International Journal of Soft Computing 11 (3): 136-139, 2016
ISSN: 1816-9503
© Medwell Journals, 2016

Ant Colony Optimization for Multiversion Software Synthesis

[1]Roman Yurievich Tsarev, [2]Igor Vladimirivich Kovalev, [1]Eugeny Valerievich Soloviev,
[1]Alexander Vladimirovich Prokopenko and [3]Alexey Nikolaevich Knyazkov
[1]Department of Informatics, Siberian Federal University, 26, ul. Kirenskogo,
ULK, 660074 Krasnoyarsk, Russia
[2]Department of System Analysis and Operation Research,
Siberian State Aerospace University, Russia
[3]Department of Transport, Siberian Federal University, Russia

Abstract: High reliability of the software can be achieved by the use of N-version or multiversion programming. In this case multiversion software modules are realized by means of several functionally equivalent versions developed to solve the same tasks. Synthesis of multiversion software based on the introduction of software redundancy with limited resources is always the optimization problem. Solution of this problem, as a rule, is a compromise between numbers of criteria. The research propose the method based on the use of ant algorithm for design of optimal multiversion software. The study presents an example of synthesis of multiversion software without redundancy. Although, there are several versions for the implementation of each software module.

Key words: Optimization, ant colony, multiversion software, resources, design

INTRODUCTION

Software reliability is considered to be one of the most important problems of safe operation of control and information processing systems in many areas of science and production (Ravichandra and Ramani, 2014). The software reliability risk needs to be improved due to significant material losses and possible danger for human's life in the case of insufficient reliability of critical software modules in control and information processing systems (Golubev *et al*., 2005). Now a days, a huge number of methodologies for the protection of the automatic systems from physical nature error, i.e. is a hardware level error, are developed.

Multiversion programming methodology as an approach to designing of fault-tolerant software systems allows raising the reliability of the system (Xie *et al*., 2014). Multiversion software composition as one of the stages of software designing determines the overall reliability parameters at the design phase (Terada and Ushio, 2010). The relatively low efficiency of used methods of the multiversion software optimal structure composition and the high complexity of the tasks along with a high practical significance resulted in the urgency of developing new methods of software design.

It was previously suggested different methods solving the problem of optimal composition of multiversion software. These are the method of screening variants and the technique for order preference by similarity to ideal situation (Behzadian *et al*., 2012). The method of screening variants proposes the presentation of the solution process as a multi-stage structure, where each stage is associated with checking the availability of certain properties of a variants subset. In case of large-scale problems it can be very complicated process.

When using the TOPSIS method is often necessary clarify, in what sequence to produce a consistent optimization criteria (Huang, 2008). Even in case of three criteria must be considered two variants. And for n criteria there are $2^{(n-1)}$ such variants. To determine which of them will be the best result is often very difficult. In addition, to find reference points used in this method, the method of full enumeration is applied. The proposed method of selection of the optimal composition of multiversion software using ant algorithms devoid of the above-mentioned shortcomings or their effect can be reduced.

MATERIALS AND METHODS

Ant colony algorithm application: The main idea of the ant algorithm consists in modeling the behavior of ant colony associated with their ability to find quickly the

Corresponding Author: Roman Yurievich Tsarev, Department of Informatics, Siberian Federal University, 26, ul. Kirenskogo, ULK, 660074 Krasnoyarsk, Russia

shortest path from nest to a food source and adapt to changing conditions, finding a new shortest path (Blum, 2005). In its motion the way is marked by ant pheromone and this information is used by other ants to choose path (Ozbakir *et al*., 2011). This is an elementary rule of conduct that determines the ability of ants to find a new shortest path, if the old one is not available (Jovanovic and Tuba, 2013).

The apparent positive feedback leads to choose the best way as the only route of movement of most ants (Chen and Zhang, 2013). Modeling evaporation of the pheromone-a negative feedback-ensures that the obtained locally optimal solution is not unique, as far as the ants will search the other ways (Crawford *et al*., 2014; Dorigo and Blum, 2005).

