Optimal Population Size for Genetic Algorithm Using Fuzzy System

Emad S. Jabber

Computer Science Department, College of science, Basrah university, Basrah / Iraq

Abstract

A genetic algorithm (GA) uses the idea of biological evolution to seek good solutions to problems with very large search spaces. It has the following parameters: population size, crossover rate, and mutation rate. The selection of the initial parameters for a GA is very difficult. Some attempts to find optimal combination of parameters used trial and error methods or combination approaches, while others used a GA to find optimal parameters for another GA.
The current work uses fuzzy system to determine optimal population size for any problem which is used the Genetic Algorithm. Two combinatorial problems with a large search space are used to test the effectiveness of the current work. The results are validated and GA is shown to be effective for the tested problems.

Keywords – genetic algorithm, adaptive GA, adaptive crossover, traveling salesman problem, knapsack problem.

أفضل حجم مجتمع للخوارزمية الجينية باستخدام النظام المضبب

عماد شعـلان جبر

قسـم علـوم الحاسـبات، كليـ ـ ـة العلوم، جامعة البصرة، البصرة / العراق

المستخلص

تستخدم الخوارزمية الجينية فكرة التطور البايولوجي للبحث عـن حلـول جيـده للمسألة مع مجالات بحثية واسعة ،حيث إنها تمتلك المعلمات التالية : حجم المجتمع، معـدل التزاوج و معدل الطفرة .

تعتبر عملية اختيار المعلمات الابتدائية للخوارزمية الجينية صعبة جـدا، حيـث هناك بعض المحاولات لإيجاد التوافق الأمثل للمعلمات منها اسـتخدام طريقـة التجربـة والخطأ وطرق توافقية، بينما تستخدم طرق أخرى الخوارزمية الجينية لإيجـاد المعلمـات المثلى لخوارزمية جينية أخرى. العمل الحالي يستخدم النظام المضبب لتحديد أمثل حجـم مجتمع لأي مسالة تستخدم الخوارزمية الجينية. استخدمت مسألتان توافقيتان تمتلكان فضـاء بحث واسع لاختبار تأثيرات العمل الحالي، حيث أظهرت الخوارزمية الجينية فاعلية للمسائل التي تم اختبارها و كانت النتائج قانونية.

1. Introduction

The Genetic Algorithm (GA) is a method of computation that simulates biological evolution [Davis, 1991]. This method is typically used to optimize functions that are intractable or have large or unknown search spaces. Despite years of successful application to a wide array of problems, little consensus has been generated concerning the optimum population size that should be used [Davis, 1991, Hinterding et al. 1997].

Typically, choosing the population size of GA is a fundamental decision faced by all GA users. On the one hand, if too small a population size is selected, the GA will converge too quickly, with insufficient processing of two few schemata. On the other hand, a population with too many members results in long waiting times for significant improvement, especially must be performed wholly or partially in serial, the population is too large to get enough mixing of building blocks per unit of computation time [Grefenststte, 1986, Stanley and Bart, 2003]

The problem of finding the optimal population size cannot be solved in isolation, since it is connected to a number of different parts of the GA and the problem at hand, some of these parts [Grefenstette, 1986, Schmidt and Stidsen, 1996, Stanley and Bart, 2003] are :

1. Chromosome length (number of genes in the chromosome).
2. The number of the constant genes in the chromosome.
3. The range of gene in the chromosome (e.g. in binary coding the range of gene =2).

The goal of this research is to determine the optimum population size for a Genetic Algorithm by the Fuzzy Logic, when using the previous parts as parameters to the Fuzzy Logic.

2. *Fuzzy Logic*

Fuzzy Logic (FL), which was formulated by Zadah (1965), was developed as a response to the fact that most of the parameters we encounter in the real would are not precisely defined. As such, it gives a framework for approximate reasoning and allows qualitative knowledge about a problem to be translated into an executable set of rules. This reasoning and rule base approach, which is known as a fuzzy inference system, is then used to respond to new inputs [Reznik, 1997, Stanley and Bart, 2003].

