A Novel Mechanism for Efficient the Search Optimization of Genetic Algorithm

Chen-Fang Tsai
Department of Industrial Management and Enterprise Information,
Aletheia University, Taiwan.
E-mail: au1204@mail.au.edu.tw

Shin-Li Lu*
[*]Department of Industrial Management and Enterprise Information,
Aletheia University, Taiwan.
E-mail: shinlilu@mail.au.edu.tw

Received 25 June 2015

Accepted 16 November 2015

Abstract

This paper proposes a Social Genetic Algorithm (SGA) that includes a transformation function that has ability to improve search efficiency. The SGA is different from the Traditional Genetic Algorithm (TGA) approaches, as it allows refinement of the TGA parameters for the selections of operators in each generation with two functions: optimization of crossover rate and optimization of mutation rate. In this paper, a new function that optimizes gene relationship has been introduced to advance the evolution capability and flexibility of SGA in searching complex and large solution space. Our proposed approach has been evaluated using simulation models. The simulation results have shown that SGA outperforms TGA in improving search efficiency. The contribution of the proposed approach is a dynamic and adaptive methodology, which has ability to improve efficiency.

Keywords: Genetic algorithm, Adaptive Crossover, Adaptive Mutation.

1. Introduction

Selection, crossover and mutation are three GA's operators used in the process of generating a set of solutions and then selecting an optimal one (Giddens, 1994; Kim and Cho 2015). Crossover is a process in which two selected individuals swap variables to produce two different individuals, which can then be added to the population. Crossover results in a random information exchange and each alternative solution created contains some of the characteristics of both parents (Moin et al., 2015). Mutation is a secondary operator in TGA and it has proven to be an essential component of the TGA paradigm (Long et al., 2015). After offspring have been produced, they are evaluated and may replace existing members in the population.

One of the key difficulties in TGA is to find equilibrium between exploitation and exploration in evolutions. Many studies have attempted to provide a better stability between them, but the equilibrium is not easy to achieve due to inversely associated factors i.e. selection pressure and population diversity (Fazlollahtabar et al., 2012; Tsai et al., 2008; Deep and Singh, 2015). Increasing population diversity will result in loss of selective pressure, whilst reducing selective pressure will increase the potential of exploration in the search space. Schaffer and Morishima (1987) introduced a mechanism of punctuated crossover in which distribution of crossover points is based on

[*]Shin-Li Lu: Aletheia University, 32 Chen-Li Street, Tamsui, New Taipei City 251, Taiwan.

performance of the generated offspring and can identify the location of better crossover points. The outcome of this approach is better than the ones produced by other traditional crossover operators, but it effectively doubles the variable's length.

The traditional optimization algorithms lack a dynamic adaptive mechanism to effectively search feasible solution pools and they also intend to adopt a fix structure in solution representation that leads to an inadequate efficiency in search. Most researchers use fixed crossover and mutation rates throughout the evolution process which could also hinder the search ability.

Considering these limitations in the traditional approaches, the proposed SGA is able to improve the efficiency of TGA evolution process. The improvement attributes to a novel transformation algorithm that is able to adaptively set crossover and mutation rates as well as optimizes gene relationships based on search states in each generation. This enhances the search capability and can be applied to optimize (Green Product Design; GPD) cost function by generating and evaluating different designs within affordable cost for the suppliers to meet dynamic market demands (Tsai et al. 2015). The rest of the paper is organized as follows. Section 2 presents literature review. Section 3 describes social genetic algorithm. Section 4 discusses simulation results of the proposed algorithm. Finally section 5 concludes the paper.

2. Literature Review

A chromosome is organized as a set of genes, which are sections of the chromosome representing individual variable (Schaffer and Morishima, 1987; Han et al., 2015; Wang 2015). It is therefore imperative to design a suitable structure of the chromosome. Individual chromosomes in population are given opportunities to reproduce, often referred to as reproductive trials for the production of offspring. The number of opportunities each individual has is proportion to its fitness, so the better individuals contribute more of their genes to the next generation. It is assumed that an individual having high fitness is due to the fact that it contains superior schemata (Ripon et al., 2007; Gibbs et al., 2008; Wang, 2015). As the composition of these genes significantly affects the TGA's performance, an improper arrangement for the chromosome structure often results in poor performance (Schaffer, 1985).

