Selfish Gene Algorithm Vs Genetic Algorithm: A Review

Norharyati Md Ariff[1], Noor Elaiza Abdul Khalid[1], Rathiah Hashim[2], Noorhayati Mohamed Noor[1]
[1]Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA Shah Alam, Selangor, Malaysia
[2]Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, Parit Raja, Malaysia

E-mail: sakurayati@yahoo.com, elaiza@tmsk.uitm.edu.my, radhiah@uthm.edu.my, noorhayati@tmsk.uitm.edu.my

Abstract. Evolutionary algorithm is one of the algorithms inspired by the nature. Within little more than a decade hundreds of papers have reported successful applications of EAs. In this paper, the Selfish Gene Algorithms (SFGA), as one of the latest evolutionary algorithms (EAs) inspired from the Selfish Gene Theory which is an interpretation of Darwinian Theory ideas from the biologist Richards Dawkins on 1989. In this paper, following a brief introduction to the Selfish Gene Algorithm (SFGA), the chronology of its evolution is presented. It is the purpose of this paper is to present an overview of the concepts of Selfish Gene Algorithm (SFGA) as well as its opportunities and challenges. Accordingly, the history, step involves in the algorithm are discussed and its different applications together with an analysis of these applications are evaluated.

Keywords: Parallel Processing, Task Partitioning, Hierarchical Heterogeneous Cluster, Multi-Core, Heuristic Testing

1. Introduction

The real-world problems may consist of unstable structure that was contributed by incomplete of noisy data and many multi-dimensional problems; thus creating flaws to the conventional algorithms to solve the real-world problems. So, evolutionary algorithm seems to be useful in solving such problems. Evolutionary algorithms (EAs) are based on a search and optimization methods that were inspired by the biological model of Nature Selection. Based on this understanding, we find a family of EAs, known as the genetic algorithm (GA) [1,2], evolutionary strategy (ES) [4], genetic programming (GP)[10], Selfish gene algorithm (SFGA)[6,7] and Memetic algorithm (MA) [8] have been developed after the Darwinian theory.

EAs attempts to solve complex problems by mimicking the processes of Darwinian evolution where individuals in the population continuously compete with each other in the process of searching for optimal solutions [3].The members of the EA family share a great number of features in common. They are all population-based stochastic search algorithms performing with best-to-survive criteria. Each algorithm commences by creating an initial population of feasible solutions, and it evolves iteratively from generation to generation towards a best solution. In successive iterations of the algorithm, fitness-based selection takes place within the population of solutions. Better solutions are preferentially selected for survival into the next generation of solutions, with a diversity being introduced to the selected solutions in an attempt to uncover even better solutions over the next generation, with an aim to search for a global optimum [2].

Since ancient time, human has been imitating the nature to survive by implementing the way and style of nature in everyday lives. Our ancestors have been using inspiration from nature to create a variety of materials and equipment for centuries. Richard Dawkins wrote a book called "The Selfish Gene" in 1976 and followed in 1982 with "The Extended Phenotype". He proposed a new theory for considering the Darwinian natural selection mechanism. He has put an evolution in a different view and provides an alternative interpretation key called the Selfish Gene Theory. In this theory, population can be seen as a pool of genes and individual's genes fight for their survival in the genotype of the vehicles. The survival of the fittest is a struggle fought by genes, not the individual. Inspired by this theory, Corno F. and his cooperative members [6,7] proposed and implemented a new evolutionary algorithm method called the Selfish Gene Algorithm(SFGA).

This paper tries to give an overall view of Selfish Gene Algorithm (SGFA). Section 2 will introduce the Selfish Gene Algorithm. The steps involve in Selfish Gene Algorithms are discussed in Section 3. Section 4 provides the applications that have been done by other researchers using Selfish Gene Theory and Algorithm. Section 5 sketches a tabular form comparison of most EAs, Genetic Algorithm with Selfish Gene Algorithm. The conclusion is drawn in Section 6.

2. The Selfish Gene Algorithm (SFGA)

The selfish gene algorithm is a new member of Evolutionary Algorithms that search stochastically through a virtual population of the genes, not a real population like Genetic Algorithm [10, 26]. It focuses on the fitness of the genes itself rather than the individuals. It does not have any crossover and mutation at all and its population is seen as a store of genetic material. It is used as a term of "Virtual Population" which models the gene pool concept. This virtual population presents the number of individuals, and their specific identity. The individual is represented here by its genome. There are explicitly distinguished between its location in the genome which is called locus and the value of the locus called allele. The successful of an alleles is calculated by its frequency appear in the virtual population. Generally evolution means, that organism which succeeds will increases its allele's frequency in population at the expense of children. An organism which fails will decreases its allele's frequency in population [1,2,10,24,25,26].

