Proceedings of the 8th WSEAS International Conference on APPLIED MATHEMATICS, Tenerife, Spain, December 16-18, 2005 (pp315-318)

Fuzzy Simulated Annealing for Fuzzy Optimization

MAHDI NAMAZIFAR[1], MOHAMMADREZA RAZZAZI[1]
and HASSAN JAVAHERI NEYESTANAKI [2]

[1]Department of Computer Engineering and Information Technology
Amirkabir University of Technology
[2]Department of Mining and Metallurgy, Amirkabir University of Technology

Department of Computer Eng. And IT, Amirkabir University of Technology, Hafez Avenue, Tehran. IRAN

Abstract: - Solving fuzzy optimization problems is more complex than solving crisp ones. The complexity of such problems motivates the use of approximation methods to solve them. In this paper, a simulated annealing technique is presented which is able to approximately solve fuzzy optimization problems.

Key-Words: - Simulated annealing; Fuzzy optimization; Neighborhood structure; Fuzzy number; Cooling schedule; Metaheuristics.

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 success has motivated a direct application of the systems in which a key role is played by human judgments, preferences, etc. Getting some good solution satisfying a level of performance quickly and reaching an acceptable solution based on human-like reasoning mechanisms are incentives for combination of fuzzy theory and optimization.

Optimization of fuzzy problems has a significant role in many fuzzy systems. In literature, one can find several kinds of fuzzy optimization problems [1-7], and also different approaches of solution have been proposed too. Among these approaches, fuzzy genetic algorithm (FGA) [4] and fuzzy tabu search (FTS) [8] can be mentioned. These methods can produce good approximate solution in solving fuzzy optimization.

Here, we introduce a simulated annealing (SA) method; a stochastic global optimization method originally developed by Kirkpatrick *et al.* [9] for very large combinatorial optimization problems, and extends it to continuous-valued functions for the fuzzy optimization problem. We denote this method as FSA in the rest of this paper, for notational convenience.

FSA can be used to optimize fuzzy equations and linear programming models, which their variables and parameters are fuzzy. It also may be used for other applications like fuzzy regression and fuzzy neural networks.

The rest of this paper is organized as follows. In next section the SA technique is briefly described. In section 3, our FSA algorithm is presented. Finally, Section 4 concludes the paper.

Now we introduce some basic notations to be used in this paper. We place a bar over a symbol if it represents a fuzzy set. So, $\underline{X}$ represents a fuzzy set. All of the fuzzy sets are subsets of real numbers. If $\underline{A}$ is a fuzzy set, then $\underline{A}_x$ denotes the membership degree of x to A. A triangular fuzzy number $\underline{N}$ is defined by three numbers $a < b < c$, where (1) $\underline{N}_x = 0$ for $x \le a$ and $x \ge c$, and $\underline{N}_b = 1$; and (2) the graph of $\underline{N}_x = y$ is a straight line segment from $(b,1)$ to $(c,0)$ on $[b,c]$. We write $\underline{N} = (a,b,c)$ for triangular fuzzy numbers. We use the standard arithmetic of fuzzy sets based on the extension principle.

2 Simulated annealing: an overview

Simulated annealing is a computational stochastic technique for achieving near global optimum

solutions to combinatorial and function optimization problems. The idea of the method is borrowed from the thermodynamic process of cooling (annealing) of molten metals to attain the lowest free energy state [9]. When molten metal is cooled slowly enough, it tends to solidify in a structure of minimum energy. This annealing process is mimicked by a search strategy. The key principle of the method is to allow occasional worsening moves so that these can eventually help locate the neighborhood to the true (global) minimum [10]. The associate mechanism is given by the Boltzman probability, which is:

$$\text{Probability}(P) = \exp\left(\frac{-\Delta E}{K_B T}\right) \qquad (2.1)$$

where ΔE is the change in the energy value from one point to the next, K_B the Boltzman's constant and T the temperature (control parameter). For the purpose of optimization the energy term, ΔE refers to the value of the objective function and the temperature, T, is a control parameter that regulates the process of annealing. The consideration of such a probability distribution leads to the generation of a Markov chain of points in the problem domain. The acceptance criterion given by Eq. (2.1) is popularly referred to as the Metropolis criterion [11]. Another variant of this acceptance criterion (for both improving and deteriorating moves) has been proposed by [12] and can be written as:

$$\text{Probability}(P) = \frac{\exp(-\Delta E/T)}{1+\exp(-\Delta E/T)} \qquad (2.2)$$

In simulated annealing search strategy: at the start any move is accepted. This allows us to explore solution space. Then, gradually the temperature is reduced which means that one becomes more and more selective in accepting new solution. By the end, only the improving moves are accepted in practice. The temperature is systematically lowered using a problem-dependent schedule characterized by a set of decreasing temperatures. In [13], Prilot discussed more about the parameters used in simulated annealing algorithms. Due to its simplicity and versatility, simulated annealing has the distinction of being one of the most widely used techniques for both function and combinatorial optimization problems.

