ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Reachability of Open Source Software

R.Kamalraj[1], R. Sujatha[2], P. Srisathya[2]

AP, Dept. of CSE, SNS College of Technology, Coimbatore[1]

Final year, Dept. of B.E CSE, SNS College of Technology, Coimbatore[2, 3]

Abstract: Open Source Software is developed by imbibing the components that the developer wants. There may be some components which the end-user may find only of minimal usage. Such components may be identified and considered for removal. Aim of the project is to identify the frequently accessed components of Open Source Software and to give suggestions on the effort that could be invested on the betterment of the software. In "Reachability of Open Source Software", the main task lies in identification of less frequently used components and to give suggestions to improve the efficiency of the component.

I. INTRODUCTION

Software Engineering is the science and art of building significant software systems on time, on budget with acceptable performance and with correct operation. The seminal definition of Software Engineering is "The establishment of and use of sound Engineering principles in order to obtain economically software that is reliable and works efficiently on real machines. IEEE Definition is "Software Engineering is the application of systematic disciplined quantifiable approach to the development, operation and maintenance of software that is the application of Engineering to Software". A Decision-Support System may be developed which helps the developer in updating mechanism. Since the Open Source Software is designed as per the requirements of the developer. The end user may or may not use all those components. The most used and less used components are identified and they may be suggested for updating or removal

II. CORPORATE METRICS

In the existing world, a concept called "Software metering" concept is used. It maintains and tracks the software that is used, Corporate check out licenses for mobile users and record all license in use. It is commonly known as "Pay-Per-Use". There is a need to make sure that only the allowed number of licenses are in use and at the same time that there are enough licenses for everyone using it. This can include monitoring of concurrent usage of Software for real-time enforcement of license limit. It is a method of Software Licensing where the licensed software automatically record how many times or for how long one or more functions in the software are used, and the user pays fees based on this actual usage. It is a fixed planning to allocate software usage to computers according to the policies a company specifies and to maintain a record of usage and attempted usage. The major demerit is that it is highly used for Proprietary software and it does not track the frequency of sub components used and it does not provide any suggestion.

III. MODULES

The modules used for accomplishing Reachability of Open Source Software are as follows

i. Identifying sub-components or functions in the software
ii. Identifying the frequency of usage
iii. Recording the results in the database
iv. Analyzing and defining suggestions

IV. MODULE DESCRIPTION

Identifying sub-components or functions in the software

The Open Source Software for which the 'reachability is tested is first studied and the Source code is analyzed. The classes and the functions are listed. All the sub-components in the software are listed.

Identifying the frequency of usage

The frequency of usage specifies the number of times a component is used. In some cases some components are highly used and some are less used. Those components are identified by setting a counter value, which keeps a record of the number of hits the function has faced.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Recording the results in the database

The component usage, user details with time and date are updated in the database. The results are stored in consolidated logs in the database.

Analyzing and defining suggestions

Based on the values in the database log a threshold value is set by considering the average value of database. If the hit ratio count of a tool is below the average value, then the tool may be suggested for removal. If the hit ratio count of a tool is above the average value, then the tool is suggested for updating schemes. On seeing the suggestion report, the developer can decide on further enhancement of the product. The effort can be improved on the tools that are most frequently used by the user. The suggestions suggested by the system are considered by the developer for developing the system.

V. REACHABILITY AND LINEAR SYSTEMS

Reachability testing is applied to semaphore based multithreaded programs that use semaphores to synchronize operations on shared data. A novel aspect of Reachability tesing is that is derives test sequences on the fly, avoiding the construction of any static models. Also, reachability testing algorithms deal with partial orders directly, avoiding the test sequence exploration problem that occurs when independent events are interleaved. A prototype tool called "RichTest".

VI. REACHABILITY FRAMEWORK

A framework for selecting synchronization sequences from java Multi thread program based on the analysis of reading and writing shared variables. It consists of a strategy for generating synchronization sequence set of reachability testing, and an approach for deterministic testing of the synchronization sequences. A prototype for Reachability testing of java multi thread program has been developed. In the prototype a dynamic proxy class is used to implement a deterministic testing framework of Java Multi thread program.

VII. REACHABILITY TESTING APPROACH OF CONCURRENT SOFTWARE

The object-oriented paradigm provides support for modular and reusable design and is attractive for the construction of large and complex concurrent systems. Reachability analysis is an important and well-known tool for static (pre-run-time) analysis of concurrent programs. However its direct application to concurrent object-oriented programs has many problems, such as incomplete analysis for reusable classes and increased computational complexity. It also seems impossible to arrive at a single general-purpose strategy that is both safe and effective for all programs. There is a tool-suite based approach for the reachability analysis of concurrent object-oriented-programs. This approach enables choice of an appropriate `ideal' tool, for the given program and also provides the flexibility for incorporation of additional tools. It is a novel abstraction-based partitioning methodology for effective reachability analysis of concurrent object-oriented programs. Using this methodology, a variety of tools has been developed, having different degrees of safety, effectiveness and efficiency, for incorporation into the tool-suite. It formally shown the safety of these tools for appropriate classes of programs and have evaluated their effectiveness and efficiency.

VIII. APPORTIONING: A TECHNIQUE FOR EFFICIENT REACHABILITY ANALYSIS OF CONCURRENT OBJECT-ORIENTED PROGRAMS

The object-oriented paradigm has been found to be useful for the construction of large and complex concurrent systems. Reachability analysis is an important and well-known tool for static (pre-run-time) analysis of concurrent programs. However, direct application of traditional reachability analysis to concurrent object-oriented programs has many problems, such as incomplete analysis for reusable classes (not safe) and increased computational complexity (not efficient). We have proposed a novel technique called apportioning, for effective reachability analysis of concurrent object-oriented programs, that integrates the techniques of abstraction (considering a reduced representation of the system) and partitioning (dividing the system into smaller units). The given program is apportioned into a reduced version of each of its classes, as well as a reduced version of the program. The error to be checked is also decomposed into appropriate sub-properties for checking in the reachability graphs corresponding to the apportioned program. We

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

have developed a number of apportioning-based algorithms, having different degrees of safety and effectiveness. In this paper, we present the details of one of these algorithms.

