Handling a Multi-Tasking Environment via the Dynamic Search Genetic Algorithm

S. P. Koh*, I. B. Aris, S. M. Bashi** and K. H. Chong*****

Abstract – A new genetic algorithm for the solution of a multi-tasking problem is presented in this paper. The approach introduces innovative genetic operation that guides the genetic algorithm more directly towards better quality of the population. A wide variety of standard genetic parameters are explored, and results allow the comparison of performance for cases both with and without the new operator. The proposed algorithm improves the convergence speed by reducing the number of generations required to identify a near-optimal solution, significantly reducing the convergence time in each instance.

Keywords: Artificial Intelligence, Genetic Algorithm, Multi-Tasking

1. Introduction

The optimization of a sequencing operation of a machine is to minimize the total motion time of its manipulators along their paths. The evolutionary algorithm is used to obtain the motion plans in the multi-tasking environment. Global optimization algorithms imitating certain principles of nature such as simulated annealing and the field of evolutionary algorithms have proven to be useful tools for the optimization of high dimensional and highly nonlinear problems. Ref. [1] explains why the evolutionary algorithm (EA) performs an effective search compared to other methods. The EA presents a continual improvement using pair selection and mutation, working as a local search where the mutation operator slightly modifies a solution. If this new solution is better than previous ones, it will be accepted with high probability by the selection mechanism. On the other hand, the pair selection and crossover avoids the process being trapped in a local minimum, executing an intelligent jump to another search space region.

2. System Setup

In this research, the following three major components would be developed: simulation package, hardware of a dual-beam optical scanner, and artificial intelligent system. The simulation package consists of a graphical user interface where it links and directs the flow of the working process. It is a medium to allow interaction between the hardware, control system, artificial intelligent, and database. In this simulation package, the input data would be stored and learned in the database. The data from the database can be extracted to be processed and executed via the hardware. For this research, a dual-beam optical scanner has been designed and developed in order to test the functionality of the proposed artificial intelligent algorithm. An evolutionary solution has been chosen to optimize the performance and solve the dual-beam scanning problem. This algorithm would segregate the task of both scanner heads and allow the multi-tasking system to operate in synchronization.

System integration of both hardware and software of the dual-beam scanning module is illustrated in Fig. 1.

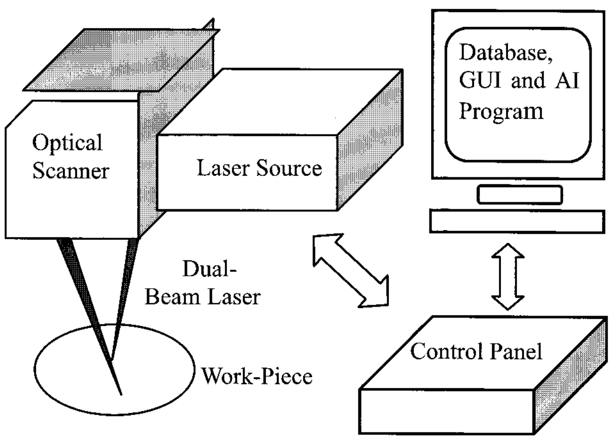


Fig. 1. System set-up

* College of Engineering, Universiti Tenaga Nasional, Malaysia. (johnnykoh@uniten.edu.my)
** Faculty of Engineering, Universiti Putra Malaysia, Malaysia. (ishak@eng.upm.edu.my, senan@eng.upm.edu.my)
*** Dept. of Physic and Science, Universiti Tunku Abdul Rahman, Malaysia. (chongkh@mail.utar.edu.my)
Received 26 February, 2007 ; Accepted 2 July, 2007

3. GA Optimization

A complete implementation of a GA needs to encompass the following major components:

i. Means of obtaining an initial population of solutions.
ii. Means of encoding solutions to the problem as integer chromosomes as used in the proposed system.
iii. Means of evaluation and fitness assignment of solutions in the solution pool.
iv. A selection procedure for individual solutions meeting the specified objective for reproduction into the next generation.
v. Reproduction operators for the encoded solutions.
vi. Appropriate settings for GA control parameters.

Some other important concepts would be needed in implementing a robust and efficient algorithm such as reinsertion, genetic drift, elitism, fitness sharing, mating viability and restrictions, and mutation rates.

A nodal-combinatorial problem is designed and tested using the following GA methods, namely:

i. Generational Replacement Genetic Algorithm (GRGA)
ii. Steady State Genetic Algorithm (SSGA)
iii. Standard Hybrid Genetic Algorithm (SHGA)
iv. Dynamic Search Genetic Algorithm (DSGA).

Comparison results are made and processing speed is rated in terms of number of generations required with different standard specifications.