So, the SA requires the following basic elements to be defined:

- *Configuration* is a solution or an assignment of values to variables.
- A *move* is a transition ($s' \to s''$) from one trial solution (s') to another (s'').
- Set of *candidate moves* (*neighborhood, or trial solutions*) is the set of all possible moves out of a current configuration.
- *Simulated annealing parameters*, such as cooling schedule and K_B .
- *Termination criteria* which determines when the SA is terminated.

Given the above basic elements, the SA schemes can be described as follows. Start with a certain configuration, evaluate the objective function for that configuration, then follow a certain candidate move. If the move improves the objective function then pick that move and consider it to be the new current configuration; otherwise, pick the move with the probability of $\text{Probability}(P)$ to be the new current configuration, and drop it with the probability of $\text{Probability}(P)$. Repeat the procedure until the termination criteria is satisfied.

On termination, the best solution obtained so far is the solution obtained by the SA approach. A recent trend in the SA field is the use of different cooling schedules [14], which use different functions for T to decrease. Because of its simplicity, geometric cooling schedule is the most frequently used cooling schedule in SA.

3 Simulated annealing for fuzzy optimization

We can express a general function F with fuzzy input and output as

$$\overline{Y} = F\left(\overline{X}\right) \qquad (3.1)$$

Where $\overline{X}$ is the input and it is a fuzzy subset of some interval $[0, M]$, $M > 0$. $\overline{Y}$ is the output from F given $\overline{X}$. In this section it is assumed that there is only one independent in the equation. It can easily be generalized to more variables, as is done in [8]. Also, any interval for $\overline{X}$ can be used instead of $[0, M]$. The interval $[0, M]$ is specified just for notational convenience.

The fuzzy optimization problems are more complicated than crisp ones. On the other hand, optimal solution of an optimization problem can not

be obtained by differentiating the objective function with respect to the fuzzy variables. So, other methods for solving fuzzy optimization problems should be introduced. In this paper we propose a SA approach for solving fuzzy optimization problems.

Here we use the same formulation for the problem as is used in [8]. We are about to find $\overline{X}$ in $[0, M]$ to maximize $\overline{Y}$. However, we can not maximize $\overline{Y}$ since it is a fuzzy set. So, the centroid of $\overline{Y}$ is used to measure the largeness of $\overline{Y}$. Now we wish to find $\overline{X}$ in $[0, M]$ to maximize the centroid ω

$$\omega = centroid(\overline{Y}) \qquad (3.2)$$

ω is the objective function in the FSA. Next, the same discretization work for the fuzzy set $\overline{X}$ is done. Supposing N a positive integer, we choose

$$z_i = i \times M / N, \qquad 0 \le i \le N \qquad (3.3)$$

Thus we discretize $[0, M]$ into $N+1$ points z_i, with $z_0 = 0$ and $z_N = M$. We will now use a vector $\overline{X}$, comprised by the $N+1$ points, to represent fuzzy variable as shown below:

$$\overline{X} = (x_0, x_1, ..., x_N) \qquad (3.4)$$

Where

$$x_i = \overline{X}(z_i) \qquad (3.5)$$

We input the discrete version of $\overline{X}$ to F and obtain ω of $F(x_0, x_1, ..., x_N)$. So, F maps $[0,1]^{N+1}$ into the real numbers. Now we are about to find a vector $\overline{X}$ in $[0,1]^{N+1}$ to maximize ω. FSA will be designed to find this vector.

Before describing the FSA, some important issues should be addressed.

In our notation $\overline{X}_{Curr}$ and ω_{Curr} denote current solution and current objective value, respectively. Similarly, $\overline{X}_{Trial}$, $\overline{X}_{Best}$ and ω_{Trial}, ω_{Best} represent trial solution, best solution, trial objective value and best objective value, respectively. One of the problems FSA faces is generating a trial solution, given a current solution $\overline{X}_{Curr}$. Here, we use a neighborhood structure as described in [15]. The solution space $S = [0,1]^{N+1}$ is partitioned into disjunct cells by division of the coordinate interval along the $x_0, x_1, ..., x_N$ axes into $p_0, p_1, ..., p_N$ parts. The problem specific partition parameter $P = (p_0, p_1, ..., p_N)$ determines a unique partition of S into cells. Thus specifies the address of each cell. Thus the address of a cell can be expressed as the following array A.

n_0	n_1	n_2	…	n_N

Where $1 \le n_i \le p_i$, $1 \le i \le N$ denotes the n_i th interval of the i th axis. We use the following strategy to choose cell at each iteration: given $\overline{X}_{Curr}$, which belongs to cell A_c, and a probability threshold P, for $i = 0,1,2,...N$, draw a random number $r \sim u(0,1)$, if $r < P$, then $A_t(i) = A_c(i)$; otherwise, draw randomly an integer l from the following set $\{l : l = 1,2,..., p_i, l \ne A_c(i)\}$, and let $A_t(i) = l$, where A_c and A_t are the address of the current and trial solutions.