The Fuzzy Inference System (FIS) is a popular methodology for implementing FISs are also known as fuzzy rule base systems, fuzzy expert systems, fuzzy models, fuzzy associative memories, or fuzzy logic controllers when used as controllers [Passiono and Yurkovich, 1998, Reznik, 1997].

The essence of the system can be represented as shown in the figure (1)

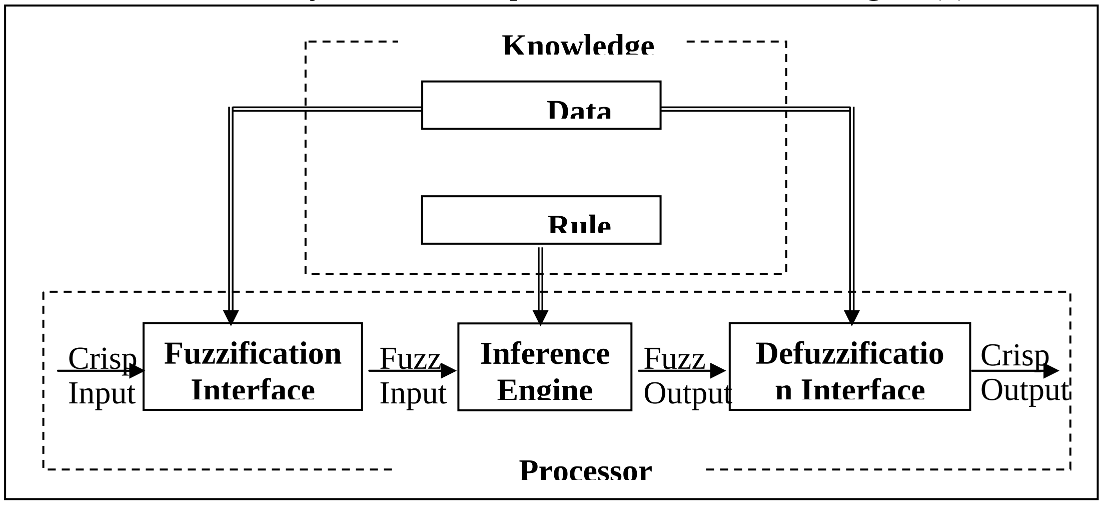

Figure (1): A Fuzzy Inference System (FIS)

As indicated in the figure (1), the FIS can be envisioned as involving a knowledge base and a processing stage. The knowledge base provides the membership function (MFs) and fuzzy rules needed for the process.

In the processing stage, numerical crisp variables are the input of the system. These variables are passed through a fuzzification stage where they are transformed to linguistic variables, which become the fuzzy input for the inference engine. This fuzzy input is transformed by the rules of the inference engine to fuzzy output. The linguistic results are then changed by a defuzzification stage into numerical values that become the output of the system [Passion and Yurkovich, 1998, Schmidt and Stidsen, 1996].

3. The Current approach

The basic idea of the current approach is to use fuzzy system to determine the optimal population size to any genetic algorithm depending on the actual number of genes in the chromosome (the chromosome length – the constant genes in this chromosome) , and the range of gene in the chromosome.

A simple scheme is to use Very High Population size when actual number of genes in the chromosome is Very High and the range of gene is High. On the other hand, the population size is Very Low when the actual number of gene is Very Low and the range of gene is Low. Typically, the population size is increased when the two above parameters are increased and a vice versa.

The fuzzy rule structure considered is the usual Mamdani type rule with one input variable (Actual Number of Gene) and two outputs variable (Range of Gene). The graphical representations of possible fuzzy partitions are stated in figure (2).