The selfish gene algorithm is different with others techniques because it has no explicit population. Meanwhile, individuals are creating and destroy only for temporary. The mutation is encoded as a random gene selection but not according to its frequencies. For its fitness function, absolute values of fitness are not assigned to generate solutions. It compares between each individual to generate solutions to others. After the criteria meets, it returns back to the algorithm.
Research by [2, 10] showed the concept of selfish gene algorithm (SGA). The chromosome is also called genotype and all living organisms have DNA molecules. Genes is part of DNA molecule and gene location in DNA is called locus. Usually for the same locus there can be different versions of gene and it is called alleles. To determine the fittest chromosome or genotype, allele's frequencies or probabilities in population are calculated.

3. Steps involve in Selfish Gene Algorithm

Simple illustration of SGA steps shown below give widely ideas how SGA works. There are 10 efficient steps.

3.1 Encoding representation

In all EAs techniques, encoding is the first stage in their steps where chromosomes are represented. This step is analogous to step in the GA [1,2]. The chromosomes can be represented as binary strings, char and vectors in GA; list of real numbers in SFGA. Each gene we will explicitly distinguish between its location in the chromosome (the locus) and the value appearing at that locus (the allele). Chromosome in GA represents a legal solution to the problem and it is composed of a string of genes which is fixed but in SFGA, selected genes from Virtual Population (VP) according their high frequencies combine as one chromosome and strive in the SFGA process. Then the gene can separate after using it in the algorithm. Figure 1 describes the illustration of the component in the selfish gene algorithms and GA based on their chromosome representation.

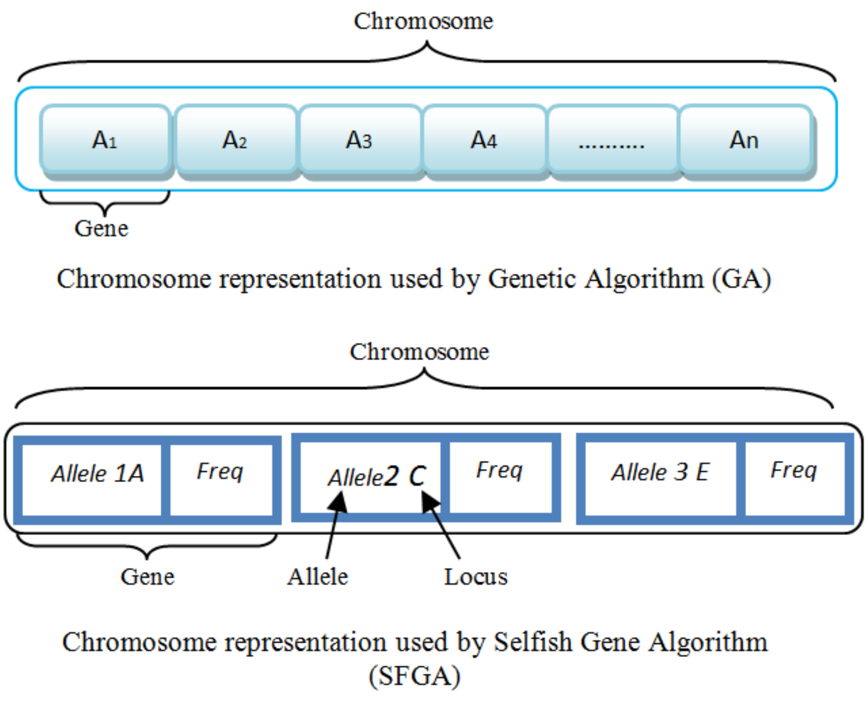
Chromosome
A1
A2
A3
A4
....
An
Gene
Chromosome representation used by Genetic Algorithm (GA)
Chromosome
Allele 1A
Freq
Allele2 C
Freq
Allele 3 E
Freq
Gene
Allele
Locus
Chromosome representation used by Selfish Gene Algorithm (SFGA)

Figure 1 Illustration of the gene in the Selfish Gene Algorithm and Genetic Algorithm.

3.2. Initialize the population (pop)

Initialization is the most steps in EAs technique. Population size is very important all variant EAs algorithms as limited population size may produce solutions in low quality [17].The conventional population of individuals that used in GAs is replaced by a virtual population (VP) where the individual is seen as only storing of genetic material [53]. The recombination or crossover stage is present in GA but not in SFGA. Among the types of crossover operation are single point, two point, uniform and arithmetic crossovers. A population is a set of individuals and each of them has associated fitness values that can measure their “goodness”.

This step is a bit different with other technique because a population (real population of individuals) is not important; it’s just a store room for genetic materials. From the population, take genes and store it in a virtual population (Gene pool).In Virtual Population, the number of individuals, and their specific identity, are not of interest, and therefore are not specified or stored. The first population is created by generating individuals randomly. In the SFGA, the VP evolves through a mechanism called tournament. Because there is no explicit definition of individuals in the VP, an individual is generated only when needed for competing in a tournament, and it is discarded immediately after [9].

