Journal of Universal Computer Science, vol. 17, no. 14 (2011), 2048-2063
submitted: 6/12/10, accepted: 28/9/11, appeared: 1/10/11

An Adaptive Genetic Algorithm and Application in a Luggage Design Center

Chen-Fang Tsai
(Department of Industrial Management & Enterprise Information, Aletheia University
New Taipei City, Taiwan R.O.C.
au1204@mail.au.edu.tw)

Weidong Li[1]
(Faculty of Engineering and Computing, Coventry University, Coventry, United Kingdom
weidong.li@coventry.ac.uk)

Anne James
(Faculty of Engineering and Computing, Coventry University, Coventry, United Kingdom
csx118@coventry.ac.uk)

Abstract: This paper presents a new methodology for improving the efficiency and generality of Genetic Algorithms (GA). The methodology provides the novel function of adaptive parameter adjustment during each evolution generation of GA. The important characteristics of the methodology are mainly from the following two aspects: (1) superior performance members in GA are preserved and inferior performance members are deteriorated to enhance search efficiency towards optimal solutions; (2) adaptive crossover and mutation management is applied in GA based on the transformation functions to explore wider spaces so as to improve search effectiveness and algorithm robustness. The research was successfully applied for a luggage design chain to generate optimal solutions (minimized lifecycle cost). Experiments were conducted to compare the work with the prior art to demonstrate the characteristics and advantages of the research.

Keywords: genetic algorithm, optimization, search
Categories: F.2.0, G.1.6, I.2.8

1 Introduction

A main challenge in evolutionary algorithms is parameter setting. For instance, an inadequate parameter setting can critically worsen the performance of Genetic Algorithms (GA) such as search efficiency [Chen and Liang 11] [Nedjah et al. 08]. However, effective rules and methodologies to choose suitable parameters have not been developed [Angelova and Pencheva 11] [Schwefel et al. 89] [Felix et al. 08]. Recently, studies on parameter setting have become active to pursue better stability between exploration (wider search) and exploitation (search refining) in a search space. However, this equilibrium is not easy to reach. An important reason is that algorithms are usually related to specific applications and problems so that it is not always appropriate to use pre-set parameters [Otman and Jaafar 11] [DeJong 07].

---

[1] Corresponding Author.

Dynamic setting in evolutionary algorithms is a relatively new approach to identify better solutions. The motivation of the dynamic setting is to achieve the trade-off between exploration and exploitation during the evolution processes of the algorithms. For instance, research of adapting operator probabilities based on the performance of the operators in GA was conducted [Kapoor et al. 10] [Davis 85] [Tahera et al. 08]. A conversion mechanism was used to adjust operator probabilities in proportion to the fitness values of chromosomes managed by the operators. The results showed a substantial improvement in terms of the performance of GA. However, there are still a number of improvement spaces such as algorithm robustness for various problems. In this research, it is aimed at developing an improved and adaptive mechanism to dynamically set parameters. In the algorithm, superior performance members in GA are preserved and inferior performance members are deteriorated during search to enhance the algorithm efficiency towards optimal solutions. The algorithm also adjusts the crossover and mutation rates in each evaluation generation in GA, and new rates are derived from the fitness values during evolution in order to prevent premature convergence. Based on the above measures of dynamic parameter setting, this new algorithm is adaptive to various problems and reinforces the population diversity and effectiveness towards optimal solutions [Kapoor et al. 11] [Lu and Xiuxia 11].

The improved GA was applied to a lifecycle cost optimization problem in a luggage design center to improve the cost-effectiveness of Small Series Production [Chao et al. 02]. Experiments showed that the improved algorithm achieved better performance than other previous research works in terms of computational efficiency, optimal solutions and robustness.

The rest of the paper is organized as follows. Section 2 is literature review. Section 3 presents the improved GA mechanism. Section 4 describes the experimental results of the algorithm. Section 5 presents the industrial application and results. Section 6 concludes the research.

2 Literature Review

GA is an important evolutionary technique to search optimal or near-optimal solutions according to the fitness (i.e., optimization objective function) computing, crossover and mutation operations on chromosomes (i.e., problem solutions) in a population during every evolution (i.e., an evolution process stands for a generation) [Dongmei et al. 11] [Goldberg 89]. A crossover rate is to manage the efficiency to exchange the members of the population and the mutation rate is to control the efficiency of random change genes. The rational of the algorithm is that the operators combine the better parts of chromosomes during evolution and randomly change parts of these in order to create solutions with a refined fitness value. More appropriate settings of crossover rate and mutation rate can improve the performance of GA significantly. On the other hand, the population size of GA is critical. A small population size will lead to premature convergence and a large population size will deteriorate the efficiency of the algorithm from finding a better alternative in a reasonable amount of time [Angelova et al. 11].

Guidelines were developed for setting the parameters but they are pre-set and static values [García-Martínez and Lozano 10] [Smith 96]. On the other hand, researchers attempted alternative approaches of dynamically varying parameter settings during the execution of GA [Choubey 10]. Improved performance of algorithms was achieved attributing to the adaptive capabilities for various applications such that chromesomes and generations were adjusted with different fitness values during evolution. In [Schaffer and Morishima 87], a punctuated crossover operator with cost penalty was applied for specific problems. This approach can identify where better crossover points are located because of the dynamic parameter setting. In [Smith 96], a varied mutation rate based on a cooling concept was devised and applied for specific problems. In [Whitley 94], the Hamming distance between parent solutions was used to improve GA performance but this work was most applicable to steady state GA with elitism. It is, therefore, limited in problem domains. In [Srinivas and Patnaik 94], the crossover and mutation rates for each individual were determined in respect to a function of its fitness. In [Tahera et al. 08], research was developed to determine the mutation rate specifically for each solution and maintained the exploration without affecting the exploitation properties. Previous techniques, however, are still relatively simple to solve more complex problems requiring more finely controlled measures.

