Uncertainty Analysis of Software Reliability: Architecture-Based Approach

Harish Mittal
Sat Priya Group of Institutions, Rohtak, INDIA
mittalberi@gmail.com

Abstract - **Many architecture – based software reliability models have been proposed in the past. The architecture - based approach for software reliability assessment considers the utilization and the reliability of components, thus allowing insight into the dynamic behavior of software executions. This paper details the state of the architecture- based approach to reliability assessment of component based software. This paper aims at presenting and comparing different methods for uncertainty analysis: entropy, method of moments, Monte Carlo simulation and perturbation theory.**

Keywords: **SRGM, DTMC, Component Failure Behaviour, Transition probability matrix, Software Behaviour.**

I. INTRODUCTION

A. Software Reliability

Software reliability is defined as the probability that software product will work without failure in a specified environment for a specified exposure period. The exposure period can be execution time or software runs. The environment usually is characterized by a set of input states along with their probabilities of occurrence. This probability distribution over the input space that represents the frequencies of occurrence of possible input states in the operation of software application is known as operational profile. Thus, the predictive quality of software reliability models is affected by the ability to estimate the correct operational profile. However, building an operational profile is not an easy task, especially for a new product. Therefore, it is of critical importance to study the sensitivity of the software reliability to the errors in the operational profile.

B. Architecture Based Approach

Architecture–based assessment of software reliability combines the dynamic information about software architecture with the failure behavior of components.

Several different models have been proposed to predict the software reliability growth (SRGM); however, none of them has proven to perform well considering different project characteristics.

1) State-based models: This class of models adapts the control flow graph principles to represent the software architecture. Control flow graph reveals the structure, decision points, and branches in program code, thus representing the interaction between modules and possible execution paths. State-based models assume that the transfer of control between modules has a Markov property which means that given the knowledge of the module in control at any given time, the future behavior of the system is conditionally independent of the past behavior. The architecture of software has been modeled as a discrete time Markov chain (DTMC), continuous time Markov chain (CTMC), or semi Markov process (SMP).

2) Path-based models: Similar to state-based models, path-based models consider the software architecture explicitly and assume that components fail independently. However, the method of combining software architecture with components and interfaces failure behavior differs from the state-based models. Path-based models consider a sequence of components executed along each path and compute the path reliability by multiplying their reliabilities. Then, the system reliability is estimated by averaging path reliabilities over all paths.

3) Additive models: This class of models does not consider explicitly the architecture of the software. Rather, they are focused on estimating the overall application reliability using the component's failure data. These models consider software testing phase and assume that each component reliability can be modeled by non-homogeneous Poisson process (NHPP).

C. Uncertainty Analysis

Two important questions arise with respect to predications of software reliability based on models. The first question addresses the appropriateness of the model. The second question addresses the accuracy of parameters values. Parameters can be estimated using the field data obtained during operational usage of the software. In practice, there is a lot of uncertainty around parameters because they rarely can be estimated accurately. Reliability is a measure of failure uncertainty [2]. Different methods can be applied for synthesizing uncertainty. These methods are shown in fig 1.

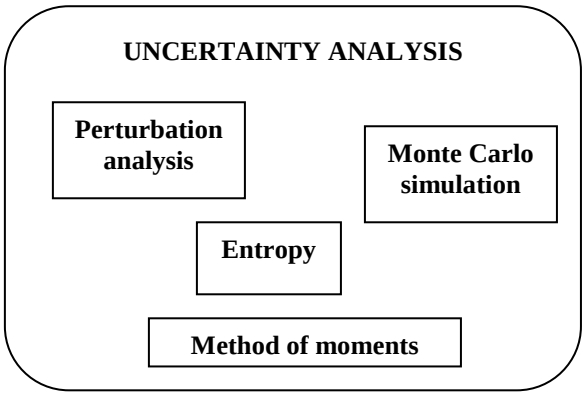

Figure1. Methods for uncertainty analysis in software reliability

In order to estimate the system reliability using architecture - based model we need to know the *software architecture* and software usage described by the *operational profile.*

