International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 3 Issue: 12 6753 - 6755

A Survey on Program Slicing

Priya Nagargoje
Computer Engineering
MAEER"S MIT Pune
PUNE, India
priya.nagargoje58@gmail.com

V. S. Jagatap
Computer Engineering
MAEER"S MIT Pune,
PUNE, India
vandana.jagtap@mitpune.edu.in

***Abstract*:**-Program slicing is an important technique for untangling programs by only focusing on selected aspects of semantics. The processing flow of slicing deletes those parts of the program that have no effect upon the semantics that are required to execute. For program slicing it is important to understand the important aspects that are related to execution and relationship of variable involved in the program. Slicing has applications in software maintenance, testing and debugging.

Program slicing is a process of extracting parts of programs by tracing the programs in which the main task is to find out all statements in a program that directly or indirectly influence the value of a variable at some point in a program. In proposed paper a detailed survey is done on various slicing techniques and understanding the applications in various areas such as debugging, program comprehension and understanding, program integration,

Keywords:- *Program slicing, debugging, software maintenance, semantics*

*****

I. INTRODUCTION

Slicing has many application areas for example area of software engineering and development, software maintenance, debugging, program comprehension in which it is helpful to extract subprograms based upon arbitrary semantic criteria are potential applications of slicing.

There are different types of slicing techniques but the main slicing techniques are static slicing and dynamic slicing.

Static Slicing
Static program slicing is an established method for analyzing sequential programs, especially for program understanding, debugging and testing.

Dynamic Slicing
Construction of a dynamic slice of object oriented programs with respect to object oriented programs is similar to dynamic slicing of procedural programs because both can be solved by dynamic control flow and data flow analysis but because of features like inheritance and dynamic binding it becomes more complex to trace dependencies in object oriented programs.

II. LITERATURE SURVEY

[1]In this paper author described a compiler and virtual machine infrastructure as the context for research in automatic program partitioning and optimization for distributed execution.

Author described a program partitioning as the process of decomposing a program into multiple tasks. The main purpose of proposed design is to enable experimenting with optimizing program execution on resource-constrained devices with respect to not only memory consumption but also CPU time, battery lifetime and communication.

Methodology Used:

Author represented program as a graph of class instances and their dependences and assign the graph with respective weights that represent resource consumption. A general graph partitioning algorithm assigns nodes to abstract processors according to the resource constraints.

[2]

In this paper author introduced a system that takes large Java programs as its input and then automatically converts them into distributed Java programs.

In this paper research is done with consideration of resource-constrained mobile devices such that the system should perform on platform with minimum energy consumption and also minimize total execution time.

Here author presented two contributions to automatic partitioning for object-oriented programs, the first contribution is a two-layer graph modeling approach that gives uniformed partitioning for multiple distribution objectives. The second contribution is the highly automation in the code distribution process without user's interaction.

Methodology Used:

This method initially constructs an object relation graph (ORG), using a combination of static analysis and offline profiling. Instead of directly partitioning of object relation graph it then transformed it into a target graph (TG). By applying this two-layer graph modeling, author achieved an uniformed strategy for different partitioning goals.

[3]
In this paper authors proposed a new principled approach to building web applications that are secure by construction. Author used java code for construction of web application but before that author mentioned that some web applications, functionality is usually implemented as client-side if code written in JavaScript and moving code and data to the client can create security vulnerabilities, so author proposed that , security-critical code and data should placed on the server but it raises performance issue so for that author stated that code and data can also be replicated across the client and server, to obtain both security and performance.

A max-flow algorithm is used to place code and data in a way that minimizes client–server communication.

Methodology Used:
Here author used Java code for application development this java code is annotated with information flow policies that specify the confidentiality and integrity of web application information. The compiler uses these policies to automatically partition the program into JavaScript code running in the browser, and Java code running on the server

[4]
In this paper author proposed an approach that automatically parallelizes Java programs at runtime. Proposed approach is purely software based and built on Java virtual machine and can achieve parallelization without profiling execution.