In design chain applications, evolutional algorithms have led to innovations [Zhao and Wang 11]. Design chains need to be considered by modularization, optimization [Qu et al. 11], and substitution of design specification [Chiu 01] [Chao et al. 02] [Liu et al. 05]. The applications of evolutional algorithms reduce the operating cost of design chains to the optimal level [Anane et al. 02] [Caldentey and Wein 03] [Erel et al. 04] [Selwyn 05]. Research is imperative to apply improved GA with dynamically adjusting parameters to solve more complex design chain problems [Hongfeng and Guanzheng 09].

3 An Adaptive Genetic Algorithm

This research proposed a novel GA to improve the performance and adaptability by introducing an adjustable parameter controller to guide the search behaviours in order to reach a better solution. The concept has been derived from the control chart theory. The theories of statistical quality control have been utilized as a control mechanism for process control in various industrial applications [Zhao 07]. Usually a control chart is divided into N divisions, which can be used to identify the causes of variations in a production process. The corrective action is then based on the cause identified by the control chart. This implies that different members in the GA evolution should be handled in different ways. So far there is no a generic widely accepted solution to solve various problems in a robust means [Alfaro-Cid et al. 09]. Therefore, it is important to develop a generic approach of dynamic parameter setting in GA to be applied to different problems.

In this research, adaptive parameter settings in GA adjust the crossover rate and mutation rate in each evolution generation. The new rates are derived from the fitness value by considering the population diversity and the level of the parents' fitness. During the evolution process it is crucial to examine whether GA is converging to an optimum. This research detects convergence by observing the range between the maximum and minimum fitness values of the population. If the population starts to converge to a local optimum, the crossover and mutation rates will be adjusted for further exploration.

In the algorithm, at the first stage, the range value of the population is divided into N search regions, and in which each region is assigned crossover and mutation rates in each generation. The rates are calculated using Procedures 1 and 2 (Tables 1 and 2). At the second stage, the members of parent chromosomes and offspring chromosomes with the best quality level are allocated to the first region, and the algorithm will keep the crossover and mutation rate of this region to protect these best fitness values. Meanwhile, a higher crossover and mutation rate will be arranged to other members of parent chromesomes and offspring chromesomes. The highest rates will be asigned to the Nth search region, which is used to store the chromesomes with the worst fitness values. At the final stage, the algorithm will combine the rates based on a transformation function. The higher search regions will be provided with lower crossover and mutation rates to conserve the superior genes, and the lower search regions will be assigned with higher crossover and mutation rates to support exploration. In more details, the optimization of the parameters adapted is defined as follows:

1. At the first stage, the range value of the population is divided into N's search regions; ($R = (Max_{value} – Min_{value}) = Z_i \times$ standard deviation (Z_i= division 1 to N)); which are each assigned a range of the crossover and mutation rates for each generation. This range is calculated by using Tables 1 and 2.
2. At the second stage, the members of parents and offspring are allocated to the first region which is called the best quality level; the controller will select the crossover and mutation rate of the first search region of to protect these best fitness values. The highest crossover and mutation rate will be provided to the worst members of parents and offspring which are ranked in the Nth search region.
3. At the final stage, the system will combine the suitable crossover and mutation rates based on a transformation function which varies the N's member positions in the search population. The higher search regions will keep the lower crossover and mutation rates for exploitation which conserve the superior genes. The lower search regions will be assigned the higher crossover and mutation rates to support exploration. This approach selects the crossover and mutation rates based on the different types of the problems before the search processing of each generation. The notation used in Procedures 1&2 are listed below:

1. CR_{1-N} = the crossover rate of the regions (1 to N)
2. MR_{1-N} = the mutation rate of the regions (1 to N)
3. D_{1-N} = the individual fitness values of the regions (1 to N)
4. R_C = the crossover rate calculated according to the position of the fitness value
5. R_M = the mutation rate is calculated according to the position of the fitness value in a region
6. C_{P1-N} = the situation factors that are selected to tune crossover rates
7. M_{P1-N} = the situation factors that are selected to tune mutation rates
8. F_{Max} = the maximum value in a population
9. F_L = the chromosome with the highest fitness value before crossover or mutation operation

10. F_S = the chromosome with the lowest fitness value before crossover or mutation operation
11. F_{mean} = the mean fitness value in a population
12. F_σ = the standard deviation in a population - ($\sigma = \sqrt{(Xi - \overline{X})^2 / i}$)

An important attempt in this research is the different rate setting for different regions. The crossover and mutation rates are calculated by the position of the individual fitness value assigned to these regions (as Procedures 1 and 2). The chromosomes with the highest and lowest fitness values in a region will be selected for the transformation function to determine the value of the crossover rate. The situation factors (C_{PN}) are parameters, which are selected to tune the mutation rates for specific problems. This will prevent the premature converge of GA.