1) Software Architecture

Software architecture reflects the way software components interact during execution. This means that we consider the dynamic information on interactions between components as a part of software architecture. The architecture is modeled with a *discrete time Markov chain (DTMC).*Two different approaches that we use to build a DTMC are:

- *Intended Approach* is used in early phases of software development. We base our estimates on historical data from similar products or on high level information about software architecture and usage obtained from specification and design documents. Since UML is rapidly becoming a standard for software development, in intended approach we are looking into the UML annotations such as use cases and sequence diagrams. Use case diagrams provide graphical description of how external objects interact with the system, while sequence diagrams show components and messages that are exchanged between them.
- *Informed Approach* is used during the late phases of the software development when testing or field data become available. Thus, component traces obtained using profilers [31] and test coverage tools [32] can be used to obtain a set of execution paths and establish the frequency count of the transition arcs.

The discrete time Markov chain proves to be a good model for software architectures and operational profiles for several reasons. From the software engineering point of view, the model can be build both in early and late phases of the software life cycle. Once the model has been built, any number of statistically typical test cases can be generated from the model [21]. From the analytical point of view, this is a tractable stochastic process with well developed theory, analytical results, and computational algorithms. Furthermore, the model provides basis for building the several different architecture-based software reliability models [8].

2) Operational Profile

Dynamic information in software architecture clearly depends on the software usage, that is, the operational profile. Software evolves during its life cycle and operational profile changes. Operational profile directly affects the reliability estimate. Therefore, it is important to conduct uncertainty analysis due to uncertainty in the operational profile estimation.

II. RELATED TERMS

DTMC: Discrete time Markov chain (DTMC) is a modeling paradigm used to represent application architecture. The transfer of control between components has a Markov property, the architecture is modeled with a discrete time Markov chain (DTMC). The Markov chain has a two-phase construction: structural phase and dynamic statistical phase**.**

SRGM: A software reliability growth model (SRGM) can be regarded to be a mathematical expression which fits the experimental data. Software modeling techniques can be divided into two subcategories: prediction modeling and estimation modeling.

Component Failure Behaviour: Component reliabilities or failure rates estimate the reliability of each component. Several techniques are: Software reliability growth models, non-failed executions and fault injection technique.

Transition probability matrix: is defined as $P = [p_{ij}]$
where, p_{ij} = Pr {program transfers the control from component i to component j}.

Software Behaviour: is the manner in which different components interact is defined through the software architecture.

III. RELATED WORK

With the growing emphasis on reuse, software development process moves toward Component-based software design. As a result, there is a need for modeling approaches that are capable of considering the architecture of the software made out of components. Assessing reliability at early stages of software development, such as at the level of software architecture, is desirable and can provide a cost-effective way of improving software system's quality. However, predicting a component's reliability at the architectural level is challenging because of uncertainties associated with the system and its individual components due to the lack of information.

Traditionally, the most common method for uncertainty analysis in software reliability is conducting sensitivity studies. Thus, sensitivity of the software reliability estimation to errors in the operational profile has been investigated in the context of black - box reliability growth models [3], [19], [20].

Sensitivity studies of software reliability estimates obtained using architecture - based models have been presented in [4]. In these studies the authors assumed fixed known values for the transition probabilities and derived the sensitivity of the system reliability with respect to the reliability of each component. However, any inaccuracy in the operational profile directly will affect transition probabilities among components. Therefore, in [7] they presented the sensitivity studies of software reliability with respect to the operational profile (i.e., transition probabilities) and component reliabilities.

An extensive survey of architecture-based software reliability models was presented in [8] and [22]. Although numerous papers were devoted to such models, only a few have actually applied the models on real case studies [13], [4], [8], [9], [10], [11], and even fewer have conducted detailed uncertainty analysis on real data [8], [9], [11]. Typically, studies of the model sensitivity to the parameters in the context of architecture–based models are conducted by measuring the change in the overall reliability as a single component reliability varies. Thus, in [2] and [16] the authors assumed fixed known values for the transition probabilities and derived the sensitivity of the system reliability with respect to the reliability of each component. The results were illustrated on simple made-up models.