3.1 Generational Replacement Genetic Algorithm

In GRGA, the entire population is simultaneously replaced by an equal number of offspring. The hope is that the offspring of the best strings carry the important building blocks from the best string forward to the next generation. The parents and offspring are not to live in the same population, and each population becomes the next generation. This replacement strategy is known as generational replacement and GAs based on this process are said to use non-overlapping populations as shown in Fig. 2.

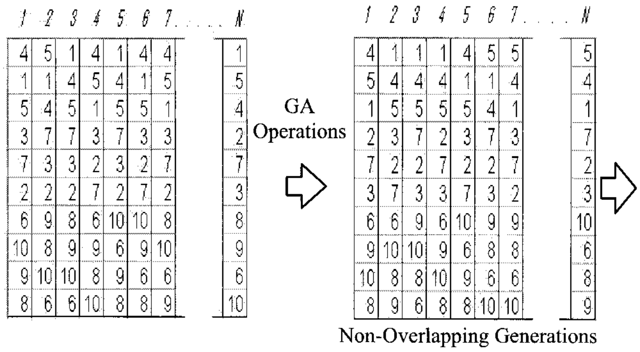


Fig. 2. GRGA concept

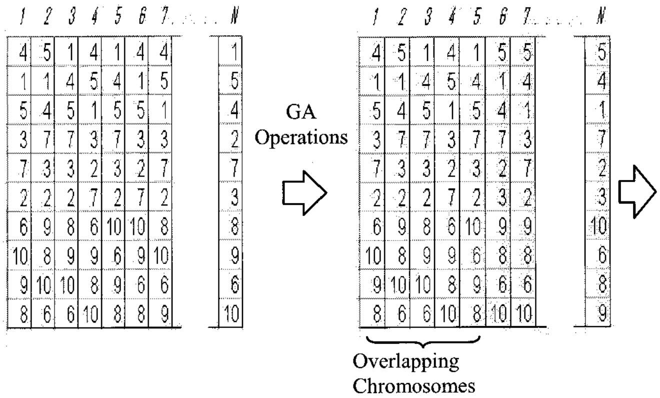


Fig. 3. SSGA concept

3.2 Steady-State Genetic Algorithm

A common variation of the generational process is some form of *elitism* strategy, which ensures the survival of highly fit members from one generation to the next at the expense of an equal number of poorly fit offspring. This process ensures that the algorithm does not forget the best that it found so far, and it helps in speeding-up the convergence.

Naturally, there is also interest in GAs where offspring have the chance to survive in the same generation and compete with at least some of their parents. A more recent development in GA theory is the use of overlapping populations in SSGA [2]. In every generation, only a few individuals are produced by recombination and mutation operators. These new individuals are then evaluated, and possibly reinserted into the population, replacing:

i. Random members of the parental population
ii. The oldest member of the parental population
iii. Their own parents
iv. The least fit member of the parental population.

This process is illustrated in Fig. 3. This algorithm has a built-in elitism since only the lowest ranked individuals are deleted while the best are automatically kept in the population. The percentage of the replaced individuals, *i.e.* overlap amount, makes the difference between the generational and the incremental GAs. At one extreme, a nearly 100% overlap is obtained by replacing one or two individuals at each generation. At the other extreme, the steady-state GA becomes a simple generational GA if the entire population is replaced, i.e. 0% overlap.

3.3 Standard and Dynamic Search GA

Literature has stated that real-parameter crossover operators are able to produce exploration or exploitation depending on the way in which they handle the current diversity of population [3]. Exploration will generate additional diversity starting from the current chromosomes

Table 1. GA Simulation Parameters

SHGA/DSGA Simulation Parameter	Value
Max. Generations Allowed	4000
Population, p_o	50
Selection Method	RW
Crossover Rate, p_c	0.80
Mutation Rate, p_m	0.05
Mutation Point, m_p	2
No. of Best Chromosomes Kept, k_b	1
Crossover Type	Fixed/Adaptive
Number of Simulation	10

and exploitation will create improved elements from the diversity [4]. Thus, an adaptive dynamic crossover operator is proposed in DSGA. Contradictory to conventional standard GAs where the size of crossover region is fixed throughout the crossover process, the size of crossover region in the DSGA is dynamically changed towards optimality, exploring at the early generations and exploiting at the latter generations. The GA parameters are set as in Table 1.

4. Experimental Results

In this section, the proposed Dynamic Search Genetic Algorithm (DSGA) has been studied and benchmarked with the conventional and standard GA, namely; GRGA, SSGA and SHGA. The advantages of DSGA in terms of its convergence time and quality of solutions have been discussed.

