# The Impact of Population Size on Knowledge Acquisition in Genetic Algorithms Paradigm: Finding Solutions in the Game of Sudoku

**Nordin Abu Bakar, Muhammad Fadhil Mahadzir**

*Faculty of Computer & Mathematical Sciences*
*Universiti Teknologi MARA (UiTM)*
*40450 Shah Alam,*
*Selangor, MALAYSIA*
*nordin@tmsk.uitm..edu.my*

## ABSTRACT

*Population size is an important component in genetic algorithms (GAs).The concept of population in GAs has contributed to a unique searching strategy which empower its search process through the massive volume of the data in a population. The purpose of this study is to see how the impact of population size on genetic algorithms in producing correct solution for a Sudoku puzzle. Sudoku is a Japanese number puzzle game that has become a worldwide phenomenon. The puzzle involves completing a grid of cells by assigning a single number to each cell. The numbers in a row or a column must consist of any one of the numbers from 1 to 9; no repetition is allowed. GA will be used to generate the correct solution of Sudoku puzzles. The mechanism to produce legal Sudoku grid will follow the requirements needed and meet all the constraints. A fitness function is designed to evaluate legal grids and GA will be tested for performance and time efficiency. The challenges lie on how GA will represent a Sudoku grid in the process and the effectiveness of its operators such as crossover and mutation. The results show how different population size can produce different solutions. The best performance is observed at 500 population size. The paper will conclude with an insight of this value and its significance to the knowledge acquisition in GA paradigm..*

**Keywords**
*Genetic algorithms, Sudoku, population size, fitness function, knowledge acquisition*

## 1. INTRODUCTION

Most researchers (Goldberg,1989;Bäck, 1996;De Jong, 1975) have agreed that the performance of GAs is dependent upon components such as the probability of the operators used, the operators themselves, and the diversification of the population. The common practice is that the parameters are set at the beginning and the process will run according to these parameters until a good solution is found. However, in most cases GAs fail to produce good results in an acceptable time frame. Since GA is modeled after the evolution process, the longer the runs take place, the results will get better.

Previous researchers have suggested the values for GAs parameters (such population size, crossover rate and mutation rate )that worked well in their research( De Jong ,1975; Grefenstette ,1986; Goldberg, 1989). The main drawback is that the problem being solved might be different and the solutions found maybe well below the optimum. The importance of the population size is due to the fact that it represents the volume of the candidate solutions that are being considered in the search. When the population size is controlled, the search space will be limited or extended. One of the components in the genetic process that will be affected by adjusting the population size is the selective pressure. If the population size is too large, the selective pressure will be reduced considerably and this will make the search ineffective. On the other hand if the population size is too small, selective pressure will be strong and this exploits certain individuals that will result in a premature convergence. The second component that is affected by changing population size is the population diversity. Whitley concluded that when the population is small, the search focuses on the top individuals in the population; as a result population diversity is lost. Therefore, it is clear that population provokes certain features whenever the population size is different. Exploitation of top individuals will occur if population size is small and exploration of new dimensions will take place if population size is large. In order to take advantage of these features, we propose the population size to be adapted as the GA runs.

## 2. METHODOLOGY

### 2.1. The Method

Genetic Algorithm is used in this study to generate the correct solution of Sudoku puzzles. The type of encoding in GA that has been used is permutation encoding. In permutation encoding, every chromosome is a string of numbers, which represents number in a sequence.

### 2.2. Initial Population

In this study, an integer array of 81 numbers is generated randomly to represents a Sudoku grid and a chromosome(Mantere & Koljonen, 2006). The array is divided into nine sub-blocks of nine numbers corresponding to the 3×3 sub squares from left to right and from top to bottom. At the starting of development process, the population size is set to 100 . But the experiments indicated that  a bigger population size is better because this population size can find a correct solution in lowest number of generation. *Figure 3.3* show the example of Sudoku grids and the integer array of 81 numbers that represent the chromosome.

