Adaptive Life-Cycle and Viability based Paramecium-Imitated Evolutionary Algorithm

MING-SHEN JIAN+, TA YUAN CHOU#, KUN-SIAN SIE#, LONG -YEU CHUNG+
+ Department of Computer Science & Information Engineering, National Formosa University, Huwei Township, Yunlin County, TAIWAN
#Department of Computer Science and Engineeing, National Sun Yat-sen University, Kaohsiung, TAIWAN
+Department of Applied Geoinformatics, Chia Nan University of Pharmacy & Science, Tainan, Taiwan
jianms@gmail.com+ tayuan@gmail.com# Chungly1@mail.chna.edu.tw+

Abstract: -This paper proposes a sim-paramesium genetic algorithm to enhance the searching and optimizing speed of classical genetic algorithms. Based upon classical genetic algorithms, the sim-paramesium genetic algorithm employs additional operators, such as asexual reproduction, competition, and livability in the survival operation. Taking the advantages of these three operators, the searching and optimizing speed can be increased. Experiments indicate that simulations with the proposed algorithm have a 47% improvement in convergence speed on the traveling salesman problem. Also, while applying the proposed method to solve the graph coloring problem, the proposed algorithm also has a 10% improvement in solution qualities. Furthermore, since these operators are additional parts to the original GA, the algorithm can be further improved by enhancing the operators, such as selection, crossover, and mutation.

Keywords: genetic algorithm, paramecium, traveling salesman problem, graph coloring problem, evolutionary algorithm

1. Introduction

Genetic algorithm (GA) was first proposed by Holland in 1975 [1] based on natural evolution selection “survival of the fittest.” In recent decades, Genetic Algorithms are widely used in many fields, such as optimization, scheduling problems which are usually proved to be NP-complete problems [1-7], *etc*. In Whitley’s research [8], the major factors hiding behind the structure of GA are selection pressure and population diversity. The selection pressure influences the fitness and the relations between chromosomes of the offspring and the parents. Mauldin [2] also confirmed the importance of maintaining the diversity of GA. In addition, in view of GA, researches with measures or extra conditions [9-12] are aspiring to not only natural way but performance.

In this paper, a new heterogeneous GA which could be compatible with most classical GAs is proposed to achieve better performance. The proposed method employs three operators, denoted as aging, competition, and reproduction. Furthermore, additional four parameters, “life bound”, “alpha value”, “beta value”, and “gamma value” are used to control those three processes. The three major contributions in this paper are:

1. Not only competition or fitness of each generation but also life cycle and reproduction based on biological resources are considered. In addition to the original competition of GA, due to the limitation of life cycle, there are not always enough individual selected for the next generation. Hence, considering the biological resources, the individuals with high competitiveness can compete for the rest biological resources and reproduce by asexual reproduction.
2. The asexual production consists of aging, competition and reproduction is proposed and independent from the crossover.
3. The paramecium-imitated which includes sexual and asexual production is proposed.
4. The proposed algorithm with new frameworks can converge faster and perform better than classical GA in solving the traveling salesman problem and the graph coloring problem.

The rest of this paper is organized as follows. In Section 2, the related work about improving GA efficiency is introduced. Section 3 defines the problem statement and presents the proposed algorithm. Performance evaluation is given in Section 4. Conclusion is drawn in Section 5.

2. Related Work

In the following, the survey of GA procedures, such as representation, selection, crossover, and mutation are described. Also, the sexual and asexual crossovers are demonstrated.

2.1 Representation

Representation, or chromosome encoding, is to transform all candidate solutions to the genotype form so that they can be operated in GAs. Also, the optimal solutions of the given problem should be covered by the representation so that GAs can have the chance to find out them. Therefore, the encoding of the problem may depend on the problem domain. In this paper, the traveling salesman problem (TSP) and the graph coloring problem (GCP) are considered in Section 3.

2.1.1 Representation of the Traveling Salesman Problem (TSP)

The goal of the traveling salesman problem (TSP) is to find the minimum trip of cities. The general presentation is to encode the permutation of cities which are visited sequentially. Take an 8-cities TSP for instance, the chromosome may be encoded as: 3 →8→1→6→7→2→4→5. Each gene number within this chromosome is the identifier of one city. The sequence of the permutation of these numbers means the order to visit cities.

2.1.2 Representation of the Graph Coloring Problem (GCP)

The graph coloring problem (GCP) is to minimize the number of colors (chromatic number, denoted by $\chi(G)$ needed to color the given graph with the basic rule – no two adjacent vertices get the same color. Concerning the essence of GCP, two main approaches differentiate by the numbers of the vertices are class numbers or color numbers. In this thesis, the latter is applied. Here is a 7-vertices GCP example, the chromosome is encoded as: S = {V_1, V_3, V_5}, {V_4, V_7}, {V_2, V_6}. Each vertex in the same bracket has the same color assignment. That means the V1, V3 and V5 have the same color.

2.2 Selection

In most selection operations, better individuals have more chances to reproduce better offspring. However, elitism is not always effective since heavy selection pressure can decrease the population diversity. Different selection methods influence the convergence speed [8]. In this paper, the roulette wheel approach [15] and Tournament approach [16] are considered.