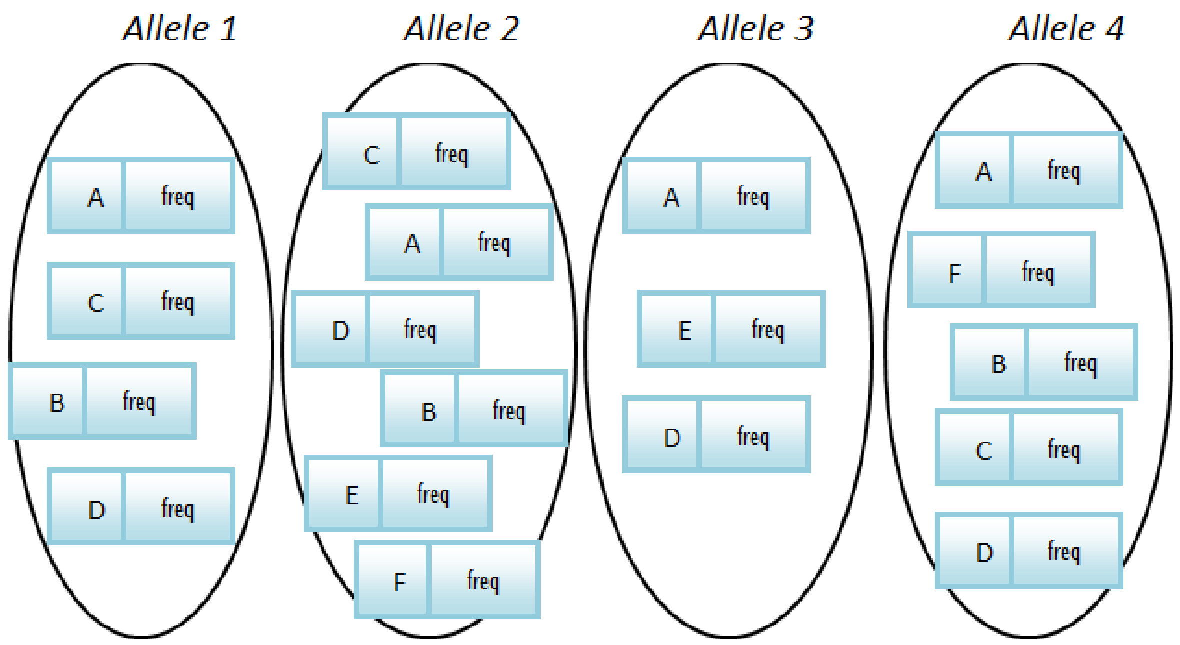

Figure 2 Example of 4 types alleles in the virtual population of the Selfish Gene Algorithm

From another point of view can only be seen as VP gene pool struggling to imitate nature called the "culture transmission”. In recent developments of SFGA the VP contains vectors of accumulated frequencies of alleles following a distribution of probabilities for each gene. In this new algorithm only a reward to the winner of the process of mating generated by an individual from VP and random immigrants is accumulated [53].

3.3. Evaluate the gene frequencies (pop)

Fitness function stage is important in EAs. It is a heuristic function that measures the performance of an individual chromosome or genotype in the problem domain. The fitness function establishes the basis for selecting chromosomes that will be mated during the reproduction in EAs. Each gene should have its own frequency and locus. Fitness function is based on the goodness of a given individual or an individual which have the highest frequency. This step is same as GA, where evaluation takes place at phenotypic level after a decoding phase [9].As with other Evolutionary

Algorithms, in SFGA an individual is represented by its genome. But what makes it differ from other techniques is, for each gene we will explicitly distinguish between its location in the genome (the locus) and the value appearing at that locus (the allele). In the VP, due to the number of possible combinations, genomes tend to be unique, but some alleles might be more frequent than others. In the SG, the success of an allele is measured by the frequency with which it appears in the VP.

3.4. Select the parent from pop based on its gene frequencies

Two individuals are randomly selected according their allele frequencies from VP and compares them in a tournament. The winner of the tournament will see its genes rewarded. In GA, parent selection stage can employ selection methods such as random, stochastic technique of roulette and tournament selection.