**Individual:**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 2 | 3 | 6 | 8 | 5 | 4 | 9 | 7 | 1 |
| 5 | 8 | 9 | 2 | 7 | 1 | 6 | 4 | 3 |
| 4 | 1 | 7 | 3 | 6 | 9 | 8 | 2 | 5 |
| 4 | 6 | 8 | 7 | 2 | 3 | 1 | 9 | 5 |
| 1 | 2 | 5 | 9 | 6 | 4 | 7 | 3 | 8 |
| 7 | 9 | 3 | 5 | 8 | 1 | 6 | 4 | 2 |
| 6 | 8 | 7 | 5 | 4 | 9 | 3 | 1 | 2 |
| 3 | 9 | 2 | 8 | 1 | 7 | 4 | 5 | 6 |
| 1 | 5 | 4 | 2 | 3 | 6 | 9 | 7 | 8 |

**Individual:**

| | | |
|---|---|---|
| *236854971* | *589271643* | *417369825* |
| | | |
| *468723195* | *125964738* | *793581642* |
| | | |
| *687549312* | *392817456* | *154236978* |

*Figure 3.3: An example of Sudoku grid and the representation of Sudoku puzzles in GA. One possible solution (individual) is an array of 81 numbers that is divided into nine sub blocks of nine numbers.*

### 2.3. Fitness Value

The fitness value will be evaluated by using some equations. Each equation represents the constraints of Sudoku Puzzle. Fitness is scaled between 0 and 1; where a value 1 represents a perfect solution. The descriptions of each equation are as follows (Mantere & Koljonen, 2006; Moraglio & Togelius, 2007;Gold, 2005):

1) The first equation ($E_1$) is to count the duplicate number in each sub block. It is required that each sub block must have all elements from set A without repetition where A = {1,2,3,4,5,6,7,8,9}.

$$E_1(x) = 0;$$
$$While(true)\{if\ (duplicate) E_1(x) + +\}$$

2) The second equation ($E_2$) is to count the duplicate number in each row and column set. It is required that each row and column must have all elements from set A without repetition where A = {1,2,3,4,5,6,7,8,9}.

$$E_2(x) = 0;$$
$$While(true)\{if\ (duplicate) E_2(x) + +\}$$

3) The third equation ($E_3$) requires that row or column sums must be equal to 45.

$$E_3(x) = \frac{|\sum_{i,j=1}^{9} x_{i,j}|}{45}.$$

4) The fourth equation ($E_4$) requires that each row and column product should be equal to 9!.

$$E_4(x) = \frac{|\prod_{i,j=1}^{9} x_{i,j}}{9!}$$

5) The last equation is to calculate the overall fitness value of the initial population.

$fitness = \frac{1}{(\sum_{j=1}^{n} E_j)/n}$ ;(where n=4 ,n is number of constraints)

After the fitness is calculated, GA will rank the solution accordingly. If a perfect solution is found (where fitness value is 1), the process will be terminated. If not the process will continue until GA reach a certain number of generations.

### 2.4. Selection

In the selection process, an individual is probabilistically selected from the initial population on the basis of its fitness and the selected individual is then copied into the next generation of the population without any change. The selection method that has been used in this study is rank selection which means all the population will be rank again each other based on their fitness. The population with the fitness value is close to 1 will be in the top ranking and so on (Joachim, 2004; Goldberg,1989).

### 2.5. Crossover

The genetic operation of crossover allows the creation of new individuals which accounts for the generation of new points in the search space to be tested. Crossover starts with two parents independently selected probabilistically from the population on the basis of their fitness (Joachim, 2004; Goldberg,1989). The crossover rate was fixed at 0.7 at the onset of this study.

### 2.6. Mutation

The operation of mutation begins with the probabilistic selection of an individual from the population on the basis of its fitness. When mutation process is applied, each value in every column and sub block will be checked. If duplicate value is found, that value will be swap with other value in that row. This operation is done sequentially. The altered individual is then copied into the next generation of the population. In this case, the mutations are applied only inside a sub-block. In the first development of this project, mutation rate has been set to 0.1. The result shows that mutation operation has helped the population fitness to increase. (Joachim, 2004; Goldberg,1989)

### 2.7. New Population

After the process of reproduction, crossover, and mutation is done, the new population is derived and the process will be continued to calculate the fitness value. All of the process will be repeated until the best solution or the correct answer is found.

