Chapter 26
Software Release Planning Using Grey Wolf Optimizer

Vibha Verma
University of Delhi, India

Neha Neha
University of Delhi, India

Anu G. Aggarwal
iD https://orcid.org/0000-0001-5448-9540
University of Delhi, India

ABSTRACT

This chapter presents the application of grey wolf optimizer in software release planning considering warranty based on the proposed mathematical model that measures reliability growth of software systems. Hence, optimal release and warranty time is determined while minimizing the overall software development cost. The software cost model is based on failure phenomenon modelled by incorporating fault removal efficiency, fault reduction factor, and error generation. The model has been validated on the fault dataset of ERP systems. Sensitivity analysis has been carried out to study the discrete changes in the cost parameter due to changes in optimal solution. The work significantly contributes to the literature by fulfilling gaps of reliability growth models, release problems considering warranty, and efficient ways for solving optimization problems. Further, the grey wolf optimizer result has been compared with genetic algorithm and particle swarm optimization techniques.

INTRODUCTION

In recent times IT-based firms focus on developing reliable software systems without bearing any financial or goodwill losses during the post-implementation phase. The technology advancements in Medical, Defence, Space, Transportation, banks, universities, homes appliances, etc. have increased the demand for qualitative software products. These products facilitate day to day task handling by reducing

the efforts and time required at both the individual and organizational level and any failure encountered during the software operations may lead to heavy financial losses and sometimes may prove hazardous to human lives also (Yamada and Tamura, 2016).

Due to the ever-increasing importance of software systems, the researchers and IT firms are continuously working to improve their reliability and hence the quality. For this, it is very necessary to assess the reliability of a software system during its development phase. The software development process, also known as the software development life cycle, through this phase developers try to enhance the software quality. Here the development process comprises of few steps i.e., planning, analysis, design, implementation, testing and maintenance. Among all these steps, testing is considered as the most decisive and essential task for improving the quality of the software by detecting and removing the faults. Faults detectability influences software reliability growth which results in the development of Software Reliability Growth Models (SRGMs).

Since 1970 numerous SRGMs have been developed for the assessment of software reliability. These models incorporate various aspects related to software development for e.g. the code size and complexity, the skill of tester, developer and programmer, testing tools and methodology, resources allocated, etc. A number of researchers and IT practitioners have proposed Non-Homogenous Poisson Process (NHPP) based SRGMs to assess the reliability of the software system (Aggarwal et al., 2019; Anand et al., 2018; Kapur et al., 2011). These models help to predict the number of faults and the time for the next failure on the basis of observed failure data.

These time-dependent SRGMs are divided into two classes: one is perfect debugging where it has been assumed that whenever a new failure occurs the faults causing it is removed immediately and no new faults are introduced in the meantime. The other one is imperfect debugging which further can be split into two types: (a) whenever originally detected faults are removed that do not remove completely this phenomenon is known as imperfect fault removal and (b) it takes several attempts to remove a fault and also some new faults which were previously non-existent may also get generated. This phenomenon re-introduction of faults is known as Error generation (Yamada et al., 1984). In this chapter, we model the failure phenomenon of software incorporating error generation.

Previously it was assumed that faults initiating the failure are removed with complete efficiency but later it was observed that all the encountered faults are not removed i.e. fault removal process is not 100% efficient (Zhang et al., 2003) i.e. only few number of faults are removed out of the overall faults spotted. It can be defined as the ability or effectiveness in the fault removal process. This measure helps the developer to predict the effectiveness of the fault detection and further effort needed. In our study, we have considered Fault Removal Efficiency (FRE) because it is immensely correlated with fault removal process i.e., as the FRE escalates fault removability escalates as well.

Also, it was stated that in practice the number of faults experienced is not the same as the faults removed during the process. Musa (1975) defined Fault Reduction Factor (FRF) as "the ratio of a number of faults removed to the number of failures experienced". This indicates that for the reliability evaluation FRF plays an important role. Experimentally, FRF takes values between zero and one. It has been discovered that FRF could be affected by issues like fault dependency, complex code, human nature, etc. This, in turn, affects the FRF curves which can be increasing, decreasing or constant. Here we consider that FRF follows Weibull distribution while error generation and FRE are considered to be constant. FRF, FRE and error generation all these factors have been incorporated because of their significant impact on the failure process.

Related Content

Developing Knowledge Management Systems from a Knowledge-Based and Multi-Agent Approach

Aurora Vizcaíno, Juan Pablo Sotoand Javier Portillo-Rodríguez (2009). *Software Applications: Concepts, Methodologies, Tools, and Applications (pp. 817-833).*

www.irma-international.org/chapter/developing-knowledge-management-systems-knowledge/29424

A Hybrid Optimal Feature Extraction for Brain Tumor Segmentation

P. Santhosh Kumar, V. P. Sakthivel., Manda Rajuand P. D. Sathya (2022). *International Journal of Software Innovation (pp. 1-15).*

www.irma-international.org/article/a-hybrid-optimal-feature-extraction-for-brain-tumor-segmentation/303578

Web Services Description and Discovery for Mobile Crowdsensing: Survey and Future Guidelines

Salma Bradai, Sofien Khemakhemand Mohamed Jmaiel (2016). *International Journal of Information System Modeling and Design (pp. 31-49).*

www.irma-international.org/article/web-services-description-and-discovery-for-mobile-crowdsensing/178563

A WYSIWYG Approach to Support Layout Configuration in Model Evolution

Yu Sun, Jeff Gray, Philip Langer, Gerti Kappel, Manuel Wimmerand Jules White (2012). *Emerging Technologies for the Evolution and Maintenance of Software Models (pp. 92-120).*

www.irma-international.org/chapter/wysiwyg-approach-support-layout-configuration/60718

Regression Testing-Based Requirement Prioritization of Mobile Applications

Varun Gupta, D. S. Chauhanand Kamlesh Dutta (2012). *International Journal of Systems and Service-Oriented Engineering (pp. 20-39).*

www.irma-international.org/article/regression-testing-based-requirement-prioritization-of-mobile-applications/89379