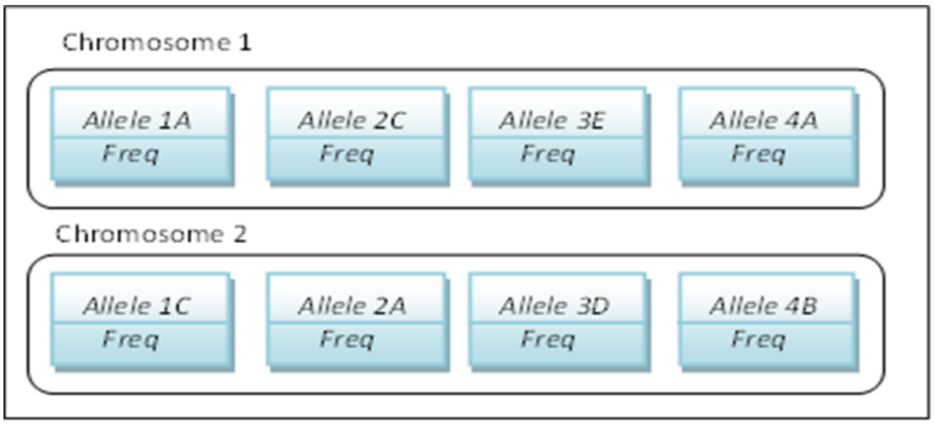

Figure 3 Example of parent selection from the Virtual Population

3.5. Reproduction parents– Mutation (If any) -takes gene randomly and compare each other.

After each competition the winner has the opportunity to reproduce it. The reproduction is very similar to genetic algorithm. Two winner chromosomes will undergo crossover to produce two new child chromosomes. These chromosomes could also undergo the process of mutation to add diversity to the population. Mutation is not a very important process because the allelles combination will be created when new chromosomes are created.

3.6. Compare two chromosomes to find the best combination by calculating its fitness function.

The two child chromosomes will be evaluated phenotypic level with the fitness functions. The one with higher fitness is chosen to be the winner.

3.7. Reward and Penalize/Punishment-updating the best solution by update the frequency vector for each allele in the chromosome

When a child chromosomes is fit or win, its genes are rewarded: the specific allele values making up the individual's genotype will see their frequency in the VP increases. Vice versa, genes belonging to unfit child chromosomes will be penalized. For good individual, reward is given to the alleles by adding 1 into its frequency and the worst one, penalize is given to the alleles by subtracting 1 from its frequency. Assume that the winner is Gbest and the looser is Wbest. Firstly, for each pair of locus l1 and l2 (l1,l2), calculate the number of the pair values of $x_{l1}x_{l1}$ and $x_{l2}x_{l2}$ (such as 00, 01) in Gbest and Wbest respectively. Then, for each pair in Gbest, increase the corresponding frequency by giving some rewards like ε; and decrease the frequency by giving the same punishment ε to the pairs in Wbest. After the change the frequencies of each pairs in the individuals, the unconditional probabilities p($x_{l1}x_{l1}$) and p($x_{l2}x_{l2}$) and conditional probability p($x_{l1}x_{l1}$, $x_{l2}x_{l2}$) can also be updated by the current value of the pair (l1,l2).

Freq(Gbest (l1,l2)) = freq(Gbest (l1,l2)) + ε;
Freq(Wbest(l1,l2)) = freq(Wbest(l1,l2)) - ε;

Based from [19], this reward scheme shows a positive feedback that gives fast algorithmic convergence by increase allele selection probability. Reward and punishment are all functional explanations of behavior depend on some notion of what is good and bad. In terms of evolutionary adaptation, good and bad boil down to values of inclusive Darwinian fitness, a measure reasonable clear in principle, but often elusive in practice [13].

Convergence speed can be tuned using the parameters of high value will make the VP moving quickly toward a good first solution, while a very small value will make the float VP for a long time before choosing a local optimum to select a target [9].

3.8. Update the best solution by arranging the separation fittest.

Separation based on the fittest will create the new fitness function and return to virtual population.

3.9. Repeat steps if not meet the required criteria.

Repeat steps 1-6 until required solution found or other needed criteria are met-for example maximum evolve cycles elapsed or solutions genes frequency is above some threshold or whatever we need.

3.10. Check termination criteria

The usual stopping criterion is a fixed amount of computing time (or, almost equivalently, of fitness computations).A slightly more subtle criterion is to stop when a user-defined amount of time has passed without improvement of the best fitness in the population [30]. Figure 5 shows flowcharts of Selfish Gene Algorithms (SFGA).

