Tamarin: Concolic Program Disequivalence for MIPS

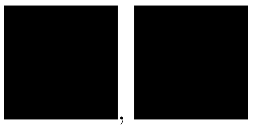
Abel Nieto

University of Waterloo
anietoro@uwaterloo.ca

Abstract. TODO

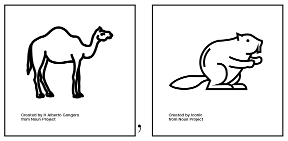
1 Introduction

We are staring at two opaque black boxes laying at our feet. Each box has a narrow slot through which we can place items in the box, but we cannot quite see what is inside. They look approximately like this:



We know each box contains an animal, but we do not know which specific animal is in each one. We would like to find out if both boxes contain the same species of animal. Our solution is simple: we take two carrots, and drop one in each box through the slots.

After a while, a chewing sound emerges from the boxes. We peer into them and, indeed, it looks like the carrots were successfully eaten. Triumphantly, we declare that the boxes contain the same species of animal. The truth is altogether different:


Created by H Alberto Gongora from Noun Project
Created by Iconic from Noun Project

The boxes are assembly programs. The animals are the functions those programs compute. The carrot is unit testing. The task was to determine whether the programs were equivalent. And we failed at it. In this paper, we show a technique that is better than the carrot.

Program equivalence. The program is the specification. The complications of assembly language.

2 Concolic Program Disequivalence

General Idea.

3 Tamarin

Overview.

3.1 Trace Collection

CPU instrumentation, PC concretization, error boxing, and fuel.

3.2 Transformations

Desugaring, simplification, trimming, and conversion to SSA.

3.3 Query Representation

Memory, jumps, arithmetic operators.

3.4 Concolic Execution Redux

Alternation. Compatibility. Soundness/Completeness. Efficiency.

4 Evaluation

5 Related Work

6 Conclusions