Send Orders for Reprints to reprints@benthamscience.ae

1094 *The Open Cybernetics & Systemics Journal,* 2015, *9,* 1094-1098

Open Access

Brittleness Analysis of Software Architecture based on Ant Colony Optimization

Hong Zhang[*,1], Changzhen Hu[2] and Xiaojun Wang[1]

[1]*School of Computer Science & Technology, Beijing Institute of Technology, Beijing, 100081, P.R. China;* [2]*School of Software, Beijing Institute of Technology, Beijing, 100081, P.R. China*

Abstract: The term software architecture (SA) intuitively denotes the high level structures of a software system. It can be defined as the set of structures needed to reason about the software system, which comprise the software components and connectors, their relations and properties. Nowadays SA has become an important factor in the process of software development and the researches mainly focus on the languages, modeling, dynamic evolution process, etc. In this article, the complex system and brittleness theory are applied into the field of SA, and the concept of brittleness graph and collapse path of SA is introduced into the analytical process. Ant colony algorithm is used for simulation. The results of simulation demonstrate that Ant colony optimization (ACO) performs well on finding out the max collapse route of the brittleness graph of SA.

Keywords: Ant colony algorithm, brittleness analysis, complex system, software architecture, software security.

1. INTRODUCTION

Software systems, as independent entities, have drawn more and more attentions in academia and industrial field. The security, reliability, usability and maintainability all have evaluated the performance of software systems from different perspectives.

The software crisis in the 1970s diverts peoples' focus from data structure and algorithms to the design of module and SA in larger-granularity. With the mature techniques, as a means of improving the quality of software, supporting the development and reusability of software, the research on SA has gained fruitful achievements. This accomplishment covers the early design phase extending to other phases of the software development life cycle, such as requirement phase, realization phase, deployment phase and post-development phase. SA styles [1] define idiomatic patterns of system organization, which help the system architects to construct the software system, specifically the styles define the terminology of the elements, the relevant configuration rules, the semantic interpret contents and the system analysis. When choosing the styles in software system designing, it is important to fully consider the characteristics of software system and its styles. Some common styles are listed in Table **1**. Current researches on SA focus on the following aspects:

1.1. Description of Software Architecture

The description aspect involves in what way and by what means to describe SA. There are mainly two aspects of the description.

(1) Software architecture languages such as UniCon, Rapide, Darwin, Aesop, C2, Acme are all description languages, each has its own features in describing SA [2].

(2) Software architecture views: Comprehensive SA views help the system architects a lot in sharing techniques, improving the quality of software system, enhancing the effciency in developing, and better views also illustrate the principle of separation of concerns. The "4+1" view model from Kruchten integrates the logical view, the development view, the process view and physical view through the senario. Views and Beyond Model from CMU-SEI includes the module view, the component-connector view and the dispatch view. In [3] the authors give a detailed summary on the multi-view representation of SA.

1.2. Design of Software Architecture

The design of SA refers to the process that meets the fixed tactics and styles according to some functional and non functional requirements. One of the designing goals is to repeatedly use the architecture styles to support the software reuse and to study the model representation in earlier stage, the analysis, identification, evaluation in mid-term, and the experience conclusions of designing in later stage.

1.3. Analysis and Evaluation of Software Architecture

Giving explicit analysis and evaluation help in checking the validity of SA model in the design phase, finding out the hidden defect and revising them in time, which guarantees the software will run normally. The evaluation of SA mostly concerns the quality attribute, which covers the performance, the security, the reliability and the usability. There are three main qualitative evaluation methods, which are based on the questionnaire or check table, based on the scenario and based on the metric respectively.

1874-110X/15

Table 1. Result of experiments and theory.

Dataflow system	**Virtual machines**
Batch sequential	Interpreters
Pipes and filters	Rule-based systems
Call-and-return systems	**Data-centered systems**
Main program and subroutine	Databases
OO systems	Hypertext systems
Hierarchical layers	Blackboards
Independent components	
Communicating process	
Event systems	

1.4. Dynamic Evolution and Reusability of Software Architecture

The dynamic evolution [4] and static scalability both cover the research category of SA. The dynamic evolution includes three aspects: the interactive dynamic, the structural dynamic and the architectural dynamic. Nowadays software systems demand for support not only in the designing phase but also in the running environment. The research on dynamic architecture includes several languages, which support the structural description and the tools supporting the dynamic evolution process. The dynamic structure can accept or deny the changes inside or outside the system according the predefined policies and adjust itself to the changing situation [2, 5].