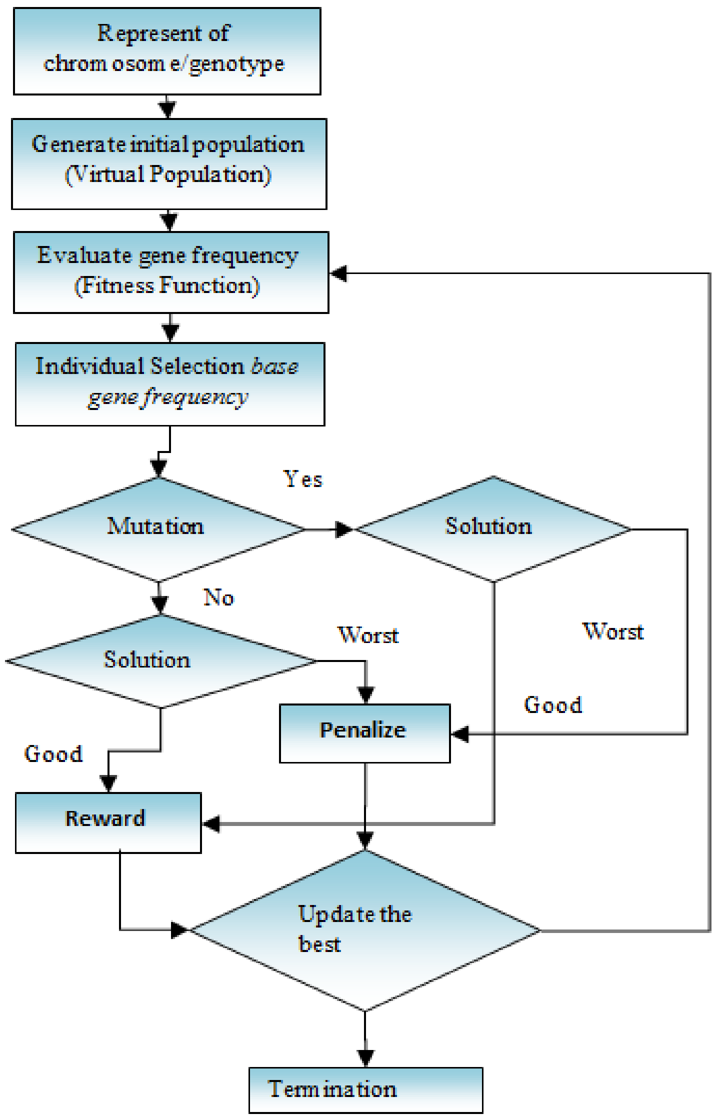
Represent of chromosome/genotype
Generate initial population (Virtual Population)
Evaluate gene frequency (Fitness Function)
Individual Selection *base gene frequency*
Mutation
Yes
Solution
No
Solution
Worst
Worst
Good
Penalize
Good
Reward
Update the best
Termination

Figure 4 shows a flowchart of Selfish Gene Algorithm (SFGA)

4. Applications using the selfish gene algorithm

There are a number of applications done by another researcher using selfish gene theory and selfish gene algorithm shown in Table 1. Corno F. et al (1998), who are the first researcher tries to implement the new evolutionary algorithm inspired by Selfish Gene Theory. They have been experimenting the algorithm on several case studies such as genetic coach problem and multiple knapsack problems. Their studies gave a big impact in the world of evolutionary algorithm and provide referral to other researchers.

Table 1. The applications are done by several researchers using Selfish Gene Theory and Selfish Gene Algorithm.

Authors, Years	**Objectives**
Corno F. et al, (1998)	Test *SFGA* on the Genetic Coach problems that simulates the dilemma of selecting the best crew for a rowing competition.
Corno F. et al (1998)	Test *SFGA* on multiple knapsack problem
Corno F. et al, (1999)	Proposed a new evolutionary mechanism of *SFGA* called SG-clans which exploits the evolution of isolated clans inside the population to quickly discover useful inter-gene linkages.
Corno F. et al, (1999)	*SFGA* is designed for addressing sequential circuits and thanks to its ability guaranteeing feasibility of solutions during the optimization process it is more effective than other approaches.
Corno F. et al, (2000)	The first application that uses *SFGA*. The *Selfish Gene* algorithm is adopted for determining the logic for a BIST architecture based on Cellular Automata (CA).
Villagra A. et al (2004)	Showed a new algorithm that combines the MCMP-SRI and *Selfish Gene* approaches. MCMP-SRI-SG algorithm is a variant of MCMP-SRI that applies the *selfish gene theory* substituting the population of individuals for a virtual population.
Zhang et al, (2004)	Enhance the spectral method of sequential circuit test generated by using a *SFGA*. The objects of evolution are the Hadamard spectral matrix, non-linear digital signal processing *(DSP)* filtering cutoff values, vector holding time, and relative input phase shifts, which are all modeled as genes.
Popa R. 2004	Analyses the performances of SFGA, and proposes a method of improvement of these performances by hybridization with the simulated annealing technique.
Davide De Caro et al, (2001)	Proposed a synthesis tool that exploits heuristic search and the "*selfish gene*" genetic algorithm to determine test pattern generator for a given test set
Sakurai Y. et al, (2008)	Developed idea in that each gene on a chromosome or an individual acts *selfish*ly to optimize only itself or the relationship between itself and its surroundings locally, without considering the totality of the individual or other genes of the same individual.
Wang F. et al, (2009)	Employed the *selfish gene theory* to construct virtual population for optimizations.
Yang C. et al, (2009)	SGEGC uses a vector of survival rate to model the condition distribution, which serves as a virtual population that is used to generate new individuals.
Wang F. et al,(2009)	The Selfish Gene Theory (SG) is deployed in this approach and a Mutual Information and Entropy based Cluster (MIEC) model with an incremental learning and resample scheme is also set to optimize the probability distribution of the virtual population.
Wang F. Et al, (2010)	Proposed a selfish gene-based approach called SGMIEC to solve the discrete optimization problems. They employ a mutual information and entropy based cluster model to test the impacts of the genes and an incremental learning method with resample scheme is also used in the mutual information cluster construction.
António C. C. **(**2012)	Instead of local search as performed in MAs, the *selfish gene algorithm (SFGA)* follows a different learning scheme where the conventional population of individuals is replaced by a virtual population of alleles.

