Development of Software Reliability Growth Models Using Big Data Approaches

Savita Bamal
Assistant Professor, Department of Computer Science and Engineering
GITM College Bilaspur (Gurgaon) Haryana, India
Savita_nc@yahoo.com

Abstract: **The effective empirical approaches of software reliability modeling with big data have been only few presented. In particular, empirical approach of big data for software reliability is managed by using several software models. We investigated the use of big data on building SRGM to estimate the expected software faults during testing process. The proposed Big data approaches consists of a collection of linear sub-models. A data set provided by big data analytics approach used in real life applications like(real time control, military and operating system applications) was used to show the potential of using big data in solving the software reliability modelling problem. In this paper, different kind of SRGM Gompertz, modified Gompertz and hazard rate models) are used to achieve software reliability.**

Keywords: Software Reliability, Reliability Models (Gompertz,Modified Gompertz), Big data analytics-A Data Driven Approach, Conclusion, References.

I. INTRODUCTION

The use of software reliability growth models plays an important role in measuring improvements, achieving effective and efficient test/debug scheduling during the course of a software development project, determining when to release a product or estimating the number of service releases required after release to reach a reliability goal.

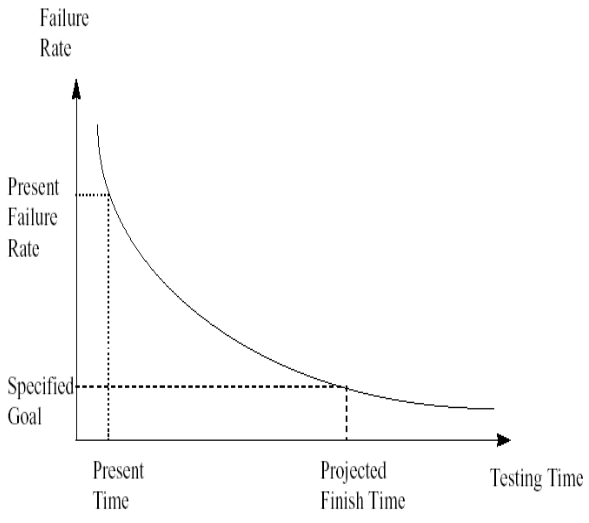
Many new trends in software development process standardization, in addition to established ones emphasize the need for statistical metrics in monitoring reliability and quality improvements. We propose different empirical techniques such as data driven and long term predictions are used to achieve software reliability growth modeling using big data analytics techniques**.**

Software reliability: Software reliability is defined as the probability of failure-free software operation for a specified period of time in a specified environment (lyu, 1996). Software Reliability Growth Model (SRGM) is a mathematical model of how the software reliability improves as faults are detected and repaired (quadri, 2010). Among all SRGMs developed so far a large family of stochastic reliability models based on a Non-Homogeneous Poisson Process known as NHPP reliability models, has been widely used. Some of them depict exponential growth while others show S-shaped growth depending on nature of growth phenomenon during testing. The success of mathematical modelling approach to reliability evaluation depends heavily upon quality of failure data collected. Software reliability is defined as the probability of failure-free software operation for specified period of time _t'in a specified environment's **R (t) =e$^{-m\ (t)}$.**

II. SOFTWARE RELIABILITY GROWTH MODELS:

SRGMs are a statistical interpolation of defect detection data by mathematical functions (Wood, 1996). They have been grouped into two classes of models-Concave and S-shaped. The only way to verify and validate the software is by testing. This involves running the software and checking for unexpected behaviour of the software output (kapur, 2009). SRGMs are used to estimate the reliability of a software product. In literature, we have several SRGMs developed to monitor the reliability growth during the testing phase of the software development. past three decades, hundreds of models were introduced to estimate the reliability of software systems (Musa, 1975; Xie, 2002). The issue of building growth models was the subject of many research work (Lyu, 1996; Musa, 2004) which helps in estimating the reliability of a software system before its release to the market Software reliability growth models were significantly used to help in computing the number of faults which is still resides in the software (Teng and Pham, 2002). Thus, it is important to specify the effort required to fix faults, the time required before software can be released and the cost of repair. Software reliability growth models employ system experimental data for testing to predict the number of defects remaining in the software. Software reliability models can be

classified to two types of models according to prediction style either from: (1) the design parameters thus called "defect density" models (2) the test data thus "software reliability growth" models. Some known SRGM are Logarithmic, Exponential, Power, S-Shaped and Inverse Polynomial models. They are typical analytical models. They normally describe the fault process as a function of execution time (or calendar time) and a set of unknown parameters. The model parameters normally estimated using least-square estimation or maximum likelihood techniques (Lye, 1996).
Reliability growth models curve represents in fig.1



III. NHPP MODEL

The Non-Homogeneous Poisson Process (NHPP) model consists of a mean value function and intensity function.

