# Breeding Software Test Data with Genetic-Particle Swarm Mixed Algorithm

Kewen Li
College of Computer and Communication Engineering, China University of Petroleum, Dongying, Shandong, 257061, China
School of Management, Tianjin University, Tianjin,300072, China
Email: likw@upc.edu.cn

Zilu Zhang
College of Computer and Communication Engineering, China University of Petroleum, Dongying, Shandong, 257061, China
Email: zzlguoyanyanyun@163.com

Jisong Kou
School of Management,Tianjin University, Tianjin,300072, China

*Abstract*—**Software test is usually costly and vital in software development lifecycle. Though genetic algorithms have the globally searching capability, premature convergence and weak local optimization are two key problems existing in the conventional genetic algorithm. This paper introduces particle swarm optimization into genetic algorithm to breed software test data automatically. The GPSMA (Genetic-Particle Swarm Mixed Algorithm) uses the individual's update mode to replace the mutation operation in genetic algorithm on the basis of population division. The experimental results show the new method can not only maintain effectively the polymorphism in the colony and avoid premature, but also greatly improve the convergent speed.**

*Index Terms*—**software test, data generation, mixed algorithm, particle swarm optimization, genetic algorithm**

## I. INTRODUCTION

Software test is the main approach to find errors and defects assuring the quality of software. The workload of software test takes up 40%~60% of the total development time. A considerable number parts of software test are mostly suitable for automation [1]. Complete software functional testing in the sense of subjecting the program to all possible input is impossible. In recent years, many researchers proposed a lot of methods to achieve this process. Reference [2] described an approach based on Genetic Algorithms to generate test data for path testing; Reference [3] proposed an approach based on Ant Colony Optimization to generate test data; Reference [4] presented a method using genetic algorithm based on graph theory to generate test cases. All these greatly promoted the development of the theory of automation test.

Different from traditional search algorithms, evolutionary computation techniques work on a population of potential solutions (points) of the search space. Through cooperation and competition among the potential solutions, these techniques often can find optimal more quickly when applied to complex optimization problems. The most commonly used population-based evolutionary computation techniques are motivated from the evolution of nature. Genetic algorithms, evolutionary programming, evolutionary strategies and genetic programming are four well-known examples [5], which have been used in many fields [6]. Evolutionary testing is characterized by the use of meta heuristic search for test case generation. The aim of considered test is transformed into an optimization problem [7] where the input domain of the test object forms the search space for test data that satisfies the respected search goal. .

Since firstly described by Holland [8], genetic algorithms have been applied to a virility of learning and optimizations. Genetic algorithm is an optimization algorithm which based on the principle of natural selection and genetic mechanism. It simulates natural evolution mechanism of life, achieving the specific goals in the artificial system. The principle behind GAs is that they create and maintain a population of individuals represented by chromosomes. By crossover and mutation on chromosomes, the population evolves from one generation to another until meeting the termination conditions. Selection, recombination, and mutation are genetic operations in each genetic algorithm and have been well studied in many literatures. Genetic algorithm is a stochastic, population-based optimization method. It is easily to get local optimal of the problems. Hence many researchers think up a lot of measures to improve the performance of genetic algorithm. Though genetic algorithm is not guaranteed to find the optimal solution, however it often finds a very good solution within the


doi:10.4304/jcp.5.2.258-265

limit time. GA is used to generate test data because their robustness and suitability for solutions of different test tasks have already been proven in previous work [9][10][11][12]. In the application of genetic algorithm, the mutation makes for expanding the search space, but the randomicity of it might wash out the potential solutions whose fitness value is relatively low. In addition, although the mutation in the early stage contributes to the expansion of search space, but the mutation might undermine the stability of the population in the latter stage, thus affecting the convergent speed.

Different from these evolution-motivated evolutionary computation techniques, a new evolutionary computation technique called Particle Swarm Optimization (PSO) is motivated from the simulation of social behavior [6]. Particle swarm optimization which was proposed by Kennedy and Eberhart in 1995[13] is commonly used to solve the problem of nonlinear optimization through the coordination between the individual to implement population convergence. As in GA and Es, PSO exploits a population of potential solutions to probe the search space. In contrast to the aforementioned methods in PSO no operators inspired by natural evolution are applied to extract a new generation of candidate solutions. The individuals in the PSO update themselves using the best value of their own and the best value of the whole population in the history. Finally, the entire population will converge to the global optimum [14].