Upper Bound For Adaptive Parameter Setting of the Crossover Rate
If $F_{\text{fitness in crossover}} >= F_{mean}$,
Then $C_{PN} = [(F_\sigma) / (F_{mean})]$;
$R_C = (F_{Max} – F_L) / (R)$ Then $CR_N = C_{PN} \times R_C$
Lower Bound For Adaptive Parameter Setting of the Crossover Rate
If $F_{\text{fitness in crossover}} =< F_{mean}$,
Then $C_{PN} = [(F_\sigma) / (F_{mean})]$;
$R_C = (F_S – F_{Min}) / (R)$ Then $CR_N = C_{PN} \times R_C$

Table 1: Adaptive parameter setting of the crossover rate (Procedure 1)

Upper Bound For Adaptive Parameter Setting of the Mutation Rate
If $F_{\text{fitness in mutation}} >= F_{mean}$,
Then $M_{PN} = [(F_\sigma) / (F_{mean})]$;
$R_M = (F_{Max} – F_L) / (R)$ Then $MR_N = M_{PN} \times R_M$
Lower Bound For Adaptive Parameter Setting of the Mutation Rate
If $F_{\text{fitness in mutation}} =< F_{mean}$,
Then $M_{PN} = [(F_\sigma) / (F_{mean})]$;
$R_M = (F_S – F_{Min}) / (R)$ Then $MR_N = M_{PN} \times R_M$

Table 2: Adaptive parameter setting of the mutation rate (Procedure 2)

An example with the following data is shown in Table 3:

1. CR_{1-6} = the crossover rate of the regions 1 to 6 (N=6)
2. MR_{1-6} = the mutation rate of the regions 1 to 6 (N=6)
3. D_{1-6} = the individual fitness value of the regions 1 to 6 (N=6)
4. C_{P1-6} = the situation factors that are selected to tune the crossover rates (0.95)
5. M_{P1-6} = the situation factors that are selected to tune the mutation rates (0.095)
6. F_{Max} = the maximum fitness value in a population (120)
7. F_{Min} = the maximum fitness value in a population (20)

8. F_L = the chromosome with the highest fitness value before crossover or mutation operation (80)
9. F_S = the chromosome with the lowest fitness value before crossover or mutation operation (30)
10. F_{mean} = the mean fitness value in a population (50)

Upper Bound For Adaptive Parameter Setting of the Crossover Rate	
If ($F_{larger\ in\ crossover}$ =80) > = (F_{mean} =50)	
Then $R_{crossover}$ = [((F_{Max} =120)- ($F_{larger\ in\ crossover}$=80))/ (($F_{Max}$ = 120) - (F_{Min} =20))] = [(120-80)/ (120-20))] = 40/100 = 0.4 CR_3 = ((C_{P1} = 0.95)×($R_{crossover}$ = 0.4)) = 0.38	
If (A > 0, A <= 1/6),	Then CR_a = ((C_{P1} = 0.95)×($R_{crossover}$ = 1/6))
If (B > 1/6, B <= 2/6),	Then CR_b = ((C_{P1} = 0.95)×($R_{crossover}$ = 2/6))
If (C > 2/6, C <= 3/6),	Then CR_c = ((C_{P1} = 0.95)×($R_{crossover}$ = 3/6))
If (D > 3/6, D <= 4/6),	Then CR_d = ((C_{P1} = 0.95)×($R_{crossover}$ = 4/6))
If (E > 4/6, E <= 5/6),	Then CR_e = ((C_{P1} = 0.95)×($R_{crossover}$ = 5/6))
If (F > 5/6),	Then CR_f = ((C_{P1} = 0.95)×($R_{crossover}$ = 6/6))
Upper Bound For Adaptive Parameter Setting of the Mutation Rate	
If ($F_{larger\ in\ Mutation}$ =80) > = (F_{mean} =50)	
Then $R_{Mutation}$ = [((F_{Max} =120)- ($F_{larger\ in\ Mutation}$=80))/ ((F_{Max} = 120) - (F_{Min} =20))] = [(120-80)/ (120-20))] = 40/100 = 0.4 CR_3 = ((C_{P1} = 0.095)×($R_{crossover}$ = 0.4)) = 0.038	
If (A > 0, A <= 1/6),	Then CR_a = ((C_{P1} = 0.095)×($R_{Mutation}$ = 1/6))
If (B > 1/6, B <= 2/6),	Then CR_b = ((C_{P1} = 0.095)×($R_{Mutation}$ = 2/6))
If (C > 2/6, C <= 3/6),	Then CR_c = ((C_{P1} = 0.095)×($R_{Mutation}$ = 3/6))
If (D > 3/6, D <= 4/6),	Then CR_d = ((C_{P1} = 0.095)×($R_{Mutation}$ = 4/6))
If (E > 4/6, E <= 5/6),	Then CR_e = ((C_{P1} = 0.095)×($R_{Mutation}$ = 5/6))
If (F > 5/6),	Then CR_f = ((C_{P1} = 0.095)×($R_{Mutation}$ = 6/6))

Table 3: Adaptive parameter setting of the crossover and mutation rate (an example)

The transformation function is followed by the ranking position of the parent and offspring chromosomes. The operating policies are that good quality solutions within a population should be conserved and poor quality solutions should be improved. The actual levels of preservation and improvement depend on the complexity of specific problems. There is no standard solution and most results are still problem dependent. This is the reason why this research proposes the adaptive parameter setting for GA evolutions. Hence, this can be considered to be a novel approach applicable to various problems.