The reusability of SA has a larger-granularity than that of the code, module and component. It can also effectively reduce the cost in software development and maintenance; to improve the quality in developing process. The above contents are about past research on SA, however, in this article we do the analysis of SA from a different perspective of view, putting the brittleness concept into the analysis of SA in combination with the complex system. In best of our knowledge, the notion of complex system and brittleness on SA is the first time introduced in this article.

2. COMPLEX SYSTEM AND BRITTLENESS THEORY

The research on complex system started from 1970s, which was composed of dissipational theory [6], synergetics [7] and catastrophe [8], and most of the study was on the system state from disorder to order and evolution from low-level to high-level. Open complex giant system [9], is characterized by its openness, complexity, hierarchy and emergent property in evolution process, has a close connection with synergetic, catastrophe, reliability and stability. According to this notion, the systems can be classified into four categories which are simple systems, simple giant systems, the complex giant systems and special complex giant systems.

In [10] the author gave the idea of judging criteria of a complex system in the doctoral dissertation. The notion of brittleness was introduced by [11], which is mainly about the system's sensitive behaviour showed in internal or external complicated environment with uncertainty. Brittleness refers to the characteristic possessed in a part or a subsystem Si of system S, which has a strong sensitivity to the environment, that is, once disturbed or attacked, Si will breakdown and lead to a chain reaction, resulting in global collapse. Thus, Si is called the brittleness source.

The brittleness possessed in a complex system can be described by "self-organized criticality" (SOC), which was proposed by P Bak. in Physical Review Letters in 1987 [12] and was described by using the sandpile model (See Fig. **1**).

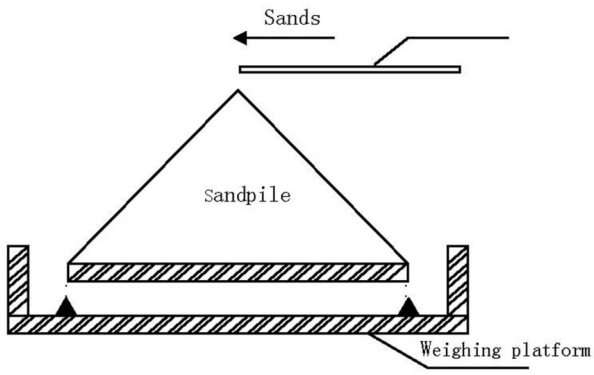

Fig. (1) The sandpile model.

Sandpile model is described as follows: drop the sands with constant speed and altitude to a platform. As time goes on, the sands fell down on the platform gradually form a sandpile. There exists one moment after which when adding a grain of sand will result in sandpile collapsing. With the help of slow-motion camera, P Bak. can calculate how many grains of sand can be affected when dropping one grain of sand at top of sandpile. Sandpile can be looked as a non-linear system with a persistent energy supply. The scale of sandpile collapse has a typical power function distribution relationship with the collapse frequency. SOC is considered as the dynamic cause which leads to power law distribution and the power law is known as the evidence of complex system having SOC.

Like a specific product which satisfies the needs of human beings, software system also meets the judging criteria of complex system. During operation, the software system changes from one state to another, being constantly affected from internal or external environment; when the integrated influence put on it equals to the load it can bare, the software system reaches a "self-organized criticality" state, under that circumstance the software system will lead to a chain breakdown and result in a global collapse if affected by any further disturbance. This process shows the brittleness characteristic of the software system and the part which leads to the global collapse is called the brittleness source.

