Annals of DAAAM for 2010 & Proceedings of the 21st International DAAAM Symposium, Volume 21, No. 1, ISSN 1726-9679
ISBN 978-3-901509-73-5, Editor B. Katalinic, Published by DAAAM International, Vienna, Austria, EU, 2010
Make Harmony Between Technology and Nature, and Your Mind will Fly Free as a Bird
Annals of DAAAM International

SENSITIVITY ANALYSIS OF A GENETIC ALGORITHM FOR THE FLOW-SHOP SCHEDULING PROBLEMS

OLAH, B[ela]

Abstract: *The main goal of this scientific work is the sensitivity analysis of a genetic algorithm (GA) for permutation flow-shop scheduling problem. This paper covers the analysis of the proportion of the different genetic operators (cloning, crossover and mutation operator) used by the algorithm as well as the comparison of the results given by choosing of the different selection strategies in function of the efficiency of the near optimal solutions. I analyze how the efficiency of the algorithm changes by some values of the genetic parameters, I evaluate the obtained results and I search for relations that help to apply the GA more effectively and efficiently. The practical importance of my research results is to determine in what setting the genetic parameters have to be used in order to supply near optimal solutions at the fastest possible time.*
Key words: *selection, cloning, crossover, mutation, population*

1. INTRODUCTION

1.1 Flow-shop scheduling problem

The flow-shop scheduling problem can be defined as follows: n jobs are given which must be processed on m machines; all jobs have the same routing. That sequence must be determined which is the best according to certain logistic conditions. The most important aspect of the scheduling problem is the identification of criteria (Evans, 1993), which are used to evaluate schedules. Common scheduling criteria are the followings:

- Minimize make-span.
- Minimize idle time machines.
- Minimize waiting time of jobs.

1.2 Genetic algorithm

Figure 1 shows the structure of a simple genetic algorithm (Pohlheim, 2009). The algorithm works on populations of individuals instead of single solutions. At the beginning of the computation a number of individuals are randomly initialized. Each individual (genotype) represents a potential solution to the problem. These individuals in a population are called also chromosomes (Goldberg, 1989). The objective function is then evaluated for these individuals. Selecting the best individuals according to their level of fitness forms a new population. Some members of this new population undergo alterations by means of crossover and mutation.

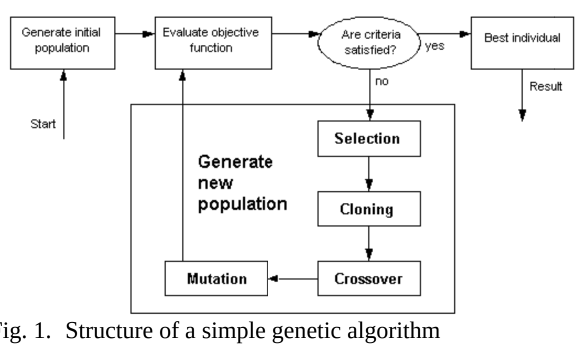

Fig. 1. Structure of a simple genetic algorithm

This cycle is performed until the optimization criteria are satisfied. The Holland's method is especially effective because the crossover operators greatly improve the capability of the algorithm to approach, and eventually find, the optimum.

2. THE COMPUTER PROGRAM

In Delphi, I implemented a program, which solves the scheduling problem using a genetic algorithm. The algorithm dynamically handles the number of jobs, machines and iterations, the size of the population, the ratio of the genetic operators as well as the objective function. So the software can be excellent applied in practice. In Figure 2 the program can be seen while running. The algorithm keeps indicating the results in the information board and drawing the value of the best individual (bottom graph) and the average fitness of the population (top graph) iteration by iteration. Because parents may be replaced by offspring with a lower fitness, the average fitness of the population can increase, but the fitness of the best chromosome decreases monotonously.

I use an integer vector as a chromosome to represent a sequence as a list of n jobs. The program uses two classical genetic operators: mutation and crossover (Michalewich, 1996). Four different mutation operators (insertion, displacement, reciprocal exchange and swap) are always used in the same proportion. I used two crossovers: *order crossover (OX)* and *cycle crossover (CX)* for the representation. These crossover operators are also in the same proportion to each other. Selection is a genetic operator that chooses a chromosome from the current population for inclusion in the next population. The following types of selection are handled by the software:

- Random: a selection operator which randomly selects the chromosomes from the population.
- Roulette-wheel: a selection operator in which the chance of a chromosome getting selected is proportional to its fitness.
- Best: selects the best chromosomes from the population.

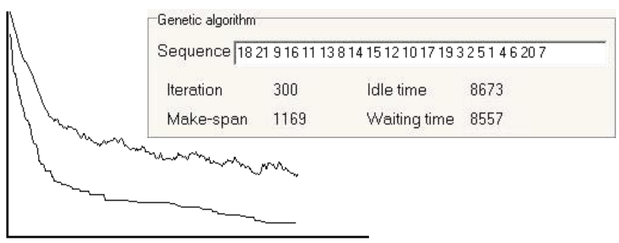

Fig. 2. Evolution of the results, information board of the GA

3. RESULTS OF THE RESEARCH

In the course of my earlier studies, when I compared the genetic algorithm to other heuristic methods for the flow-shop scheduling problem (Olah, 2009), parameters of the GA were basic settings. The following question was raised: how does the efficiency of the algorithm change by the modification of the values of the genetic parameters?