This paper presents the GPSMA (Genetic-Particle Swarm Mixed Algorithm) to breed software test data for path testing. The triangle program is a benchmark for many researchers in software test. The performance of the GPSMA method in generating test data for the triangle program is investigated. The result of experiments is discussed in comparison with results obtained by other methods. In the next section, test adequacy criteria and test data generation are briefly introduced. In section Ⅲ the GPSMA is detailed described. In section Ⅳ the method of population division is presented, and, in section Ⅴ, the impact factor of mutation and individual's update mode are introduced, and, in section Ⅵ, the experimental results are reported. The paper ends with conclusions and ideas for future research.

## II. TEST ADEQUACY CRITERIA AND TEST DATA GENERATION

As is mentioned above, software testing automation can relieve software engineers from their tedious daily task, and reduce the cost of software developing significantly [15]. There are several approaches have been proposed for automated test data generating, including random method, UML based method, program specification based method and symbolic evaluation method.

Evolutionary Algorithms have proved a powerful optimization algorithm for the successful solution of software testing. They are especially used in those cases where the characteristics of the system are not or only partly known or where classical mathematical methods are not applicable [16].

There are two main testing techniques: white box testing and black box testing. The former considers the internal structure and behavior of the software when generating test data, while the latter generates test data for software from its specification without considering the behavior of the program under test. The objective of black box testing is to find out when the input-output behavior of the program dose not agree with its specification. It is also called functional or specification based testing. In contrast to this is white box testing. In white box testing, the structure and the behavior of the software is examined by execution of the source code. Test data are derived from the program's input domain [17]. It is also called program-based or structure testing. Automating test data generation is an important step in reducing the cost in software development cycle and when automating, which is important to look at two aspects: the test data generation algorithms and test adequacy criterion [18] .Empirical results show that tests selected on the basis of test adequacy criteria are useful for uncovering faults and errors.[Horgan et al. 1994, Chilenski and Miller,1994,Demillo and Mathur,1992]. Test adequacy criteria are objective measures by which the quality of test cases can be judged. Neither of these benefits can be gotten unless adequate test data (almost impossible) can be found. Manual generation of such tests can be quite time-consuming, so obtaining test cases automatically attracts many researchers' interest. There is no doubt that automatic test will significantly reduce the test cost and improve the efficiency of software test.

However, for almost any realistic system an exhaustive test suite will contain infinitely many test cases, so that a test suite can never be executed. Therefore a finite selection from the infinite exhaustive test suite is necessary. Unfortunately, test selection is a difficult task. A simple solution is to make a random selection, although it may be quite satisfactory. It is better to apply selection criteria. A criterion aims at detecting as many errors or faults as possible within a restricted period of time: it should maximize the chance of detecting an error or a fault while minimizing the cost of executing the test suite [19].

Most test adequacy criteria require certain features of a program's source code to be exercised. A simple example says: Each statement in the program should be executed at least once when the program is tested. The methodologies that use such criteria are usually called coverage analyses, because certain features of the source code are to be covered by the test cases [20].

There is a hierarchy of increasingly complex coverage criteria having to do with the conditional statements in a program. We shall refer to this hierarchy as defining levels of coverage. At the top of hierarchy is multiple condition coverage, which requires the tester to ensure that every permutation of values for the Bollen variables in every condition occurs at least once. At the bottom of the hierarchy is function coverage which requires only that every function be called once during testing. Somewhere between these extreme is path coverage,

which is the criterion we use in our test-data experiments [20].

Path coverage is the most complete of all the path testing methods. If every possible path in the program is executed, the path coverage could achieve 100%. It means that from the start of the program all possible branches of reaching the end will be passed, but this is usually impossible and impractical [21]. Testing only a small set of input values and some representative paths is the solution. The experiment in this paper chooses the path that satisfies outputting the isosceles triangle as the target path, using the GPSMA to generate test data for it. The explicit introduction of the experiment is shown in part VI.

## III. THE GPSMA (GENETIC-PARTICLE SWARM MIXED ALGORITHM)

In biology, niche refers to the role of organizational functions in specific environment. Niche in nature provides the possibility for the formation of new species and it is a key reason to maintain the biological diversity. Many researches bring the niche technology into genetic algorithm in order to maintain evolutionary potential of GA to deal with the multimodal function optimization problems [22]. On the basis of population division, drawing on the idea of niche, this paper brings forward the GPSMA method to generate test data in each sub-population. In order to avoid forming of the island of evolution, this paper uses the "excellent rate of production" (denoted by $\delta i$) to dynamically adjust the update speed of individual within each sub-population. Under the premise of biological diversity, the evolutionary process can be controlled, so as to achieve the purpose of asynchronous and efficient evolution.