4 Experiments

Experimental cases from [Jason and konstantinos 02] [GEATbx 06] were selected for benchmarking study (see Table 4).T_1 is Ackley's Path that is a famous multimodal test function (Table 4: Illustrated on Ackley's Path function plot in X-Y axis from -30 to 30). T_2 is AxisParallelHyperellip that is eminent as a weighted bubble model. It is a convex, continious, and unimodal model (Table 4: Illustrated on Moved Axis Parallel Hyper-ellipsoid function; design of the function in X-Y axis from -5 to 5). T_3 is the Rastrigin's function which is a vastly multimodal. The site of the local minima is recurrently dispersed. It is also known as a ball model. It is continious, convex and unimodal (Table 4: Visualization of Rastrigin's function in X-Y axis from -5 to 5). T_4 is MovedAxisParallelHyperellipsoid that is a special function designed from the axis parallel hyper-ellipsoid. This function is a more complicated model than the previous function (Table 4: Illustrated on Moved Axis Parallel Hyper-ellipsoid function; draw of the different variables, the goal function values were designed from the multi-dimensional function with others variable set to 0). T_5 is Rosenbrock's valley that is a typical optimal problem, also known as Banana function. The global optimal value is an elongated, and extended shaped basin. It is tricky and has been frequently applied in evaluating the presentation of optimization algorithms (Table 4: Illustrated on Rosenbrock's function that is full designation range of the function). T_6 is a Six-hump camel back function with a comprehensive optimal function. The surrounded areas include six local minima, and two of them are global minima (Table 4: Illustrated on Six-hump camel back function schemed of the region surrounding the minima [GEATbx 06]).

T_1 = AckleyPath = $f(x) = 20.0 + e^1 - 20.0 * e^{-0.2\sqrt{\frac{\sum_{i=1}^{n} x^2}{n}}} - e^{\frac{\sum_{i=1}^{n} 2*\pi*x}{n}}$ $-32.768 \leq x \leq 32.768$
T2 = AxisParallelHyperellip = $f(x) = \sum_{i=1}^{n} i * x^2$ $-5.12 \leq x \leq 5.12$
T3 = Rastrigin = $f(x) = 10 * n + \sum_{i=1}^{n}(x^2 - 10 * \cos(2 * \pi * x))$ $-5.12 \leq x \leq 5.12$
T4 = MovedAxisParallelHyperellipsoid = $f(x) = \sum_{i=1}^{n} 5 * i * x^2$ $-5.12 \leq x \leq 5.12$
T5 = Rosenbrock = $f(x_1, x_2) = 100 * (x_1^2 - x_2)^2 + (1 - x_1)^2$ $-2.048 \leq x \leq 2.048$
T6 = Six- humpCamelBack = $f(x_1, x_2) = (4 - 2.1x_1^2 + x_1^{\frac{4}{3}}) * x_1^2 + x_1x_2 + (-4 + 4x_2^2)) * x_2^2$ $-3 \leq x_1 \leq 3, -2 \leq x_2 \leq 2$

Table 4: The benchmark functions of [T_1-T_6]

These experimental benchmarking functions were used to make a comparison between simple adaptive and dynamic parameter settings with different GA operators and parameters. The operator settings chosen are the combinations of the best experimental results from previous studies [Alfaro-Cid et al. 09] [Jason and konstantinos 02]. The selection operator is of either uniform or tournament. The mutation operator can have 1 or 8 cross points. The mutation operator has a uniform distribution probability for its rate. The replacement operator can be set in either uniform, simulated annealing or elitism. A range of ‘rule-of-thumb’ configurations were tested with population sizes from 25 to 150 with 25 generations. Selection is based on the roulette selection. Crossover rates were set from 0.6 to 0.9 with an incremental step 0.05. Mutation rates (Gaussian) were set from 0.05 to 0.45 with an incremental step 0.05. The numbers of generations were set from 30 to 300. All results presented were averaged over 150 trials. In published research, GA's operators and parameters presented the experimental configuration for the simple adaptive setting. This research was tested on six kinds of different functions and test-suites. The experimental results were used to compare the simple adaptive and dynamic adaptive setting in terms of the converge generation, the best value, the mean value and the standard deviation in the population evolutions. Figures 1-4 show that this research achieved better results than those of the previously developed simple adaptive methods.

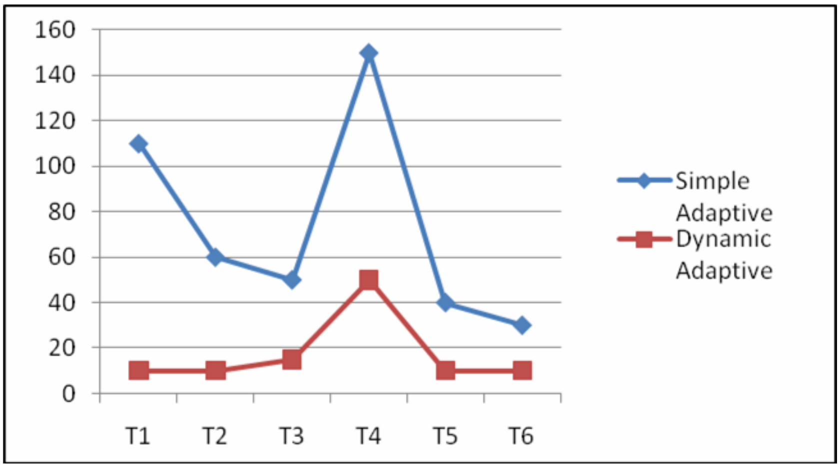

Figure 1: The comparison of the converge generations based on the benchmarking functions