As it is well-known the fitness of the best chromosome decreases monotonously by the increase of the number of iterations. Naturally similar graph has to be effected performing the analysis of the same (20-machine, 25-job) permutation flow-shop scheduling problem several times. The program has been run 30 times to this problem for each iteration-number represented in Figure 3. The average values of the objective function (idle time) resulted by the genetic algorithm with fix (150) population-size using by 4% cloning, 16% mutation and 80% crossover in case of three different selection operators (best, roulette and random) are the followings.

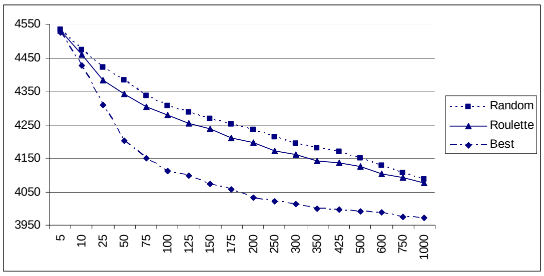

Fig. 3. Results of the different selections & iteration-numbers

The chart shows the idle time of the machines along the vertical axis and the increasing number of iterations from 5 to 1000 along the horizontal axis. It can be declared that the best selection, contrary to the other two selection strategies, excellently approaches to the optimal solution. It can be seen that the random selection produces the worst results among the selection operations while the roulette-wheel selection is about one per cent better than the random after the same iterations. After 100 iterations the GA which uses the best selection is 9,2 per cent better than after 5 iterations.

The new question is raised: what do these graphs look like by the modification of the probabilities of cloning, mutation and crossover? The next investigation leaves out the crossover operator from the optimization, so the populations will be generated by applying only two genetic reproduction operations: cloning and mutation of the best individuals. The software has been run 30 times to the previous scheduling problem for each proportion of the cloning represented in Figure 4. The average values of the idle time provided by the GA in view of 100 population-size and the best selection mechanism in function of the ratio of the mutation after 150 iteration are the followings.

On the vertical axis the idle time is given, the other axis represents the ratio of the cloning. From this diagram we can see that the genetic algorithm generates better and better solutions at first (up to 30% cloning), but more increase in the ratio of the cloning gives worse and worse values of the idle time. Consequently the minimum value (4013) occurs when the ratio of the mutation is 70% (and the cloning is 30%).

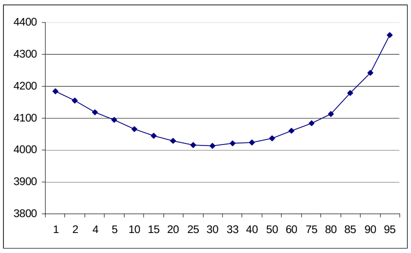

Fig. 4. Results in function of the ratio of the cloning-mutation

In the next analysis the genetic algorithm applies only for the cloning and the recombination (crossover) operations, accordingly the mutation operator will be left out. It can be stated from Figure 5 that the smallest values occur when the ratio of the crossover is between 75 and 15 per cent.

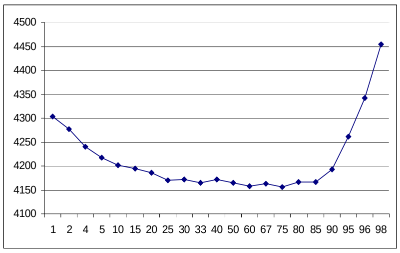

Fig. 5. Results in function of the ratio of the cloning-crossover

The minimum value (4156) is at 75% cloning. It can be established that the GA without crossover converges toward the optimum much (about 4%) better than without mutation.

Finally the genetic (crossover and mutation) operators will be checked for correlation with each other. The average values of the objective function resulted after 30 execution of the algorithm for each proportion of the mutation from 0 to 90 represented on horizontal axis of Figure 6 in function of the ratio of the crossover and mutation using by fix 10% cloning are the followings:

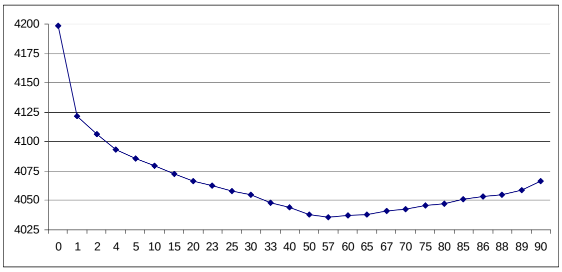

Fig. 6. Results in the ratio of the mutation-crossover

From this chart we can see that the program generates better and better solutions with increasing in the ratio of the mutation up to 57%, but following this the efficiency of the algorithm is decreasing in a small extent. Consequently the optimum (4036) occurs when the ratio of the mutation is 57% (and the crossover is 33%).

4. CONCLUSION

I can claim that a higher proportion of the mutation and the best selection results much better solutions than a higher proportion of the crossover or the other selection mechanisms.

In the future I would like to analyze the efficiency of each crossover and mutation operator. Because the job-shop and the open-shop scheduling problems have a considerably larger search space than the flow-shop, and both are an important and ubiquitous problems, I am going to enlarge the algorithm over these scheduling problems.

5. REFERENCES

Evans, J. R. (1993). *Applied Production and Operations Management*, West Publishing Company, ISBN 0-314-01038-6, Minneapolis

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, ISBN 0-201-15767-5, Reading

Michalewich, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, ISBN 978-3-540-60676-5

Oláh, B. (2009). Genetic Algorithm vs. Reinforcement Learning, In: *DAAAM International Scientific Book*, Katalinic, B. (Ed.), pp. 831-844, DAAAM International, ISBN 978-3-901509-69-8, Vienna

Pohlheim, H. (2009). Evolutionary Algorithms, *Available from:* http://www.geatbx.com/docu/algindex-01.html#TopOfPage *Accessed:* 2009-08-28