5. Comparative Analysis

This comparative analysis is to compare between Genetic Algorithm (GA) which is most EA method with Selfish Gene Algorithm (SFGA). Table 2 shows a comparison of five performance evaluation features of the algorithms that is the convergence rate, the algorithm complexity, the accuracy in finding solutions, the processing speed and the rate of achieving the optimal solutions.

Table 2. The performance evaluation matrix of the evolutionary algorithms.

Performance features	**GA [19,20]**	**SGA [6,7,9,21]**
Convergence	Difficult	Fast
complexity	Simple	Simple
Accuracy	Low	Reliable
Speed	Slow	Fast
Optimum solution	Slow	Faster

GA has difficulties in converging as the probability of making progress decreases rapidly as the minimum/maximum is approached. Thus, these algorithms are often hybridized with other techniques to improve their performance. Staddon[13] uses meta-heuristic population in GA and successfully resolved many optimization problems. However, premature convergence narrows down its ability to find many solutions. In the bid to reduce premature convergence possibility an algorithm that hybridized the classical GA with local search technique and named as Memetic [15]. Corno F. *et al.* [24, 25] prove that SFGA is able to outperform a GA on a test problem. SFGA converge very fast towards a local optimum. Because it is a new algorithm, it still explores the neighborhood of an upward sloping path. The success of SFGA is depending on its VP and the rewards and punishment scheme. For SFGA, the rewards and punishment are a good control parameter but GA always faces many limitations and one of the limitations is the improper choice of control parameters in solving real-world problems due to its detrimental influence on the trade-off between exploitation and exploration [17].

6. Conclusions

Selfish Gene Algorithm is similar to Genetic Algorithm in terms of terminology such as population, crossover and mutation. However, Selfish Gene Algorithm use population as Virtual Population where the population is a place to store all the information needed temporarily. Selfish gene algorithm also evolves genes itself or its characteristic that provide higher fitness rather than evolving individuals with higher fitness. Thus, gives better solution to other methods.

Acknowledgments
Authors would like to thank Universiti Tun Hussein Onn Malaysia for the support in publishing this article and also Gates IT Solution for the incentive to publish.

REFERENCES

[1]. Pignalberi G., Cucchiara R., Cinque L. and Levialdi S.: Tuning range segmentation by genetic algorithm, EURASIP Journal Appl. Sig. Proc., vol. 8, pp. 780-790 (2003)
[2]. Huang C.F. and Rocha L. M.: A systematic study of genetic algorithms with genotype editing. In Proc. Of 2004 Genetic and Evolutionary Computation Conference, volume 1, pages 1233{1245 (2004)
[3]. Jones G. 1998, Genetic and evolutionary algorithms. In Paul von Rague, editor, Encyclopedia of Computational Chemistry. John Wiley and Sons. (1998)

[4]. Yuan X., Zouridakis G., and Situ N., : Automatic Segmentation of Skin Lesion Images Using Evolution Strategies, Preprint submitted to Elsevier(2008)

[5]. Poli R., Langdon W.B, and McPhee N.F.: A Field Guide to Genetic Programming. Lulu Enterprises,UK (2008) (ISBN 978-1-4092-0073-4) (2008)

[6]. Corno F., Reorda M. S., Squillero G.: Exploiting the Selfish Gene Algorithm for Evolving Cellular Automata, IJCNN2000: IEEE-INNS-ENNS International Joint Conference Neural Networks, Como (I), July 2000, pp. 577-581(2000)

[7]. Vasiliauskas A.: Selfish Gene Algorithm, Available from: http://coding-experiments.blogspot.com/2008/04/selfish-gene-algorithm.html. (2008)

