Abstract Model Repair

George Chatzieleftheriou[1], Panagiotis Katsaros[1],
Borzoo Bonakdarpoor[2], and Scott A. Smolka[3]

[1] Department of Informatics, Aristotle University of Thessaloniki
54124 Thessaloniki, Greece
gchatzie, katsaros@csd.auth.gr
[2] School of Computer Science, University of Waterloo,
200 University Avenue West Waterloo N2L3G1, Canada
borzoo@cs.uwaterloo.ca
[3] Department of Computer Science, Stony Brook University
Stony Brook, NY 11794-4400, USA
sas@cs.sunysb.edu

Keywords: Model Repair, Model Checking, Abstraction Refinement

1 Abstract

Given a Kripke structure M and a CTL formula ϕ, *model checking* is the problem of determining if the formula ϕ is satisfied in M. An extended problem of model checking is that of *model repair*. Given a Kripke structure M and a CTL formula ϕ, such that ϕ is not satisfied in M, the problem of model repair is to find a new model M' which satisfies ϕ. Additionally, the aim of model repair is to find repaired models with the minimal changes with respect to the initial model M.

State explosion problem is the main limitation of applying automated formal methods such as model checking in large systems. The most promising method for fighting state explosion problem is the use of *abstraction*. The main idea is to use a smaller abstract model $\hat{M}$ for which if $\hat{M} \models \phi$ then it holds that $M \models \phi$. This means that the construction of abstract model should be done in order to fill this requirement.

Using as inspiration the success of abstraction-based model checking, we proposed a framework for using *abstraction refinement* in the model repair problem with the aim of making the repair process feasible for complex systems with a large state space. We have started proposing an Abstract Model Repair (AMR) methodology in [1].

The main contributions of our previous and ongoing work on Abstract Model Repair are the following:

– Our AMR framework uses Kripke Structures (KSs) as the concrete model, Kripke Modal Transition Systems (KMTSs) as the abstract model, and 3-valued semantics of CTL for interpreting formulas over KMTSs. Refinement takes place repeatedly on the abstract model until one abstract KMTS is found such that it violates the CTL property. In this case, the property is also false in the concrete KS and the repair process for the abstract model starts.

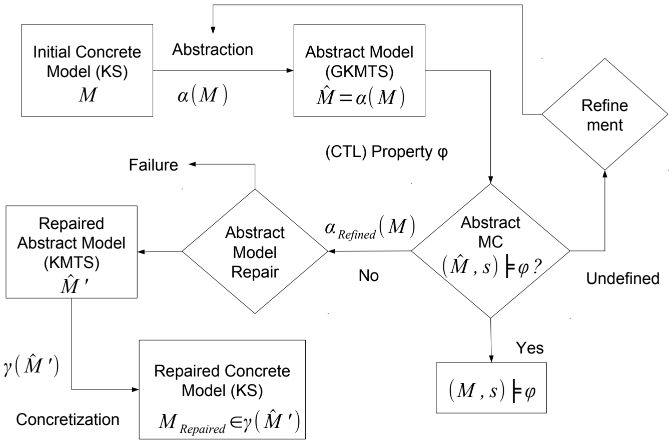

Fig. 1: Abstract Model Repair Framework.

- In order to take into account the minimality of changes criterion of the model repair problem, we formulate a distance metric for KSs based on the number of differences in the state space, the number of differences in their transition relation and the number of common states with alter labeling.
- We present a recursive, syntax-directed AMR algorithm for KMTSs, where the repair of an abstract KMTSs is achieved by successive calls of basic repair functions for atomic formulas, logical connective and CTL operators.
- We prove that AMR algorithm is sound for the full CTL and complete for one large fragment of CTL. We also analyze algorithm's complexity reaching the conclusion the it is only based on the size of the smaller abstract KMTS.
- We implement a prototype tool for applying our method to case studies and we get results that shows experimentally a significant speed-up of the repair process with respect to concrete model repair solutions.
- We try to introduce the use of Generalized Kripke Model Transition Systems [2] as the abstract model in our AMR framework, with the objective of eliminating cases that the refinement process leads to failure. Thus, our approach will be complete with respect to the refinement.

References

1. G. Chatzieleftheriou, B. Bonakdarpour, S. A. Smolka, and P. Katsaros. Abstract model repair. In *Proceedings of the 4th international conference on NASA Formal Methods*, NFM'12, pages 341–355, Berlin, Heidelberg, 2012. Springer-Verlag.
2. S. Shoham and O. Grumberg. Monotonic abstraction-refinement for CTL. In K. Jensen and A. Podelski, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, volume 2988 of *Lecture Notes in Computer Science*, pages 546–560. Springer Berlin / Heidelberg, 2004.