The roulette wheel approach (RWA) selects individuals in proportion to the related position of fitness in the entire population. The RWA probability (P_i) of each individual would be

$$P_i = \frac{F_i}{\sum_{j=1}^{n} F_j} \times 100\%$$

where F_i indicates the fitness value of the ith individual. Then, candidate solutions with better fitness would have more chances to be selected than worse ones. Therefore, RWA would let the better individuals get more chance to generate their offspring.

Tournament approach takes the rule of ranking approach with racing. First, two candidates are selected randomly. Then, compare the fitness of these two selected individuals according to the objective function. Only the winner (or better one) can be selected to be the parent. The procedure is described as follows:

Step 1. Randomly select two individuals, denoted as a and b.

Step 2. Compare these two individuals. If the fitness of a is better than the fitness of b, then a is selected. Otherwise, b is selected.

Step 3. If there are enough parents, go to crossover, otherwise go to **Step 1**.

2.3 Crossover

The effect of different crossover operators is investigated for many years [13]. Even with the same crossover operation, the performance still varies with the crossover rate [14]. In the following, the definitions of crossover used in this paper are presented.

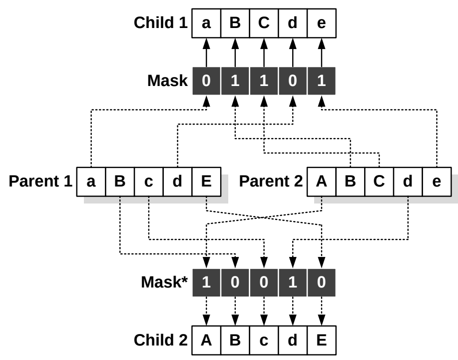

Fig. 1. Example of the UX. The child 1 and child 2 are generate by using the given mask and the inverse mask, named as mask*

2.3.1 Uniform Crossover
Uniform crossover (UX) is also a famous crossover. A binary mask, which has the same length of bits with all candidate solutions, is firstly constructed. Each bit on the binary mask is 0 or 1, is used to be the selection basis. According the mask, the i^{th} gene of parent one is copied to the i^{th} gene of child one when the i^{th} bit of the mask is 0, and vice versa. A simple example for UX is shown in Fig 1.

2.3.2 Greedy Partition Crossover
Greedy partition crossover (GPX) is proposed by Galinier and Hao [19] for solving the graph coloring problem. This method constructs the partial color classes to build the child successively in a greedy fashion. The class with the largest cardinality of one selected parent is copied to the child. The vertices of this class are eliminated from the other parent. Figure 2 shows a simple example with 10 vertices and 3 color class. The characters A, B, C, …, J are the identifiers of all vertices. Each column is one color class.

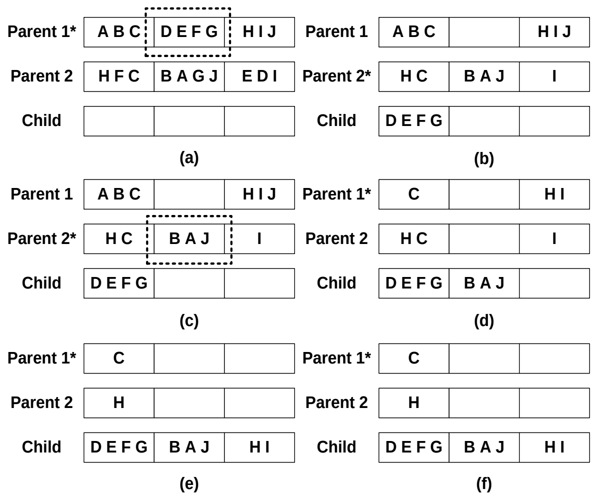

Fig. 2. Example of GPX. (a) Find the group with maximum number of vertices in parent one. (b) Take this class to the child and remove those vertices of this class in parent two. (c) Find the group with maximum number of vertices in parent two. (d) Do the same operation as (b). (e) Then, three color classes of child are filled. (f) The rest vertices are put in a color class randomly.

2.4 Mutation
There are many mutation methods have been developed. Here, we describe two famous mutations, such as reciprocal exchange mutation and inversion mutation.

2.4.1 Reciprocal Exchange Mutation

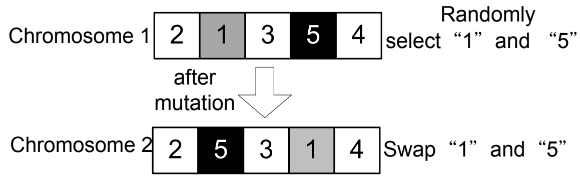

Fig. 3. A simple instance of Reciprocal Exchange Mutation

In this type of mutation operation, two genes are randomly selected and interchanged, as shown in Fig. 3.

2.4.2 Inversion Mutation
As literally, inversion mutation reverses a section of genes. So this method causes more variations than reciprocal exchange mutation. Figure 4 is an example of inversion mutation.