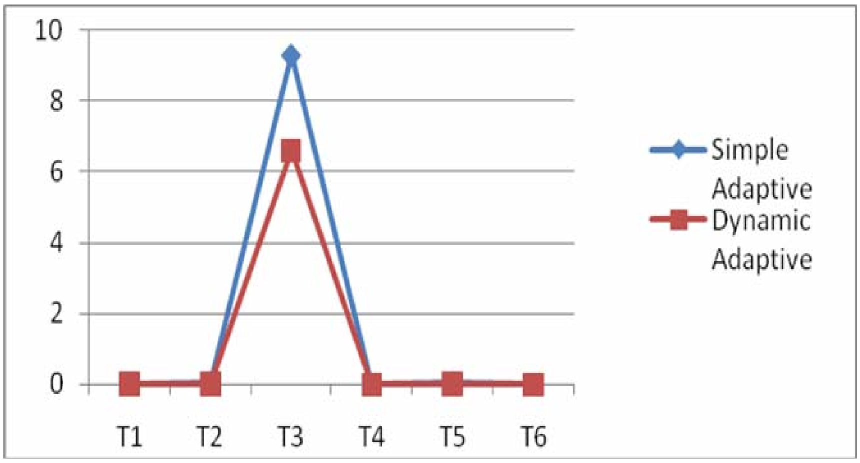

Figure 2: The comparison of the best values based on the benchmarking functions

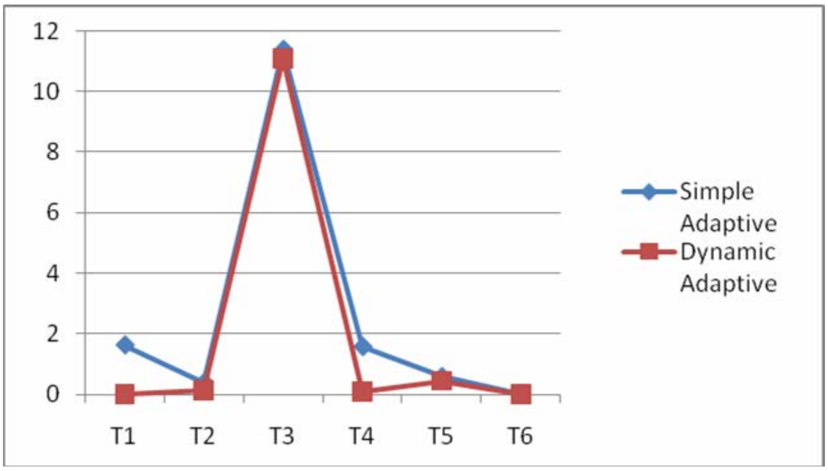

Figure 3: The comparison of the mean values based on the benchmarking functions

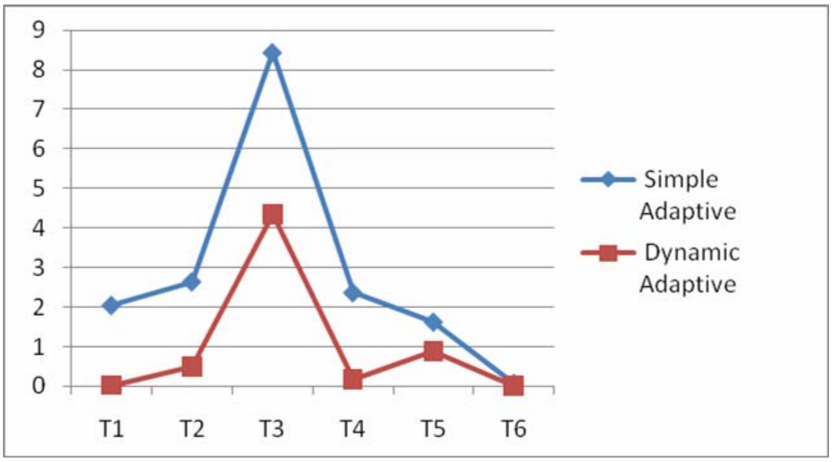

Figure 4: The comparison of the standard deviations based on the benchmarking functions

5 An Industrial Application

Companies DCI (China and US), AGILE (Taiwan and China) and THERBLIG (Taiwan) form a global chain. The companies have more than 30-year history in designing and manufacturing luggage and this supply chain is one of the leading providers in the world. In today's competitive environment, the companies need to be able to respond to the market with the lowest cost. The improved GA was applied to minimize the total cost of Product Lifecycle Cost (PLC) in this chain.

PLC = Investment Cost + Setup Cost + Inventory Cost + Manufacturing Cost+ Shipping Cost, that is

$$
\begin{aligned}
\mathrm{PLC} = & \sum_{s=1}^{S}\sum_{f=1}^{F}\sum_{p=1}^{P}\sum_{o=1}^{O}\left(\mathrm{IV}_{sfpo}\right)\left(\frac{1}{\mathrm{AR}_{sfpo}}\right)\left(\mathrm{Q}_{P}\right) \\
& +\sum_{s=1}^{S}\sum_{f=1}^{F}\sum_{p=1}^{P}\sum_{o=1}^{O}\left(\mathrm{SC}_{sfpo}\right)\left(\frac{1}{\mathrm{AR}_{sfpo}}\right)\left(\mathrm{Q}_{P}\right) \\
& +\sum_{s=1}^{S}\sum_{f=1}^{F}\sum_{p=1}^{P}\sum_{o=1}^{O}\left(\mathrm{IC}_{sfpo}\right)\left(\frac{1}{\mathrm{AR}_{sfpo}}\right)\left[\bar{d}\times\overline{\mathrm{LT}}+Z\times\sqrt{\overline{\mathrm{LT}}\times\sigma_{d}^{2}+d^{-2}\times d_{\mathrm{LT}}^{2}}\right] \\
& +\sum_{s=1}^{S}\sum_{f=1}^{F}\sum_{p=1}^{P}\sum_{o=1}^{O}\left(\mathrm{MC}_{sfpo}\times\frac{1}{\mathrm{AR}_{sfpo}}\times\mathrm{MQ}_{sfpo}\right) \\
& +\sum_{s=1}^{S}\sum_{f=1}^{F}\sum_{p=1}^{P}\sum_{o=1}^{O}\left(\mathrm{ShC}_{sfpo}\times\frac{1}{\mathrm{AR}_{sfpo}}\times\mathrm{ShQ}_{sfpo}\right)
\end{aligned}
$$