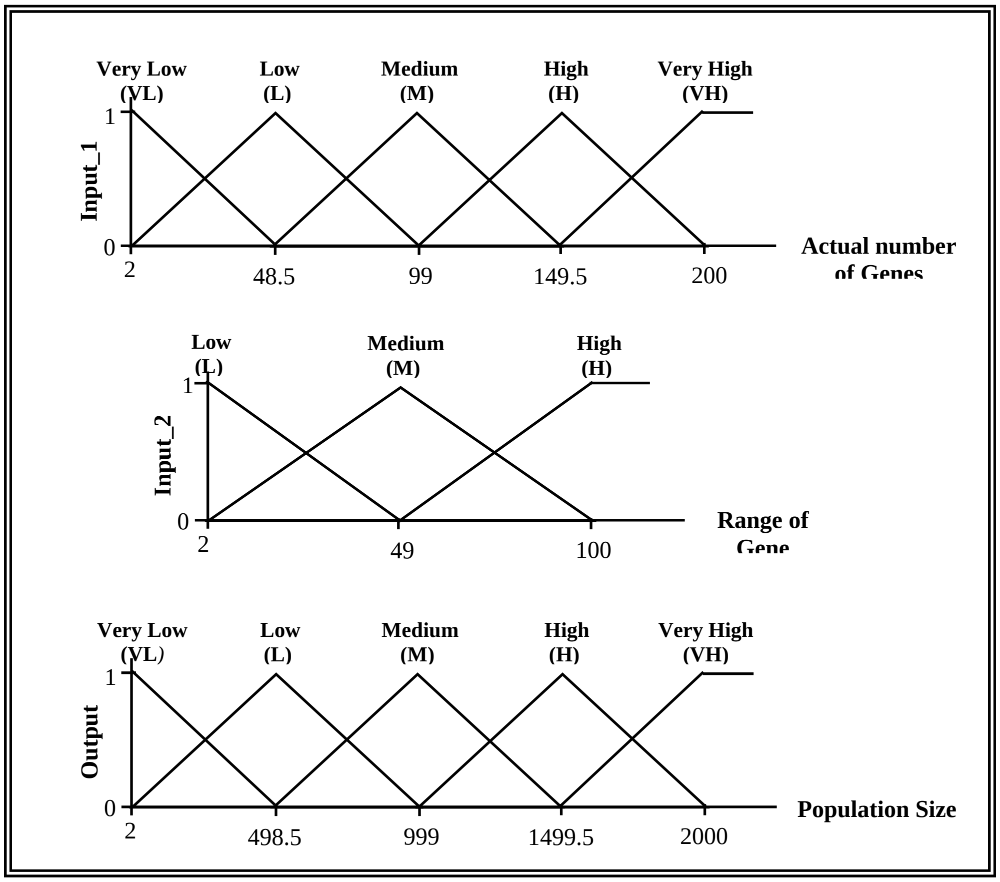

Figure (2) The representations of the inputs and output fuzzy partitions

Table (1) represents the rules that give the previous information.

Input_1 / **Input_2**	**VL**	**L**	**M**	**H**	**VH**
L	***VL***	***L***	***M***	***M***	***H***
M	***L***	***L***	***M***	***H***	***VH***
H	***L***	***M***	***H***	***H***	***VH***

Table (1) Fuzzy Rules for the Current Approach

4. Test results

To show the efficiency of current work, we used two well known combinatorial optimization problems: 0/1 knapsack problem and traveling salesman problem .These problems are chosen because they are well known hard NP-complete problem, they allow us to compare results with previous work available [Julstrom, 1998, Michalewicz, 1992], and uses two different encoding for chromosome (binary, integer) respectively [Obitko, 1998] as state below:

5. 0/1 Knapsack problem

This problem is chosen because it is well known to be hard and non polynomial – time algorithm exists to be solved optimally. So this falls under NP-Complete set of problem. This problem is a combinatorial problem with a large search space that can be attacked from GA. In this problem there exists a set of N items available to be packed into knapsack with capacity C units. Each item has values (v) and takes up (w) units of weights. The idea is to find a set of L items that should be packed into the knapsack to maximize the total value without exceeding the knapsack's weight capacity [Julstrom, 1998, Rossiter, 1998].

