Evaluation of Software Consistency for Component Based System Through Soft Computing Technique

Ravi Kumar Sharma
Research Scholar
Manav Rachna International University
Faridabad, India
E-mail: ravirasotra@yahoo.com

Parul Gandhi
Assistant Professor
Manav Rachna International University
Faridabad, India
E-mail: parul.fca@mriu.edu.in

Abstract: Many algorithms and techniques have been developed for estimating the reliability of component-based software systems (CBSSs), but still there is a need to perform vast research in this field. It is very difficult to estimate the reliability of a CBSS accurately due to the involvement of two factors: component reliability and glue code reliability. Moreover, real time problems are directly related to real world phenomenon i.e. reliability. There are many Soft computing techniques to solve such problems with uncertain solutions. Many proposed techniques, learn from the past and capture existing patterns in data. Neural networks and fuzzy logic are two basic elements of soft computing. In this paper, we are using a model for estimating CBSS consistency, identified as an adaptive Neuro fuzzy inference structure (ANFIS), that is based on above mentioned elements of soft computing, and comparative study is performed on its results with that of a plain FIS (fuzzy inference system) for different data sets.

Keywords: *Component-based Software systems, Fuzzy, Neuro fuzzy, Reliability model.*

I. INTRODUCTION

Software generally has two customer requirements: reliability and availability. Reliability is required when the product's default will have the greatest impact, while availability is required when interruption will have the greatest impact. Although it is difficult to formally define reliability. Precisely, software reliability is well-defined as a software structure's probability of failure-free process for the indicated period of time in a specified environment. As the difficulty of software applications lasts to grow, greater stress has been placed on reuse. Therefore, component-based software system (CBSS) applications have come into reality. Component-based software engineering (CBSE) is an [4] absorbed form of software reuse troubled with building software from standing components, including commercial off-the-shelf (COTS) components, by gathering them together in an interoperable method. Succeeding a highly consistent software application is a challenging task, even when high-quality, protested, and trustworthy software components are combined. More than a few techniques have therefore arisen to analyze the reliability of component-based applications. This collapse into two groups:

- System-level reliability estimation: reliability is expectable for the application as a complete.
- Component-based reliability estimation: application reliability is predictable founded on the reliability of the separate components and their interconnection tools.

Old-style methods to the software reliability investigation treat the software as entire and use the test information during the software investigation phase to model only the software's connections with the outside world. These are recognized as black-box prototypes. Though, black-box prototypes ignore the arrangement of software assembled from components as well as the reliability of separate components and are so not suitable for demonstrating CBSS presentations. Lately, soft computing methods have developed. Since reliability is a real-world problem, several run-time constraints related with reliability. The soft computing methodsultimate for estimating CBSS reliability, as these methods are deal mostly with uncertainity. The basic soft computing methods are neural networks and fuzzy logic. In this paper we relate these two methods to estimate CBSS reliability.

In a fuzzy inference system (FIS), the if-then rules are expressed on the basis of knowledgeable instruction. Still, this is a fairly time unbearable method. The improvement of an ANFIS above the FIS model is that it combines fuzzy logic with the knowledge, abilities of neural networks (NNs) to solve this challenging. The purpose of our planned faultless is to integrate the best structures of fuzzy logic and neural networks in a CBSS reliability estimation model. Our outcomes show how this is an improvement over an FIS.

II. LITERATURE SURVEY

Fuzzy logic has been broadly calculated in altered fields of engineering with changing radiation of achievement[5]. An FIS frequently contains if-then guidelines. These guidelines are fired in equivalent: when if- situations are met, the consequents of those guidelines fire to the step to which the antecedents of the guidelines are met. One and only of the main tasks in making an FIS is responsible the fuzzy sets and fuzzy guidelines to be recovered. The fuzzy sets and guidelines can be actual stage time consuming. The explanation of this difficulty is to chain an FIS with the learning capabilities of NNs, resultant in an ANFIS. The future model is an improvement over the fuzzy model.