Holland's schema theorem (Holland, 1975) was the first attempt using formal descriptions to describe how a TGA works. A schema is a pattern of genes consisting of a subset of genes at certain positions in a chromosome (Moon, 1994). Schemata are also known as similarity subsets because they represent subsets of strings with similar patterns at certain fixed position. By passing more of these superior schemata on to the next generation, it increases the likelihood of finding better solutions. Schemata possess two essential properties: the order of the schema and its length.

The power of a TGA lies in it being able to find high-quality building blocks. These are schemata with short defining length consisting of bits which work well together, and they tend to lead to improved performance when they are incorporated into a chromosome (Schaffer, 1985). A successful coding scheme is one which encourages the formation of good building blocks by ensuring that related genes are close together in the chromosome, whilst there is only a little interaction between well separated genes (Rosca, 1995). However, when large numbers of interactions between genes are required or they cannot be avoided, Gibbs's recommendations (Gibbs, 2008) should be taken into consideration in the design of the schemes to ensure that TGA performs well. In this research, we try to propose a transformation function to refine the chromosome structure in a dynamic adaptive TGA which conforms to Gibbs's recommendations. Such a modification usually leads to an improvement in TGA's search. The goal is to regain beneficial genes that were lost through poor selection of mates in the evolution process.

Alfaro et al. (2009) proposed an effective approach of adapting operator probabilities based on the performance of the operators. The conversion mechanism varies operator probabilities in proportion to the fitness value of strings created by the operators. The simulation results have shown a substantial improvement in the performance of a TGA. Hence, the dynamic demands for parameter settings have grown rapidly in recent years.

Eshelman et al. (1989) utilized a mechanism of mutation in which mutation rate is a dynamic parameter according to the Hamming distance between the parent solutions. The simulation outcomes have demonstrated significant improvements in TGA's performance.

Several researchers (Srinivas and Patnaik, 1994; Rosca, 1995) found that the crossover rate and mutation rate has great correlation with its fitness and the larger the crossover rate in combination with smaller mutation rate can get TGA better in search efficiencies (Thierens, 2007). They proposed different methods to determine the mutation rate specifically for each solution and maintain the exploration without affecting the exploitation properties. Hence, the study attempts to provide an interactive and analytic controller to find an optimal solution. It modifies the cost variable relationships and TGA parameters settings after each evolution according to the resulting fitness. In their research, they only carried out sensitivity analysis on either mutation rate or crossover rate in relation to fitness function, so they did not examine both rates at the same time due to lack of a transformation function to relate the behaviour of the rates to the outcome of fitness function.

We have designed a self-adjusting mechanism to enhance the versatility and generality of a TGA. The mechanism allows refinement of TGA parameters for the selections of operators in each generation (Tsai et al., 2008; Tsai et al., 2011). The parameters setting carried out by a transformation function are to tune the trade-off between exploration and exploitation in search space. The function varies the crossover and mutation rate at each evolving generation and their new rates are derived based on the assessment of the fitness value in order to prevent premature convergence. The transformation function uses evolution evidence, which includes the population diversity and the level of the parents' fitness, to set TGA parameters dynamically on each generation. This function is the main body of Social Parameter Controller as part of the proposed SGA. The next section reports an overview on SGA with two controllers Social Parameter Controller and Social Relation Controller to optimize TGA social behaviors.

3. Social Genetic Algorithm

SGA can vary all of the process parameters in order to locate the global optimum in search space. However, an inherent problem of TGA is that much more time is wasted in tuning the parameter values (i.e. crossover and mutation rate). Hence, this research proposes a dynamic social function, which assists users in selecting optimal parameters for better solution of the problems. It attempts to balance the relationship between the exploration and exploitation search of TGA evolutions, which is aimed at improving the efficiency of TGA management. The main steps for TGA and SGA are summarized in Figure 1.