4.1 GRGA Experimental Results

A test has been conducted to examine whether the GRGA is able to reach good quality solutions. Fig. 4 plots the average and best results from ten different experiments to optimize a twenty-node scanning task, but each time employing purely random genetic parameter values,

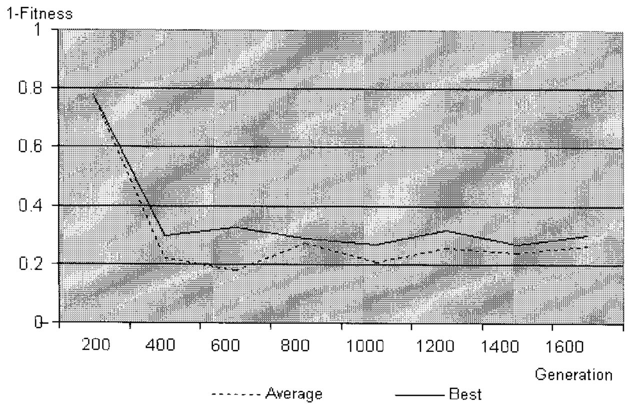


Fig. 4. Average and best cost using GRGA

population size, chromosome length, and stopping criteria. It is observed that the algorithm indeed optimizes the cost and that the rate in which improvements are introduced is impressive.

4.2 SSGA Experimental Results

SSGA was more robust and showed a tendency to converge after less than 500 generations. As shown in Fig. 5, in conjunction with uniform crossover and mutation, the SSGA reached a near optimal solution with a cost of 0.15, which represents a 51.0 % improvement compared to the initial cost, a 3.0 % improvement over the GRGA. This is mainly due to an enforced diversity and small population strategies. After crossover, members of the population were compared, and any duplicate members would be mutated. It is found that about 30% of the chromosomes are identical to another in the population. Since duplicate members do not improve the solution, they can be removed. As a result, diversity leading to better solutions is maintained and the algorithm is prevented from premature convergence. It is found that there was no significant difference between using either different crossover operators or crossover probabilities.

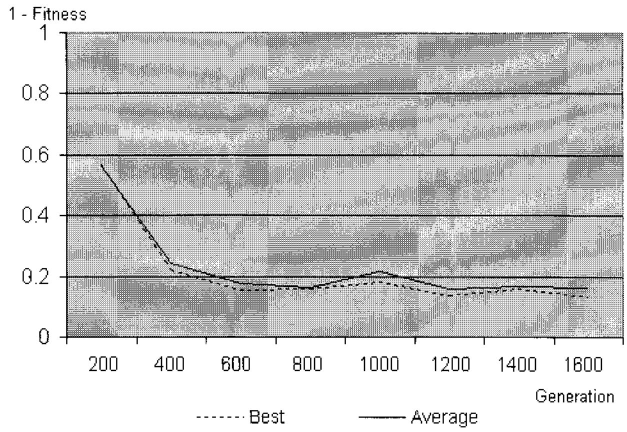


Fig. 5. Average and best cost using SSGA

4.3 SHGA Experimental Results

Genetic drift is the culprit that causes a GA to fail to find only a single solution at a time, and cause over-specialization [5]. In order to reduce the effect of genetic drift, attention has been turned to speciation "niching" techniques, which have been developed and tested against GRGA and SSGA. First, results that were obtained by applying SHGA with standard test parameters outperformed both the GRGA and SSGA in terms of solution quality and convergence time. Out of ten trials, the SHGA reached an optimal solution within 500 generations as shown in Fig. 6.

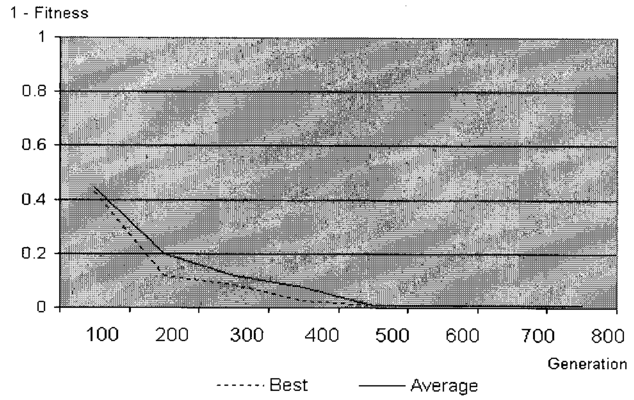


Fig. 6. Average and best cost using SHGA

4.4 DSGA Experimental Results

Convergence rate is used as the main criteria to compare the efficiency of DSGA and SHGA in obtaining a solution for dual-beam scanning. Fig. 7 presents the convergence rate comparison of different optimization techniques. Ten simulations were run and the best result of every optimization was obtained. The best fitness function value for the simulation scenario was calculated manually and used as a benchmark, which is indicated as "ideal" for the simulation. It can be found that DSGA converged faster compared to SHGA as it converged to ideal fitness function value at the range of 400 generations. However, SHGA can only converge to the ideal value at the range of 500 generations. Also, it can be seen that most of the fitness function values with SHGA are more than 0.4. However, most of the fitness function values with DSGA are less than 0.4. This indicates that the convergence rate of DSGA is faster than that for SHGA.