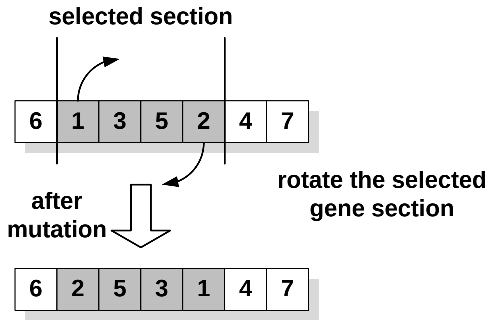

Fig. 4. An example of inversion mutation

2.4.3 Sexual and Asexual Genetic Algorithm
Freisleben and Merz [20] developed genetic local search operators incorporating domain knowledge into a genetic algorithm which would almost lead fitness to the best answer. However, they mentioned that additional local search operators can decrease the efficiency of GA algorithm, so the complexity of their method must be reduced to speed up the computational time. Besides, their algorithm is specially designed for the traveling salesman problem. Transforming the local search method of their algorithm to other problem may be very difficult or unsuitable for that problem. In addition, fast convergence speed usually does not lead to the best answer. In other words, the disadvantage of local search is the quick premature convergence of population. Local searches would decrease the diversity of whole population. However, to keep the diversity of population is always a highlighted point

for researchers of GA. Mutation plays a key role to maintain diversity of population and avoids risk of getting premature convergence in natural evolution. Mutation is a key point to keep the variety of population, since it produces the genes different than its parents even though parents are the same. Furthermore, Mauldin [2] developed an important genetic search to maintain the diversity. He confirmed that ability of guaranteeing genetic diversity is equal to the robustness of genetic search and degree of decreasing uniqueness. Besides the improvements on standard GA operations, there are other researches working on modifying GA frameworks. Multi-parents of GA is proposed in the 1960s [1] [20] [21]. Tsutsui [22] points out that multi-parent recombination would drive a better answer than traditional two-parent recombination.

The word “sexual” has several meanings in GA. Here, we employ the meaning of “sexual” that signifies the way to generate next generations that could be sexual or asexual by adjusting the GA parameter “crossover rate.” Sexual crossover takes more than two parents to generate the children. In the other way, asexual crossover is only to duplicate the parent to the children.

3. The Proposed Algorithm

Although GA is recognized as a powerful and heuristic algorithm, efficiency is still an important issue. The algorithm proposed in this research presents another view of improving GA’s efficiency from the natural point of view. The proposed algorithm with the new framework of genetic algorithm, called paramecium-imitated evolution algorithm focuses on enhanced livability and competition, by adding additional procedures to a standard GA’s procedure. The motivation of combining livability and competition is that GA is a simulation of natural species to evaluate the most suitable pattern for environment, and the evaluation speeds varies according to their species. The first idea of this thesis is to intensify the evolution speed. The stronger the survival is, the better the remained individuals will be. Therefore, the first idea of this research is to identify a different survival method, which is the competition element in this research. The competition method is proposed to improve the convergence speed (The details of the competition method would be described afterward). But the population diversity would be fast reduced by the competition method in the experiments. Even GA with the competition method has better fitness behavior earlier, but sooner or later the population would be premature. Figure 5 shows this situation.

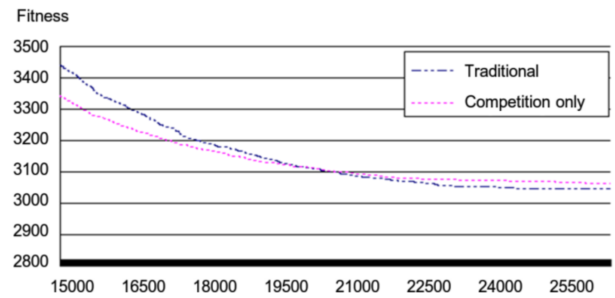

Fig. 5. Simulation result of TSP “a280” shows influences of the competition method (population size: 100, reproduction rate: 0.8, mutation rate: 0.5, alpha value: 1, beta value: 1)

Therefore, the life method is the idea added into the thesis to improve the diversity and prevent the population from fast pre-maturating when the competition method always duplicates the same individual. The method proposed here is to combine these two methods to improve the effect of GA. Therefore, adding livability and competition is the solution to make the simulation of nature more practical.

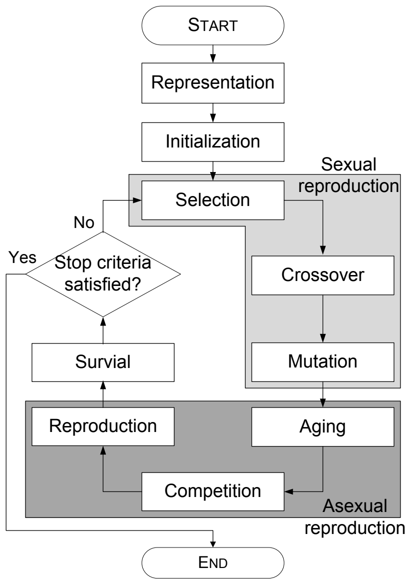

Fig. 6. The flowchart of the proposed algorithm

Figure 6 shows the flowchart of the proposed algorithm. Obviously, there are three added functional blocks in the rear of the standard GA functions. The middle block “sexual reproduction” in the flowchart is the classical GA flow except the survival function is taken to the end of all procedures. The main characteristic of the proposed algorithm is the asexual reproduction block after the sexual reproduction.