The first step of the GPSMA is to select a random sample of individuals from the search space for each sub-population and set the initial value of $\delta i$ for each sub-population, in this experiment, the value of $\delta i$ is 0.14. The next step is to compute the fitness value for each individual in order for the selection and crossover operations. The GPSMA will automatically compute the value of $T_{ij}$ and $\delta i$ for each sub-population in order for updating individuals in the next step. If the terminal conditions can not be satisfied after these steps, the above steps should be continued. The main process of GPSMA is shown in Fig. 1.

Here are some details about the process. After computing the fitness of each test case in the current population, the algorithm will execute the selection and crossover operations. This paper uses the roulette wheel method to implement the selection operation and uses the single-point crossover method to accomplish the crossover operation. It should be noted that the selection and crossover operation are completed in each sub-population separately. The terminal conditions can be set according to specific environment. Take the first experiment in this paper for example, the terminal condition is the maximize number of iteration.

If there is the individual that meets the constraints in the interactive process, it will be output and replaced by new individual in order to maintain the size of population. The problem of using any method to generate test data is different from optimization problem where there is only a single goal. In test data generation problem, the goal is to find a set of data to satisfy the adequacy criteria. The random method has a very low efficiency in test data generation, but in many evolutionary methods, the random method may help to increase the quantity of data generated. In the GPSMA, the initial individuals of each sub-population is randomly generated, therefore, they may contain the data that satisfies the coverage criteria. In order to generate the data as more as possible, the GPSMA will continuously examine if there is such data, output it and replace it using new one. In this paper, the random method is not predominant. The test data generation is mainly relying on the GPSMA.

Further more, there is two crossover strategies, namely single-point and double-point crossover. In order for the simple principle, the experiment in this paper uses the single-point crossover strategy and the position of crossover is chosen by random. Whether the double-point one has the better performance or not will be discussed in the future work.

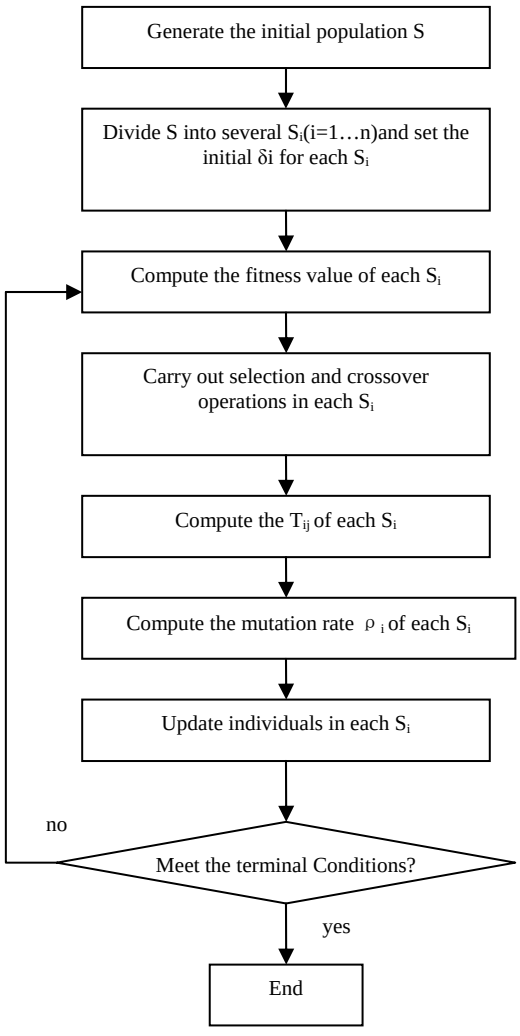


Figure 1. The main process of GPSMA

## IV. THE STRATEGY OF GROUP DIVISION

In the study of genetic algorithm, researchers have put forward a large number of different strategies for dividing the population. Reference [23] used the similarity denoted by Hamming distance to divide the population. Reference [24] presents a method using the constraints to divide population. In order to preventing formation of the evolution of island, it uses the coverage density to achieve the communion between sub-populations. Reference [25] proposed an even partition strategy based on the code of each individual. The method can divide the population into several sub-populations which could reflect the overall nature of the solution space. This paper adopts the population division strategy in reference [25]. The GPSMA uses a binary vector as a chromosome to represent values of the program input variables. The length of the vector depends on the required precision and the domain length for each input variable. Supposing the initial population is S. Based on the theory of literature [25], if the rank of S is s, then S could be divided into $2^s$ sub-populations. In short, in each sub-population, the s bits code in the far right of the individual is identical. For example, if there is a population A and the length of chromosome of each individual in A is 8. The rank is 2, then A could be divided into 4 sub-populations and the style for each sub-population is:

######00
######01
######10
######11

Each "#" denotes a 0 or 1.

