PROCEEDINGS OF SCIENCE

Test Data Generation with A Hybrid Genetic Tabu Search Algorithm for Decision Coverage Criteria

Xin Fan[1], FengYu Yang, Wei Zheng, QiJun Liang
School of Software
Nanchang Hangkong University
Nanchang, China 330063
E-mail: `fanxinzh2@126.com`

In order to improve the efficiency of test data generation during the software test, many studies have been focusing on the automation of test data generation with metaheuristic techniques such as genetic algorithm and tabu search etc. This paper proposes a hybrid algorithm in combination with genetic algorithm with tabu search to generate the test data based on decision coverage criteria. The hybrid algorithm may improve the global search ability of optimizing the solution of tabu search and local search ability of genetic algorithm; at meanwhile, an approach was described in detail to insert fitness function as stubs in source code, control the prematurity during search iteration, avoid falling into the local optimum and speed up the convergence speed. By applying the hybrid algorithm to the instance program under test, the final result shows that both the decision coverage and the calculation speed have been improved.

CENet2015
12-13 September 2015
Shanghai, China

[1] Speaker

1. Introduction

The test data generation is one of the most time-consuming tasks during software testing, especially for the manual testing. With the rapid development of metaheuristic searching techniques, many optimization algorithms such as genetic algorithm(GA), tabu search(TS) and hill climbing(HC) have been applied to solve the problem of automatic test data generation. Recently, many scholars have studied the metaheuristic algorithm to generate test data. Genetic algorithm[1-2], Tabu search have also been applied to such fields as picture coloring and resource allocation[3-4]. Eugeniad proposed that the tabu search can be applied to software test [5]. A improved tabu search was proposed to make the test case generation to be more efficient [6]. some hybrid strategy is also studied but not used to generate the test data [7-9].

Prematurity of and converging to the local optimum solution are the two main problems to solve in the previous study. In order to improve the searching efficiency and solve the two problems as mentioned above, the paper proposes a hybrid algorithm of combing GA with TS, presents the approach to build the fitness function in genetic algorithm(GA) based on the decision coverage criterion and introduces TS when GA comes to the premature status. The paper also introduces the steps to generate initial population and sets up crossover and mutation parameter in detail.

The paper is organized as follows. In Section 2, we describe the definition and features of GA and TS, analyze the advantages and disadvantages of both algorithms. In Section 3, we introduce the approach of building the fitness function based on the decision coverage criteria and insertion fitness function into source code of program under testing; furthermore, the hybrid genetic tabu search algorithm is present in detail. Section 4 describes the experiments carried out in this study and presents discussion on the result of the experiments. Section 5 concludes the whole work of this paper and points out the future improvement of the hybrid algorithm.

2. Genetic Algorithm and Tabu Search

This section makes a description of both genetic algorithm and tabu search, introduces the advantages and disadvantages of both algorithms and tries to find the merging point to combine GA and TS together.

2.1 Genetic Algorithm

Genetic algorithm is a model of simulating biological genetics and natural selection to search the optimized solution as proposed by John Holland (1975) and students of Holland (e.g. DeJong, 1975). In the genetic algorithm, a population of candidate solutions (called individuals) to an optimization problem is evolved for better solutions. Each candidate solution has a set of properties (its chromosomes or genotype) which can be mutated and altered. Now, the genetic algorithm has been applied to many fields such as manufacturing, computational science, mathematic analysis and economics, etc. The advantages and disadvantages of GA are listed as below:

1) GA has the parallel search ability, which searches the optimization solution from different ways in the solution space.

2) GA preserves historical solution information to some extent and is suitable for solving the global optimization problem with large scale fitness functions.

3) GA employs fitness functions and object functions rather than differential functions, which make it more general for the application.

4) If the probability of mutation is too small, the opportunity of introducing new individual will become much less; GA leads to prematurity during the local search and becomes more prone to fall into the local optimum solution.

2.2 Tabu Search

The basic concept of Tabu search is created by Fred W. Glover in 1986 and formalized in 1989[9]. Tabu search is a metaheuristic search method employing local search methods used for mathematical optimization. Tabu searches take a potential solution to a problem and check its immediate neighbors (that is, solutions that are similar except one or two minor details) in the hope of finding an improved solution. The overall approach is to avoid entrainment in cycles by forbidding or penalizing moves which take the solution, in the next iteration, to points in the solution space previously visited. The advantages and disadvantages of TS are listed as below:

1) Since TS has the memory structure, these memory structures form what is known as the tabu list, a set of rules and banned solutions used to filter which solutions will be admitted to the neighborhood to be explored by the search. TS has strong ability of "climbing" and is suitable for local search.