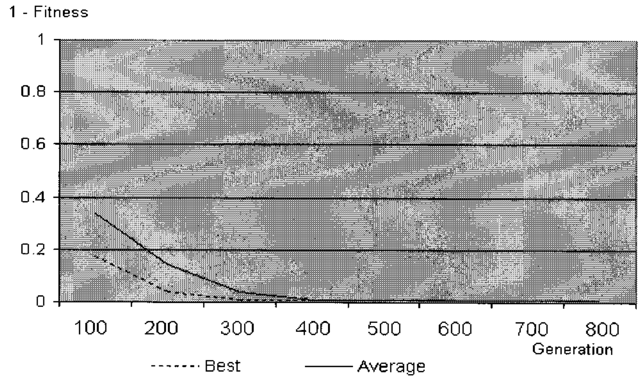


Fig. 7. Average and best cost using DSGA

5. Conclusions

The experiment has reviewed GA and related design formulations extensively and has developed and tested numerous alternative GA based implementations. The possibilities of using GRGA, SSGA, SHGA, and DSGA to solve the optimization problems were investigated. SHGA attempts to improve the solution quality or at least prevent premature convergence problems. DSGA has been proposed inheriting new adaptive crossover GA operators. It is believed to better sample the search space, improve its exploitation power and find good quality solutions. Solutions obtained using DSGA have surpassed those obtained using GRGA, SSGA, and SHGA.

References

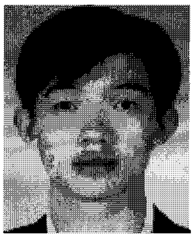
[1] Goldberg, D. E., "*The Design of Innovation: Lessons From Genetic Algorithms, Lessons for the Real World*," Internal Report 98004, Illinois Genetic Algorithms Laboratory, Department of General Engineering, University of Illinois at Urbana-Champaign, Illinois, 1997.

[2] Whitley D., "*A Genetic Algorithm Tutorial,*" Technical Report CS-93-103, Colorado State University, 1993.

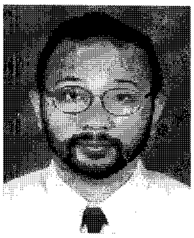
[3] Herrera, F., Lozano, M., and Sanchez, A. M., "*Hybrid Crossover Operators for Real-Coded Genetic Algorithms*," An Experimental Study, Department of Computer Science and Artificial Intelligence, University of Granada, 18071, Granada, Spain. 2003.

[4] Beyer, H. G. and Deb, K., "*On self-Adaptive Features In Real-Parameter Evolutionary Algorithms*," IEEE Transactions on Evolutionary Computation 5(3), 2001. pp. 250-270, 2001.

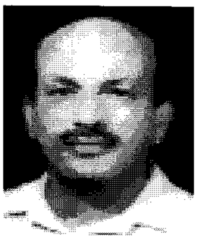
[5] Darwen P., Yao X., "*A Dilemma for Fitness Sharing with a Scaling Function*," In Proceeding of IEEE Conference on Evolutionary Computation, pp. 166-171, 1995.



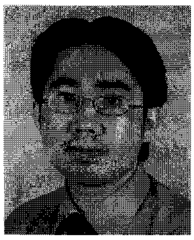
S. P. Koh
He received his B.Eng (Hons) in Electronics and Electrical, and his M.Sc from the Universiti Putra Malaysia in Control and Automation. His research interests include Artificial Intelligence, Lasers, Advanced Mechatronics, and Control Systems.



Ishak Bin Aris
He received his B.Sc in Electrical Engineering from the George Washington University, USA in 1988. He also received his M.Sc and Ph.D. in Power Electronics Engineering from the University of Bradford, United Kingdom in 1991 and 1995, respectively. His areas of interest include power electronics and drive system, robotics, artificial intelligence, and automotive electronics.



S. M. Bashi
He graduated from the University of Mosul in Electrical and Electronics Engineering (1969). He received his Ph.D. in Simulation of Power Transmission Systems from Loughborough University of Technology, England (1980). His areas of research interest include power system analysis and design, quality of power supply, simulation and application of power electronics systems, and machine drives.



K. H. Chong
He received his B. Eng (Hons) in Electronics and Electrical, and his M. Sc from the Universiti Putra Malaysia in Electrical and Electronic Engineering respectively. His current research interests include artificial intelligence and industrial process control.