## V. THE IMPACT FACTOR OF MUTATION AND THE UPDATE MODE OF INDIVIDUAL

This paper introduces the "excellent rate of production" (denoted by δi) as the impact factor of mutation. Assuming that in the i-generation, the sub-population j produces the total quantity of optimal individuals is $T_{ji}$ and the total quantity of optimal individuals of all the sub-populations is $\sum_{j=1}^{n} T_{ji}$, then $\delta i = T_{ji} / \sum_{j=1}^{n} T_{ji}$.

The "excellent rate of production" reflects impact degree of the history speed on the current speed. The role of δi is considered important for the GPSMA's convergence behavior. δi is employed to control the impact of the previous history of velocities on the current velocity. Thus, the parameter δi regulates the trade-off between sub-populations. If δi is large, it facilitates exploration (searching new areas); if δi is small, it facilitates exploration, i.e. fine-tuning the current search area.

PSO is a stochastic global optimization method which is based on simulation of social behavior. As in GA and ES, PSO exploits a population of potential solutions to probe the search space [26]. In each iteration, the particle update itself by tracking two extreme values, the first is personal best value in history (denoted by $p_{best}$), the second is the best value of sub-population in history (denoted by $g_{best}$). The particles are manipulated according to the following equations:

$$v_i(t+1) = v_i(t) + c_1 r_1 \times (p_{best} - x_i(t)) + c_2 r_2 \times (g_{best} - x_i(t)) \quad (1)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (2)$$

Where i =1,2,..,N, and N is the size of the population. $c_1$ and $c_2$ are two positive constants, called the cognitive and social parameter respectively. In the experiment of this paper ,they are set 2.0; $r_1$ and $r_2$ are random numbers uniformly distributed within the range [0,1].

There is a significant difference between test data generation and optimization, that is, whether from history or from the current generation, there may be more than one optimal solution to meet the constraints. This paper uses the latest optimal solution as the parameters in the formula. Consider $p_{best}^i$ as the personal best value and $g_{best}^j$ as the best value of sub-population. In order to avoid forming the island of evolution, this paper uses 1-δi as inertia weight, so that the speed of individual in sub-population with lower δi has larger changes, and vice versa. Then the amended formula is:

$$V_i(t+1) = \rho_j V_i(t) + c_1 r_1 (p_{best}^i - X_i(t)) + c_2 r_2 (g_{best}^j - X_i(t)) \quad (3)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (4)$$

This paper uses the above formula to replace the mutation operation in traditional genetic algorithm. It is easily to find that every individual has the chance to expand its searching scope, contributing to find the optimal solution.

## VI. EXPERIMENTAL RESULTS

The validity of any technique can only be ascertained by means of experimental verification. The triangle program（Fig. 2 Triangle Classify） is a benchmark for many researchers in software testing. A lot of researchers usually use the program to validate the performance of their methods. The purpose of the program is to classify a triangle given the lengths of the three sides into one of four categories: scalene, isosceles, equilateral or invalid triangle [27].

The genetic algorithm only depends on the evaluation function to accomplish the evolution of each individual. In the experiment, we use the return value of the Triangle Classify Program as the fitness value of individual, using a specific function to output the best individual and replace it with a new one.

Because there is only three variables in this program, namely a,b and c, so there is three dimensions in each chromosome. According to the particle swarm algorithm, each individual in the population will using two extreme values to update its position. In this experiment, each dimension of the individual will update its value according to the best individual of sub-population in history and the best value of its own.

Fig. 2 shows the source code of Triangle Classify Program. The value of "tringle" denotes the three parameters a, b and c could compose a scalene, isosceles, equilateral or invalid triangle. In the experiment, the data generated is used to drive the program to run and the return value of the program is used to deicide whether the data should be output or not.

Further more, from the source code it is easily to see that the three parameters a, b and c are integers. So in this paper's experiments, the operations on chromosomes are very simple.