2) TS greatly depends on the quality of initial solution. If the initial solutions are good in solution space, it will increase the calculation speed to find the optimum solution; otherwise, it will lead to bad solutions and decrease the convergence speed.

3. Hybrid Algorithm of Genetic Algorithm and Tabu Search

Many attempts have been tried to put forward some improvement strategy of GA and TS in practical applications. In this section, a hybrid strategy in combination of GA and TS is introduced to generate the test data satisfying decision coverage criterion. In the proposed hybrid genetic and tabu search algorithm (HGATS), GA is applied to search the whole solution space to ensure the diversity of the population and TS is applied to the local search to avoid converging to the local optimum.

3.1 Configuring Genetic Algorithm and Tabu Search

3.1.1 Fitness function

A test criterion is often used to ensure the adequacy of white box testing. The general test criteria are statement coverage(SC), decision coverage(DC), condition coverage(CC), condition/decision coverage(C/DC) and modified condition/decision coverage(MC/DC). In this paper, we create the fitness function consisting of Korel function to drive the search iteration of HGATS. Korel function is defined as the computation of branch distance for different conditions (as shown in Table 2).

No.	Decision to make	Branch distance
1	a<=b	a-b
2	a>=b	b-a
3	a= =b	abs(a-b)
4	a!=b	abs(a-b)
5	a&&b	a+b
6	a||b	min(a,b)

Table1: Branch Distance Computation

When the value of *a* and *b* satisfies the logic relation, the fitness function is 0. When the value of *a* and *b* do not satisfy the relation, the fitness function will take the branch distance as the value. The fitness function of HGATS is described as Equation (3.1).

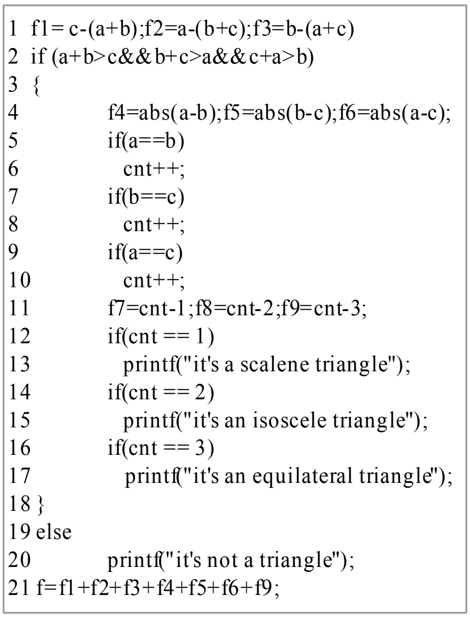
$$f(x) = f(BC_1) + f(BC_2) + ... + f(BC_n) = \sum_{i=1}^{n} f(BC_i) \tag{3.1}$$

in which, *f(BC$_i$)* is the fitness function for every decision. In this paper, we choose the triangle program as the instance program under test. The fitness function has been inserted into the program as shown in Fig. 1 with the flowchart in respect of the triangle program shown in

Fig. 2. If we want to cover the decisions that belongs to the path 2-4-5-6-7-8-9-10-11-12-14-16-17-18-20-21, the whole fitness function is shown in Fig. 1.

3.1.2 Encoding Strategy

In this paper, we use the binary encoding method to encode the individual. If the range of a variable is $[x_{min}, x_{max}]$, a binary code with the length L is used to represent the value. If the individual a is encoded as: $a_L a_{L-1} a_{L-2} \dots a_2 a_1$, we can decode and compute the value of



```
1  f1= c-(a+b);f2=a-(b+c);f3=b-(a+c)
2  if (a+b>c&&b+c>a&&c+a>b)
3  {
4            f4=abs(a-b);f5=abs(b-c);f6=abs(a-c);
5            if(a==b)
6              cnt++;
7            if(b==c)
8              cnt++;
9            if(a==c)
10             cnt++;
11           f7=cnt-1;f8=cnt-2;f9=cnt-3;
12           if(cnt == 1)
13             printf("it's a scalene triangle");
14           if(cnt == 2)
15             printf("it's an isoscele triangle");
16           if(cnt == 3)
17             printf("it's an equilateral triangle");
18 }
19 else
20           printf("it's not a triangle");
21 f=f1+f2+f3+f4+f5+f6+f9;
```

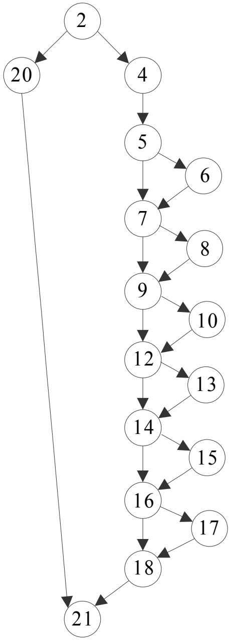


Figure1: Inserting Fitness Function in the Triangle Program

Figure 2: Flowchart of Triangle Program

a by Equation (3.2). The parameter *L* stands for the length of the binary code of individual *a*; and in the experiment presented in Section 4, *L* is set as 8,16 or 32.