(Equation 1)

where, S = Number of suppliers; F = Number of factories; P = Number of products; Q =Quantity of design part; IV = Unit Investment Cost of design part ; SC = Unit Setup Cost of design part; IC = Unit Inventory cost of design part ; MC = Unit Manufacturing Cost of design part; S_hC = Unit Shipping Cost of design part; AR = Availability Ratio of design part; z = service level of the demand of design part during lead-time of Part i ; σ_d = the standard deviation of the demand of design part;

σ_{LT} = the standard deviation of lead-time of design part.

The process of the application of the improved GA for this optimization problem is below and shown in Table 5:

(1) Determine initial populations through GA selection.
(2) Define chromosomes and their fitness functions for this application
(3) Execute the GA to find the optimal cost

(4)Conduct the comparisons of the simple adaptive and dynamic adaptive function based on practical experimental results

Initial Populations (Chromosomes)

Fitness Function Computing: (Chromosome)

PLC = Investment Cost + Setup Cost + Inventory Cost + Manufacturing Cost + Shipping Cost.

PLC = (IV) + (SC) + (IC) + (MC) + (S_hC)

$$\text{Example: PLC} = (\text{IV}); \sum_{s=1}^{S}\sum_{f=1}^{F}\sum_{p=1}^{P}\sum_{o=1}^{O}(IV_{sfpo})\left(\frac{1}{AR_{sfpo}}\right)(Q_P)$$

The Chromosome is (Quantity Amount (Q_p)):((IV(Q_p); SC(Q_p); IC(Q_p); MC (Q_p); S_hC(Q_p)): (136,82,43,62,372)

The Chromosome is: 10110110010001000011110110

The optimal settings of GA parameters

The simple adaptive function (Based on previous research work)	**Dynamic Adaptive Function (Based on this research work)**
Crossover rates ranging from 0.6 to 0.9 with an incremental step 0.05. Mutation rates (Gaussian) ranging from 0.05 to 0.45 with an incremental step 0.05.	Adaptive Parameter Setting of the Crossover Rate $R_C = (F_{Max} – F_L) / (R)$ Then $CR_N = C_{PN} \times R_C$ Adaptive Parameter Setting of the Mutation Rate $R_M = (F_{Max} – F_L) / (R)$ Then $MR_N = M_{PN} \times R_M$

The comparisons on the simple and dynamic adaptive function on experimental results optimal solution on GA evolutions

Table 5: The architecture of the improved GA for the luggage design centre problem

The simulation results of the simple and dynamic adaptive function are shown in Table 6. The experimental results have shown that the adaptive function (The total cost: 75376) produces a better value than the than simple adaptive function (The total cost: 90707).

The simulation results on The Simple Adaptive Function
EXAMPLE:
The Chromosome is (Quantity Amount (Q_p)): 427,82,43,364,372,52,146,128,70,469,39,239,243,221,140,512,364,337,251,567,243,525,212,27,35,196,244,496,77,397,415,38,420,232,165,182,358,333,173,372,
Vendor: v1
The product item: p2 The unit Cost:6.5 Quantity Amount: 82 Total Cost:533.0
The product item: p3 The unit Cost:8.5 Quantity Amount: 43 Total Cost:365.5
Vendor: v2
The product item: p1 The unit Cost:20.5 Quantity Amount: 364 Total Cost:7462.0
The product item: p2 The unit Cost:7.5 Quantity Amount: 372 Total Cost:2790.0
The product item: p3 The unit Cost:7.5 Quantity Amount: 52 Total Cost:390.0
The total cost: 90707

The simulation results on Dynamic Adaptive Function
EXAMPLE:
The Chromosome is (Quantity Amount **(**Q_p**)**): 136,82,43,62,372,52,146,128,70,469,39,509,243,221,479,76,364,337,251,567,243,525,212,27,35,196,244,496,77,397,415,38,420,232,165,182,358,333,173,372,
Vendor: v1
The product item: p1 The unit Cost:10.5 Quantity Amount: 136 Total Cost:1428.0
The product item: p2 The unit Cost:6.5 Quantity Amount: 82 Total Cost:533.0
The product item: p3 The unit Cost:8.5 Quantity Amount: 43 Total Cost:365.5
Vendor: v2
The product item: p1 The unit Cost:20.5 Quantity Amount: 62 Total Cost:1271.0
The product item: p2 The unit Cost:7.5 Quantity Amount: 372 Total Cost:2790.0
The product item: p3 The unit Cost:7.5 Quantity Amount: 52 Total Cost:390.0
The total cost : 75376

Table 6: The simulation results of the simple adaptive and dynamic adaptive parameters in GA

To understand the individual cost change during the execution process of the algorithm, a cost reduction index is defined below:

$$\text{Change ratio} = \frac{(\text{Setup Cost})\,\mathit{or}\,(\text{Investment Cost})}{\text{Total Cost}} \qquad \text{(Equation 2)}$$

If the investment cost and inventory cost are fixed, the change rates of investment cost/inventory cost vs. the total cost (TC) are shown in Figures 5 and 6. It can be interoperated that the fixed part of the cost in the total cost is increased, and other variable parts of the cost such as setup, machining and shipping will be optimized to lower levels.