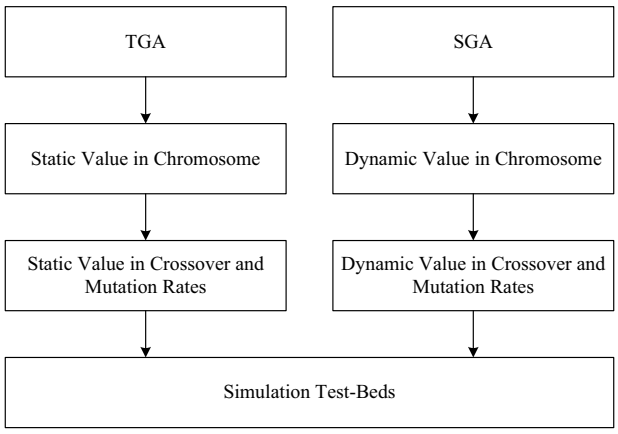

Figure 1. Process of TGA and SGA in simulation test-beds

3.1. *Social-Parameter Controller for SGA Search Optimizer*

The proposed SGA is able to optimize crossover and mutation rates to preserve beneficial factors in chromosomes. The preservation of beneficial cost variables is important, which are often necessary for complex problems with a rugged fitness landscape. It is therefore imperative to design an effective structure for modelling cost variables. Hence, the proposed approach employs dynamic social abilities and problem domain knowledge to adjust positions of related genes within variable structure. The transformation function in the Social Parameter Controller alters crossover and mutation rates in every evolution, and a new rate being established on a shifted fitness value. The algorithm applies evidence of shifted value, which presents the population diversity and the level of fitness deviation to manage TGA parameters. We also design an algorithm to optimize gene structure that is able to preserve positive building blocks. The detailed descriptions and analysis on Social Parameter Controller is omitted in this paper, as they can be found in our previous work (Tsai et al., 2008; Tsai et al., 2011). The next subsection will report Social Relation Controller, a new function in SGA.

3.2. *Social-Relation Controller for SGA Search Optimizer*

One of the main capabilities of the proposed mechanism is that it is capable of optimizing cost variables of fitness (cost) function by identifying their internal variables social-relations to form better chromosome structures in the evolution process (Han et al., 2015). Each cost variable in the fitness (cost) function is represented as strings to form part of chromosome, so a chromosome consists of a list of cost variables, which are also factors in the fitness function. It is necessary to group the important variables or critical factors of the fitness function together in the chromosome in order to identify and preserve beneficial ones.

We have designed a Social Relation Controller to automatically and systematically identify and preserve beneficial genes or cost variables. The controller includes three operations: Social Classification, Social Association and Social Sequencing to evaluate variables importance, give the appropriate weightings and re-order them. Social Classification Operation (SCO) evaluates the relative effect of each variable on the fitness (cost) function. Social Association Operation (SAO) assigns different weightings based on their relative effects. Social Sequencing Operation (SSO) optimizes the sequence of the cost variables to produce a refined chromosome structure to preserve positive building blocks (See Table 1). These three operations are invoked in sequence in each generation in the evolution process. After a number of generations, the controller is expected to preserve beneficial chromosomes. When the values of variables in the population are similar to each other, the evolution process converges. It means that parents and off-springs have similar chromosomes and an optimised solution is expected.

SCO evaluates the relative effect of each variable on the fitness function. It attempts to identify the beneficial variables within the cost variable structure in three steps. First, it calculates a ratio, called Affect Ratio (AR) of each variable, representing the relative fitness of the variables to the cost function (fitness function). It identifies and selects important variables based on the AR value of each variable. The second step sort the variables in descending order according to the AR values derived in Step 1.

The above fitness values derived from the SCO can be used in the next operation, SAO. The controller employs ratios of the fitness values to define the association relationships between variables. Step three is to divide variables into groups. Initially only two groups are designed to accommodate all variables. The important variables have higher AR. SAO uses AR values to re-sequence the order of variables as (V_6, V_4, V_2, V_3, V_1, and V_5). The control variables produce their ratio through the multiplication of their original variable ratios by the weights. After SCO and SAO have been carried out, it applies SSO to optimize the variable sequence of the cost variables to adjust the chromosome structure to preserve positive building blocks.