The proposed algorithm has the same procedures as a simple GA, except that there are three extra procedures, such as "aging," "competition," and "reproduction" added after the mutation procedures. Four parameters, such as "*life bound*," "*alpha*," "*beta*," and "*gamma*" are added in these three new procedures.

The "*life bound*" is literally the maximum number of generations for the individual. The "aging" process relating to the "*life bound*" let individuals be dropped when its age exceeds the "*life bound*" parameter, but oppositely the "reproduction" allows the better individual to live again.

The individual of the better fitness has the chance to duplicate itself when it is selected in the process of "competition." If there is only "aging" function, the best fitness of entire population would be reduced greatly due to the best individual would be swapped out when it reaches the life limit. With the combination of those two functions, the best fitness would have chances to duplicate itself. Details of these functions are described in the following paragraphs.

3.1 Aging

The "aging" block with the parameter - "*life bound*" would like to age the individual and allow those individuals that get older than "*life bound*" to be dropped. The diversity of the population can be kept by constantly swapping out some individuals that exist in the population very long time. Figure 7 is the flowchart of aging function.

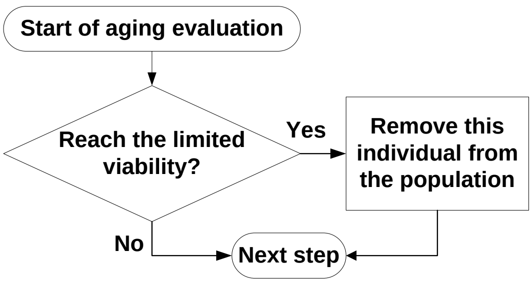

Fig. 7. The flowchart of aging function

The aging function may even remove the current best solution if it reaches the life limit. So that, the current best solution would usually become worse than the former generation. Additional procedures, "competition" and "reproduction" are used to counteract the effect produced by the "aging" procedure.

3.2 Competition

The "*alpha*" value is to determine how many individuals in the top of entire population are picked to proceed the competition and the reproduction functions. On the contrary, the "*beta*" value is to choose those genes in the bottom of total population. Figure 8 shows the flowchart of comparing individuals in entire population. In this diagram, α individuals in front of the population are taken to be compared with β individuals in the rear.

The number of individuals after aging operation is called "*alive number.*" Before operating the real competition operation, here comes the estimation of "alive *number*" first. The total number of population size before "aging" procedure is *populationsize* $\times$ (1+*reproduction rate*).

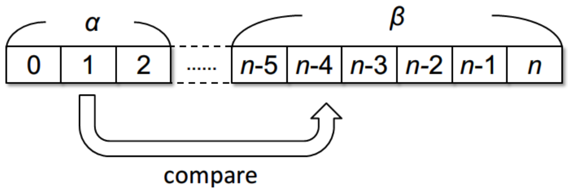

Fig. 8. The flowchart of comparing individuals in entire population

Due to the operator of "aging," the "*alive number*" would not be more than the total number of population size after dropping some older individuals. So "*alive number*," may have two cases: **1**. "*alive number*" is larger than the number could enforce the natural law "survival of the fittest." **2**. the "*alive number*" is insufficient.

"The number which could enforce natural law" mentioned here means the number is enough to proceed the survival operator in GA operators. Because amounts of creatures in the natural environment would keep on a quantity related to the food and the living space supplied by the environment. On the contrary, it is no doubt that with enough food and space, species would keep growing with no competition until the amount of "population size" reach the bound of the natural supply. So clearly, "The number which could enforce natural law" is the "population size" parameter of GA. Then, when case one happens, the proposed algorithm would execute the right flow of the competition procedure as shown in Figure 9. If case two happens, the "alive number" is smaller than the parameter "population size," the proposed algorithm would go the left flow and then go to the next procedure.

After entering the right-hand side flow of the competition flowchart, the operation described in the front of this section is taken. The values of "*alpha*" and "*beta*" mean the groups of individuals to compare and to be compared respectively. The competition operation takes the third parameter "*gamma*," which is a threshold value to compare those individuals of two sides. If the fitness of the better individuals subtracts the fitness of the worse one is larger than "*gamma*" value, the worse individuals are marked to perform the replacement.

In Fig. 9, the competition flowchart, the sorting operation is to sort all individuals in order to allow quick process in the next step. Block 1 is to take the better individuals in the population according to the value of "*alpha*" to be compared in block 2. If the value of "*alpha*" is 1, the best individual would be selected. The result of comparing is the mark of individuals to decide which individuals has the quality to reproduce itself as shown in block 3.