Software reliability modeling has gained a lot of importance in the recent years. The use of intelligent neural network and hybrid techniques in place of the traditional statistical techniques has shown a remarkable improvement in the prediction of software reliability. In [14] they develop models to accurately forecast software reliability**.** In [15] they introduce a model that can be used for software reliability prediction. In [23] they proposed a fuzzy based approach to measure uncertainty.

IV. UNCERTAINTY ANALYSIS METHODS

A. Perturbation Analysis

Perturbation theory provides mathematical means to study how the stationary distribution of a Markov chain containing an irreducible set of states changes as the transition probabilities of the chain vary. Using perturbation theory we can assess the sensitivity of stationary probabilities $\pi = [\ \pi_i\ ]$ to perturbations in the operational profile (i.e., transition probability matrix P). Since π_i can be interpreted as the expected execution rate of component i in the long run, it represents a measure of component usage which can be used to identify critical components.

B. Entropy

It is used to quantify the uncertainty of the operational profile, the overall software reliability, and software components.

$$H = -\sum_i \pi_i \sum_j p_{ij} \log p_{ij}$$

where, π_i represents the usage distribution and the p_{ij} the transition probabilities.

Uncertainty of component *i* is estimated using the conditional entropy

$$H_i = -\sum_j p_{ij} \log p_{ij}$$

C. Method of Moments

It is an approximate analytical method that allows us to estimate the moments of the system reliability based on the knowledge of the moments of component reliabilities. Method of moments involves the following steps:

1) Obtain the expression for the system reliability using the architecture-based software reliability model.
2) Expand the expression for system reliability using Taylor series.
3) Determine the moments of the components reliabilities.
4) Estimate the mean and the variance of the system reliability using the parameter moments and Taylor series coefficients.

Mean reliability is $E\ [\mathrm{R}] = a_0$

Variance of the Reliability is

$$\sigma_R^2 = \sum_{i=1}^{n} a_i^2 \sigma_i^2$$

where, $\sigma_i^2 = Var\ [\ R_i\ ]$ is the variance of component reliability.

D. Monte Carlo Simulation

The fourth method for uncertainty analysis is based on simulation. Monte Carlo simulation involves the following steps:

1) Obtain the expression for the system reliability using the architecture-based software reliability model.
2) Assign probability distributions to the transition probabilities and components reliabilities.
3) Sample the distributions.
4) Compute the reliability of the system using the sampled values.
5) Repeat steps 3&4 until the desired number of values of system reliability has been generated.
6) Calculate the moments, frequency chart and percentiles for the system reliability, do the distribution fitting.

V. COMPARISION of the METHODS of UNCERTAINTY ANALYSIS

The choice of the most suitable method for uncertainty analysis for a particular application may be based on the following criteria:

- data requirements
- reliability measures derived
- accuracy of the solutions
- scalability with respect to the number of components.

Here we briefly summarize the methods of uncertainty analysis that we have considered so far with an emphasis on the above criteria.

A. Perturbation Analysis

Perturbation theory provides mathematical means to study how the stationary distribution of a Markov chain containing an irreducible set of states changes as the transition probabilities of the chain vary. Its characteristics with respect to the different criteria are the following:

- Data requirements
  - Low: Point estimates of the transition probabilities.
- Reliability measures derived
  - NA: Reliability measures are not derived. Instead, we study the sensitivity of the expected execution rates of software components to perturbations in the operational profile.
- Accuracy of the solutions
  - Analytical solution; bounds for the absolute and relative change of components execution rates.
- Scalability
  - Scales well. Could be used for large systems.