The value of each parameter used in experiments is introduced in the next paragraph.

```
int TringleClassify(int a, int b, int c)
{
int tringle = 0;
if ((a + b > c) & (b + c > a) & (c + a > b))
{
    if ((a != b) & (b != c) && (c != a))
    {
        tringle = 1;
    }
    else
    {
        if ((a == b) & (b != c) || (b == c) &
           (c != a) || (c == a) & (a != b))
        {
            tringle = 2;
        }
        else
        {
            tringle = 3;
        }
    }
}
return tringle;
}
```

Figure 2 . Triangle Classify Program

In this experiment, the rank is 3, so that the entire population is divided into 8 sub-populations. The size of each sub-population is 100, the encoded string length of the individual is 24, the probability of crossover is 0.75, set c1=c2=2.0, and the GPSMA runs for 10 iterations. The solutions that could satisfy the isosceles triangle are considered to be the optimal solutions.

We use C#.Net instrument to implement our algorithm, and through 100 times duplicate experiments to get the data that satisfies the coverage criteria. The changes of quantity and the concrete number in each generation are in the following figures 3-10.

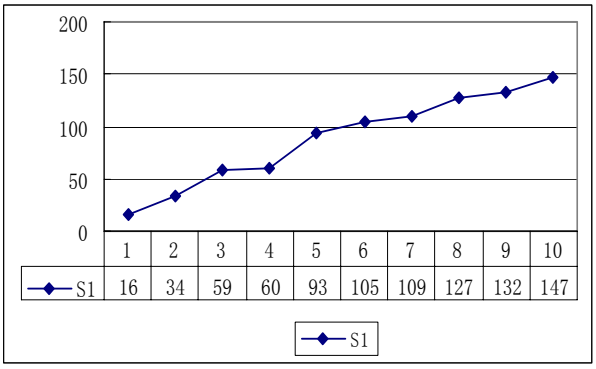


Figure 3. Test data generation in sub-population 1

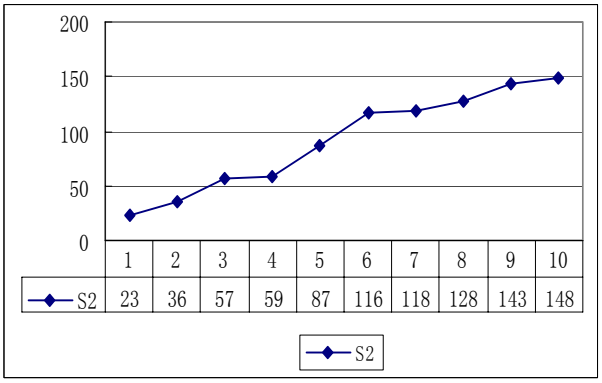


Figure 4. Test data generation in sub-population 2

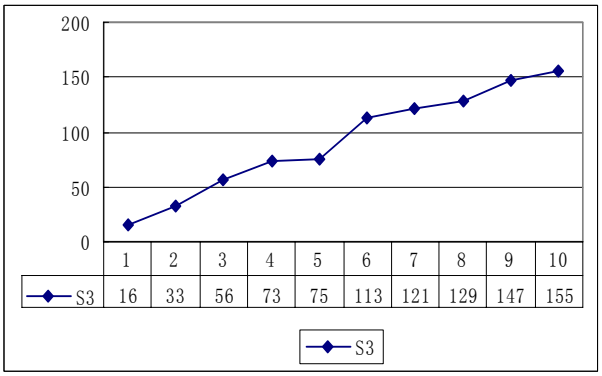


Figure 5. Test data generation in sub-population 3

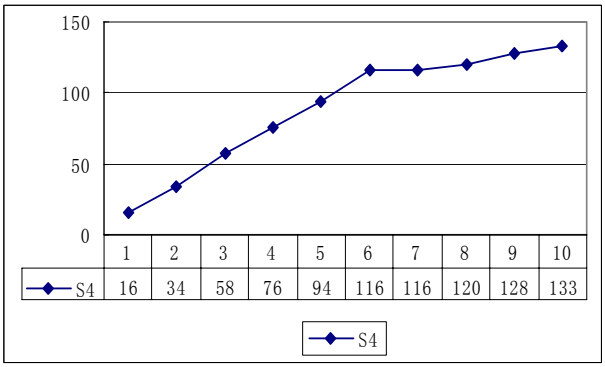


Figure 6 .Test data generation in sub-population 4

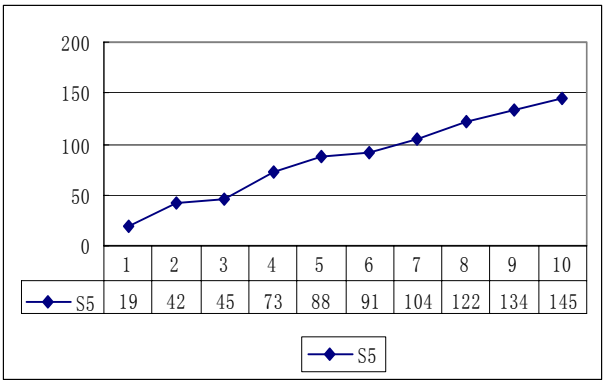


Figure 7. Test data generation in sub-population 5

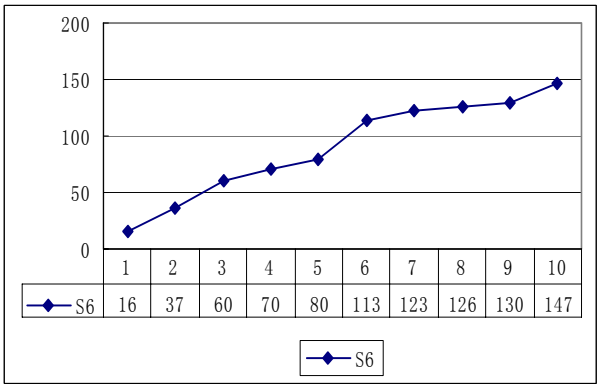