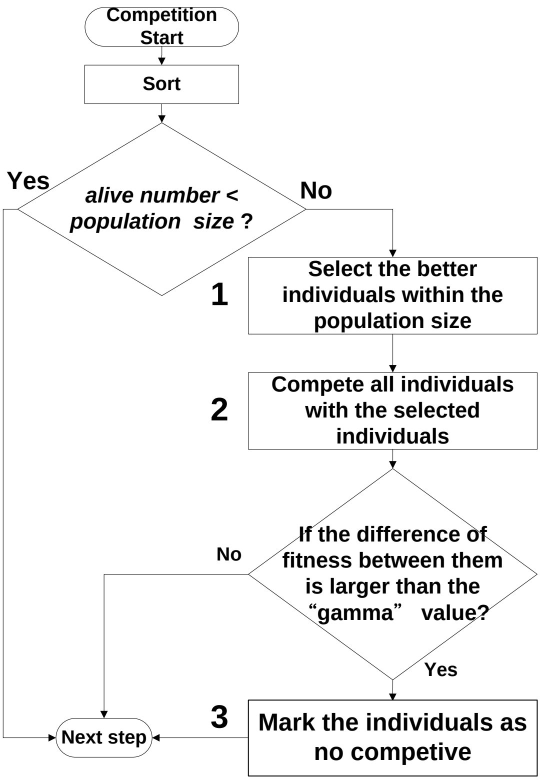

Fig. 9. The flowchart of competition operation

3.3 Reproduction

The flowchart of reproduction is shown in Fig. 10. After the aging procedure, there are two cases about the "alive number." If the "alive number" is smaller then the "population size," the left-hand side of reproduction procedure is applied. This left procedure would duplicate some of better individuals to fix the insufficient quantity of population. On the other aspect, if the number of individuals is larger than the "population size," the right-hand side procedure is triggered. Those worse individuals would be replaced by some better individuals.

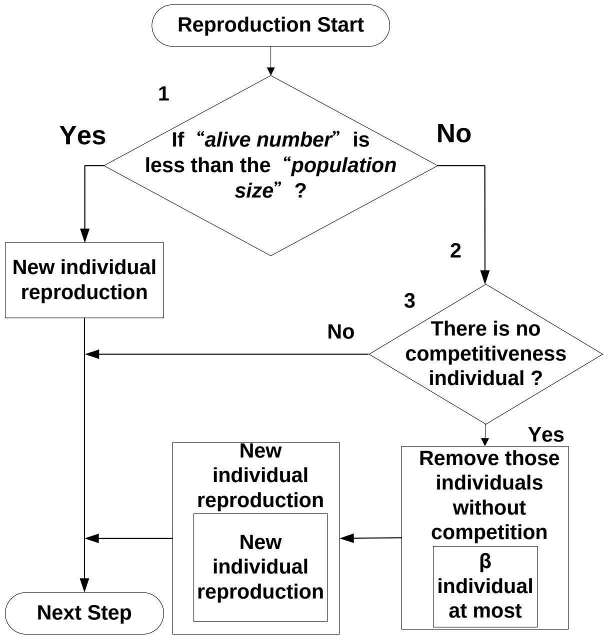

Fig. 10. The flowchart of reproduction

In Fig. 10, the decision block 1 determines if the "alive number" is enough to replacement procedure in the right-hand side of the flowchart. On the other aspect, if the "alive number" is less than the "population size," the decision block 2 in the right-hand side is taken.

Considering decision block 2, it is to decide if there are any marked individuals. When there exists marked individuals, the process would replace those marked individuals with the best individuals. Otherwise, the process goes to next step.

3.4 The Parameters Setting Method

Based on experience a setting method which gets better results than the blind searching is given. The values of "*alpha*" and "*beta*" are set to 1. The value of "*gamma*" can be yielded according to the following function:

$$Gamma = \frac{fitness\ of\ the\ best\ individual\ in\ the\ first\ generation}{population\ size ?\ (1 + reproduction\ rate)}$$

Experiments shows that the "life bound" greatly influences the performance and is also problem dependent. The value of "life bound" could be assigned a large value (such as 20, 50…) during initialization. The dynamic aging adjusting function is given below.

"*life bound*" = "*life bound*" – 1 until

$$Aging\ number > \frac{reproduction\ rate ?\ population\ size}{"life\ bound"}$$

3.5 The Complexity Analysis
There are three operations in the flow–aging, competition, and reproduction. It is obvious that the complexity of aging operation is O(n), n is the number of the total population size, i.e. *population size* × (1 + *reproduction rate*). Sorting operation and comparing operation are two basic operations of the competition operation. Sorting operation is O($n$log$n$). In addition, the complexity of comparing operation is according to the values of "*alpha*" and "*beta.*" The complexity of comparing operation is O($n$log$n$) at most because the value "*alpha*" plus "*beta*" should be less than n. Because the reproduction operation is to replace the marked individuals, the complexity of it is obviously O(n). So it is clearly that the complexity of three operations proposed in this research is O($n$log$n$).

4. Performance Evaluation
This section presents the performance of the proposed algorithm. To make a comparison, Table 1 describes the GAs used in this simulation.