Probability-proportional rule used in ant algorithms determines in this case the probability of choosing by ant k the version j of the module i to iteration t. In view of what is required to optimize the result of both the cost and the reliability the rule takes the following form:

$$P_{ij,k}(t)=\frac{\left[t_{ij}(t)\right]^{\alpha}\left[\frac{1}{C_{ij}}\right]^{\beta}\left[R_{ij}\right]}{\sum_{l=1}^{m_i}\left[t_{il}(t)\right]^{\alpha}\left[\frac{1}{C_{il}}\right]^{\beta}\left[R_{il}\right]}$$

Where:

- τ = The artificial pheromone value (Srikanth *et al*., 2013)
- c_{ij} = The cost of the version j of the module i
- R_{ij} = The reliability of the version j of the module i
- α = adjustable parameter that specifies the weight of the pheromone trail, β and
- φ = parameters that can change the solution or in the direction of reducing the cost, either in the direction of improving the reliability
- m_i = The number of versions of the module i

It is necessary to have the compromise between the adjustable parameters in order to mitigate the greedy algorithm and do not get stuck in local minima. Note that probability-proportional rule only determines the probability of choosing one or another version. Proper choice of the version performed on a «roulette wheel». Each available version has its own sector with an area proportional to the probability. To select the version we need to throw the ball on the roulette (generate a random number) and to identify sectors where the ball stopped.

Pheromone update rule in this case, taking into account the optimization of cost and reliability, takes the following form:

$$t_{ij}(t+1)=(1-\rho)t_{ij}(t)+$$
$$+K_{ij}\left(\frac{R_{ij}}{\max(R_{il},l=\overline{1,m_i})}\right)\left(\frac{\min(C_{il},l=\overline{1,m_i})}{C_{ij}}\right)$$

Where:

- ρ = The pheromone evaporation intensity (Abduljabbar *et al*., 2013)
- K_{ij} = The number of ants that chose the version *j* of the module i
- m_i = The number of versions of the module i

At the initial stage, the level of pheromone is initialized to a small positive number so that at the initial step the probabilities of transition to the next version were not zero.

RESULTS AND DISCUSSION

Selection of the best versions of the modules is shown in the example of single-functional model of a software system without redundancy. The program consists of a set of modules. The program structure is shown in Fig. 1.

It is available more than one version of each module, but because of the severe restrictions on the cost (and this case should not be excluded from consideration) or non-critical parts or whole the software, saving of multiversions of the modules is undesirable. The model developed for this situation can choose an optimal set of software module versions optimizing reliability and cost, by which the total cost of development is acceptable and the project is realizable. Thus, the software system consists of eight modules. For each module were originally available three versions. Initial data on the cost

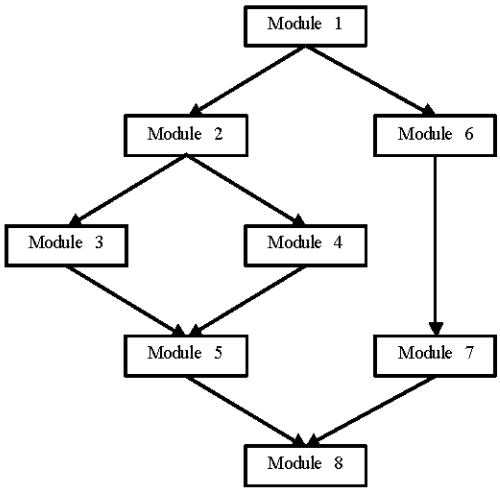

Fig. 1: Structure of multiversion software

Table 1: Cost and reliability of versions

Modules	Version	Cost	Reliability
1	1	34	0.9970
	2	43	0.9978
	3	46	0.9980
2	1	100	0.9994
	2	110	0.9995
	3	125	0.9998
3	1	56	0.9988
	2	58	0.9990
	3	70	0.9995
4	1	67	0.9980
	2	69	0.9985
	3	75	0.9992
5	1	90	0.9990
	2	94	0.9994
	3	99	0.9999
6	1	400	0.9987
	2	445	0.9992
	3	470	0.9998
7	1	156	0.9988
	2	160	0.9991
	3	169	0.9995
8	1	78	0.9980
	2	90	0.9990
	3	113	0.9996

Table 2: Selected versions of multiversion software.

Module	Version	Cost	Reliability
1	1	34	0.9970
2	2	110	0.9995
3	3	70	0.9995
4	1	67	0.9980
5	3	99	0.9999
6	2	445	0.9992
7	3	169	0.9995
8	1	78	0.9980

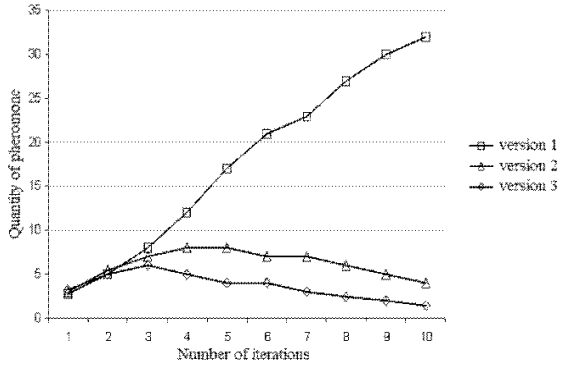

Fig. 2: Changing of pheromone quantity while determining the optimum version of module 1

and reliability for each version of each module are shown in Table 1. The results of algorithm choosing the optimal versions of the module 1 and module 2 are shown at Fig. 2 and 3. As can be seen, even if the number of pheromone differs slightly in the initial iterations, the optimal version clearly stands out after several iterations. Similar results were obtained for other modules.