Author evaluated some experimental result which indicates that good speedup can be achieved for real-world Java applications that exhibit data parallelism with very small overhead.

After evaluating the experimental results, author concluded that the proposed approach has low overheads and achieves competitive speedups compared to manually parallelized code.

Methodology Used:
The approach collects online trace information during program execution, and dynamically recompiles methods that can be executed in parallel.

Proposed approach also described a cost benefit model that makes intelligent parallelization decisions, as well as a parallel execution environment to execute parallelized code.

[5]
This paper describe the slicing technique which is language-independent called Observation-Based Slicing (ORBS)

In ORBS a potential slice obtained through repeated statement deletion which is validated by observing the behavior of the program: if the slice and original program behave the same under the slicing criterion, the deletion is accepted.

Here author evaluated five variants of ORBS on ten programs of different sizes and languages which shows that ORBS is less expensive than existing techniques.

The main advantage is that it is not only perform slices for systems written in multiple languages, including systems which may contain third party binary components or libraries

Methodology Used:
ORBS use statement deletion as its primary operation and observation as its validation criteria. The approach leverages existing tool chains, making it better suited to execution reality than previous slicing approaches based on models of dependence and semantics.

III. CONCLUSION

In this paper we have studied different slicing methods with different precision and accuracy. Each slicing methods has its own advantage for example object relation graph contains two phases which helps to achieve uniformed strategy for different partitioning goals. Also Observation-Based Slicing (ORBS) which perform slices for systems written in multiple languages

In conclusion this paper shows that while slicing algorithms should be precise and are attractive option instead of manually parallelization of programs.

ACKNOWLEDGMENT

I wish to express my sincere thanks to the guide Prof. V.S. Jagtap and Head of Department, Prof. V.Y. Kulkarni , as well as our principal DR L. K KShirsagar, also Grateful thanks to our PG Coordinator M.V. Bedekar and last but not least, the departmental staff members for their support.

REFERENCES

[1] Automatic Distribution of Java Byte-Code Based on Dependence Analysis Roxana E. Diaconescu, Lei Wang, Michael Franz University of California, Irvine Department of Computer Science roxana, wangglei, franz @ics.uci.edu

[2] Automatic Partitioning of Object-Oriented Programs for Resource-Constrained Mobile Devices with Multiple Distribution Objectives Lei Wang and Michael Franz Department of Computer Science, University of California, Irvine Irvine, CA 92717, USA {leiw,franz}@uci.edu.

[3] Secure Web Applications via Automatic Partitioning Stephen Chong Jed Liu Andrew C. Myers Xin QiK. Vikram Lantian Zheng Xin Zheng Depar tment of Computer Science Cornell University {schong,liujed,andr u,qixin,kvikram,zlt,xinz}@cs.cor nell.edu

[4] On-line Trace Based Automatic Parallelization of Java Programs on Multicore PlatformsYu Sun and Wei Zhang Department of ECE, Virginia Commonwealth University wzhang4@vcu.edu

[5] ORBS: Languag e-Independent Program Slicing David Binkley Nicolas Goldy Mar k Har many Syed Islamy Jens Kr inkey Shin Yooy Loyola University Mar yland Baltimore, USA yUniversity College London London, UK

[6] Cost and Precision Tradeoffs of Dynamic Data Slicing Algor ithms XIANGYU ZHANG and RAJIV GUPTA The University of Ar izona andYOUTAO ZHANG University of Texas

[7] Automatically Partition Software into Least Privilege Components using Dynamic Data Dependency Analysis Yongzheng Wu and Jun Sun Singapore University of Technology and Design fyongzheng wu,sunjung@sutd.edu.sg Yang Liu Nanyang Technological University yangliu@ntu.edu.sg Jin Song Dong School of Computing National University of Singapore dongjs@nus.edu.sg IEEE Press, Dec. 2007, pp. 57-64, doi:10.1109/SCIS.2007.357670

[8] Specialization Slicing MIN AUNG, SUSAN HORWITZ, and RICH JOINER , U niversity o f Wisconsin THOMAS REPS , U niversity o f Wisconsin and G ramma Tech, Inc.

.