$$a = x_{\min} + (\sum_{i=1}^{L} a_i \cdot 2^{i-1}) \cdot (x_{\max} - x_{\min}) / (2^L - 1) \tag{3.2}$$

3.1.3 Selection Strategy:

We use the roulette wheel selection. If the fitness value of some individual is f(xi), the chosen probability is computed as Equation (3.3).

$$p_s = f(x_i) / \sum_{j=1}^{n} f(x_j) \tag{3.3}$$

3.1.4 Other parameters in GA:

Set crossover probablity P_C as 0.85; set the mutation probablity P_m as 0.12. The population size is from 20,30,40,…to 100 and the max number of iteration is 1000.

3.1.5 Neighborhood calculation in TS

If the test data consists of n variables $<x_1,x_2,\dots,x_n>$, we use the following strategy described in Equation (3.4) to get the neighborhood solutions. $j \in [1,n]$, $x^{(1)}{}_j$, $x^{(2)}{}_j$, $x^{(3)}{}_j$, $x^{(4)}{}_j$ stands for possible values when the algorithm searches the neighborhood x_j.

$$x^{(1)}{}_j = x_j + \lambda; x^{(2)}{}_j = x_j - \lambda; x^{(3)}{}_j = x_j + \mu; x^{(4)}{}_j = x_j - \mu \tag{3.4}$$

λ and μ stands for the adjusted dose. In the our algorithm, we set λ as $0.2\times x_j$, μ as $0.1\times x_j$. The calculation addition or subtraction depend on the randomly generated value T within the range between 0 and 1. If T is greater than 0.5, the addition is performed; otherwise, the subtraction is performed.

3.1.6 Neighborhood calculation in TS

We design two types of tabu lists in the TS algorithm, i.e. the short term tabu list and the long term tabu list, in which, the short term tabu list stores the global optimum solution that can avoid repeating search for the global optimum solutions and the long term tabu list is designed to store local optimum solution to avoid falling into the local optimum.

3.2 Hybrid Genetic Tabu Search Algorithm

The basic thought of hybrid genetic tabu search algorithm (HGATS) is that let GA generate individuals with the high fitness value and keep them survive during the selection. When GA is hovering at the old status individuals and become prone to be premature. TS is employed to local search to increase the probability of getting the global optimum solutions. The detail process of HGATS is listed below:

Step 1: set the maximum iteration number as M, the populations size as N, crossover and mutation parameter as p_c and p_m, δ is the threshold indicating premature.

Step 2: generate the population of GA, label every individual as $x^1_m,x^2_m,\ldots,x^N_m$; $m=0$ indicates the m^{th} generation, $c=0$, $k=0$

Step 3: calculate the fitness value for each individual, label as $f(x^1_m),\ldots,f(x^N_m)$; assume $x_{min} = x^1_m$ be the best individual that has the minimum fitness value;

Step 4: sort all the individuals by descending way according to the value of its own fitness. If x_{min} equals to x^1_m , set $c=c+1$; else set $x_{min} = x^1_m$, c=0;

Step 5: if c $>\delta$, copy some of the individuals to the next generation and mutate the rest of the individuals in the current generation. Go to Step 7; else go to Step 6;

Step 6: select N individuals as the parent for the next generation with the probability $p_i = f(x^i_m)/ (f(x^1_m)+f(x^2_m)+\ldots+f(x^N_m))$;

Step 7: crossover and mutate the individuals by p_c and p_m to generate new population.

Step 8: if $k=10$, call TS algorithm to proceed local search among the new generated individuals, then set $k=0$; else go to Step 9.

Step 9: if $m<M$, set $m=m+1$, $k=k+1$, go back to step 3; else go to Step 10.

Step 10: stop the calculation and output the final result.

4. Experiment and Result Analysis

In this section, two different experiments have been carried out to validate the effectiveness and performance of HGATS. Firstly, we decode the individuals of GA by 8 bits, 16 bits and 32 bits respectively, retrieve the same data mentioned in Eugenia's work[5], make a comparison with the pure tabu search and random generation method as described in Eugenia's work[5]. The result is shown in Table 2, in which, the Decision coverage of HGATS can also reach 100%, but the elapsed time has been decreased when compared to the TS. In 8 bits case, the elapsed time decreases from 0.156 to 0.117, which shows the speed is increased by 25%. In 16 bits case, the time decreases from 0.503 to 0.378, which shows the speed is increased by 24%. In 32 bits case, the ratio of speeding up is 10%. To sum up the three cases above, the average speed of generating test data by using HGATS has been increased by 19.6% .