[8]. Garg P., : A Comparison between Memetic algorithm and Genetic algorithm for the cryptanalysis of Simplified Data Encryption Standard algorithm, International Journal of Network Security & Its Applications (IJNSA), Vol.1, No 1, April 2009 (2009)

[9]. F. Corno, M. Sonza Reorda, G. Squillero, The Selfish Gene Algorithm: a New Evolutionary Optimization Strategy, SAC98: 13th Annual ACM Symposium on Applied Computing, Atlanta, Georgia (USA), February 1998, pp. 349-355

[10]. Rui Tavares, António Teófilo, Paulo Silva, Agostinho Cláudio da Rosa, Infected Genes Evolutionary Algorithm, SAC 1999, Pages: 333-338 , DOI: 10.1145/298151.298374

[11]. Zhang J, Bushnell M.L., Agrawal V.D, 2004, On Random Pattern Generation with the Selfish Gene Algorithm for Testing Digital Sequential Circuits, Test Conference, 2004. Proceedings. ITC 2004, Pages 617 – 626.

[12]. Sherphed, John G., John G Pope, 2002, Dynamic pool models I: Interpreting the past using Virtual Population Analysis (VPA), In P. J.B. Hart, J.D. Reynolds (ed.). Handbook of Fish Biology and Fisheries. Vol.2. Fisheries. Oxford, UK: Blackwell Science. Pp.127-136.

[13]. J. E. R. Staddon, 1983, Adaptive Behaviour and Learning By J. E. R. Staddon, books

[14]. Kent Simon and Patel N. (2006), Artificial Intelligence makes computers lazy. International Journal of Industrial and Systems Engineering. 1 (4) 519-532

[15]. Eick. C.F and Toto E., 1994, Evaluation and enhancement of Bayesian Rule-sets in genetic algorithm learning environment for classification tasks.ISMIS94

[16]. (Sedighizadeh D. and Masehian E., 2009, Particle Swarm Optimization Methods, Taxonomy and Application, International Journal of Computer Theory and Engineering, Vol 1, No 5, Pages 1793-8201.

[17]. El-Mihoub T.A, Hopgood A.A., Nolle L. and Battersby A.: Hybrid Genetic Algorithms: A Review. Engineering Letters, 13:2, EL_13_2_11 (2006)

[18]. Yong Chen, Yan Jiao, Chi-Lu Sun and Xinjun Chen, 2008, Calibrating virtual population analysis for fisheries stock assessment, EDP Sciences, IFREMER, IRD 2008, Aquat. Living Resour. 21, 89–97 (2008)

[19]. Clow B. and White T.: An evolutionary race: A comparison of genetic algorithms and particle swarm optimization for training neural networks. In Proceedings of the International Conference on Artificial Intelligence, IC-AI '04, Volume 2, pages 582–588. CSREA Press, (2004)

[20]. Ciesielski V. and Mawhinney D.: Prevention of early convergence in genetic programming by replacement of similar programs. In Xin Yao, editor, Proceedings of the 2002 Congress on Evolutionary Computation,(2002)

[21]. Popa R.: Hybridated Selfish Gene Algorithm. Artificial Intelligence Systems, 2002. (ICAIS 2002). 2002 IEEE International Conference on 2002. (2002)

[22]. Eberhart RC and Shi Y.: Comparison between genetic algorithms and Particle Swarm Optimization. In: Porto VW, Saravanan N, Waagen D and Eiben AE (eds) Evolutionary Programming VII, pp. 611–616. Springer(1998)

[23]. Clow B. and White T.: An evolutionary race: A comparison of genetic algorithms and particle swarm optimization for training neural networks. In Proceedings of the International

Conference on Artificial Intelligence, IC-AI ’04, Volume 2, pages 582–588. CSREA Press, (2004)

[24]. F. Corno, M. Sonza Reorda, G. Squillero, 1999, Optimizing Deceptive Functions with the SG-Clans Algorithm, IEEE.

[25]. F. Corno, M. Sonza Reorda, and G. Squillero, A new Evolutionary Algorithm Inspired by the Selfish Gene Theory, IEEE International Conference on Evolutionary Computation, 1998, pp. 575-580

[26]. A. Villagra, M. De San Pedro, M. Lasso, and D. Pandolfi,2004, Multirecombined Evolutionary Algorithm Inspired in the Selfish Gene Theory to Face the Weighted Tardiness Scheduling Problem

[27]. Corno F., Sonza Reorda M., Squillero G.; Evolving Effective CA/CSTP BIST Architectures for Sequential Circuits SAC2001: ACM Symposium on Applied Computing, March 2001, Las Vegas (USA), pp 345-350.

[28]. Corno F., Sonza Reorda M., Squillero G.; Exploiting the Selfish Gene Algorithm for Evolving Hardware Cellular Automata CEC2000: Congress on Evolutionary Computation, San Diego (USA), July 200o pp 1401-1406.