Equation1:

$$P\{N(t)=y\}=\frac{[m(t)]^y}{y!}e^{-m(t)}, \quad y=0,\ 1,\ 2,\ \ldots$$

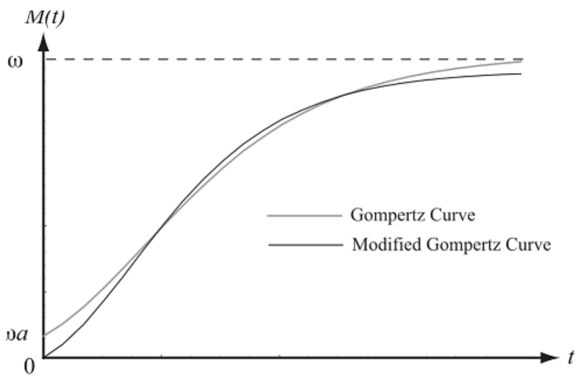
Where m (t) =a (1-e$^{bt)}$

λ (t) = m' (t) = abe^{-bt}

These time-area models based on NHPP were distinguished using a probability of the failure-time. In Equation (2), the mean value function m(t) is parameter. The failure intensity function λ(t) can be characterized using the relationship of Equation (1). A finite model has the assumption no new fault has occurred at the time of each repair. But, new failure at the time of repair from the real circumstances may happen. To reflect this state of affairs, a NHPP model using the RVS (Record Value Statistics) can be used. The mean value function was known to next condition10. A pattern of the Equation (3) is mean value function about the infinite failure NHPP model.

IV. GOMPERTZ SOFTWARE RELIABILITY MODEL

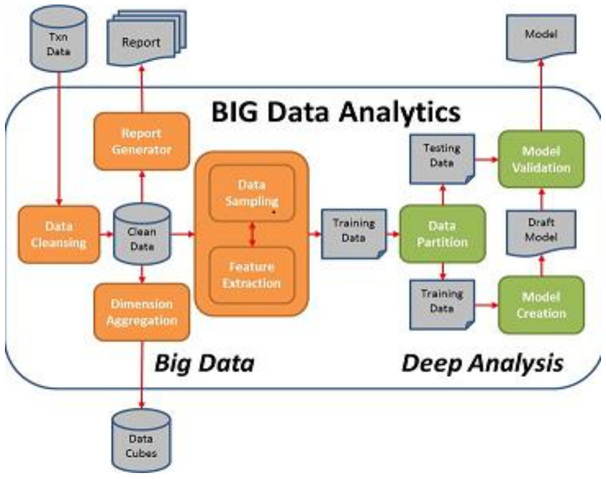
The simplest form of a software reliability growth model is an exponential one. However, S-shaped software reliability is more often observed than the exponential one. Some models use a non-homogeneous Poisson process (NHPP) to model the failure process. The NHPP is characterized by its expected value function, m (t). This is the cumulative number of failures expected to occur after the software has executed for time t. Gompertz SRGM is based on an NHPP. In fact, many Japanese computer manufacturers and software houses have applied the Gompertz curve model, which is one of the simplest S-shaped software reliability growth models (Kececioglu, 1991). The Gompertz curve model gave good approximation to cumulative number of software faults observed (Satoh, 2000). It takes the number of faults per unit of time as independent Poisson random variables. The Gompertz model equation for software reliability is **u (t) =ab^{ct}** Where, u(t) is software reliability ai time t and is dimensionless percent. a is the upper limit approached the reliability. band c are also gompertz constant, which provide shape parameter to gompertz equation. A relatively small value of c promulgates rapid early reliability growth, while large value of c indicates slower reliability.



Gompertz curve has been used to estimate the number of residual faults in testing phases of software development.Since the Gompertz curve is a deterministic function, the curve cannot be applied to estimating software reliability which is the probability that software system does not fail in a prefixed time period. In this article, we propose a stochastic model called the Gompertz software reliability model based on non-homogeneous Poisson processes. The proposed model can be derived from the statistical theory of extreme-value, and has a similar asymptotic property to the deterministic Gompertz curve.