## 3. THE RESULTS

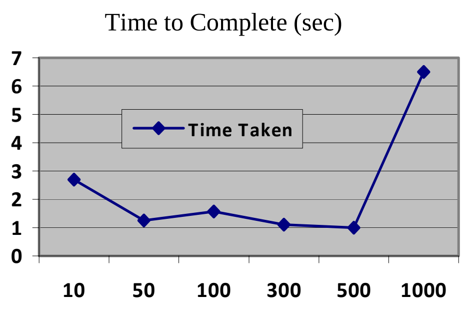
*Table 3.1* shows the results that have been obtained.. The GA is run with several different population sizes. The experiments are done a few times for each population and the means of the results are taken. In order to illustrate the findings more clearly the graphs are produced.

*Table 3.1: The results obtained from the experiment.*

| **No. of Population** | **Time (sec)** | **No. of Generation** | **No. of Crossover Applied** | **No. of Mutation Applied** |
|---|---|---|---|---|
| 10 | 2.692 | 11337 | 1131 | 7934 |
| 50 | 1.255 | 1274 | 128 | 881 |
| 100 | 1.568 | 784 | 80 | 549 |
| 300 | 1.104 | 156 | 13 | 129 |
| 500 | 0.995 | 88 | 9 | 58 |
| 1000 | 6.499 | 224 | 21 | 155 |

### 3.1. Time taken to find correct solution

The graph in Figure 3.1 shows the time taken by the GA to find the correct Sudoku grid in different population size. As the population grew, the GA was able to find the correct solution in lesser time. However, when the population was at 1000, the time taken to find good solution had dramatically increased.



Size of Population

*Figure 3.1: The time versus population size.*

### 3.2. Where the correct solutions are found

The graph in *Figure 3.2* shows where the correct solutions are found. A run with small population would find the solution in later generations and as the size of population increases the correct solutions are found in the early generations.

Size of Population

*Figure 3.3: Correct solutions in generation versus population size.*

The overall results are depicted in *Figure 3.4* The average best fitness increases as the number of generation increases but at the generation 50, the fitness value has decreased. This is because of crossover operation that causes the value in the column set or sub block become duplicates again. This is clearly the results of genetic operators used such as crossover and mutation.

The results are stated as follows:

*Table 3.2: The results obtained for popsize=500*

| **No. Of Generations** | **Best Fitness** |
|---|---|
| 1 | 0.088 |
| 10 | 0.17 |
| 20 | 0.303 |
| 50 | 0.679 |
| 80 | 0.493 |
| 83 | 1.0 |

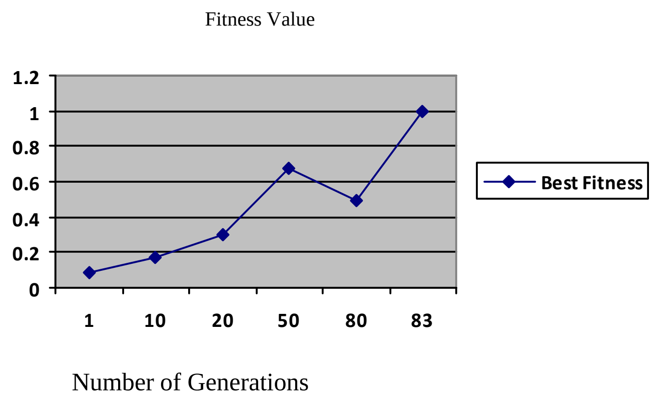


*Figure 3.4: The best fitness versus number of generations.*

## 4. DISCUSSION

### 4.1. Population Size

An important choice of population size for the Genetic Algorithms must be carefully considered. This is because it will affect the result of the problem. A small population size will make it converge quickly and the result may not be optimized, but if the size it too big it may cause the process to be too long.