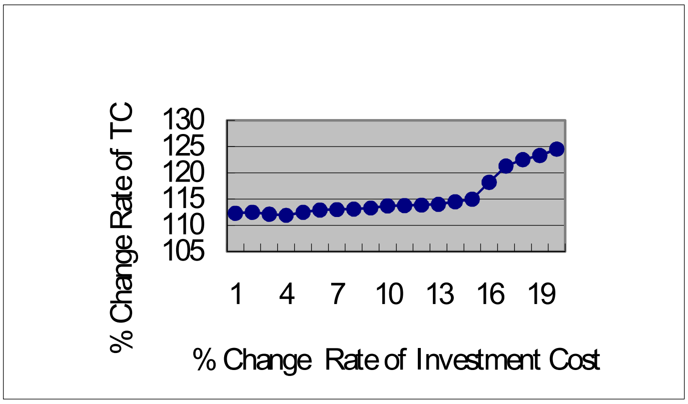

Figure 5: Change rate of investment cost vs. total cost (TC)

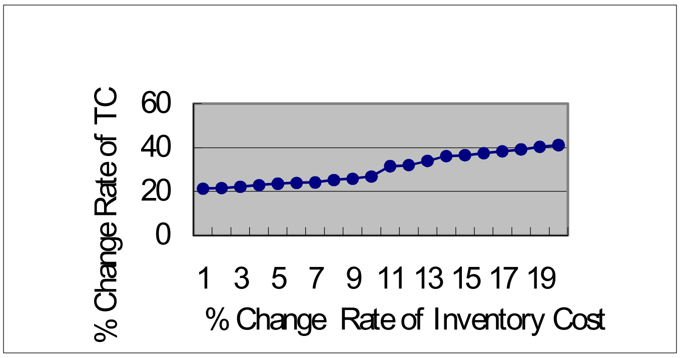

Figure 6: Change rate of inventory cost vs. total cost (TC)

6 Conclusions

In order to improve the performance of GA, this research proposed dynamically adaptive parameter setting. Adaptive crossover and mutation strategies were designed and applied in GA based on transformation functions to pursue better stability between exploration and exploitation in a search space so as to improve both search effectiveness and algorithm robustness. Experimental results showed that the new algorithm was enhanced signficantly in terms of computational efficiency, optimal solutions and robustness. This research was successfully applied to a luggage design centre for lifecycle cost reduction in new product development in luggage chains. Simulation results for the application showed that this approach can obtain the lower cost selections for supply chain management. It therefore concludes that this dynamic adaptive GA can perform better than the one with a simple adaptive GA.

Future research will be conducted to explore non-linear models to improve GA search in complex optimization problems. Meanwhile, further experiments are expected to apply the algorithm to wider applications such as the electronics industry, in which various high precision processes need to be optimized.

References

[Alfaro-Cid et al. 09] Alfaro-Cid, E., McGookin, E.W. and Murray-Smith, D. J.: "A comparative study of genetic operators for controller parameter optimization"; Control Engineering Practice, 17, 1 (2009), 185-197.

[Anane et al. 02] Anane, R. M., and Younas, Tsai, C-F., and Chao, K-M.: "Agent-based Framework for the Supply Chain"; Conference Proceedings of IEEE 2002 International Conference on Machine Learning and Cybernetics, 4, (2002), 1956 - 1961.

[Angelova et al. 11] Angelova M., Tzonkov S., and Pencheva T.: "Genetic algorithms based parameter identification of Yeast Fed-Batch cultivation"; in Proceedings of the Conference on ”Numerical Methods and Applications”, vol. 6046 of Lecture Notes in Computer Science, 2011, 224–231.

[Caldentey and Wein 03] Caldentey, R. and Lawrence W.M.: "Analysis of a Decentralized Production Inventory System"; Manufacturing & Service Operations Management, 5, 1 (2003), 1-10.

[Chao et al. 02] Chao, K-M., Anane P., Norman, R., and James, A.: "An agent-based approach to engineering design"; Journal of Computers in Industry, 48, 1 (2002), 17-28.

[Chen and Liang 11] Chen C.S. and Liang J.Y.: "Large effcient searching algorithms for multi-dimensional space data using hilbert space flling curves"; Journal of Information & Computational Science, 8, 1, (2011), 41-50.

[Chiu 01] Chiu, C.C.: "The Analysis of Related Total Cost of Postponement Strategies for Supply Chain Management"; Master Thesis, Chau-Tong University (2001).

[Choubey 10] Choubey N.S.: "A Novel Encoding Scheme for Traveling Tournament Problem using Genetic Algorithm"; IJCA Special Issue on Evolutionary Computation, 2, 2010, 79–82.

[Davis 85] Davis, L.: "Applying Adaptive Algorithms to Epistatic Domains"; Processings of 9th International Joint Conference on AI (1985), 162-164.

[DeJong 07] DeJong, K.: "Parameter Setting in EAs: a 30 Year Perspective, Parameter Setting in Evolutionary Algorithms"; Springer Verlag (2007), 1-18.

[Dongmei et al. 11] Zhang D.M., Liu W., Wang A. and Deng D.: "Parameter Optimization for Support Vector Regression Based on Genetic Algorithm with Simplex Crossover Operator"; Journal of Information & Computational Science 8, 6, (2011), 911-920.