IX. Reachability Analysis in Privacy-Preserving Perturbed Graphs

Many real world phenomena can be naturally modeled as graph structures whose nodes representing entities and whose edges representing interactions or relationships between entities. The analysis of the graph data has many practical implications. However, the release of the data often poses considerable privacy risk to the individuals involved. In this paper, we address the edge privacy problem in graphs. In particular, we explore random perturbation for privacy preservation in graph data, and propose an iterative derivation process to analyze node reachability within the graph. We specifically focus on deriving the probability that the shortest path linking two nodes in a directed graph is of a particular length. This allows us to determine the expected length of the shortest path between two nodes, and determine whether they are linked or not. The performance of the proposed method is demonstrated via extensive experiments on both real and synthetic

X. Quantifying and Querying Network Reachability

Quantifying and querying network reachability is important for security monitoring and auditing as well as many aspects of network management such as troubleshooting, maintenance, and design. Although attempts to model network reachability have been made, feasible solutions to computing network reachability have remained unknown. In this paper, we propose a suite of algorithms for quantifying reachability based on network configurations (mainly ACLs) as well as solutions for querying network reachability. We present a comprehensive network reachability model that considers connectionless and connection-oriented transport protocols, stateless and stateful routers/firewalls, static and dynamic NAT, PAT, etc. We implemented the algorithms in our network reachability analysis tool called Quarnet and conducted experiments on a university network. Experimental results show that the offline computation of reachability matrices takes a few hours and the online processing of a reachability query takes 0.075 seconds on average.

XI. Bounded reachability checker for compositional linear hybrid systems

Existing reachability analysis techniques are easy to fail when applied to large compositional linear hybrid systems, since their memory usages rise up quickly with the increase of systems' size. To address this problem, we propose a tool BACH 2 that adopts a path-oriented method for bounded reachability analysis of compositional linear hybrid systems. For each component, a path is selected and all selected paths compose a path set for reachability analysis. Each path is independently encoded to a set of constraints while synchronization controls are encoded as a set of constraints too. By merging all the constraints into one set, the path-oriented reachability problem of a path set can be transformed to the feasibility problem of this resulting linear constraint set, which can be solved by linear programming efficiently. Based on this path-oriented method, BACH 2 adopts a shared label sequence guided depth first search (SLS-DFS) method to perform bounded reachability analysis of compositional linear hybrid system, where all potential path sets within the bound limit are identified and verified one by one. By this means, since only the structure of a system and the recently visited one path in each component need to be stored in memory, memory consumption of BACH 2 is very small at runtime. As a result, BACH 2 enables the verification of extremely large systems, as is demonstrated in our experiments.

XII. A Combined Approach for Reachability Analysis

This paper presents the principle of an approach that allows analyzing most important system properties like reachability, executable systems, deadlock-freeness etc. The approach is based on two concepts: analysis purpose-directed analysis and specification unfolding. The first concept drives the analysis to the aimed specification part whereas the second allows presenting the specification behavior in a suitable form for the analysis. In contrast to the common analysis methods, the present approach considerably alleviates the state-explosion problem.

XIII. Reachability Analysis in Dynamically Routed Networks

A novel approach to reachability analysis of dynamically routed networks is introduced. The goal is to determine the network-wide reachability using static analysis of configuration files gathered from forwarding devices. We describe a method that can compute the reachability in networks with a mix of static routing configurations, distance

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

vector routing protocols, filtering routing updates and redistributions. The method computes a network-wide approximation of distributed routing information using the standard graph algorithms. Thus, for any network state, we can determine a set of active paths used for packet delivery. The outcomes of the method can be, for instance, used during the conformance checking of distributed access control lists against network security policies.

REFERENCES

[1] Jiacum Wang,"Reachability analysis of real-time systems using Petrinets",IEEE Transactions on Cybernetics ,Volume 30 , Issue 5.

[2] A. Biere, A. Cimatti, E. Clarke, and Y. Zhu, "Symbolic model checking without BDDs", In Tools and Algorithms for the Construction and Analysis of Systems,(TACAS'99),Springer, 1999.

[3] Ahmed Bouajjani, Javier Esparza, Stefan Schwoon, "Reachability Analysis of multithreaded software with Asynchronous Communication",Institute for Formal Methods in Computer Science.

[4] http://www.sciencedirect.com/science/article/pii/S1571066110000526

[5] Naeem Esfahani,Sam Malek,"A learning Based Framework for Engineering Feature Oriented Self Adaptive Software Systems", IEEE Transactions on Software Engineering, Vol.39,No.11,November 2013.

[6] Dongsun Kim, Sunghun Kim,"Where Should We Fix This Bug? A Two-Phase Recommendation Model", IEEE Transactions on Software Engineering, Vol.39,No.11,November 2013

[7] Hafedh Mili, Fatma Mili and Ali Mili , "Reusing Software: Issues and Research Directions", IEEE Transactions on Software Engineering, Volume 21 No. 6, June 1995.

[8] Negar Hariri,Jane Cleland,"Supporting Domain Analysis through Mining and Recommending Features from Online Product Listings", IEEE Transactions on Software Engineering, Vol.39,No.12,December 2013