A. Neural Networks and Fuzzy Logic

NNs plus fuzzy logic are matching concepts. NNs have knowledge learning capabilities: they can absorb from facts and response. They are black-box prototypes. Fuzzy logic representations are rule-based [9] prototypes; everywhere the guidelines frequently have an if-then procedure. Fuzzy prototypes do not have knowledge, capabilities, so for knowledge, they necessity adopts procedures from additional methods. It is consequently accepted to marry the two tools. Fuzzy logic is based on the mishmash of four models shown in Fig. 1.

B. ANFIS

ANFISs were principal recommended [2] in 1992. These are adaptive systems whose determination is like to FISs. The goal of an ANFIS is to fit the best structures of fuzzy structures and NNs. The benefits of an ANFIS concluded an FIS are as surveys:

- An ANFIS can simulate and examine the charting relation between contribution and production data through a learning algorithm to improve the parameters of an agreed FIS.

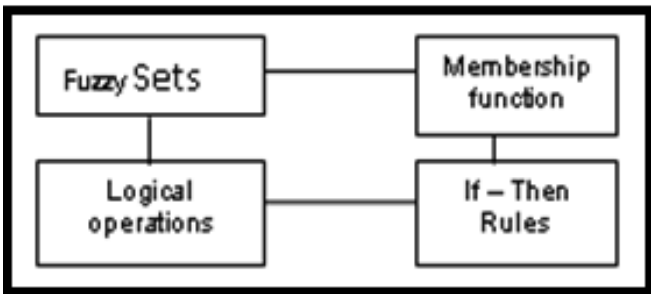

Fig. 1: Working of fuzzy system.

- An FIS is fully in need of on its related functions. An ANFIS includes systems which combine nodes and indicator links, and some knowledge instructions are related to these systems. The systems learn relationships between contributions and productions.
- An ANFIS can be qualified without any essential for the skilled information, usually compulsory for the normal fuzzy logic strategy.

III. FEATURE SELECTION AND ANALYSIS

In this paper, we recommend computerized model for CBSS reliability approximation that uses the ANFIS method. An ANFIS is a lesson of adaptive networks, which are functionally comparable to FISs. This stands an extension lead of the fuzzy model [1]. We yield into deliberation all four features future in that prototypical to equate the results of both models.

The four features scheduled in [1] are as surveys. There are two main issues for assessing component reliability, the reusability of the component and its operational profile.

A. Reusability of the component

Reusability is unique of the greatest basic representations of component-based development (CBD). As the term recommends, reusability denotes for how often a component is used in different applications. We nominated reusability as an issue of valuing component reliability because it is held that components that have been used in numerous applications are extremely reliable. Hence the consistency of a factor is directly relative to its reusability: component reliability ∝ reusability.

B. Operational profile for the component

The operational profile (OP), a measurable description of exactly how the software determination be used, is important in any software reliability industrial application. An OP is a whole set of processes with their possibilitiesof existence. The reliability may be dissimilar for changed OPs. The OP for any factor describes the responses provided to the factor.Extra [7] factors may donate to component reliability, but these are the two issues that have the highest influence on a component's reliability. As the components are stage self-determining, the actions of these two issues will be the similar for all applications.

C. Component dependency

In a CBSS, various mechanisms are related to one additional to form a greater application. The mechanisms are independent of each other to achieve [10] their functions, that is, the production of one component may help as acontribution for additionalcomponent. This dependency plays an important role for approximating the reliability of the complete application. If components are highly helpless on one another, the reliability of the system will be low.There are numerous mechanisms for describingdependencies between components. For example, an adjacency matrix illustration of dependencies specifies which dependencies exist between components. However, this representation does not explain the exchanges between components. These exchanges play an important role in relation to the dependencies of components and sotheir reliability, and so we want a different way to characterize the interactions. [3] A planned dependency illustration which is based on linked listsandexecuted using Hash Map in Java. This illustration can store dependencies along with other information like delivered and required information, which can be used to evaluate several interface- and dependency-related issues.