3. APPLICATION OF BRITTLENESS ON SOFTWARE ARCHITECTURE

A brief introduction of some relevant terms on brittleness graph will be given in order to give a detailed description of the application of brittleness on SA. Brittleness graph G of SA:

$$G = \{V, E, D\} \quad (1)$$

Vertex set V is composed of components and connectors, the edge-set E is composed of the pairs of components and

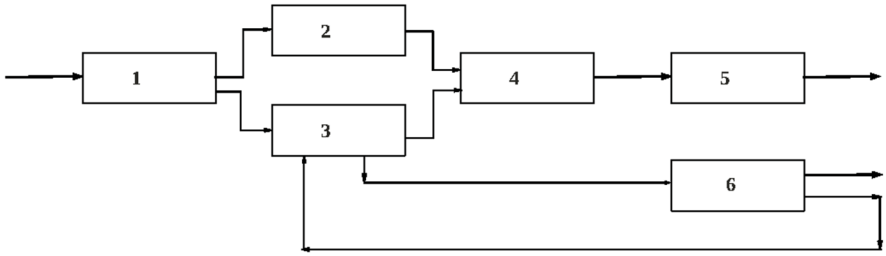

Fig. (2) Pipes and filters.

connectors which have direct relations. D represents the relations value set which is composed of brittleness value, represented with {rij , i, j ∈ V, i, j ≥ 1}. Brittleness Value: It represents the strong or weak relations among each two vertexes those have direct brittleness relations. Collapse Path: It is also called Hamilton path in brittleness graph of SA. Supposing that G = {V, E, D} is the brittleness graph of SA, if in the process of the software system collapses, there exist H and has following two equations:

$$V(H) = V(D) \tag{2}$$

$$E(H) = E(D) \tag{3}$$

then the directed path H which goes through each vertex once and only once is called a collapse path of G, or Hamilton brittleness path. Max Collapse Path: It is defined as a Hamilton path which has the max brittleness product among all the product values. Brittleness Source: It refers to the component or the connector which can lead to the system collapse, and it can be looked as the starting point of the collapse path in brittleness graph of SA.

4. BRITTLENESS SIMULATION ON SOFTWARE ARCHITECTURE

In section 1, a brief introduction on the common SA styles is given. In order to illustrate the application of brittleness theory on SA, we choose the pipes and filters as an example. This style is applied on a predened independent computing in ordered data in which lters achieve related functions and the pipes are used in the data input, output and transition. Fig. (**2**) depicts a typical layout of this type. The modeling procedure and simulation process are given as follows:

4.1. Transition from Software Architecture to Brittleness Graph

On the basis of Fig. (**2**), we put the topology of this style into the coordinate plane and change it into the corresponding directed graph, in which the vertexes represent the filters and pipes and the edges represent the brittleness relations between filters and pipes. We can get coordinates in the figure for each node, which are represented by Matrix C:

$$C = \begin{pmatrix} 10 & 30 \\ 20 & 40 \\ 20 & 20 \\ 30 & 30 \\ 40 & 30 \\ 40 & 10 \end{pmatrix}$$

The edge-set is {(1, 2), (1, 3), (2, 4), (3, 4), (4, 5), (3, 6), (6, 3)}. Supposing after Delphi Method which is a structured communication technique relying on a panel of experts, the brittleness values are given to each edge represented in set D:

$$D = \begin{bmatrix} \varepsilon & 0.8 & 0.6 & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & 0.3 & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & 0.6 & \varepsilon & 0.8 \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & 0.8 & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 0.2 & \varepsilon & \varepsilon & \varepsilon \end{bmatrix}$$

4.2. Simulation

Based on the brittleness graph made from above procedures, we adopt the ant colony optimization algorithm to find out the max collapse path of software system. As one of the mathematical optimization algorithms, the ant colony optimization algorithm was proposed by M. Dorigo in his doctoral dissertation, which effectively solved the Travelling Salesman Problem(TSP) and Job scheduling Problem. The ant has the ability to find out the shortest path between food and ants' nest without any tips, no matter how the environment changes. The ability depends on the pheromone released on the path it goes through, and the subsequent ants can perceive the existance and strength of the pheromone. As a result, this process can form a positive feedback. Ant Colony Optimization can be classified into three types according to the ways of the pheromone updates in one cycle, which are antcycle system, ant-quantity system and ant-density system.

The algorithm in our experiment is antcycle system, the parameters are NCmax = 100, m=6, α = 1, β = 5, ρ =0.7, Q=100, respectively represent the max iteration, the number of ants, the importance of the pheromone, the importance of heuristic factor, pheromone evaporation coefficient and intensity coefficient of pheromone increment. Fig. (**3**) shows the result of the simulation. Furthermore, the number of ants, the iteration times and other parameters are changed, that is, NCmax = 3000, m=20, α = 1, β = 2, ρ =0.1, a new result is shown in Fig. (**4**).

