1979 3-18.

[Zhu92] Q. Zhu, E.S. Lee, Comparison and Ranking of Fuzzy Numbers. In: J. Kacprzyk and M. Fedrizzi, Eds., Fuzzy Regression Analysis. Omnitech Press Warsaw and Physica-Verlag, Heidelberg, 1992, 21-44.