[29]. Corno F., Sonza Reorda M., Squillero G.; Exploiting the Selfish Gene Algorithm for Evolving Hardware Cellular Automata IJCNN2000: IEEE-INNS-ENNS International Joint Conference Neural Networks, Como (Italy), July 2000 pp 577-581.

[30]. A.E. Eibena,∗, M. Schoenauer,; Evolutionary computing, Information Processing Letters 82 (2002) 1–6

[31]. Levialdi Stefano, Cinque Luigi, Cucchiara Rita, Pignalberi Gianluca(2003), Tuning Range Image Segmentation by Genetic Algorithm, EURASIP Journal on Advances in Signal Processing 01/2003

[32]. Huang, C F. and Rocha L M (2004).A Systematic Study of Genetic Algorithms with Genotype Editing.*Springer-Verlag Berlin Heidelberg.* (2004)page. 1233–1245.

[33]. Meijs B.F.V.D.; Implementation of the parallel Selfish Gene Algorithm, Master thesis,2004.

[34]. Mauricio G.C. Resende, 2012, Biased random-key genetic algorithms with applications in telecommunications, AT&T Labs Research Technical Report.

[35]. Nikolaus Hansen, 2006, An Analysis of Mutative -Self-Adaptation on Linear Fitness Functions, Journal Evolutionary Computation, Volume 14 Issue 3, Pages 255-275

[36]. Farzad A. Sadjadi, 2004, Comparison of fitness scaling functions in genetic algorithms withapplications to optical processing

[37]. Vassilios Petridis, *Member, IEEE*, Spyros Kazarlis, and Anastasios Bakirtzis, *Senior Member, IEEE, 1998,* Varying Fitness Functions in GeneticAlgorithm Constrained Optimization: TheCutting Stock and Unit Commitment Problems

[38]. Hajime Kita and Yasuhito Sano, Genetic Algorithms for Optimization of Noisy FitnessFunctions and Adaptation to Changing Environments

[39]. Vin´ıcius L. Silva, Andr´e R. da Cruz, Eduardo G. Carrano,Frederico G. Guimar˜aes and Ricardo H. C. Takahashi, 2006,On Nonlinear Fitness Functions for Ranking-Based Selection, IEEE Congress on Evolutionary Computation

[40]. Uyar S., Sariel S. and Eryigit G., 2004, A gene based adaptive mutation strategy for Gas, Genetic and Evolutionary Computation – GECCO 2004Lecture Notes in Computer Science Volume 3103, 2004, pp 271-281

[41]. Ping-Hung Tang, Ming-Hseng Tseng, 2013, Adaptive dierected mutation for real coded genetic algorithm, Applied Soft computing 13, pp 600-614

[42]. Sivaraj R. and Ravichandran T, 2011, A review of selection methods in genetic algorithm, International Journal of Engineering Science and Technology (IJEST)

[43]. Elnima Ali, Elgasim Elamin, 2006, A proposed genetic algorithm selection method, Proceedings of the First National Symposium (NITS2006).

[44]. K.S. Tang, K.F Man, S. Kwong and Q.He, 1996, Genetic Algorithm and their application, IEEE SIGNAL PROCESSING MAGAZINE
[45]. *Goldberg, D. E. and Deb, K.*: A Comparative Analysis of Selection Schemes Used in Genetic Algorithms. In Foundations of Genetic Algorithms. San Mateo, California, USA: Morgan Kaufmann Publishers, 1991, pp. 69-93, 1991.
[46]. *Baker, J. E.:* Reducing Bias and Inefficiency in the Selection Algorithm. In [ICGA2], pp. 14-21, 1987.
[47]. Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems*. 2nd Edn. Cambridge, MIT Press.
[48]. *Blickle, T. and Thiele, L.*: A Comparison of Selection Schemes used in Genetic Algorithms (2. Edition). TIK Report No. 11, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH) Zürich, Switzerland, 1995.
[49]. Baker, J.E, Adaptive selection method for genetic algorithms, Proc. Of the 1st International Conference on Genetic Algorithm and Their Applications, Lawrence Erlbaum Associates, (1985), 101-111.
[50]. Goldberg D.E, 1990, A Note on Boltzmann Tournament Selection for Genetic Algorithms and Population-OrientedSimulated Annealing, Complex Systems Publications, Inc., pages: 445-460
[51]. Davis, L. (1991): *Handbook of Genetic Algorithms.* Van Nostrand Reinhold. New York, NY.
[52]. Ali M. S. Zalzala Peter J. Fleming, Genetic Algorithms in Egineering Systems, pages: 9-12
[53]. Carlos Conceição António, 2012 Selfish Gene theory and Memetic Algorithms: A fusion of concepts for robust design of hybrid composites