LANGUAGE-BASED INTERACTIVE TESTING

Yishuai Li

A DISSERTATION

in

Computer and Information Science

Presented to the Faculties of the University of Pennsylvania

in

Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

2022

Supervisor of Dissertation

Benjamin C. Pierce
Professor of Computer and Information Science

Graduate Group Chairperson

Mayur Naik, Professor of Computer and Information Science

Dissertation Commitee

Steve Zdancewic, Professor of Computer and Information Science, Chair
Mayur Naik, Professor of Computer and Information Science
Boon Thau Loo, Professor of Computer and Information Science
John Hughes, Professor of Computing Science, Chalmers University of Technology

LANGUAGE-BASED INTERACTIVE TESTING

Acknowledgments

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

ABSTRACT

LANGUAGE-BASED INTERACTIVE TESTING

Yishuai Li
Benjamin C. Pierce

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Contents

List of Figures

CHAPTER 1

Introduction

The security and robustness of networked systems rest in large part on the correct behavior of various sorts of servers. This can be validated either by full-blown verification or model checking against formal specifications, or less expensively by rigorous testing.

Rigorous testing requires a rigorous specification of the protocol that we expect the server to obey. Protocol specifications can be written as (i) a *server model* that describes *how* valid servers should handle messages, or (ii) a *property* that defines *what* server behaviors are valid. From these specifications, we can conduct (i) *model-based testing* [1] or (ii) *property-based testing* [3], respectively.

When testing server implementations against protocol specifications, one critical challenge is *nondeterminism*, which arises in two forms—we call them (1) *internal nondeterminism* and (2) *network nondeterminism*:

(1) *Within* the server, correct behavior may be underspecified. For example, to handle HTTP conditional requests [2], a server generates strings called entity tags (ETags), but the RFC specification does not limit what values these ETags should be. Thus, to create test messages containing ETags, the tester must remember and reuse the ETags it has been given in previous messages from the server.

(2) *Beyond* the server, messages and responses between the server and different clients might be delayed and reordered by the network and operating-system buffering. If the tester cannot control how the execution environment reorders messages—*e.g.,* when testing over the Internet—it needs to specify what servers are valid as observed over the network.

These sources of nondeterminism pose challenges in various aspects of testing network protocols: (i) The *validation logic* should accept various implementations, as long as the behavior is included in the specification's space of uncertainties; (ii) To capture bugs effectively, the *test harness* should generate test cases based on runtime observations; (iii) When *shrinking* a counterexample, the test harness should adjust the test cases based on the server's behavior, which might vary from one execution to another.

To address these challenges, I introduce symbolic languages for writing specifications and representing test cases:

(i) The specification is written as a reference implementation—a nondeterministic program that exhibits all possible behavior allowed by the protocol. Inter-implementation and inter-execution uncertainties are represented by symbolic variables, and the space of nondeterministic behavior is defined by all possible assignments of the variables.

The validation logic is derived from the reference implementation, by *dualising* the server-side program into a client-side observer.

(ii) Test generation heuristics are defined as computations from the observed trace (list of sent and received messages) to the next message to send. I introduce a symbolic intermediate representation for specifying the relation between the next message and previous messages.

(iii) The symbolic language for generating test cases also enables effective shrinking of test cases. The test harness minimizes the counterexample by shrinking its symbolic representation. When running the test with a shrunk input, the symbolic representations can be re-instantiated into request messages that reflect the original heuristics.

Thesis claim. Symbolic abstract representation can address challenges in testing networked systems with uncertain behavior. Specifying protocols with symbolic reference implementation enables validating the system's behavior systematically. Representing test input as abstract messages allows generating and shrinking interesting test cases. Combining these methods result in a rigorous tester that can capture protocol violations effectively.

This thesis is structured as follows:

Bibliography

[1] Manfred Broy et al. “Model-based testing of reactive systems”. In: *Volume 3472 of Springer LNCS*. Springer. 2005.

[2] Roy T. Fielding and Julian Reschke. *Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests*. RFC 7232. June 2014. DOI: `10.17487/RFC7232`. URL: `https://rfc-editor.org/rfc/rfc7232.txt`.

[3] George Fink and Matt Bishop. “Property-Based Testing: A New Approach to Testing for Assurance”. In: *SIGSOFT Softw. Eng. Notes* 22.4 (July 1997), pp. 74–80. ISSN: 0163-5948. DOI: `10.1145/263244.263267`. URL: `https://doi.org/10.1145/263244.263267`.