The test data used for all tests between the current and Rossiter approaches is the same and it is shown in the Table (2). There are 50 items and the knapsack capacity is 625.

item	*Vi*	Wi	*item*	*Vi*	Wi	*item*	*Vi*	Wi	*item*	*Vi*	Wi	*item*	*Vi*	Wi
1	49	21	11	11	45	21	32	18	31	28	21	41	15	8
2	23	1	12	4	42	22	32	3	32	42	47	42	48	48
3	21	4	13	23	29	23	21	38	33	22	9	43	39	5
4	37	2	14	27	36	24	28	11	34	35	29	44	3	0
5	28	28	15	15	43	25	0	35	35	13	26	45	40	45
6	27	47	16	27	14	26	25	13	36	14	47	46	28	20
7	21	3	17	17	1	27	30	29	37	46	46	47	16	22
8	6	23	18	17	38	28	5	3	38	2	19	48	17	40
9	38	12	19	5	4	29	11	32	39	34	30	49	31	4
10	7	12	20	41	33	30	41	26	40	32	34	50	19	49

Sum of values :	1192
Sum of weights :	1193

Table (2) Test data for 0/1 knapsack problem with 50 items [Rossiter,1998]

Table (3) shows the parameters and the crossover form used by the current and Rossiter approaches. The length of chromosome is equal to the number of items. The binary encoding is used for chromosome , each gene represent one item and the value of gene determine if this item in knapsack or not . The fitness function is the total value of selected items if its weight is less or equal knapsack capacity else zero. The mutation operator selects two points at random to exchange them. The algorithm runs until it has arrived the maximum generations or the best chromosome was repeated more than 50 sequence generations.

Approach	**Population size**	**Pc**	**Pm**	**Crossover form**	**Max. generation**
Rossiter[Rossiter,1998]	20000	%70	%30	UX	100
The Current	28	%85	%15	UX	250

Table (3) Parameters and Crossover form used by the current and Rossiter approaches

The results of the current approach compared with Rossiter approach for ten tests are shown in table (4).

Approach		**Best**	**TEST NO.**										**Avg V**	**Avg W**
			1	2	3	4	5	6	7	8	9	10		
Rossiter[Rossiter,1998]	GA with elitism	***Value***	***982***	***982***	***982***	***982***	***982***	***982***	***982***	***982***	***982***	***982***	982	623
		weight	***623***	***623***	***623***	***623***	***623***	***623***	***623***	***623***	***623***	***623***		
	GA without elitism	*Value*	***974***	***974***	***973***	***974***	***974***	***976***	***977***	***972***	***976***	***976***	975	621
		weight	***620***	***620***	***623***	***619***	***619***	***623***	***620***	***621***	***623***	***623***		
	Greedy heuristic	*Value*	***966***	***966***									966	606
		weight	***606***	***606***										
The Current		*Value*	***982***	***982***	***982***	***982***	***982***	***982***	***982***	***982***	***982***	***982***	982	623
		weight	***623***	***623***	***623***	***623***	***623***	***623***	***623***	***623***	***623***	***623***		

Table (4) Results of the current approach compared with Rossiter approach for the 0/1 knapsack problem

The items are chosen by the current approach for all tests are:
1 , 2 , 3 , 4 , 5, 7 , 9 , 13 , 16 , 17 , 19 , 20 , 21 , 22 , 24 , 26 , 27 , 28 , 30 , 31, 32 , 33 , 34 , 37 , 39 , 40 , 41 , 42 , 43 , 44 , 45 , 46 , 47 , 49 .

For insurance the ability of the current approach to find optimal solution for test problem with small population size, it runs three times, each run has different test data by randomly shuffles the original test data that is shown in Table (2). For all runs, the current approach gives the same optimal solution with maximum value 982 and weight 623, but the chosen items are different for each run because the order of items is changed.

- ***Traveling salesman problem (TSP)***

The traveling salesman problem (TSP) is considered a standard hard NP-complete test problem for search techniques. Given a salesman who has to visit a number of towns exactly once and return to the start city with a shortest tour [Michalewicz, 1992].

We choose three well known problems for testing the effectiveness of the current approach. These test problems were chosen because there was data available in the literature to compare the results. The original city data and the optimal solution for all problems are available via the web site (http://www.informatik.uni-heidelberg.de/groups/compot /software / TSPLIB 95 / STSP.html). The error percentage was calculated by equation (1).