Figure 8. Test data generation in sub-population 6

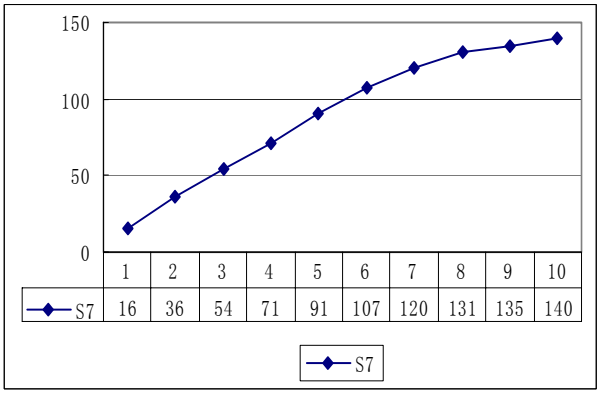


Figure 9. Test data generation in sub-population 7

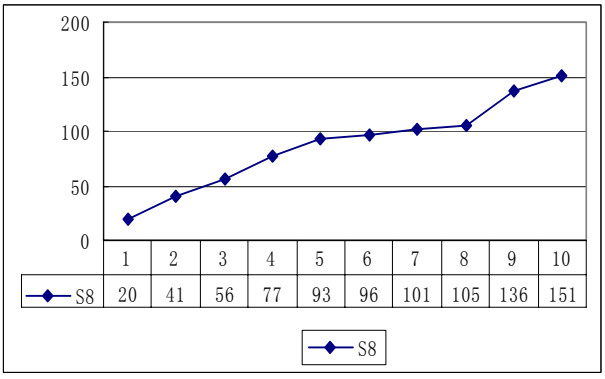


Figure 10. Test data generation in sub-population 8

The results show that the GPSMA can perform well in generating test data. The total amount of data increases continuously against the number of generations in each sub-population. The whole variety trend of the graph is ascending, which means the quantity of data increases gradually. It is easily to infer that the population division can maintain the diversity of colony effectively, and each sub-operation could evolve the superior individuals independently.

In the above experiment, there is no complicated path in the Triangle Classify Program. Further more, path cover is an effective software structural testing criteria. If testing can be designed to force execution of all paths, every statement in the program will have the chance to be executed in its true and false sides [28]. The reason for the Triangle Classify Program is used so regularly as a benchmark is even when using a large range of integers only a few combinations will satisfy particular branches in the code. So many researchers use it to verify their methods have a higher performance than others.

Under the premise of getting the concrete run times of genetic algorithm and ant colony optimization [29] in generating 10 test data. At last, in order to compare the efficiency of this method with other methods, we use the GPSMA to get 10 test data and record its average run times for comparison.

The comparative result is shown in Fig. 11.