Table 1. Process of Social-Relation operation

(1). Affect Ratio (V_{iAR}) = V_i / V_{TC}). ((V_{iAR} =Set a ratio of cost of a variable to the total cost, V_i (individual cost) and V_{TC} (total cost value))
(2). Calculate Social Classification Operation (SCO) from AR of V_i (V_{iAR}) and place the best one in the front and the worst in the end. They can be expressed as (V_6, V_4, V_2, V_3, V_1, and V_5). In this case, V_6 is the best and V_5 is the worst one. (see Table 2).
(3). Perform Social Association Operation (SAO) to transform the total fitness (cost) value. (TV_n) = W_n (weighting) ×V_i (variables cost value). Select a suitable weighting vector (Transformation Value) (see Table 2). (TV_n) = W_n × V_n =TV_1; TV_2; TV_3; TV_4; TV_5; and TV_6) and define the largest value TV_4 to the best one (V_4) and the smallest value TV_1 to the worst one (V_1) and calculate the adjusted ratio for the N variables.
(4). Control variables (CV_n) = V_n, V_{n-i}, V_{n-j}. Choose control variables according to chain social-ability relation knowledge V_1, V_5, V_3 (see Table 2).
(5). (W_n × CV_n) provides CV_n with different weightings which are derived from expert knowledge (e.g. importance index of profit). This produces the control variable ratio by multiplying the original variable ratios by these different weights.
(6). V_{nmean} = (TV_n + CV_n)/2 (Calculates the mean ratio of transformation value and control variable value).
(7). Social Sequencing Operation (SSO) optimizes the sequence of the cost variables to produce a refined chromosome structure to preserve positive building blocks and check final Social Classification for the variables with their evaluated ratio values. The final result is (V_3, V_2, V_4, V_6, V_1, and V_5) and the best one as (V_3) and the worst one (V_5) (see Table 2).

This approach employs the fitness ratios to sort the variables sequence within each variable according to descending order. Then SSO calculates the mean of the adjusted ratio and the control variable ratio to obtain a final fitness values for the variables for example V_1, V_5, and V_3. After considering the importance of variables, the order of V_6 and V_3 and the order of V_4 and V_2 can be swapped, and the new sequence of their ratio is (V_3, V_2, V_4, V_6, V_1, and V_5); it shows that (V_3) is the best one and (V_5) is the worst one. Over a number of evolutions, the beneficial variables will be recognized and the positions will be fixed to produce a refined structure for the cost variables to adjust the chromosome structure to preserve positive building blocks. Table 2 illustrates using an examples the steps involved in Variable ($V_{i=1,\ldots,n}$) Social-Relation operation.

Table 2. An example of Social-Relation operation

(1). $V_{TC} = (V_1\ (150) + V_2\ (350) + V_3\ (250) + V_4\ (450) + V_5\ (50) + V_6\ (750) = V_{TC}\ 2000)$.
(2). SCO: Affecting Ratio (V^n_{AR}) = ($V_1 = 0.075$, $V_2 = 0.175$ $V_3 = 0.125$, $V_4 = 0.225$, $V_5 = 0.025$, $V_6 = 0.375$).
(3). It can be expressed as (V_6, V_4, V_2, V_3, V_1, and V_5). In this case, V_6 is the best and V_5 is the worst one.
(4). SAO :(Transformation Value (TV_n) $=W_n \times V_n$ = Weighting × Cost Value). Select a suitable weighting vector [($V_1 = 0.075$, $V_2 = 0.175$ $V_3 = 0.125$, $V_4 = 0.225$, $V_5 = 0.025$, $V_6 = 0.375$) × ($W_1 = 0.025$, $W_2 = 0.3$, $W_3 = 0.2$, $W_4 = 0.25$, $W_5 = 0.15$, $W_6 = 0.075$,)] => ($TV_1 = 0.001875$, $TV_2 = 0.0525$, $TV_3 = 0.025$, $TV_4 = 0.05625$, $TV_5 = 0.00375$, $TV_6 = 0.028125$) and (V_4, V_2, V_6, V_3, V_5, and V_1) define the largest value TV_4 to the best one (V_4) and the smallest value TV_1 to the worst one (V_1).
(5). Control variables (CV_n) = V_1, V_5, V_3.
(6). CV_{CVn} = [($V_1 = 0.075$, $V_2 = 0.175$ $V_3 = 0.125$, $V_4 = 0.225$, $V_5 = 0.025$, $V_6 = 0.375$) × ($W_1 = 0.9$, $W_2 = 0.3$, $W_3 = 0.7$, $W_4 = 0.2$, $W_5 = 0.8$, $W_6 = 0.1$)] => ($CV_1 = 0.0675$, $CV_2 = 0.0525$, $CV_3 = 0.0875$, $CV_4 = 0.045$, $CV_5 = 0.02$, $CV_6 = 0.0375$).
(7). $V_{nmean} = (TV_{ARn} + CV_{CVn})/2$ = [(V_{1mean}=0.069372/2=0.0346, V_{2mean}=0.105/2=0.0525, V_{3mean}=0.125/2=0.075, V_{4mean}=0.1025/2=0.0506, V_{5mean}=0.017/2=0.0085, V_{6mean}=0.065625/2=0.0328125)].
(8). SSO calculates (V_3, V_2, V_4, V_6, V_1, and V_5) and defines the best one as (V_3) and the worst one (V_5).