B. Entropy

Entropy is a well known concept from the information theory and evidence theory, source entropy, conditional entropy and Shannon entropy, to quantify the uncertainty of the operational profile, the overall software reliability, and software components. The characteristics of the entropy as a measure for uncertainty are the following:

- Data requirements
  - Low: Point estimates of the transition probabilities and components reliabilities.
- Reliability measures derived
  - NA: Reliability measures are not derived. Instead, we derive the uncertainty of an operational profile and software reliability, as well as components uncertainties.
- Accuracy of the solutions
  - Exact analytical solution for the uncertainty.
- Scalability
  - Scales well. Could be used for large systems.

C. Method of Moments

Method of moments is an approximate analytical method that allows us to estimate the moments of the system reliability based on the knowledge of the moments of component reliabilities. Its characteristics with respect to the different criteria are the following:

- Data requirements
  - Low: Only the moments of components reliabilities are needed, that is, no distribution function must be specified.
- Reliability measures derived
  - Moments of the system reliability (usually the mean and variance)
- Accuracy of the solutions
  - Approximate method.
  - The accuracy may be increased by using higher order Taylor series.
  - The accuracy is not readily quantifiable.
  - Generation of random numbers is not required; therefore there is no sampling error.
- Scalability
  - Due to the computational cost for evaluating the Taylor series coefficients and system moments the use of the method is limited to medium size systems.

D. Monte Carlo simulation

The fourth method for uncertainty analysis is based on simulation. The basic characteristics of the Monte Carlo simulation used for uncertainty analysis in software reliability are the following:

- Data requirements
  - High: Probability distribution functions of modeling parameters (transition probabilities and components reliabilities).
- Reliability measures derived
  - Many characteristic of system reliability can be derived, including frequency charts, moments, percentiles, and distribution functions.
- Accuracy of the solutions
  - Approximate method.
  - The accuracy may be increased by increasing the number of simulations (sample size).Although with the hardware available today it is not critical, it is worth mentioning that the computational cost increases with the sample size.
  - Sampling errors may be involved in case of long tail distribution
- Scalability
  - Scales very well, that is, it is not very sensitive to the number of components in the system. Could be used for large systems.

TABLE I. COMPARISON of DIFFERENT METHODS for UNCERTAINITY ANALYSIS

Method	Data requirements	Uncertainty of the operational profile	Reliability measures derived	Accuracy of the solution	Scalability
Perturbation analysis	Point estimates of transition probabilities	Bounds on the absolute and relative change of components execution rates	NA	Analytical solution	Large systems
Entropy	Point estimates	1. Uncertainty of operational profile 2. Uncertainty of components 3. Expected execution rate of components	NA	Exact analytical solution	Large systems
Method of moment	Moments of components reliabilities	NA	Moments of system reliability	Approximate solution: 1. accuracy may be increased by higher order Taylor series 2. No sampling errors	Medium systems
Monte Carlo simulation	1. Probability distribution functions of components reliabilities and transition probabilities 2. Generation of random numbers	NA	Many characteristics of system reliability: 1.Frequency chart 2.Distribution 3. Moments 4. Percentiles	Approximate solution: 1. accuracy maybe increased by increasing the sample size 2. Sampling errors may be involved in case of long tail distributions	Large systems

In Table I we summarize the basic characteristics of different methods of uncertainty analysis. This is so called Make a choice table provides a sound guideline for choosing the most appropriate method for a given software application.

Now, we compare the four methods for uncertainty analysis that we have studied so far. Perturbation theory as a method for uncertainty analysis allows us to study the sensitivity of software usage due to the changes in the operational profile. In particular, we estimate the bounds on the absolute and relative changes of the component execution rates. The method for uncertainty analysis based on entropy allows us to characterize the uncertainty in the operational profile and software reliability with a single value. Thus, the operational profile with higher entropy value will have exponentially greater number of statistically typical paths. We have also derived the uncertainty and expected execution rate of software components. Although the entropy approach and perturbation theory provide useful results that can be used for identification of critical components and allocation of design and testing efforts, they do not provide estimates of the software reliability measures. Thus, we can say that entropy and perturbation theory as methods for studying uncertainty are complementary with the method of moments and Monte Carlo simulation.

