Runtime Tests for Dafny

Authors

Institutes

Abstract. Dafny is a verification-aware programming language that can automatically prove that the implementation of a particular method satisfies its specification. Dafny provides compilers for generating executable code in different target languages, such as C# or Java. A target-language executable generated from Dafny source must, in theory, exhibit the runtime behavior proven by Dafny verifier. In practice, however, compiler bugs and incorrect specification of external methods can lead to the executable producing a result that is different from the one expected. To help ensure that the verified behavior carries over to the target language, we propose to introduce runtime tests to Dafny. Supported features include mocking and parameterized tests.

Keywords: Dafny · Testing

1 Introduction

2 Motivation

An effective testing framework supports several strategies and levels of testing, which makes it versatile enough to be used at different stages of the development process. We built our testing framework around the need to support all of unit-, integration-, and system-level testing, with both automatic and manual test generation in mind.

In *unit-level testing*, the developer is only concerned with the correctness of a single method - all the outgoing method calls can be stubbed to mimic a particular scenario. We introduce this kind of testing to Dafny by supporting the automatic compilation of external methods that return mocked objects. The compiler uses the postconditions constraining the returned object in Dafny to set up the mock in the target language using the appropriate mocking library.

In *integration-level testing*, the developer is usually concerned with how several objects of different types interact with each other. The way these objects are constructed is irrelevant - only their internal state at the moment of interaction is. We introduce this kind of testing to Dafny by supporting the automatic compilation of external methods that return fresh objects of a requested type. This allows the developer to bypass the process of choosing an appropriate constructor and constructor arguments for initializing an object. Inferring appropriate constructor arguments is not a trivial task and the ability to bypass it is, therefore, also crucial for automatically-generated tests.

In *system-level testing*, the developer typically provides a sequence of API calls to ensure that they produce some desired result at runtime. This kind of testing does not require the addition of any special features to the Dafny language aside from those features that are shared by all runtime tests, as described below.

All runtime tests for Dafny make use of the *assertion library*, which is designed so that the behavior tested at runtime is also verified by Dafny before compilation.

3 Design and Implementation

4 Conclusions and Future Work

References