Bounded Exhaustive Search of Alloy Specification Repairs

Simón Gutiérrez Brida[2,4], Germán Regis[2], Guolong Zheng[1], Hamid Bagheri[1], ThanhVu Nguyen[5], Nazareno Aguirre[2,4], and Marcelo Frias[3,4]

[1] University of Nebraska-Lincoln, USA
[2] Universidad Nacional de Río Cuarto, Argentina
[3] Instituto Tecnológico de Buenos Aires, Argentina
[4] Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Argentina
[5] George Mason University, USA

Abstract

The rising popularity of declarative languages and the hard to debug nature thereof have motivated the need for applicable, automated repair techniques for such languages. However, despite significant advances in program repair for imperative languages, there is a dearth of repair techniques for declarative languages. We present BeAFix, an automated repair technique for faulty models written in Alloy, a first-order relational logic language.

BeAFix has a number of distinguishing features. Firstly, it supports *any* kind of oracle, including assertions typically found in formal specifications, as well as "specification tests". This is important since unit tests, widely available for programs, are not typically found in formal specifications. Secondly, given a defined set of mutation operations, a set of suspicious expressions and a maximum number of mutations to apply per expression, BeAFix's bounded-exhaustive approach will either find a fix, or guarantee that such a fix is not possible, within the provided bounds. With respect to fault localization, BeAFix is not tightly integrated to any specific technique. In fact, our technique is independent of the fault localization technique used, and the fault localization is run only once before the repair process begins.

To support a bounded-exhaustive approach while keeping repair times reasonable, sound state pruning techniques (i.e., those that guarantee that no valid fixes are removed) are introduced. When a faulty model has more than one suspicious expression, we use both syntactic analysis and dynamically generated scenario-based assertions to check the *feasibility* of a particular repair candidate. A failing check will determine that this candidate would never lead to a fully repaired model, allowing BeAFix to prune significant parts of the search space.

We evaluated our technique on two Alloy benchmarks, including one previously used in a state-of-the-art technique for Alloy repair. The results show that BeAFix is able to repair thousands of real-world faulty models, corroborating its ability to effectively, and efficiently generate correct repairs while also being less prone to overfitting than previous techniques.

This work was accepted for publication at the *43rd International Conference on Software Engineering ICSE 2021*, in May 2021. `https://doi.org/10.1109/ICSE43902.2021.00105`