The probability threshold (P) controls the shake-up that is performed on a certain solution to choose a neighbor. The higher the value of P, the less shake-up is allowed and consequently the closer the neighbor to the current solution and vice versa.

We choose geometric cooling schedule for our FSA with the coefficient of 0.9. As the termination criteria, we choose number of iterations and select 1000 iterations as the termination criteria.

Now, we present our FSA algorithm:

1. Initialization: Let $\overline{X}_{Curr}$ be an arbitrary initial solution generated randomly form $[0,1]^{N+1}$ and ω_{Curr} be the corresponding objective function value. Let $\overline{X}_{Best} = \overline{X}_{Curr}$ and $\omega_{Best} = \omega_{Curr}$. Select values for the following parameters: P (probability threshold), cooling schedule and termination criteria. Go to step 2.
2. Using $\overline{X}_{Curr}$, generate the neighbor of $\overline{X}_{Curr}$, $\overline{X}_{Trial}$, and evaluate its objective function value, ω_{Trial}. Go to step 3.
3. Do
   If $\omega_{Trial} > \omega_{Best}$, set $\overline{X}_{Curr} = \overline{X}_{Trial}$, $\overline{X}_{Best} = \overline{X}_{Curr}$, and $\omega_{Best} = \omega_{Curr}$. Otherwise,

with the probability of $\exp(-\Delta E/T)$, set $\overline{X}_{Curr} = \overline{X}_{Trial}$, $\overline{X}_{Best} = \overline{X}_{Curr}$, and $\omega_{Best} = \omega_{Curr}$. Update the temperature (T) in accordance with the cooling schedule.

Until the termination criteria is satisfied.

The above algorithm is designed for solving fuzzy optimization problems when the problem includes fuzzy variables which are fuzzy subsets of $[0, M]$. As described above, this algorithm can easily be modified to solve optimization problems including fuzzy variables which are fuzzy subsets of any other interval.

4 Conclusion

In this paper a simulated annealing method for approximately solving fuzzy optimization problems is introduced. We named this algorithm FSA. This algorithm can be used to optimize a general equation. Also, FSA can be used for optimizing fuzzy linear programming models, which their variables and parameters are fuzzy. It also may be used for other applications like fuzzy regression in which a fuzzy error function should be minimized. Fuzzy neural networks, also, can use this algorithm.

References:

[1] J.J. Buchley, T. Feuring, Linear and non-linear fuzzy regression: evolutionary algorithm solutions, Fuzzy Sets Systems, 112 (2000) 381–394.

[2] J.J. Buckley, T. Feuring, Y. Hayashi, Neural net solutions to fuzzy linear programming, *Fuzzy Sets Systems*, Vol. 106, 1999, pp. 99–111.

[3] J.J. Buckley, H. Hayashi, Applications of fuzzy chaos to fuzzy simulation, *Fuzzy Sets Systems*, Vol. 99, 1998, pp. 151–157.

[4] J.J. Buckley, Y. Hayashi, Fuzzy genetic algorithm and applications, *Fuzzy Sets Systems*, Vol. 61, 1994, pp. 129–1436.

[5] M. Delgado, J.L. Verdegay, M.A. Vila, A general model for fuzzy linear programming, *Fuzzy Sets Systems*, Vol. 29, 1989, pp. 21–29.

[6] E. Lee, R.J. Li, Fuzzy multiple objective programming and compromise programming with Pareto optimum, *Fuzzy Sets Systems*, Vol. 53, 1993, pp. 275–288.

[7] H.R. Maleki, M. Tata, M. Mashinchi, Linear programming with fuzzy variables, *Fuzzy Sets Systems*, Vol. 109, 2000, pp. 21–33.

[8] Chunguang Li, Xiaofeng Liao, Juebang Yu, Tabu search for fuzzy optimization and applications, *Information Sciences*, Vol. 158, 2004, pp. 3–13.

[9] Kirkpatrick, S., Gelatt, C.C., Vecchi, M.P., Optimization by simulated annealing. *Science*, Vol. 220, 1983, pp. 671–679.

[10] Balram Suman, Study of simulated annealing based algorithms for multi-objective optimization of a constrained problem, *Computers and Chemical Engineering*, Vol. 28, 2004, pp. 1849–1871

[11] Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A. and Teller, E., Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, Vol. 21, 1953, p. 1087.

[12] Galuber, R. J., Time dependent statistics of the using model. *Journal of Mathematical Physics*, Vol. 4, 1963, p. 294.

[13] Prilot, M., General local search method. *European Journal of Operation Research*, Vol. 92, 1996, p. 493.

[14] E. Triki, Y. Collette, P. Siarry, A theoretical study on the behavior of simulated annealing leading to a new cooling schedule, *European Journal of Operational Research*, Vol. 166 2005, pp. 77–92.

[15] D. Cvijovic, J. Klinowski, Taboo search: an approach to the multiple minima problem, *Science*, Vol. 267, 1995, pp. 664–666.