V. Big Data Analytics

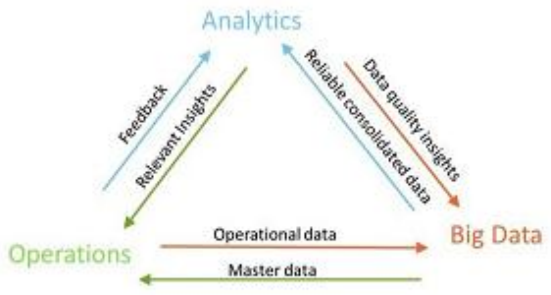
Big data refers to the dynamic, large and disparate volumes of data being created by people, tools and machines; it requires new, innovative and scalable technology to collect, host and analytically process the vast amount of data gathered in order to derive real-time business insights that relate to consumers, risk, profit, performance, productivity management.It can be represented by fig3:



VI. Big Data Drivers

Big data poses both opportunities and challenges for businesses. In order to extract value from big data, it must be processed and analyzed in a timely manner, and the results need to be available in such a way as to be able to effect positive change or influence business decisions. The effectiveness also relies on an organization having the right combination of people, process and technology. The quality of data sets and the inference drawn from such data sets are increasingly becoming more critical and organizations need to build quality and monitoring functions and parameters for big data. For example, correcting a data error can be much more costly than getting the data right the first time — and getting the data wrong can be catastrophic and much more costly to the organization if not corrected.

Fig4:



Big Data Approaches: Decisions can be made with a structured approach through data-driven insight, including Customer and product profitability, Customer acquisition and retention strategies, Customer satisfaction strategies Marketing

segmentation, Operations and performance management, Supply chain and delivery channel strategy. If you want to take a Big Data approach to software development, the SLIM Suite can provide you with the tools to estimate, track, and analyze software projects. SLIM allows you to collect data on your completed projects, analyze it so you can learn your strengths and weaknesses, use that information to set goals for future projects by designing custom templates, and then track the progress of your project throughout the software lifecycle.While there may be little incentive to take the time to conduct a postmortem for your previous project instead of immediately starting the next one, taking time to reflect on what you did well and what needs improvement is valuable for planning future projects.
Once you have a database of projects (the more the better) you can analyze them using SLIM-Metrics. Perhaps you're interested in looking at how team size affects productivity. Using SLIM-Metrics, you can build queries to group your projects into small teams *(i.e. <5 FTE)* and large teams *(> 10 FTE).* You can then plot these groups on a Size vs. PI chart to determine which trends show greater Reliability.

If your organization has at least 30 completed projects, you can use that data to create custom trend lines to use in your analyses. Creating custom trends is beneficial to estimating future projects because it will give a more accurate representation of what your organization has accomplished in the past. You can even take this customization one step further by building custom templates. Custom templates are regular SLIM-Estimate files that have already been calibrated to your organization's software lifecycle. They usually are based upon your typical projects as a model for estimating future projects.If you collect data throughout the course of your project, you can even use it to track its progress. SLIM-Control allows you to forecast the most likely delivery date, cost, and reliability of a project. At various points during the project you can enter actual data and compare it with the project plan. If the project begins to veer off schedule you can reforecast and use the updated information to adjust/ update a new plan. Collecting metrics while the project is in progress will allow you to make more accurate predictions about the project outcome.

VII. CONCLUSION

The developed models provided high performance modelling capabilities. To validate the proposed approach, the parameter estimation is carried out on the data sets collected from different sources. Parameters of the model are estimated by MLE method using cumulative failure data against time. It is observed that with the considered model the data set having AIC value high is exhibiting high Reliability and the data set having AIC value low is exhibiting low Reliability at nth time failure.

1. REFERENCES

[1].Musa, J.D., Iannino, A., Okumoto, K. Software Reliability: Measurement, Prediction, Application, New York: McGraw-Hill, 1987.
[2] Pham. H., System software reliability, Springer, 2006.
[3] Wood, A., Software Reliability Growth Models, Tandem Computers, Technical report 96.1, 1996.
[4] Kapur, P.K., Sunil kumar, K., Prashant, J. Ompal, S. Incorporating concept of two types of imperfect debugging for developing flexible software reliability growth model in distributed development environment, Journal of Technology and Engineering sciences, 2009; Vol.1, No.1; Jan-Jun.
[5] Xie, M., Goh. T.N., Ranjan.P., ―Some effective control chart procedures for reliability monitoring‖ - Reliability engineering and System Safety, 2002; 77; 143 -150.
[6] Kuei-Chen C, Yeu-Shiang H, Tzai-Zang L. A study of software reliability growth from the perspective of learning effects. Reliability Engineering and System Safety. 2008; 93:1410–21.
[7]. Yoo TH. The Infinite NHPP software reliability model based on Monotonic Intensity Function. Indian Journal of Science and Technology. 2015 Jul;
[8]. Kuo L, Yang TY. Bayesian computation of software reliability. Journal of the American Statistical Association. 1996;