ICSE Workshop on Dynamic Analysis (WODA 2003)

Jonathan E. Cook
Department of Computer Science
New Mexico State University
Las Cruces, NM 88003 USA
jcook@cs.nmsu.edu

Michael D. Ernst
MIT Lab for Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139 USA
mernst@lcs.mit.edu

Abstract

Dynamic analysis of software systems has long proven to be a practical approach to gain understanding of the operational behavior of the system. This workshop will bring together researchers in the field of dynamic analysis to discuss the breadth of the field, order the field along logical dimensions, expose common issues and approaches, and stimulate synergistic collaborations among the participants.

1. Introduction

Dynamic analysis is a term that captures all the many techniques that attempt to reason about the observed dynamic behavior of systems [1]. Some examples of proven and common dynamic analysis techniques include assertion checking, memory allocation monitors, and profilers. Dynamic analysis techniques have a proven track record of being useful and practical across the software engineering field.

Dynamic analysis includes both offline techniques that might take intensive compute time and operate on a trace or some other captured representation of the system's behavior, and runtime techniques that operate while the system is producing its behavior. Some techniques may combine these two approaches, performing preliminary runtime data reduction computations for later offline, compute-intensive analyses. The thread that binds analyses under the dynamic analysis title is the use of observed behavior of the system in the analyses.

2. Workshop topic areas

Throughout the various dynamic analysis techniques run common themes and common needs, some of which are: instrumentation; data collection, description, and management; behavior descriptions; uncertainty; partial information; and reasoning with partial models. Issues that the workshop may cover in these areas are detailed below.

2.1. Data collection and management

In order to reason about the behavior of a system, some mechanism must exist to observe the behavior of the system and either collect those observations for later processing or reason about them in real time. Thus, dynamic analysis begins and depends on instrumentation and data collection techniques. Instrumentation carries with it the potential to modify the behavior of the system, and heavyweight instrumentation can produce incredible amounts of data very quickly, enough to overwhelm even today's storage capacities and processor speeds. Thus, adaptive and specialized instrumentation mechanisms are still needed for effective broad-based data analysis capabilities, and optimized algorithms for handling large amounts of data in real-time will need developed.

2.2. Data description

Virtually all data collected as observations of an executing system can be characterized as *events*. Methods which analyze program behavior within the context and knowledge of their implementation language may have events that correspond to expression evaluation, assignment, condition testing, function or method calls, and other language constructs. These events relate to each other both sequentially and hierarchically, within the language's grammar constraints. Other analysis methods may only look at a distributed system's messages on the network, which relate to each other according to some protocol model, but not hierarchically.

Proceedings of the 25th International Conference on Software Engineering (ICSE'03)
0270-5257/03 $17.00 © 2003 IEEE

However, different dynamic analysis methods can require different levels of richness in their event data, and some reduction methods may aggregate the event data into other non-event forms. Common data descriptions and sharing methods still need developed.

2.3. Relationship to testing

Dynamic analysis shares a unique relation to testing, in that it generally works best when used with observed behaviors of the system that capture a representative sample of the whole domain of possible behavior. Testing techniques often aim to provide exactly that sample, in the most efficient manner possible. Testing can benefit from dynamic analysis in that not only can the output of the system be verified, but the observed intermediate behaviors can also be verified, providing more confidence in the system without requiring more testing. Thus, testing research and dynamic analysis research can benefit from each other.

2.4. Symbolic versus probabilistic reasoning

Dynamic analysis techniques can be split between symbolic and theoretically exact reasoning, and statistical and probabilistic reasoning. The former category produces exact, justifiable results over its input, similar to static analysis techniques but only over the observed behaviors. The latter attempts to characterize its results in terms of what is statistically likely to happen, or how probable some aspect of the behavior is. While not exact, these approaches can offer large data reduction and computation reduction benefits. Each approach offers its own unique contributions to system understanding.

2.5. Incorporating reasoning over models

In addition to data analysis, many dynamic analysis methods attempt to understand the behavior within the context of a model of the system. This might be in the form of hard-coded rules (e.g., array bounds checking, memory allocation monitors), or might be derived from static analysis of the system (e.g., path coverage testing). In either case, reasoning in the presence of either a partial model or partial behavioral information is needed. A partial model might be used because only one aspect of the system is being investigated, or it is too expensive to construct a complete model. Partial behavioral information might be used because it is too expensive or impossible to collect full information. Whether working with full or partial models or information, efficient algorithms for dynamic analysis are important, and results from the static analysis research field can often be leveraged.

3. Workshop scope and format

This workshop will focus on achieving a consensus among the participants as to the structure of the field, the important research directions this field should take, inputs needed from other research domains, and outputs that could benefit other research domains. It will cover a topical spectrum from enabling technologies; framework and common tool support; event type definition, classification, and specification; symbolic and theoretically exact reasoning techniques; statistical and probabilistic reasoning techniques; research foundations; relationships to static analysis; relationships to testing; and other potential topics.

Selected authors from those with accepted position papers will be asked to present their position as an opening for a discussion period over one of the subtopics. Focus in the workshop will be on discussion and the illumination of needed directions for future research and on recognizing potential collaborations.

4. Workshop output

This workshop will result in participants having new ideas for future research, identifying new potential collaborations, and creating a better general understanding of the scope and structure of the field of dynamic analysis. All accepted position papers will be available on the Web at

http://www.cs.nmsu.edu/∼jcook/woda2003/

and the workshop will produce a report to be printed in the ACM Software Engineering Notes.

Acknowledgements

This work was supported in part by the National Science Foundation under grants EIA-9810732 and EIA-0220590. The content of the information does not necessarily reflect the position or the policy of the Government and no official endorsement should be inferred.

References

[1] T. Ball. The Concept of Dynamic Analysis. In *Proceedings of the Seventh European Software Engineering Conference / Seventh ACM SIGSOFT Symposium on the Foundations of Software Engineering*, pages 216–234. Springer-Verlag, Sept. 1999.