[Erel et al. 04] Erel, E., Sabuncuoglu, I., and Gurhan K.O.: "Analyses of Serial Production Line Systems for Interdeparture Time Variability and WIP Inventory Systems"; International Journal of Quantitative Operations Management, 10, 4 (2004), 43-63.

[Felix et al. 08] Felix, T., Chan, S., Chung, S. H., and Subhash W.: "A Hybrid Genetic Algorithm for Production and distribution"; Omega, 33, 4 (2005), 345-355.

[García-Martínez and Lozano 10] García-Martínez C. and Lozano M.: "Evaluating a Local Genetic Algorithm as Context-Independent Local Search Operator for Metaheuristics"; Soft Computing 14,10 (2010), 1117-1139.

[GEATbx 06] GEATbx: "The Genetic and Evolutionary Algorithm Toolbox"; http://www.geatbx.com/docu/fcnindex-01.html#P129_5426 (2006).

[Goldberg 89] Goldberg, D.E.: "Zen and the Art of Genetic Algorithms"; in J. D. Schaffer, Editor, Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann (1989), 80-85.

[Hongfeng and Guanzheng 09] Hongfeng X., Guanzheng T.: "High-dimension simplex genetic algorithm and its application to optimize hyper-high dimension functions"; WRI global congress on intelligent systems, 2, (2009), 39–43.

[Jason and konstantinos 02] Jason, G. D., and konstantinos G. M.: "An Experimental Study of Benchmarking Functions for Genetic Algorithms"; IJCM, 79, 4 (2002), 403-416.

[Kapoor et al. 10] Kapoor V., Dey S. And Khurana A.P.: "Empirical analysis and random respectful recombination of crossover and mutation in genetic algorithms "; International Journal of Computer Applications. Special issue on Evolutionary Computation for Optimization. ECOT, (2010), 5-30.

[Kapoor et al. 11] Kapoor V., Dey S. and Khurana A.P.: “An Empirical Study of the Role of Control Parameters of Genetic Algorithms in Function Optimization Problems"; International Journal of Computer Applications, 31, 6 (2011), 20 – 26.

[Liu et al. 05] Liu J.X., Zhang S. And Hu J.M.: "A case study of an inter-enterprise workflow-supported supply chain management system", Information & Management , 42, 3 (2005), 441-454.

[Lu and Xiuxia 11] Lu L. and Quan X.: "The experimental parameters optimization approach using a learning genetic algorithm"; International Journal of the Physical Sciences., 6, 7 (2011), 1803-1807.

[Angelova and Pencheva 11] Maria Angelova and Tania Pencheva: "Tuning Genetic Algorithm Parameters to Improve Convergence Time"; International Journal of Chemical Engineering, Volume 2011, Article ID 646917,doi:10.1155/2011/646917 (2011), 1-7.

[Nedjah et al. 08] Nedjah, N., Mourelle, L., and Macedo D.: "Evolutionary Optimization for Intelligent Systems Design"; Journal of Universal Computer Science, 14, 15 (2008), 2453-2455.

[Otman and Jaafar 11] ABDOUN Otman and ABOUCHABAKA Jaafar: "A Comparative Study of Adaptive Crossover Operators for Genetic Algorithms to Resolve the Traveling Salesman Problem"; International Journal of Computer Applications , 31, 11 (2011), 49-57.

[Qu et al. 11] Qu L.D., He D.X. and Huang Y.: "Improved diferential evolution for the maximum module of the roots of polynomials and its application"; Journal of Information & Computational Science, 8, 1 (2011), 169-177.

[Schaffer and Morishima 87] Schaffer J. D. and Morishima A.: "An Adaptive Crossover Distribution Mechanism for Genetic Algorithms"; in J. J. Grefenstette, Editor, Proceedings of the Second International Conference on Genetic Algorithms, Lawrence Erlbaum Associates (1987), 36-40.

[Schwefel et al. 89] Schwefel, J.D., Caruna, R.A. and Eschelman L.J: "A study of control parameters aecting online performance of genetic algorithms for function optimization"; Proceedings of the 3rd ICGA (1989), 61–68.

[Selwyn 05] Selwyn, P.: "Machine Learning for Dynamic Multi-product Supply Chain Formation"; Expert Systems with Applications, 29, 4 (2005), 985-990.

[Smith 96] Smith, P.: "An Introduction to Knowledge Engineering"; University of Sunderland, International Thomson Computer Press (1996).

[Srinivas and Patnaik 94] Srinivas M., and Patnaik L. M.: "Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms"; IEEE Transactions on Systems, Man and Cybernetics, 24, 4 (1994), 656-667.

[Tahera et al. 08] Tahera K., Ibrahim R.N. and Lochert P.B.: "GADYM - A Novel Genetic Algorithm in Mechanical Design Problems"; Journal of Universal Computer Science, 14, 15 (2008), 2566-2581.

[Whitley 94] Whitley D.: "A Genetic Algorithm Tutorial"; Statistics and Computing, 4, 2 (1994), 65-85.

[Zhao 07] Zhao, H.: "A multi-objective genetic programming approach to developing Pareto optimal decision trees"; Decision Support Systems, 43, 3 (2007), 809–826.

[Zhao and Wang 11] Zhao J.Q. and Wang L.: "Center based genetic algorithm and its application to the stiffness equivalence of the aircraft wing"; Expert Systems with Applications, 38, 5 (2011), 6254-6261.