%Error = ________________________ ***. . . (1)***

The algorithm encodes tours in the obvious way, as permutations. The chromosome length is exactly the number of cities. The integer encoding is used for chromosome that represent the sequence of cities that salesman visit. Every gene is an integer $\in$[1, no. of cities]. The crossover operator is uniform crossover and mutation operator selects two points at random to exchange them. The fitness function was a summation of distance of the tour. The algorithm runs until it has arrived the maximum number of generations. Table (5) summarizes the results of these tests compared with optimal known solutions.

Chapter 2 Problem			**Population size**	**generation**	**Current approach**	
Name	**No. cities**	**Optimal**			Best	**%Error**
Bier127	127	118282	1500	2500	118636	0.002
Lin105	105	14379	780	2000	14602.66	0.015
Berlin52	52	7542	610	1500	7544.37	0.0003

Table (5) Results of the current approach for TSPs

6. *Conclusions*

In this paper, we introduce a new approach to find optimal population size for any GA. It used fuzzy system to add good characteristic to GA to preserve population diversity by finding the optimal population size, as this helps to avoid premature convergence. We tested two well known combinatorial optimization problems (0/1 knapsack problem and TSP). The experimental results have proved the strength of work in terms of solution quality and speed of conversion, as state below:

For the 0/1 knapsack problem , the current approach produces results compared with the best solution for another genetic algorithm and greedy heuristic on the same data. The current approach gives better and faster results for all ten tests with small population size 28 compared with Rossiter that used very large population size 20000 , which did not obtain optimal solution when he used greedy heuristic .The current approach takes more generations than Rossiter because the big difference between both the population size that used .

When we shuffles the items order of the test data the current approach gives the same results with maximum value 982.This ensures the ability of it to find optimal solution .

The current approach proved that it is able to find optimal result with small population size for problems with large search space.

For the TSP we used the current approach, the results are compared with the optimum or the best known solutions. It gives good error compared with the optimal .It gives error between 0.0003%-0.015. The results have proved the effectiveness of the current approach to find near optimal solution with small size of population and little generations.

References

1. [Davis, 1991] L. Davis, (1991), ***"Handbook of genetic algorithms "***. New York: Van Nostrand Reinbold.

2. [Grefenstette, 1986] J.J.Grefenstette, (1986),***"Optimization of control parameters for genetic algorithms"***. IEEE Trans. on System, Man, and Cybernetics, vol. 16, no. 1, pp.122-128.

3. [Hinterding *et al.,* 1997] R. Hinterding , Z. Michalewicz , and A. E. Eiben, (1997) , "***Adaptation in evolutionary computation : a survey " .*** In IEEECEP: proceeding of the IEEE conference on EC, IEEE world congress on computational intelligence.

4. [Julstrom, 1998] B. Julstrom , (1998),***" Description of the 0-1 knapsack problem"***. First Genetic Algorithm project handout.

5. [Michalewicz, 1992] Z. Michalewicz, (1992), ***"Genetic algorithms + Data structures = Evolution Programs"***, Berlin: Springer-Verlag.

6. [Obitko, 1998] M. Obitko, (1998), ***"Genetic algorithms"***. Available at (www.cs.felk.cvut.cz/~xobitko/ga).

7. [Passino and Yurkovich, 1998] K. M. Passino and S. Yurkovich, (1998), ***"Fuzzy control "***. Adison Wesley Longman Inc.

8. [Reznik, 1997] L. Reznik , (1997), ***"Fuzzy controllers "***. Britain Biddles ltd.

9. [Rossiter, 1998] B. Rossiter, (1998), ***"0-1 knapsack Genetic Algorithm versus Greedy Heuristic "***. Output of internet.

10. [Schmidt and Stidsen, 1996] M. Schmidt and T. Stidsen, (1996), ***"Hybrid system: Genetic Algorithms, Neural Networks, and Fuzzy Logic"***. Denmark.

11. [Stanley and Bart, 2003] Gotshall and Rylander, (2003), ***"Optimal Population Size and the Genetic Algorithm"***. School of Engineering University of Portland, U.S.A.