Item	8Bits		16Bits		32Bits	
	TS	HGA TS	TS	HGA TS	TS	HGA TS
Test data	231	211	801	764	21367	20584
Decision coverage	100	100	100	100	100	100
Elapsed time	0.156	0.117	0.503	0.378	23.713	21.267

Table 2: Comparison with the TS Algorithm

The second experiment is carried out to compare the average fitness value and DC coverage of GA and HGATS. It is shown in Fig. 3 that, with the same population size, HGATS is much quicker to get the minimum of the fitness value and reach 100% decision coverage than the standard GA.

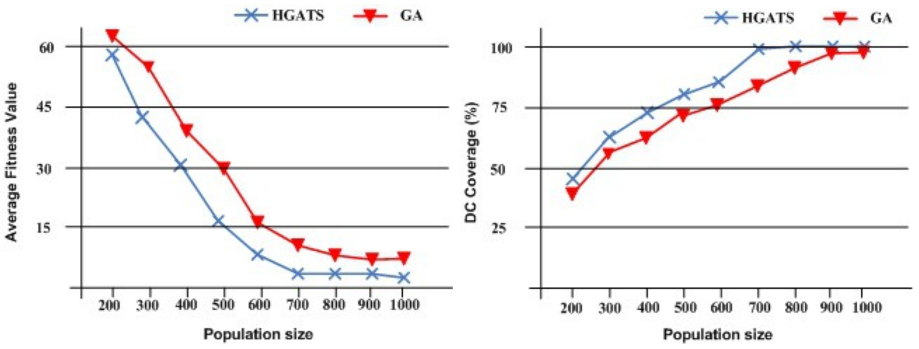


Figure 3: Plot of Population Size vs Average Fitness Value and DC Coverage

5. Conclusion

Even though there are many test data automatic generation proposed recently based on the standard GA, TS, PSO or other evolution algorithms, only a small number of systems appear to use a hybrid GA and TS algorithm. This paper proposes a hybrid algorithm to generate the test data. The creation of fitness function is the basis of the whole algorithm, which ensures the generated data satisfy decision coverage criteria. Based on the experiment result, HGATS improves the converging speed to get the optimum resolution by 19.6% when compared to the former study which applies TS. HGATS is proven to be effective and applicable. Future work lies in the improvement of the performance of this algorithm so as to deal with more complicated program and meet the requirements of MC/DC.

Finally, the authors would like to appreciate Professor Shu and his team for supporting the study in the paper. This paper is also supported by Natural Scientific Foundation of JiangXi Province of China (No.: 20142BAB217017) and Foundation of Education Committee of Jiangxi Province of China (No.: GJJ12457).

References

[1] A. Pachauri, G. Srivastava. *Automated test data generation for branch testing using genetic algorithm: An improved approach using branch ordering, memory and elitism*[J],The Journal of Systems and Software , 86(9),pp: 1191–1208(2013)

[2] Y. Chen, Y. Zhong, T. Shi , J. Liu, *Comparison of Two Fitness Functions for GA-Based Path-Oriented Test Data Generation*[C]. In: In Proceedings of the 2009 Fifth International

Conference on Natural Computation, vol. 4, Washington, DC, USA (ICNC 09), pp: 177–181.(2009)

[3] H. A. Cagdas, H. Gulsum, A. B. Murat. *The effect of neighborhood structures on tabu search algorithm in solving course timetabling problem* [J]. Expert Systems with Applications,36 (10):12349-12356(2009)

[4] R. Alan, J. McKendall. *Improved Tabusearch heuristics for the dynamic space allocation problem*[J]. Computers & Operations Research, 35(10):3347-3359(2008)

[5] D. Eugenia, T. Javier, B. Raquel. *A tabu search algorithm for structural software testing* [J]. Computers & Operations Research, 35(10):3052-3072(2008)

[6] X. M. Zhong, X. F. Zhao. *Automated test case generation based on improved tabu search algorithm*[J].Computer engineering and design,32(6),pp:2058-2060,2081(2011).(in Chinese)

[7] Y. F. Sun. *A hybrid strategy based on genetic algorithm and tabu search*[J].Journal of Beijing University of Technology,32(3),pp:258-262(2006).(in Chinese)

[8] J. Zhao , J. S Kong. *Hybrid optimzation strategy based on genetic algorithm and tabu search*[J]. Computer Engineering and Design.30(23),pp:5489-5491(2009).(in Chinese)

[9] F. Glover. *Tabu search—Part I*. ORSA Journal on Computing,13(1),pp.190–206(1989).