Information and Computation 247 (2016) 169–202

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

A framework for compositional verification of multi-valued systems via abstraction-refinement [☆]

Yael Meller [a], Orna Grumberg [a], Sharon Shoham [b,*]

[a] *Technion – Israel Institute of Technology, Israel*
[b] *Academic College of Tel Aviv Yaffo, Israel*

ARTICLE INFO

Article history:
Received 16 December 2013
Received in revised form 24 September 2015
Available online 7 January 2016

Keywords:
Multi-valued model checking
Mu-calculus
Bilattices
Mixed simulation
Refinement
Compositional model checking

ABSTRACT

We present a framework for fully automated compositional verification of μ-calculus specifications over multi-valued systems, based on abstraction and refinement.

In a multi-valued model of a system, both the system transitions and the state labels are assigned values from a lattice. We formalize our framework based on bilattices, consisting of a truth lattice and an information lattice. Formulas are interpreted on the truth lattice. The information lattice determines how *definite* the value is, in terms of the concrete system being modeled.

Our compositional approach views each component as an *abstraction* of the entire system and checks it separately. Only if all individual checks return *indefinite* values, the *parts of the components* which are responsible for these values, are composed and checked. If the latter check is still indefinite, a *refinement* of the multi-valued system is needed. Refinement is aimed at increasing the information level of model details.

1. Introduction

In this work we present a framework for fully automated compositional verification of μ-calculus specifications over multi-valued systems, based on multi-valued abstraction and refinement. Our interest in such a framework stems from the fact that multi-valued modeling is widely used in many applications of model checking. It is used both to model concrete systems more precisely and to define abstract models.

Multi-valued models enable a more precise modeling of concrete systems by distinguishing between several levels of uncertainty and inconsistency [5,10,6,24,39,37]. These models have been widely used for abstraction as well [54,55,2,31,36,32].

For example, 3-valued models are used to describe models with partial information [5], using *true* and *false* to model known values, and *indefinite* to model an unknown value. Such models can also be the result of *abstraction* which collapses together multiple states. In case one state has an outgoing transition in the concrete system and another does not, the transition will have an indefinite value in the abstract (3-valued) model. 4-valued models can model disagreement and their generalizations are used to handle inconsistent views of a system [24,39], by considering tuples of values, where each index in the tuple represents one view. Temporal logic query checking [10,6,37] can also be reduced to multi-valued model checking.

[☆] A preliminary version of this paper appeared in [49].
[*] Corresponding author.
E-mail addresses: ymeller@cs.technion.ac.il (Y. Meller), orna@cs.technion.ac.il (O. Grumberg), sharon.shoham@gmail.com (S. Shoham).

http://dx.doi.org/10.1016/j.ic.2016.01.001
0890-5401/© 2016 Elsevier Inc. All rights reserved.

Multi-valued models, both abstract and concrete, may still suffer from the *state explosion problem*. Two of the most successful approaches for fighting this problem in classical (2-valued) model checking are abstraction-refinement and compositional verification. *Abstraction-refinement* is an iterative process, in which a model is abstracted by removing or simplifying details, model checked, and if an inconclusive result is obtained due to the abstraction, the abstract model is refined by adding more details into it. In *compositional* model checking, parts of the system are verified separately in order to avoid the construction of the entire system. Typically, some information needs to be exchanged between these checks in order to enable verification of the system.

In this work, we develop a compositional multi-valued model checking based on abstraction and refinement.

The first step we take in formalizing our multi-valued framework is to consider bilattices [26] as part of our framework. A bilattice defines two lattices over a given set of elements: the *truth lattice* and the *information lattice*, each accompanied with an order. Formulas interpreted over a multi-valued model are evaluated with respect to the truth lattice. On the other hand, the relation of "more abstract" over models is based on the information lattice: Roughly, a model M_2 is more abstract than a model M_1 if values of atomic propositions labeling states and values of transitions between states in M_2 are smaller or equal by the information order than the corresponding values in M_1. Consequently, the valuation of a formula in M_2 will be smaller or equal by the information order than its value in M_1. In fact, since we consider the full μ-calculus (which combines existential and universal quantifiers), a bidirectional correspondence between transitions of M_1 and M_2 is needed. To capture this bidirectional correspondence, we define a mixed-simulation relation, based on the information lattice.

Bilattices provide a natural way to identify lattice elements that are *consistent*, meaning that they represent some concrete elements of the bilattice (to be formalized later). We can also identify elements that are *definite*. Those are the elements that represent conclusive results and need not be refined anymore. In most of the work we restrict the discussion to Consistent Partial Distributive Bilattices (CPDB), which consist of exactly all the consistent elements. In Section 7 we consider also full distributive bilattices. In particular, we discuss the interesting special case of the 4-valued Belnap bilattice.

We attempt to address compositional verification for our context in a similar manner to [58]. There, abstraction and compositional verification are joined in the context of 3-valued abstraction: each component M_i of a composed system M is lifted into a 3-valued model $M_i\uparrow$ which forms an abstraction of M. Model checking a formula φ on $M_i\uparrow$ can result in either a definite value *true* or *false*, or an *indefinite* value. In the former case, it is guaranteed that the result is also the value of φ on M. In the latter case, however, nothing can be deduced about the composed system. If the checks of all individual components return *indefinite* values, then the *parts of the components* which are responsible for these values are identified, composed, and model checked. Thus, the construction of the fully composed system is avoided. Finally, if the composed system is in itself abstract, the check of the partially composed system might still be indefinite, in which case a *refinement* is applied to each component separately.

For our multi-valued framework, once we establish our setting by means of bilattices, we can fill in the rest of the framework's ingredients. First, we define the notion of *composition* of multi-valued systems. Next, for model checking, we use the model checking algorithm for multi-valued systems and the alternation-free μ-calculus, suggested in [57]. We also show, in case the checks on individual components are indefinite, how to identify, compose, and check the parts of the models that are needed for the checked formula. As we exemplify later, the resulting composed system is often much smaller than the full composed system. Finally, we develop a heuristic for finding a *criterion for refinement*, in case the model checking of the composed system returns an indefinite result.

In the framework above we do not discuss the construction of multi-valued abstract models. This is investigated for instance in [38], which presents a methodology for a systematic construction of an abstract model from a given concrete one.

Other works deal with several aspects of multi-valued model checking (as discussed in Section 8), but to the best of our knowledge none investigates a compositional approach. Our framework for compositional multi-valued model checking is applicable to any multi-valued model defined over a CPDB. We also show the applicability of our approach to multi-valued models defined over full distributive bilattices, with STE [54] and YASM [36] as concrete examples. We consider specifications given as μ-calculus formulas, but with certain adaptations, different logics can be handled as well.

To summarize, the main contributions of this work are:

- We present a framework for fully automated compositional verification of multi-valued systems with respect to μ-calculus specifications. The framework is based on multi-valued abstraction-refinement. To the best of our knowledge, this is the first compositional approach for multi-valued model checking.
- We apply our framework to the alternation-free μ-calculus model checking algorithm. In particular, we develop an algorithm for refinement in this context.
- We formalize our framework based on bilattices, consisting of a truth lattice and an information lattice. This allows to naturally define the consistent and definite elements in the bilattice. It also provides a clear definition of abstraction and refinement in the multi-valued context. It thus provides a better understanding of the multi-valued framework.
- Based on the information order of a bilattice, we define a mixed simulation relation over multi-valued models, preserving μ-calculus specifications.