After each cost variable has been refined, the evidence about fitness ratios for the variable social-ability and sub-variable societies can be collected. These can be used to assign weightings to individual objective functions, which have been ranked during the operations of TGA evolutions. This method presents an alternative approach to the refinement of cost variable structure based upon the building block theory (Schaffer, 1985; Rosca, 1995; Gibbs et al., 2008) to preserve beneficial cost variable.

4. Simulation Experiments

In this section, we will give an overview on the selected simulation test-beds and analyze the simulation results. Various benchmark functions (Fogarty, 1989; Giddens, 1994; Jason and Konstantinos, 2002; DeJong, 2007; Eiben et al., 2007) were adopted to test the TGA and the proposed SGA such as [T_1-T_8] unimodal functions in Table 3, and [T_9-T_{16}] multimodal functions in Table 4. T_1 is a smooth, unimodal, continuous, and strongly convex function. T_2 is a sharp and parabola function. T_3 is a continuous, convex and unimodal function. T_4 is a flat step function. It is also a unimodal function (Quadratic with Noise function), which is more complicated model than the previous functions. T_5 is a three dimensional function: (continuous, convex, low-dimensional quadratic, unimodal function). T_6 is a devilish two-dimensional function: (continuous, non-convex, low-dimensional quadratic, unimodal function). T_7 is a five dimensional function that has flat step surfaces surrounded by discontinuities (discontinuous, non-convex, step, unimodal function). T_8 is a 30 dimensional function that evaluation is modified by Gaussian noise: (continuous, convex, low-dimensional quadratic, unimodal function) (DeJong, 2007).

T_9 is a DeJong Foxholes function that extended shaped basin has been frequently applied in evaluating the presentation/efficiency of optimization algorithms. T_{10} is a comprehensive optimal function (N-hump camel back function) (Eiben et al., 2007). T_{11} (Rastrigin function) is a continuous, convex and unimodal function that is a highly multimodal with the dispersed local minima (Jason and Konstantinos, 2002). T_{12} is a Griewank function with a global minimum value to zero. T_{13} is a famous multimodal function (Ackley's Path function plot in X-Y axis from -30 to 30) (DeJong, 2007). T_{14} is a two dimensional function that has many local minimum: (continuous, non-convex, non-quadratic, two-dimensional function). T_{15} is a Rosenbrock function that is a non-convex function; the global minimum is inside a long, narrow, parabolic

shaped flat valley. T_{16} is a many oscillations and peaks testing function which is difficult for hill-climbing techniques.