An advantage of the method of moments over Monte Carlo simulation is the lower data requirements. This method only requires the moments of transition probabilities and components reliabilities that can be calculated easily directly from test data (i.e., no distribution function has to be specified). Further, method of moments is an analytical method and therefore generation of random numbers is not required. A limitation of the method of moments is that accuracy may be increased only by including higher order terms in the Taylor series expansion. This is in contrast to the Monte Carlo simulation where, in principle, the accuracy may be arbitrary increased simply by increasing the number of simulations. Also, the accuracy of method of moments is not readily quantifiable. Thus, if a determination of the precise accuracy of an uncertainty analysis is required, the Monte Carlo method should be used. Also, Monte Carlo simulation provides a richer set of reliability measures such as moments, distribution function, and percentiles of system reliability, as well as sensitivity ranking of the parameters accordingly to the contribution to the variance.

VI. CONCLUSION AND FUTURE WORK

Traditional reliability techniques can have serious difficulties when used on software engineering data. Even though effort has been done to propose fuzzy based techniques, yet there is a vast scope to find better fuzzy approaches. In this paper we studied different methodologies for uncertainty analysis of software reliability that can be applied throughout the software life cycle. The uncertainty analysis of software reliability is not only important but also necessary, especially if we want to make predictions early in the life cycle and keep track of software evolution. The uncertainty analysis provides richer measures of software reliability than the traditional point estimate. The main focus of our future work is to explore other methods for uncertainty analysis and compare them accordingly to different criteria. This comparison will help us to develop sound guidelines for choosing the most appropriate method.

REFERENCES

[1]. T. Adams, "Total variance approach to software reliability estimation," *IEEE Trans. Software Engineering*, vol. 22, no.9, pp.687-688, 1996.

[2]. K. Y. Cai, Introduction to fuzzy reliability. Kluwe Academic publishers, 1996.

[3]. M. Chen, A. P. Mathur, and V. J. Rego, "A case study to investigate sensitivity of reliability estimates to errors in operational profile," *Proc. 5th International Symposium on Software Reliability Engineering*, pp.276-281, 1994.

[4]. S. S. Gokhale, W. E. Wong, K. Trivedi, and J. R. Horgan, "An analytical approach to architecture–based software reliability prediction," *Proc. 3rd International Computer Performance and Dependability Symposium*, pp. 13–22, 1998.

[5]. L. Cheung, L. Golubchik, N. Medvidovic1, G. Sukhatme, "Identifying and addressing uncertainty in architecture-level software reliability modeling." 2007 IEEE International Parallel and Distributed Processing Symposium. IEEE, 2007.

[6]. R. C. Cheung, "A user-oriented software reliability model," *IEEE Trans. Software Engineering*, vol.6, no.2, pp.118-125, 1980.

[7]. K. Goseva - Popstojanova, A. P. Mathur, and K. S. Trivedi, "Comparison of architecture-based software reliability models," *12th International Symposium on Software Reliability Engineering*, pp.22-31, 2001.

[8]. K.Goseva – Popstojanova, and K. S. Trivedi, "Architecture-based approach to reliability assessment of software system," *Performance Evaluation*, vol.45, no.2-3, pp.179-204, 2001.

[9]. K. Goseva-Popstojanova and S. Kamavaram, "Software reliability estimation under uncertainty: Generalization of the method of moments," *Proc. 8th IEEE Int'l Symposium on High Assurance Systems Engineering*, pp. 209-218, 2004.

[10]. K. Goseva-Popstojanova, M. Hamill, and R. Perugupalli, "Large empirical case study of architecture based software reliability," *Proc. 16th IEEE International Symposium Software Reliability Engineering*, pp. 43-52, 2005.

[11]. K. Goseva-Popstojanova, M. Hamill, and X.Wang, "Adequacy, accuracy, scalability and uncertainty of architecture-based software Reliability: Lessons learned from large empirical case studies," Proc. 17th IEEE International Symposium Software Reliability Engineering, pp.197- 203, 2006.