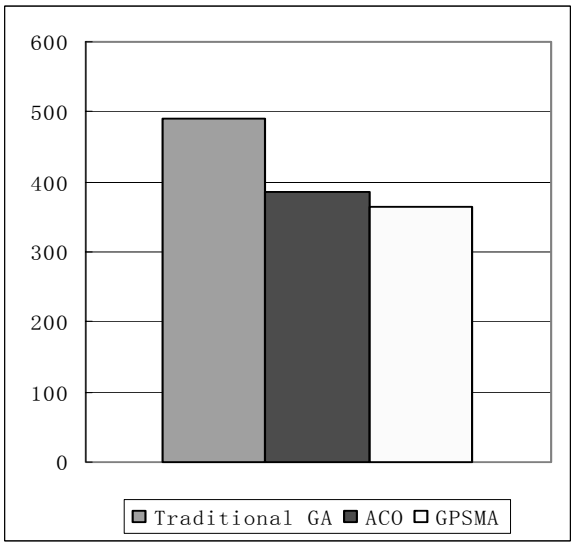


Figure 11. Experimental comparison diagram

From Fig. 11, it is easily to find the ACO and the GPSMA are notable better than traditional GA. This indicates that through replacing the mutation operation in GA with the individual update mode in PSO, each individual has the chance to achieve updating in each generation, so that there is a higher possibility to find the optimal solution. However, there is great difference between traditional genetic algorithm and modified genetic algorithms. So the result of the comparison can not indicate the GPSMA is superior to all GAs. Take reference [27] for example, it introduces the concept of

relative fitness function. Novelty, proximity and severity are developed for the fitness value. The method in reference [27] shows high efficiency in experiments.

## VII. CONCLUSIONS AND FURTHER WORK

Software test is an expensive and difficult task. if software test phase could be automated, the cost of software development would be significantly reduced [30]. This paper proposes a new method to breeding software test data called GPSMA for structure data test generation, introducing a new strategy to replace the mutation operation in traditional genetic algorithm, and using the "excellent rate of production" to implement the interaction between sub-populations. The GPSMA can search and generate certain test data in a domain to satisfy the test criteria. The capability of the GPSMA method to address test data generation problems was investigated through the Triangle Classify Program. Theoretical analysis and practical testing results show that the approach is simpler, easier, and more effective in generating test data automatically. The comparison with ant colony optimization and traditional genetic algorithm shows that the GPSMA is a good alternative for test data generation problems.

The method also has its weakness. Future work will focus on testing the GPSMA on a comprehensive set of benchmarking functions and the applications to real-world optimization problems, as well as fine-tuning of the parameters that may result in better solutions and experiment the proposed algorithm using more complex programs with loops and arrays using different data types(floats, characters, strings and so on).

## REFERENCES

[1] Xia Yun,Liu Feng, "Automated software test data generation based on immune genetic algorithm" , Computer Applications，2008，3（28）：723-725.

[2] Jin-Cherng Lin, Pu-Lin Yeh, "Using Genetic Algorithms for Test Case Generation in Path Testing" , Test Symposium,2000. (ATS 2000). Proceedings of the Ninth Asian, 2000, pp.241-246.

[3] Huaizhong Li, C.Peng Lam, "Software Test Data Generation using Ant Colony Optimization", Proceedings Of World Academy Of Science, Engineering And Technology, January 2005，Volume.1, pp.1-4.

[4] Dr.Vrlur Rajappa, Arun Biradar, Satanik Panda, "Efficient Software Test Case Generation Using Genetic Algorithm Based Graph Theory" , First International Conference on Emerging Trends in Engineering and Technology, 2008, pp.298-303.

[5] Wegener J, Baresel A, Sthamer H, "Evolutionary Test Environment for Automatic Structural Testing [J]", Information and Software Technology, 2001, 43(14), pp: 843-854

[6] Yuhui Shi, Russell C. Eberhart, "Parameter Selection in Particle Swarm Optimization" , Lecture Notes In Computer Science; Vol. 1447 Proceedings of the 7th International Conference on Evolutionary Programming VII, 1998 pp: 591-600.

[7] H.Sthamer, "The Automatic Generation of Software Test Data Using Genetic Algorithms", PhD.Thesis, University of Glamorgan, Wales, Grea Britain,1996

[8] J. Holland, "Adaptation in natural and artifical systems", Ann Arbor:The University of Michigan Press,1975

[9] Harmen Sthamer, Joachim Wegener, Andre Baresel. "Using Evolutionary Testing to improve Efficiency and Quality in Software Testing", Daimler ChryslerAG, Research and Technology, Alt-Moabit 96a, D-10559 Berlin, Germany

[10] M.R. Girgis, "Automatic Test Data Generation for Data Flow Testing Using Genetic Algorithm", Journal of University Computer Science, vol.11,no,6,2005

[11] C.Mihael, G. McGraw, M.Schatz, C.Walton, "Genetic Algorithm for Dynamic Test Data Generation", Proceedings of the 12th International Conference of Automated Software Engineering, vol.1, issue,5, Nov 1997,pp:307-308

[12] INCE, D.C (1987), "The Automatic Generation of Test Data", The Computer Journal, vol.30, no.1, pp: 63-69

[13] Kennedy,J. and Eberhart,R.C.(1995), "Particle swarm optimization", Proc.IEEE Int'l.Conf.on Neural Networks,IV,1942-1948. Piscataway, NJ: IEEE Service Center.