Table 1 The standard GAs with different methods

Name	Selection method	Crossover method	Mutation method
Standard GA1	Tournament	UX	Inversion
Standard GA2	Tournament	GPX	Reciprocal Exchange

In addition to the four parameters (population size, reproduction rate, mutation rate, and crossover rate) used by the standard GA, the proposed algorithm would use four external parameters (alpha, beta, gamma, and life bound). Those four external parameters have to be set properly in order to obtain the best performance. Otherwise, as the simple GA, the performance may be down because the inappropriate values. The parameters setting method is discussed in Section 3.

4.1 Traveling Salesman Problem
The traveling salesman problem is one of the NP-complete problem and also a classic combinatorial optimization problem which is difficult to solve. Here is the performance post of the benchmark "a280" (280 cities) in [24]. Table 2 lists the parameters used in this simulation.

Table 2 Parameters and values used in the traveling salesman problem simulation

Parameters	Values	Parameters	Values
Population size	100	Alpha	1
Reproduction rate	0.8	Beta	1, 3, 5
Crossover rate	1.0	Gamma	500, 1000, … , 2000
Mutation rate	0, 0.1, 0.2, … , 0.5	Life bound	1, 3, … , 100

The attribute "Parameters" shows the name of the parameter. The attribute "Values" shows the value of related parameter. And the attribute of "Definitions" indicates the definition of the corresponding parameter. The four parameters in first four rows of the table are the parameters used in GA. The rest of parameters are used by the proposed algorithm. The parameter of "beta" takes three values. The parameters of "gamma" and "life bound" take five values.

The test of performance in different mutation rates is shown in Fig. 11. The best performance of standard GA1 in Table 4.1 is with the mutation rate = 0.5 that will be used in the comparisons.

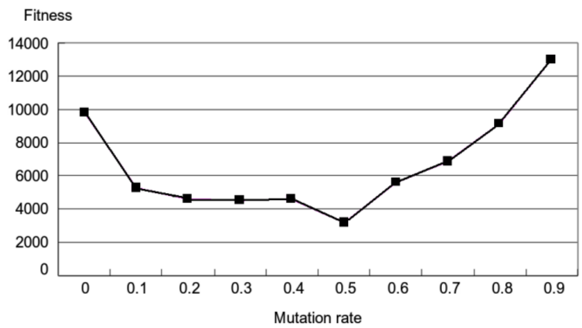

Fig. 11. The performance of different mutation rates in TSP for the standard GA1

The standard GA to be compared in Fig. 11 is the *Standard GA1* described in Table 1. From Fig. 12, among different "*life bound*" it performs the best when the "*life bound*" = 1. Among different "*gamma*" values it performs the best when "*gamma*" = 500. The improvement ration among all simulations ranges from -0.1% to 47%.

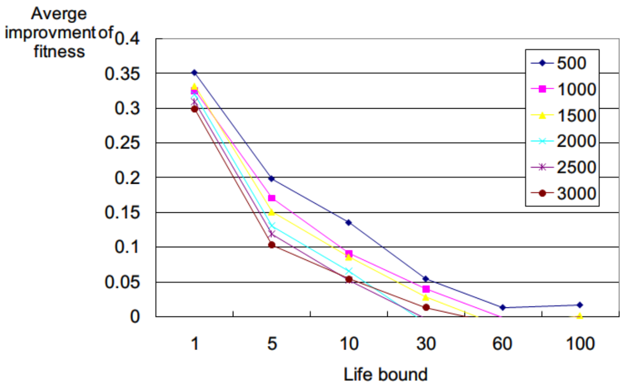

Fig. 12. Simulation result of TSP shows influences between *life bound* and *gamma*. (*population size*: 100, reproduction *rate*: 0.8, *mutation rate*: 0.5, *alpha* value: 1, *beta* value: 1)

Figure 13 is the convergences of the proposed algorithm, the *standard GA*1 runs for 30000 generations. Since "*life bound*" = 1 and "*gamma*" = 500 yield the best solution in Fig. 10. The proposed algorithm using the setting method takes the parameter values - "*life bound*" = 50 and "*gamma*" = 175. In Fig. 12, the convergence speed of the proposed algorithms is obviously faster than the *standard GA1*. In addition, the average improvement ratio of the setting method is 32.06%. Compared with Fig. 10, this improvement ratio is not the best result found in whole simulations but still a better one.

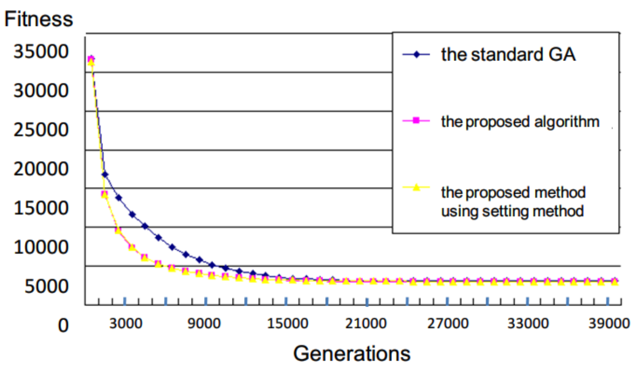

Fig. 13. The convergences of the *standard GA*1, the proposed algorithm (*population size*: 100, *reproduction rate*: 0.8, *mutation rate*: 0.5, *alpha* value: 1, *beta* value: 1, *life bound*: 1, *gamma* value: 500) and the proposed algorithm using the setting method (*population size*: 100, *reproduction rate*: 0.8, mutation *rate*: 0.5, *alpha* value: 1, *beta* value: 1, *life bound*: 50, *gamma* value: 175)