[12]. S. Kamavaram and K. Goseva - Popstojanova, "Entropy as a measure of uncertainty in software reliability", 13th International Symposium on Software Reliability Engineering, November 2002.

[13]. K. Kanoun and T. Sabourin, "Software dependability of the telephone switching system," Proc. 17th International symposium on Fault Tolerant Computing, pp. 236–241, 1987.

[14]. N. Raj Kiran, and V. Ravi, "Software reliability prediction by soft computing techniques", The Journal of Systems and Software, 2007.

[15]. K. Khatatneh, and T. Mustafa, "Software reliability modeling using soft computing technique," European Journal of Scientific Research, vol.26, no.1, pp.154-160, 2009.

[16]. K. Siegrist, "Reliability of system with markov transfer of control," IEEE Trans. Reliability, vol.14, no.7, pp. 1049–1053, 1988.

[17]. K. W. Miller, L. J. Morell, R. E. Noonan, S. K. Park, D. M. Nikol, B. W. Murrill, and J. M.Voas, "Estimating the Probability of failure when testing reveals no failures," IEEE Trans. Software Engineering, vol.18, no.1, pp.33-43, 1992.

[18]. Y-W Leung, "Software reliability allocation under an uncertain operational profile," Journal of the Operational Research Society, vol. 48, pp.401-411, 1997.

[19]. J. D. Musa, "Operational profiles in software reliability engineering," IEEE Software, vol. 10, pp.14-32, 1993.

[20]. A. Pasquini, A. N. Crespo, and P. Matrella, "Sensitivity of reliability - growth models to operational profile errors vs. testing accuracy," IEEE Trans. Reliability, vol.45, no.4, pp.531-540, 1996.

[21]. J. A. Whittaker, and J. H. Poore, "Markov analysis of software specifications," ACM Trans. Software Engineering and Methodology, vol. 2, no. 1, pp. 93-106, 1993.

[22]. S. Khokhar and H. Mittal, "Modelling uncertainty in software reliability estimation using fuzzy entropy," 6th International Conference on Upcoming Trends in IT, May 2010.

[23]. S. Khokhar and H. Mittal, "Modelling uncertainty in software reliability estimation," National Conference on Latest Trends in IT, April 2010.

[24]. C. V. Ramamoorthy and F. B. Bastani, "Software reliability – status and perspectives," IEEE Trans. on Software Engineering, vol.8, no.4, pp. 354-371, 1982.

[25]. B.Littlewood and D.Wright, "Some conservative stopping rules for operational testing of Safety –critical software," IEEE Trans. Software Engineering, vol.23, no.11, pp. 673-683, 1997.

[26]. P. S. Jackson, R. W. Hockenbury, and M. L. Yeater, "Uncertainty analysis of system reliability and availability assessment," Nuclear Engineering and Design, vol.68, pp. 5-29, 1981.

[27]. C. D. Meyer, "Sensitivity of the stationary distribution of a Markov chain," SIAM J. Matrix Anal. Application, vol.15, no. 3, pp. 715-728, July 1994.

[28]. G. Booch, J. Runbaugh and I. Jacobson, The Unified Modeling Language User Guide, Addison-Wesley, 1998.

[29]. J. M. Voas, "Certifying Off-the-shelf Software Components," IEEE Computer, vol.31, no.6, pp. 53-59, 1998.

[30]. C. Smidts and D. Sova, "An architectural model for software reliability quantification", Reliability Engineering and System Safety, vol.64, pp.279-290, 1999.

[31]. http://www.gnu.org/manual/gprof-2.9.1/html mono/gprof.html

[32]. http://xsuds.argreenhouse.com

[33]. Mittal, Harish, P. K. Bhatia, and Puneet Goswami. "Software Quality Assessment Based on Fuzzy Logic Technique." International Journal of Software Computing Applications 3 (2008): 105-112.

[34]. Gupta, Deepak, Vinay Kr Goyal, and Harish Mittal. "Analysis of clustering techniques for software quality prediction." 2012 Second International Conference on Advanced Computing & Communication Technologies. IEEE, 2012.