Table 3. [T_1-T_8] functions are unimodal functions

$T_1 = \sum_{i=1}^{n} 5 \cdot i \cdot x^2$
$T_2 = a(x_2 - bx_1^2 + cx_1 - d)^2 + e(1-f)\cdot\cos(x_1) + e$
$T_3 = \sum_{i=1}^{5} \text{int}.(x_i)$
$T_4 = \sum_{i=1}^{n} \lvert x_i \rvert^{(i+1)}$
$T_5 = \sum_{i=1}^{3} X_i^2$
$T_6 = \sum_{i=1}^{N-1} 100(X_1^2 - X_2)^2 + (1 - X_1)^2$
$T_7 = 6 \cdot n + \sum_{i=1}^{n} \lvert x_i \rvert$
$T_8 = \sum_{i=1}^{n} iX_i^4 + Gauss(0,1)$

Table 4. [T_9-T_{16}] functions are multimodal functions

$T_9 = \sin(x_1 + x_2) + (x_1 - x_2)^2 - 1.5x_1 + 2.5x_2 + 1$
$T_{10} = 10V + \sum_{i=1}^{10} (-x_i \cdot \sin(\sqrt{\lvert x_i \rvert}))$
$T_{11} = 10 \cdot n + \sum_{i=1}^{n} (x^2 - 10 \cdot \cos(2\pi x))$
$T_{12} = 1 + \sum_{i=1}^{10} (\frac{x_i^2}{4000}) - \prod_{i=1}^{10} (\cos\left(\frac{x_i}{\sqrt{i}}\right))$
$T_{13} = 20 + e - 20e^{-0.2\sqrt{\frac{\sum_1^n x^2}{n}}} - e^{\frac{\sum_1^n 2\pi x}{n}}$
$T_{14} = 0.002 + \sum_{j=1}^{25} 1/[j + \sum_{i=1}^{2} (x_i - a_{ji})^6]$
$T_{15} = \sum_{i=1}^{d-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$
$T_{16} = 0.5 + \frac{\sin\sqrt{(x^2 + y^2)} - 0.5}{[1.0 + 0.001(x^2 + y^2)]^2}$

We have chosen parameters settings for our experiments in accordance with previous studies (Fogarty, 1989; Giddens, 1994; Jason and Konstantinos, 2002; DeJong, 2007; Eiben et al., 2007) and we were able to produce better simulation results for the TGA. Parameters in our experiments were set as follows (1): Population Sizes: 30, (2): Crossover rates: 0.85, (3): Mutation rate: 0.05, and (4): Generations: 150. The TGA and SGA were tested on three different groups of test-suite functions and the results are average outputs over 50 trials. (see Tables 5 and 6)

Table 5. [T_1-T_9] test-beds are unimodal functions

	TGA	SGA	
	Minimum		IR%
T_1	0.0532500	0.0113520	78.68
T_2	4.7979530	0.7010180	85.38
T_3	35.0000000	32.9842900	5.75
T_4	12.2162300	11.4282500	6.45
T_5	0.0010000	0.0001000	90.00
T_6	0.0020000	0.0000100	99.50
T_7	-4.0000000	-4.0000000	00.00
T_8	-3.5200000	-3.7700000	7.10
	Mean		IR%
T_1	1.5688650	0.0814150	94.81
T_2	5.7122530	3.7382520	34.55
T_3	36.2304900	35.1077500	3.10
T_4	16.9344300	14.0009700	17.32
T_5	0.0050000	0.0020000	60.00
T_6	0.0016000	0.0008000	50.00
T_7	-3.1300000	-3.4600000	10.54
T_8	-2.0500000	-3.2400000	58.05
	Maximum		IR%
T_1	26.94588	1.76492	93.45
T_2	9.679126	13.90071	-43.61
T_3	42.28102	37.5	11.30
T_4	40.26505	26.91952	33.14
T_5	0.02	0.0026	87.00
T_6	0.045	0.003	93.33
T_7	-2.3	-3.1	34.78
T_8	-0.891	-2.4	169.36

The simulation results show that SGA performs better than TGA on (T_1 and T_5 and T_6 and T_8) unimodal functions. SGA performs similar results to TGA on (T_2 and T_3 and T_4 and T_7) functions.

The simulation results show that SGA achieves better performance on the high-dimensional functions than TGA (T_9 and T_{10} and T_{12} and T_{13}) functions. SGA performs similar results to TGA on (T_{11} and T_{14}) functions. TGA performs better than SGA on (T_{15} and T_{16}) functions. A powerful search model for exploration normally has ability to cope with multimodal functions. The results indicate that SGA is a strong exploration model and is particularly good at coping with unimodal functions and multimodal functions search environments. From our simulation results we can conclude that SGA outperforms the traditional TGA in this case.