4.3. Analysis of Simulation

There are some aspects need to explain:

(1) selection of algorithm Antcycle system is chosen to simulate and evaluate the performance of path selection after a comprehensive study on the three models. As for this

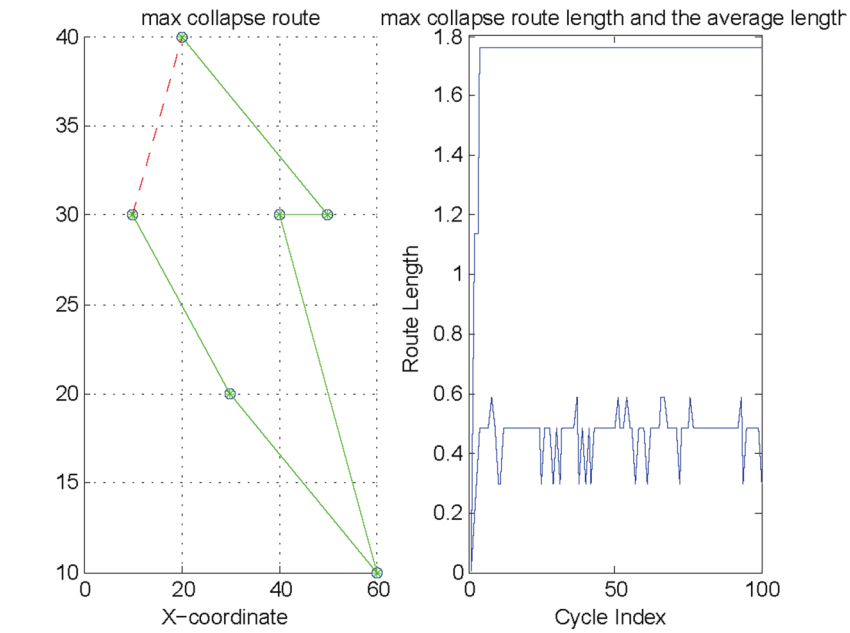

Fig. (3). Ant colony optimizaion simulation.

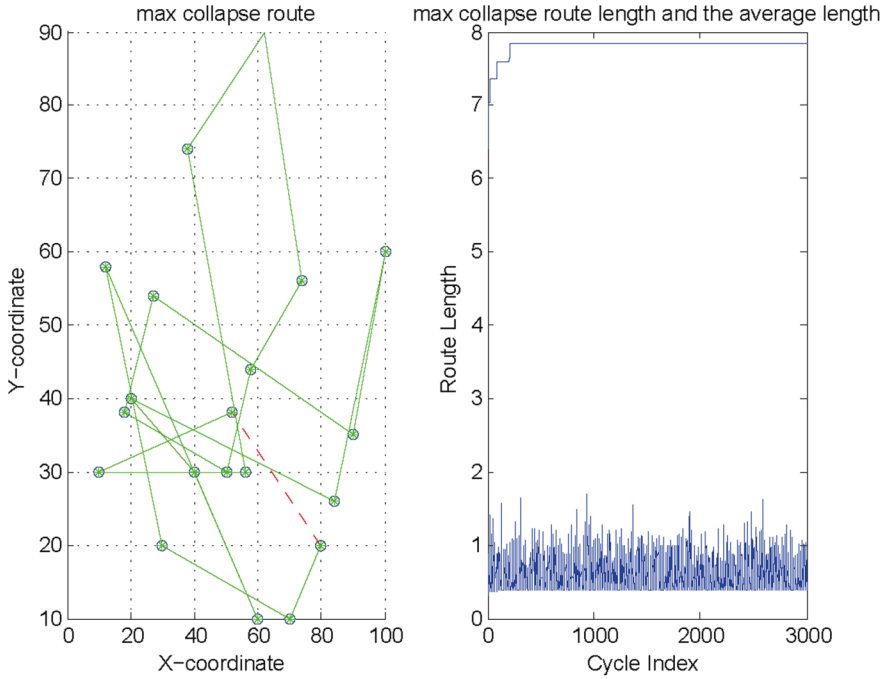

Fig. (4). Result with parameters changed comparing to Fig. (**3**).

model, the probability of the ant choosing a path mainly depends on the importance of the pheromone α and the importance of heuristic factor β. The optimal value are these: α is about 1, β is from 5 to 10, and ρ is around 0.7.