### 4.2. Fitness Function

A fitness function is a particular type of objective function that measures the optimality of a solution in a genetic algorithm so that particular chromosome may be ranked against all the other chromosomes. An ideal fitness function correlates closely with the algorithm's goal, and yet may be computed quickly. Speed of execution is very important, as a typical genetic algorithm must be iterated many times in order to produce a correct result. The fitness value in this project is evaluated by using some equations based on the constraints of Sudoku Puzzle.

### 4.3. Selection

The selection of individuals to participate in the operation of crossover and mutation on the basis of their fitness is an essential aspect of this study. When an individual is selected on the basis of its fitness to be copied into the next generation of the population, the effect is that the new generation contains the characteristics it represents. The suitable methods for selection process also need to be chosen. As for this study, a chromosome is ranked based on their fitness value before it will be selected to participate in the next operation.

### 4.4. Crossover Rate

The crossover operator performs the mating of two selected parent chromosomes to produce a better generation of chromosomes. In this study, crossover operation has caused the fitness to become far from targeted goal. So low crossover rate is chosen as high crossover will make the chromosome contain the duplicate value in the sub block and column set. A crossover rate of 0.7 is considered as a parameter value for this system.

### 4.5. Mutation Rate

In this study, a high mutation rate is chosen because mutation operation is more helpful for the system to find the correct solution. A mutation rate of 0.1 is considered as a parameter value for this study.

## 5. CONCLUSION

Using Genetic Algorithm technique to solve constrained satisfaction problem, such as this one, is indeed a challenging problem both technically as well as theoretically. There are many enhancements that can be made to the current research. This research can also be used to solve similar constrained satisfaction problems. Hopefully it can be used as a milestone for further studies and research. The choice of using games as the platform to study the insights of GA has shown a worthy experience. The multi-objectivity in games parameter fits the versatility of GA in solving such problems.

## REFERENCES

Ardel, D.H. (1994). *TOPE and magic squares, a simple GA approach to combinatorial optimization. In J.R. Koza (ed.) Genetic Algorithms* , Stanford bookstore, Stanford, CA.

Bäck, T. (1996). *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*: Oxford University Press, USA.

De Jong , K. A.(1975). *Analysis of the Behavior of a class of genetic adaptive systems*, PhD Dissertation, University of Michigan, Ann Arbor, AAC 76-09381.

Edgington, J. (2006). *Solving Sudoku Puzzles*,Department of Computer Science, University of Denver.

Ghoshray, S.,Yen, K. K. (2004). *More Efficient Genetic Algorithm For Solving Optimization Problems*. Department of Electrical and Computer Engineering, Florida International University.

Gold, M. (2005). *Using genetic algorithms to come up with Sudoku puzzles*: Sep.

Goldberg, D. (1989). *Genetic Algorithms in and Optimization*: Addison-Wesley.

Grefenstette,J. J. .(1986).*Optimization of Control Parameters for Genetic Algorithms*, IEEE Trans.

On Systems, Man, and Cybernetics, Vol. SMC-16,No.1.

Jason Hoelscher-Obermaier, (2008). *Genetic Algorithm*. Universit¨at Regensburg.

Joachim, S. (2004). *Introduction to Genetic Algorithms.* Brainware GmbH, Berlin, Brainware Ltd., London.

Mantere, T., & Koljonen, J. (2006). *Solving and rating Sudoku puzzles with genetic algorithms. New Developments in Artificial Intelligence and the Semantic Web*, 86.

Moraglio, A., Togelius, J., & Lucas, S. (2005). *Product geometric crossover for the Sudoku puzzle*.

Moraglio, A., & Togelius, J. (2007). *Geometric Particle Swarm Optimization for the Sudoku puzzle*.

Runarsson, T., & Yao, X. (2000). Stochastic ranking for constrained evolutionary optimization. *IEEE Transactions on Evolutionary Computation, 4*(3), 284-294.

Tristan, C. (2004). *A search based Sudoku solver*. Labo IA Dept. Informatique Universit´e Paris 8, 93526, Saint-Denis, France.