Table 6. [T_9-T_{16}] test-beds are multimodal functions

	TGA	SGA	
	Minimum		IR%
T_9	0.8065100	0.1217130	84.90
T_{10}	0.0000037	0.0000003	91.55
T_{11}	9.3071540	6.6021360	29.06
T_{12}	0.0003220	0.0000092	97.15
T_{13}	0.0266140	0.0117890	55.70
T_{14}	0.0010000	0.0010000	00.00
T_{15}	0.278013	0.736386	-164.87
T_{16}	0.064051	0.102043	-59.31
	Mean		IR%
T_9	1.3769850	0.3270320	76.25
T_{10}	0.0000044	0.0000004	91.11
T_{11}	11.0838000	11.3969600	-2.82
T_{12}	0.0235310	0.0000319	99.86
T_{13}	1.6230610	0.0144760	99.10
T_{14}	0.0020000	0.0015000	25.00
T_{15}	0.622548	1.596364	-156.42
T_{16}	0.113383	0.284378	-150.81
	Maximum		IR%
T_9	3.192121	0.655709	79.45
T_{10}	0.00008553	0.00000772	90.97
T_{11}	23.79152	33.43348	-40.52
T_{12}	0.631663	0.0004838	99.92
T_{13}	14.42809	0.288858	97.99
T_{14}	0.003	0.002	33.33
T_{15}	1.990089	5.021993	-152.35
T_{16}	0.25251	0.449802	-78.13

5. Conclusions

The contribution of our proposed SGA is to design two novel controllers, which enable better selections of chromosome structure and TGA's search parameters for searching optimization. The first controller is a social relation controller, which includes the transformation functions to identify the gene's social-relation abilities of chromosome structure. It enables to select better gene structure to preserve positive building blocks for chromosome structure optimization. The second controller is an adaptive social-parameter controller, which enables to select the better parameters for TGA's search. We verify the performance of our novel SGA functions comparing with the TGA in simulation test-beds successfully.

The SGA has been implemented as a prototype system and tested on simulation test-beds with different search landscapes formed by multimodal and unimodal functions. The results show that SGA performs better than TGA in non-linear and high-dimensional spaces. This is due to ability of our proposed methodology that systematically and effectively varies crossover and mutation rates as well as refines chromosome structures in TGA to achieve the better optimum. In future study, further experiments will be carried out by applying the proposed approach to wider applications such as Service-Oriented Computing (Immonen and Pakkala, 2014; Huber et al., 2014; Grolinger et al., 2014).

References

1. E. Alfaro-Cid, E. W. McGookin, and D. J. Murray-Smith, A comparative study of genetic operators for controller parameter optimization, *Control Eng. Pract.* **17** (2009) 185-197.
2. J. H. Cheng, S. W. Chen, and F. Y. Chen, Exploring how inter-organizational relational benefits affect information sharing in supply chains, *Inf. Tech. Manage.* **14** (2013) 283-294.
3. K. DeJong, Parameter setting in EAs: a 30 year perspective, *Parameter Setting in Evol. Algorithm.* **54** (2007) 1-18.
4. K. Deep, and P. K. Singh, Design of robust cellular manufacturing system for dynamic part population considering multiple processing routes using genetic algorithm, *J. Comput. Chem.* **35** (2015) 155-163.
5. A. E. Eiben, Z. Michalewicz, M. Schoenauer, and J.E. Smith, Parameter control in evolutionary algorithms, *Parameter Setting in Evol. Algorithm.* **54** (2007) 19-46.
6. L. J. Eshelman, R. A. Caruana, and J. D. Schaer, *Biases in the crossover landscape*. ICGA, George Mason University, 1989.
7. H. Fazlollahtabar, R. Hassanzadeh, I. Mahdavi, and N. Mahdavi-Amiri, A genetic optimization algorithm and perceptron learning rules for a bi-criteria parallel machine scheduling, *J. Chi. Inst. Ind. Eng.* **29** (2012) 206-218.
8. T. C. Fogarty, Varying the probability of mutation in the genetic algorithm, *Proc. 3rd Int. Conf. Gene. Algorithm.* 1989, 104-109.
9. M. S. Gibbs, G. C. Dandy and H. R. Maier, A genetic algorithm calibration method based on convergence due to genetic drift, *Inf. Sci.* **178** (2008) 2857-2869.
10. T. D. Giddens, *The determination of parameters and operators of the traditional genetic algorithm using an adaptive genetic algorithm generator*. PhD Thesis, Texas Tech University. 1994.
11. K. Grolinger, MAM. Capretz, A. Cunha, and S. Tazi, Integration of business process modeling and Web services: a survey, *Serv. Oriented Comput. App.* **8** (2014) 105-128.
12. X. Han, Y. Liang, Z. Li, G. Li, X. Wu, B. Wang, and G. Zhao, An Efficient Genetic Algorithm for Optimization Problems with Time-Consuming Fitness Evaluation. *Int. J. Comp. Meth-Sing.* **12** (2015). DOI: 10.1142 /S0219876213501065.
13. J. H. Holland, *Adaptation in Natural and Artificial Systems*, The MIT Press. 1975.