[14] Zhong Wen Liang,Wang Hui Sen,Zhang Jun, Xu De Jian, "Novel particle swarm optimization with heuristic mutation," Computer Engineering and Desgn. July.2008,pp:3402-3405.

[15] Jifeng Chen, LiZhu, Junyi Shen, Zhihai Wang, and Xinjun Wang, "An Approach on Automatic Test Data Generation with Predicate Constraint Solving Technique".

[16] Hartmut Pohlheim, "Competition and Cooperation in Extended Evolutionary Algorithms".

[17] Harmen-Hinrich Sthamer, "The Automatic Generation of Software Test Data Using Genetic Algorithms"

[18] Bruno T.de Abreu, Eliane Martins,Fabiano L.de Sousa, "Automatic test data generation for path testing using a new stochastic algorithm", http://www.sbbd-sbes2005.ufu.br/arquivos/16-%209523.pdf

[19] Tretmans, G.J. and Brinksma, H, "TorX: Automated Model-Based Testing", First European Conference on Model-Driven Software Engineering, December 11-12, 2003, Nuremberg, Germany. pp: 31-43.

[20] Michael, C.C.; McGraw, G.; Schatz, M.A, "Generating software test data by evolution" , Software Engineering, IEEE Transactions on Volume 27, Issue 12, Dec 2001 pp:1085 - 1110.

[21] Jifeng-Chen, LiZhu, Junyi Shen, LingChen, "Path-Based Automatic Test Data Generation Approach",Control and Decision, vol.20, no.9, pp:1065-1068

[22] Lin Yan, Hao Ju Min, Ji Zhuo Shang, Dai Yin Sheng, "A study of genetic algorithm based on isolationniche technique" ,Journal Of Systems Engineering. March 2000, pp:86-91.

[23] Wu Yan,Li Ru Yun, "Immune Genetic Algorithm Based On Group Division and Hybridization", Computer Engineering. February 2008:220-222.

[24] Wang Wen Yi, Qin Gurang Jun, Wang Ruo Yu, "Research on Genetic Algorithm Based on Particle Swarm Algorithm", Computer Science. 2007 Vol.34 No.8:145-147.

[25] Liu Shou Sheng, Yu Sheng Lin, Ding Yong, Zhong Jie, "Multipopulation Parallel Genetic Algorithm Based on Even Partition", Journal of Data Acquisition & Processing. Jun.2003:142-145.

[26] Konstantinos E. Parsopoulos and Michael N. Vrahatis, "'Particle Swarm Optimization Method for Constrained Optimization Problems" , Frontiers in Artificial Intelligence and Application, 2002 (3).

[27] D. Berndt, J. Fisher, L. Johnson, J. Pinglikar, A. Watkins,

"Breeding Software Test Cases with Genetic Algorithms" , 36th Annual Hawaii International Conference on System Sciences (HICSS'03) - Track 9, January 2003.
[28] Maha Alzabidi, Ajay Kumar, A.D.Shaligram, "Automatic Software Structural Testing by Using Evolutionary Algorithms for Test Data Generations", IJCNS International Journal of Computer Science and Network Security, VOL.9 No.4,Arpil 2009,pp:390-395
[29] FU Bo, "Automated software test data generation based on ant colony algorithm" , Computer Engineering and Applications, 2007,43(12):97-99
[30] Nguyen Tran Sy, Yves Deville. "Automatic Test Data Generation for Programs with Integer and Float Variables".

**Kewen Li**, vice professor, College of Computer and Communication Engineering, China University of Petroleum, Dongying, Shandong, 257061, China; School of Management, Tianjin University, Tianjin,300072, China.

**Zilu Zhang**, College of Computer and Communication Engineering, China University of Petroleum, Dongying, Shandong, 257061, China.

**Jisong Kou**, School of Management,Tianjin University, Tianjin,300072, China.