4.2 Graph Coloring Problem

Given an undirected graph $G=(V,E)$ that $V=\{v_1,v_2,\ldots,v_N\}$ is the set of vertices and $E=\{e_{ij}|\exists$ an edge between v_i and $v_j\}$ is the set of edges. The GCP is to find the partition of V with a minimum number of color classes so that every edge, $e_{ij} \in E$, v_i and v_j are not put in the same class. The Leighton graphs (le450) are used. Those parameters required for this simulation are listed in Table 3.

Table 3 Parameter used in the simulation (GCP)

Parameters	Values	Parameters	Values
Population size	100	Alpha	1
Reproduction rate	0.8	Beta	1, 3, 5
Crossover rate	1.0	Gamma	5, 10, …, 30
Mutation rate	0, 0.1, 0.2, … , 0.9	Life bound	1, 2, … , 100

The test of performance in different mutation rates is shown in Figure 14. The settings of *the Standard GA2* in Table 4.1 are used in this simulation. The selection function is tournament selection. The GPX and the reciprocal exchange mutation are applied. The *standard GA*2 has the best performance when mutation rate is 0.9. Figure 14 shows the improvement performance of the proposed method with mutation = 0.9.

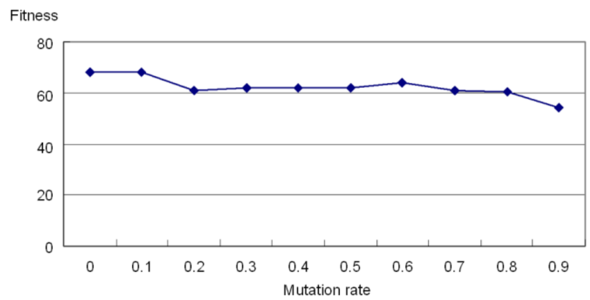

Fig. 14. The performance of different mutation rates in GCP for the *standard GA2*

The average improvement ratio compared with the Standard GA2 is given in Fig. 15. Though the performances of these GCP ratios in Fig. 14 are not as significant as that of TSP, it still gains the best 10% improvement when gamma and life bound are 8 and 15, respectively. In general, the improvement rates vary from -0.6% to 10%.

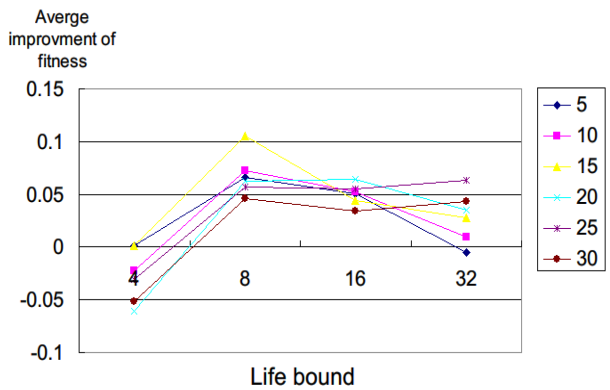

Fig. 15. Simulation result of GCP shows influences between *life bound* and *gamma* (*population size*: 100, *reproduction rate*: 0.8, *mutation rate*: 0.9, *alpha* value: 1, *beta* value: 1)

Figure 16 shows the speed of convergence. In this simulation, the improvement of the proposed algorithm is not only on the convergence speed but also on the best fitness. The best improvement rate is 10%, when *alpha* = 1, *gamma* = 15 and *life bound* = 8.

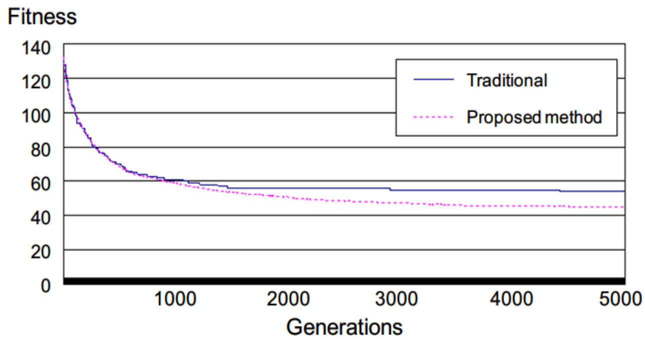

Fig. 16. The convergences of the proposed algorithm and *standard GA*2 (*population size*: 100, reproduction *rate*: 0.8, *mutation rate*: 0.9, *alpha* value: 1, *beta* value: 1, *life bound*: 10, *gamma* value: 30)

5. Conclusions

We have proposed a Paramecium-Imitated Evolutionary Algorithm. The proposed algorithm consists of sexual and asexual modules. Based on simulations, if the parameters are set well, the performance of the proposed algorithm gains even 20% better than *the Standard GA1* in TSP. Although the best fitness is the same between the proposed algorithm and the *Standard GA1*, the convergence speed of the proposed algorithm is evidently faster than the *Standard GA1*. The proposed algorithm also provides a better fitness that is
about 10% improvement in GCP compared with the *Standard GA2*.