D. Application complexity

The complication of any CBSS application can be exact in terms of the amount of mechanisms in that request and their interconnectivity. Arequest that has extra components is said to be more difficult and hence less dependable. Therefore, the difficulty of an application is inversely proportional to its reliability:

Application difficulty can be distinguished in terms of the amount of components in the application. Where N is the amount of components in a CBSS, we [5] define the application difficulty AC. Our planned model is an additive model. It does not clearly take the design of the CBSS into reflection, but the amount of application complexity will be platform helpless because it comes from the attach code (addition code).

The FIS calculated in this work has four input issues as described above, namely, reusability, application complexity, component dependency, and operational pro-file. For a first-order Sugeno fuzzy prototypical, a concept shown in the Figure 2 guideline set with fuzzy if-then guidelines is as follows:

Rules:

(1) R = reusability
(2) CD = component dependency
(3) AC = application complexity
(4) OP = operational profile

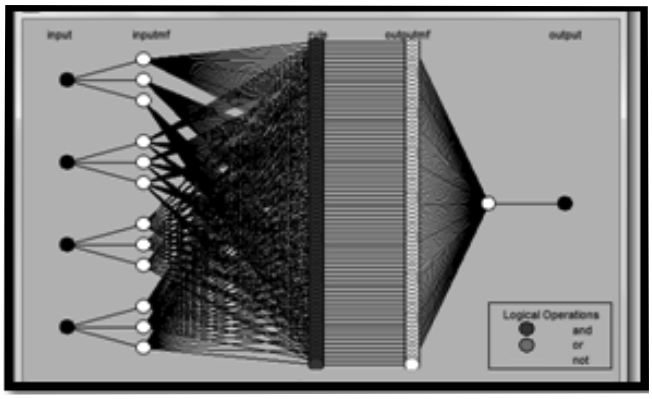

Fig. 2: Proposed architecture of the system.

One of the most important limitations of this ANFIS is that it uses only Sugeno-style fuzzy reading systems. The dissimilarity between Sugeno [6] and Mamdani inference machines lies in the defuzzificationtechnique. A Mamdani-type FIS uses the defuzzificationprocess for fuzzy outputs, and a Sugeno-type FIS uses a one-sided average to compute hard outputs. A Mamdani FIS also has output related functions, while a Sugeno FIS doesn't have output relationship functions. We calculated our ANFIS for a Sugeno FIS. We composed data from 47 classroom-based projects. Some of these schemes are difficult CBSSs and some are simple CBSSs. The ANFIS was expert and experienced using the data gained from these projects. Some of the data were used for preparation and challenging, and some were used for the results. A total of 81 guidelines was twisted on the basis of the

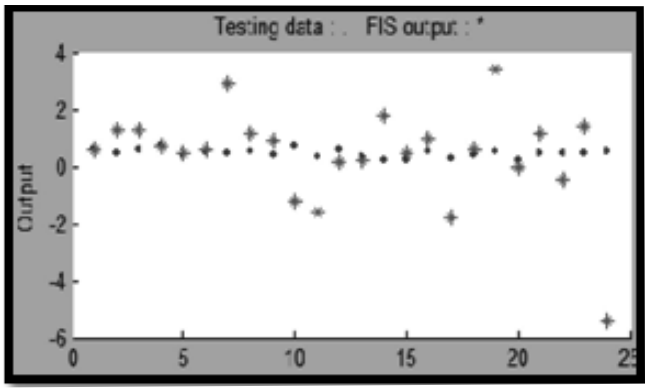

Fig. 3: Testing error mapping sugeno FIS to ANFIS.

existing data. The guidelines divide the input issues into three variables: low, medium, and high. To evaluate our recommended [8] model, the preparation, data set was loaded, and the ANFIS was accomplished using that data set. To check the model, we loaded the testing data set and then calculated the testing error.