(2) The left side of both figures show the max collapse path under different parameters, the starting point and terminal point are directly connected by dotted lines. The max collapse path length at right side comes to a stable state at very early time. Both of those are conformed as the antcycle system model.

(3) The antcycle algorithm can avoid numerous invalid searches, so it can quickly find out a good solution but not in a stagnation state. This is advantageous in finding the max collapse path.

(4) Comparing the two figures, when we increase the ants, iteration times and other parameters the algorithm is robust to the changing parameters. So the algorithm is suitable for our problem resolving.

CONCLUSION

Brittleness gives us a new perspective to analyse the software system, which is treated as a complex system. The brittleness graph is transformed from the software architectural topology on the basis of the analytical process with

experts' experience. Through simulation with the ant colony optimization algorithm the brittleness source and the global collapse path of SA can be found. We should also focus on the following aspects in the future research:

(1) Developing practical tools in finding out the brittleness source and collapse path of software system It is necessary to develop a brittleness analytical tool which can help the system architect and analyst to locate the brittleness source and the max collapse path automatically.

(2) Taking effective measures after finding out the brittleness source and the max collapse path The brittleness source and max collapse path of software system based on the brittleness graph can be obtained, but how to effectively prevent the system from collapse? This is a big challenge.

(3) Putting the brittleness analysis into design phase The collapse path of software system got from the simulation should be feedback to the design phase, based on which we can combine the notion of identity, difference and antagonism in Set-Pair Analysis (SPA) with the brittleness analysis to guide the designing of SA in early phase.

CONFLICT OF INTEREST

The author confirms that this article content has no conflict of interest.

ACKNOWLEDGEMENTS

Declared none.

REFERENCES

[1] D. Garlan, R. Allen, and J. Ockerbloom. "Exploiting style in architectural design environments", *ACM SIGSOFT Software Engineering Notes*, vol. 19, pp. 175-188, Dec. 1994.

[2] N. Medvidovic and R. N. Taylor, "A classification and comparison framework for software architecture description languages", *IEEE Transactions on Software Engineering*, vol. 26, pp. 70-93, Jan. 2000.

[3] P. Clements, D. Garlan, R. Little, Robert Nord, and J. Stafford, Documenting software architectures: views and beyond, NJ: Pearson, 2003, pp. 740-741.

[4] J. Cámara, P. Correia, R. D. Lemos, G. David P. Gomes, B. Schmerl, and R. Ventura, "Evolving an adaptive industrial software system to use architecture-based self-adaptation", SEAMS '13 Proceedings of the 8th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, San Francisco, CA, 2013, pp. 13-22.

[5] S. Kang and D. Garlan, "Architecture-Based Planning of Software Evolution", *International Journal of Software Engineering and Knowledge Engineering*, vol. 24, pp. 211-242, Nov. 2013.

[6] I. Prigogine and G. Nicolis, Self organization in non-equilibrium systems, NY: Willey, 1977, pp. 1-15.

[7] H. Haken, Synergetik, NY: Springer, 1983, pp. 1-44.

[8] R. Thom, Stabilité structurelle et morphogénèse, NY: Benjamin, 1972, pp. 1-20.

[9] H. S. Tsien, J. Y. Yu and R. W. Dai, "A new discipline of science-the study of open complex giant system and its methodology", *Chinese Journal of Nature*, vol. 13, pp. 3-10, 1990.

[10] Q. Wei, "*Brittleness Theory of Complex System and its Application to Crisis Analysis*", PhD thesis, Harbin Engineering University, Harbin, China, 2004.

[11] A. A. Fouad, Z. Qin, and V. Vittal, "System vulnerability as a concept to assess power system dynamic security", *IEEE Transactions on Power Systems*, vol. 9, pp. 1009-1015, May 1994.

[12] P. Bak, C. Tang, K. Wiesenfeld, "Self-organized criticality: An explanation of 1/f noise", *Physical Review Letters*, vol. 59, pp. 381-384, Mar. 1987.

Received: June 10, 2015 Revised: July 29, 2015 Accepted: August 15, 2015