References

[1] J. H. Holland, *Adaptation in Natural and Artificial Systems*, First Published by University of Michigan Press, 1992.

[2] M. L. *Mauldin*, Maintaining Diversity in Genetic Search, *Proceedings of the National Conference on Artificial Intelligence AAAI-84*, pp.247-250, 1984.

[3] L. A. Rendell, A *Doubly* Layered Genetic Penetrance Learning System, *Proceedings of the Third National Conference on Artificial Intelligence AAAI-83*, pp.343-347, 1983.

[4] M. Kavian and R. G. McLenaghan, Application of Genetic Algorithms to the *Algebraic* Simplification of Tensor Polynomials, *Proceedings of the 1997 international symposium on Symbolic and algebraic computation*, pp.93-100, 1997.

[5] J. Frausto-Solis, J. Mora-Vargas, M. Larre, J. L. Gomez-Ramos, "A New Genetic Algorithm for the University TimeTabling Problem Using Forced Diversity," *WSEAS Transactions on Systems*, Vol.5, Issue 10, pp. 2398-2403, 2006.

[6] M. Rafiei, "Genetic Algorithms Based Approaches to Verify Global Optimality of the OFC Active Power Filters," *WSEAS Transactions on Systems*, Vol.5, Issue 10, pp. 2404-2410, 2006.

[7] O. Ladoukakis, S. Tsitmidelis, A. Ktena, "A New Genetic Algorithm for Motor Parameter Estimation," *WSEAS Transactions on Systems*, Vol.5, Issue 10, pp. 2430-2433, 2006.

[8] D. Whitley, The *GENITOR* algorithm and selection pressure: Why Rank-Based Allocation of Reproductive Trials Is Best, *Proceedings of 3rd International Conference on Genetic Algorithms*, pp.116-121, 1989.

[9] C. K. Ting, S. T. Li, and C. N. Lee, On the harmonious mating strategy *through* tabu search, *Information Sciences*, pp.189-214, 2003.

[10] C. H. Su and C. N. Lee, A Multi-Parent Crossover for Combinatorial Optimization *Problems*, Master Thesis of National Sun Yat-Sen University, Kaohsiung, Taiwan, 2006.

[11] S. Tsutusi, Multi-Parent Recombination in Genetic Algorithm with *Search* Space Boundary Extension by Mirroring, *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature*, pp.428-437, 1998.

[12] A. E. Eiben, *Multiparent Recombination in*

Evolutionary Computing, Advances in Evolutionary Computing: theory and applications, Springer-Verlag New York, 2003.

[13] W. Rand, R. Riolo and J. H. Holland, The Effect of Crossover on the Behavior of the GA in Dynamic Environments, *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pp.1289-1296, 2006.

[14] Y. Hong, Q. Ren, and J. Zeng, Adaptive Population Size for Univariate Marginal Distribution Algorithm, *Proceedings of the Evolutionary Computation*, pp.1396-1402, 2005.

[15] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.

[16] C. R. Houck, J. A. Joines, and Michael G. Kay, A Genetic Algorithm for Function Optimization: A Matlab Implementation, North Carolina State University, Raleigh, NC, Technical Report, 1995.

[17] D. E. Goldberg and R. Lingle, Alleles, Loci and the Traveling Salesman Problem, *Proceedings of the First International Conference on Genetic Algorithms*, pp.154-159, 1985.

[18] E. Falkenauer, The Worth of the Uniform, *Proceedings of the Evolutionary Computation*, pp. -782 Vol. 1, 1999.

[19] P. Galinier and J. K. Hao, Hybrid Evolutionary Algorithms for Graph Coloring, *Journal of Combinatorial Optimization*, pp.379-397, 1998.

[20] B. Freisleben and P. Merz, New Genetic Local Search Operators for the Traveling Salesman Problem, pp.890-899, *Proceedings of the Fourth International Conference on Parallel Problem Solving from Nature*, 1996.

[21] J. T. *David* and P. Sampat, Mathematical Programming in a Hybrid Genetic Algorithm for Steiner Point Problems, *Proceedings of the ACM symposium on Applied computing*, pp.357-363,, 1995.

[22] S. Tsutusi, Multi-Parent Recombination in Genetic Algorithm with Search Space Boundary Extension by Mirroring, *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature*, pp. 428-437, 1998.

[23] S. V. Jose and A. B. John, Sexual Selection with Competitive/co-operative Operators for Genetic Algorithms, *Proceedings of the IASTED International Conference on Neural Networks and Computational Intelligence*, pp.191-196, 2003.

[24] http://elib.zib.de/pub/mp-testdata/tsp /tsplib/tsplib.html