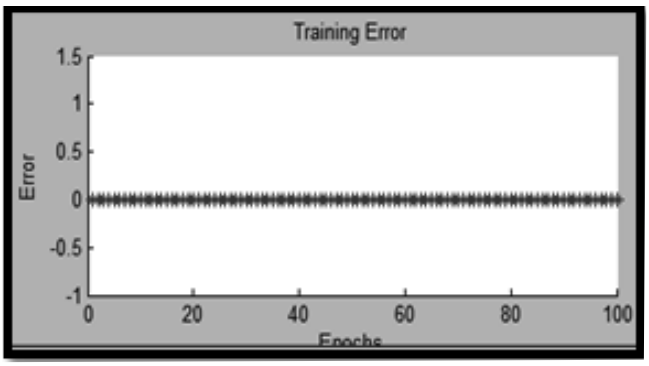

Fig. 4: Training error mapping sugeno FIS to ANFIS.

IV. RESULT AND DISCUSSION

After creation of ANFIS model, we paralleled the output dependability values for different input sets with the excluded values. We planned the root mean square error (RMSE) for the output attained by the FIS and the output obtained by the ANFIS with the creative output. The unit of relationship had important. The dissimilarity between the two RMSEs can be detected. Using only the FIS, there was an 11.74% mistakes in the results, but the ANFIS shrinks the error to 6.66%. Hence, the ANFIS implements better than the FIS. Using the ANFIS, we first skilled the FIS, and the guidelines were designed on the basis of preparation, data to create the output of the expert model. The only limit of this model is that its implementation is complex for large data sets. Our outcomes show that the ANFIS model gives a more perfectamount of reliability than the FIS model.The testing error and training error attained from the matlab fis toolbox is shown in Figs. 3 and 4.

V. CONCLUSION

For any software system, there are two user supplies: reliability and availability. Reliability is essential when the product's performance will have the maximum Fig. 3 Testing error representing sugeno FIS to ANFIS. An adaptive neuro unclear model for approximating 49 impacts. Many approaches to CBSS reliability have been planned, some based on mathematical methods and others based on soft computing methods. This is a cross method that needs less computational time than old-style approaches and the beforeplanned FIS approach.

REFERENCE

[1] Tyagi, K., Sharma, A., 2012. A rule-based approach forestimating the reliability of component-based systems.Adv. Eng. Softw. 54, 24–29.

[2] Jang, J.R., 1992. ANFIS: adaptive-network-based fuzzy inference system, IEEE Trans. Systems, Man, and Cybernetics.

[3] Sharma, A., Grover, P.S., Kumar, R., 2009. Dependency analysis for component based software systems. ACMSIGSOFT Software. Engg. Notes 34 (4), 1–6.

[4] Parul Gandhi, Ravi Kumar Sharma, , "Quality Assurance of Components Based Software Engineering" Proceedings IEEE, 2016.

[5] Huang, N., Wang, D., Jia, X., 2008. FAST ABSTRACT: an algebra-based reliability prediction approach forcomposite web services, 19th International Symposium on Software Reliability Engineering, pp. 285–286.

[6] M.R.Lyu (ed.), Handbook of Software Reliability Engineering, McGraw-Hill, New York, 1996.

[7] I.Jacobson, M. Christerson, P.Jonsson, G. Overgaard, "Object-Oriented Software Engineering: A Use Case Driven Approach," Addison-Wesley Publishing Company, 1992.

[8] Dong, W., Huang, N., Ming, Y., 2008. Reliability analysis of component-based software based on relationships of components, IEEE Conference on Web Services, pp. 814–815.

[9] Fiondella, Lance, Rajasekaran, Sanguthever, Gokhale, Swapana, 2013. Efficient software reliability analysis with correlated component failures. IEEE Trans. Reliab. 62 (1), 244–255.

[10] Zhang, F., Zhou, X., Chen, J., Dong, Y., 2008. A novel model for component-based software reliability analysis, 11th IEEE High Assurance Systems Engineering Symposium, pp. 303–309.

❐