14. N. Huber, A. Hoorn, A. Koziolek, F. Brosig, and S. Kounev, Modeling run-time adaptation at the system architecture level in dynamic service-oriented environments, *Serv. Oriented Comput. App.* **8** (2014) 73-89.
15. A. Immonen and D. Pakkala, A survey of methods and approaches for reliable dynamic service compositions, *Serv. Oriented Comput. App.* **8** (2014) 129-158.
16. G. D. Jason and G. M. Konstantinos, An experimental study of benchmarking functions for genetic algorithms. *Int. J. Comput. Math.* **79** (2002) 403-416.
17. D. H. Kim, and J. H. Cho, Advanced intelligence tuning using hybrid of clonal selection and genetic algorithm, GM and PM. *Int. J. Comp. Intel. Appl.* **14** (2015) doi: 10.1142/S1469026815500042.
18. T. Long, O. M. McDougal, and T. Andersen, GAMPMS: Genetic algorithm managed peptide mutant screening. *J. Comput. Chem.* **36** (2015) 1304-1310.
19. N. H. Moin, O. C. Sin, and M. Omar, Hybrid Genetic Algorithm with Multiparents Crossover for Job Shop Scheduling Problems. *Math. Problems Eng.* (2015) doi: 10.1155/ 2015/ 210680.
20. B. R. Moon, *Hybrid genetic algorithms with hyperplane synthesis: A theoretical and empirical study*, PhD Thesis, The Pennsylvania State University, 1994.
21. KSN. Ripon, S. Kwong, and K. F. Man, A real-coding jumping gene genetic algorithm (RJJGA) for multiobjective optimization, *Inf. Sci.* **177** (2007) 632-654.
22. J. P. Rosca, Towards automatic discovery of building blocks in genetic programming. *In Working Notes for the AAAI Symp. Gene. Progr.* 1995, 78-85.
23. J. D. Schaffer, Learning Multiclass Pattern Discrimination, *Proc. 1st Int. Conf. Gene. Algorithm. Lawrence Erlbaum Associates*, 1985, 74-79.
24. J. D. Schaffer and A. Morishima, An adaptive crossover distribution mechanism for genetic algorithms. *Proc. 2ed Int. Conf. Gene. Algorithm.* 1987, 36-40.
25. M. Srinivas, and L. M. Patnaik, Adaptive probabilities of crossover and mutation in genetic algorithms. *IEEE T. Syst. Man CY. B.* **24** (1994) 656-667.
26. D. Thierens, Adaptive strategies for operator allocation. *Parameter Setting in Evol. Algorithm.* **54** (2007) 77-90.
27. C. F. Tsai, S. L. Lu, J. H. Chen, K. M. Chao, N. Shah, An adaptable optimizer for green component design, *Inf. Syst. E-Bus. Manage.* **13** (2015) 193-210.
28. Y. W. Wang, An artificial chromosome embedded genetic algorithms for smart grid power demand forecast. *Journal of Industrial and Intelligent Information* **3** (2015) 69-74.