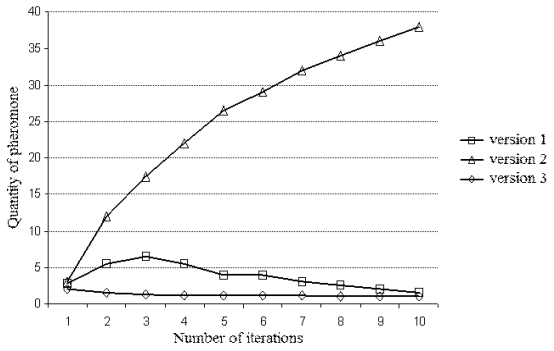

Fig. 3: Changing of pheromone quantity while determining the optimum version of module 2

CONCLUSION

Selected versions of all modules are shown in Table 2. Note that initially there were three versions for each module. As the result we have got the multiversion software without redundancy. Thus, the example shows that using the proposed method based on the use of ant algorithm for selection of the optimal multiversion software we can effectively solve posed optimization problem.

REFERENCES

Abduljabbar, Z.A., M.S. Khalefa and M.A. Jabar, 2013. Comparison between ant colony and genetic algorithm using traveling salesman problem. Int. J. Soft Comput., 8: 171-174.

Behzadian, M., S.K. Otaghsara, M. Yazdani and J. Ignatius, 2012. A state-of the-art survey of TOPSIS applications. Expert Syst. Applic., 39: 13051-13069.

Blum, C., 2005. Ant colony optimization: Introduction and recent trends. Phys. Life Rev. J., 2: 353-373.

Chen, W.N. and J. Zhang, 2013. Ant colony optimization for software project scheduling and staffing with an event-based scheduler. IEEE Trans. Software Eng., 39: 1-17.

Crawford, B., R. Soto, F. Johnson, E. Monfroy and F. Paredes, 2014. A max-min ant system algorithm to solve the software project scheduling problem. Ex. Syst. Appl., 41: 6634-6645.

Dorigo, M. and C. Blum, 2005. Ant colony optimization theory: A survey. Theor. Comput. Sci., 344: 243-278.

Golubev, I.M., R.J. Tsarev and T.I. Semenko, 2005. N-version software systems design. Proceedings of the 11th International Scientific and Practical Conference of Students, Post-graduates and Young Scientists Modern Technique and Technologies MTT 2005, March 21-27, 2005, IEEE, Tomsk, Russia, ISBN: 978-0-7803-8877-2, pp: 147-149.

Huang, J., 2008. Combining entropy weight and TOPSIS method for information system selection. Proceedings of the 2008 IEEE Conference on Cybernetics and Intelligent Systems, September 21-24, 2008, IEEE, Chengdu, China, ISBN: 978-1-4244-1673-8, pp: 1281-1284.

Jovanovic, R. and M. Tuba, 2013. Ant colony optimization algorithm with pheromone correction strategy for the minimum connected dominating set problem. Comput. Sci. Inf. Syst., 10: 133-149.

Ozbakir, L., A. Baykasoglu, B. Gorkemli and L. Gorkemli, 2011. Multiple-colony ant algorithm for parallel assembly line balancing problem. Appl. Soft Comput., 11: 3186-3198.

Ravichandra, M. and A.V. Ramani, 2014. Measuring software reliability using fuzzy logic. Int. J. Soft Comput., 9: 314-317.

Srikanth, G.U., V.U. Maheswari, A.P. Shanthi and A. Siromoney, 2013. Scheduling of real time tasks using ant colony optimisation. Int. J. Soft Comput., 8: 50-55.

Terada, S. and T. Ushio, 2010. Optimal configuration for multiversion real-time systems using slack based schedulability. IEICE. Trans. Fundam. Electron. Commun. Comput. Sci., 93: 2709-2716.

Xie, M., C. Xiong and S.H. Ng, 2014. A study of N-version programming and its impact on software availability. Int. J. Syst. Sci., 45: 2145-2157.