Specification-based Error Recovery: Theory, Algorithms, and Usability

Sarfraz Khurshid
University of Texas at Austin

FEBRUARY 2013
Final Report

DISTRIBUTION A: Approved for public release.
This project laid the foundation for a novel methodology for correcting erroneous program executions using specifications at run-time. The basis of the methodology is a view of the specification as a non-deterministic implementation, which may permit a high degree of non-determinism. The key insight is to use likely correct actions by an otherwise erroneous execution to prune the non-determinism in the specification, thereby transmuting the specification to an implementation at run-time and reducing the performance overhead. A suite of techniques and tools were designed, developed, optimized and rigorously evaluated in this project. It leveraged the Alloy specification language and its SAT-based tool-set as an enabling technology for specification-based analysis. The ideas, techniques, tools, and evaluation results from this project contributed in part to archival publications, Masters theses, and PhD dissertations.


Specification-based Error Recovery: Theory, Algorithms, and Usability
FA9550-09-1-0351
Final Report (May 1, 2009 to June 30, 2012)

Principal Investigator: Sarfraz Khurshid
PI Email: khurshid@ece.utexas.edu
PI Tel: (512) 471-8244
Program Manager: Dr. Robert Bonneau

1 Summary

This project laid the foundation for a novel methodology for correcting erroneous program executions using specifications at run-time. The basis of the methodology is a view of the specification as a non-deterministic implementation, which may permit a high degree of non-determinism. The key insight is to use likely correct actions by an otherwise erroneous execution to prune the non-determinism in the specification, thereby transmuting the specification to an implementation at run-time and reducing the performance overhead. A suite of techniques and tools were designed, developed, optimized and rigorously evaluated in this project. It leveraged the Alloy specification language and its SAT-based tool-set as an enabling technology for specification-based analysis. The ideas, techniques, tools, and evaluation results from this project contributed in part to 44 archival publications, 4 completed Masters theses, and 3 completed PhD dissertations. This project funded in part 8 graduate students, including 3 female students.

2 Annual summaries

2.1 Reporting period: 05/01/2009 – 04/30/2010

During the first year of the project, the following research contributions were made:
• Contract-based data structure repair – Introduced the idea of using rich behavioral contract specifications including invariants, pre- and post-conditions as the basis of systematic data structure repair.

• Repair algorithms – Developed four algorithms that embody the idea. The algorithms leverage MIT’s Alloy tool-set to provide systematic repair, and employ heuristics to optimize performance.

• Similarity metric – Used a distance metric for graph similarity to compute the effect of repair on an erroneous program state and to evaluate different algorithms for effectiveness.

• Evaluation – Conducted an experimental evaluation of the feasibility of contract-based repair and demonstrated the promise it holds.

A basic technique embodying these ideas and and experimental evaluation were presented at the 24th European Conference on Object-Oriented Programming (ECOOP) in June 2010; a pre-print version of the paper is submitted along with this report.

2.2 Reporting period: 05/01/2010 – 04/30/2011

During the second year of the project, our primary research contribution was on program repair using data structure repair. A key element of the “Usability” thrust of our project is to design a repair feedback mechanism to help users debug their code or specifications. We developed a novel mechanism for translating repair actions performed on an erroneous program state into code that abstracts those actions using assignment statements that may replace existing program statements or be added as new statements. These statements serve as debugging suggestions, which the user can choose to apply or ignore. Details of this approach and an experimental evaluation were presented at the IEEE 4th International Conference on Software Testing, Verification and Validation (ICST) in March 2011; a pre-print version of the paper is submitted along with this report.

2.3 Reporting period: 05/01/2011 – 04/30/2012

During the third year of the project (May 1, 2011 to April 30, 2012), our primary research contribution was to develop a new technique to enhance our core approach for data structure repair to scale better. Our insight into scalability is two-fold: (1) the dynamic program trace of field writes and reads provides useful guidance to repair incorrect state mutations by a
faulty program; and (2) unsatisfiable cores generated by SAT can capture the history of previous runs, which can be used in an efficient iterative approach on successive problems with increasing state spaces. Details of this technique and an experimental evaluation were presented at the 18th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS) in March 2012; a pre-print version of the paper is submitted along with this report.

Additionally, we utilized unsatisfiable cores in another novel technique, which was for fault localization – the problem of locating faults in the source-code of a buggy program. Specifically, we developed a specification-based technique that utilized correct and erroneous executions of the buggy program to more accurately locate faults. Our insight is that unsatisfiability analysis of violated specifications, enabled by SAT technology, can help (1) compute unsatisfiable cores that contain likely faulty statements; and (2) generate tests that help spectra-based localization. Details of this technique and an experimental evaluation were presented at the 27th IEEE/ACM International Conference on Automated Software Engineering (ASE) in September 2012; a pre-print version of the paper is submitted along with this report.

2.4 Reporting period: 05/01/2012 – 06/30/2012

During the final two months of the project (May 1, 2012 to June 30, 2012), we focused on enhancing the ideas, analyses, and implementations we developed in this project to integrate them as parts of doctoral dissertations – we expect three future doctoral dissertations to use the work done in this project at their foundation.

3 Archival publications

3.1 Published after the end of the funding period


3.2 Published during the funding period


28. Muhammad Zubair Malik. Dynamic Shape Analysis of Program Heap using Graph Spectra (NIER Track). In Proc. 33rd International Con-
ference on Software Engineering (ICSE), pages 952–955, Waikiki, Honolulu, May 2011


4 Masters theses – finished during the funding period


5 PhD theses – finished during the funding period

