Science of Computer Programming 96 (2014) 354–376

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

Property-dependent reductions adequate with divergence-sensitive branching bisimilarity ☆

Radu Mateescu [a,*], Anton Wijs [b]

[a] Inria Grenoble – Rhône-Alpes and LIG/CONVECS team, 655, av. de l'Europe, F-38330 Montbonnot Saint Martin, France
[b] Technische Universiteit Eindhoven/MDSE section, Faculteit Informatica, Den Dolech 2, 5612 AZ Eindhoven, The Netherlands

HIGHLIGHTS

- We propose a method for reducing an LTS model w.r.t. a μ-calculus formula.
- The maximal set of actions is hidden without changing the formula meaning.
- We define a new μ-calculus fragment adequate with ds-branching bisimilarity.
- We apply maximal hiding and ds-bb reduction before checking formulas of this fragment.
- This yields important performance gains (speed and memory) for model checking.

ARTICLE INFO

Article history:
Received 29 March 2013
Received in revised form 28 March 2014
Accepted 7 April 2014
Available online 18 April 2014

Keywords:
Divergence-sensitive branching bisimulation
Labeled transition system
Modal μ-calculus
Model checking
On-the-fly verification

ABSTRACT

When analyzing the behavior of finite-state concurrent systems by model checking, one way of fighting state space explosion is to reduce the model as much as possible whilst preserving the properties under verification. We consider the framework of action-based systems, whose behaviors can be represented by labeled transition systems (LTSs), and whose temporal properties of interest can be formulated in modal μ-calculus (L_μ). First, we determine, for any L_μ formula, the maximal set of actions that can be hidden in the LTS without changing the interpretation of the formula. Then, we define L_μ^{dsbr}, a fragment of L_μ which is adequate w.r.t. divergence-sensitive branching bisimilarity. This enables us to apply the maximal hiding and to reduce the LTS on-the-fly using divergence-sensitive τ-confluence during the verification of any L_μ^{dsbr} formula. The experiments that we performed on various examples of communication protocols and distributed systems show that this reduction approach can significantly improve the performance of on-the-fly verification.

© 2014 Elsevier B.V. All rights reserved.

1. Introduction

Model checking [2] is a technique to systematically verify whether a system specification meets a given temporal property. Although successfully applied in many cases, its usefulness in practice is still hampered by the state space explosion phenomenon, which may entail high memory and CPU requirements in order to carry out the verification.

☆ This article is an extended version of the conference article [1].

* Corresponding author.
E-mail addresses: Radu.Mateescu@inria.fr (R. Mateescu), A.J.Wijs@tue.nl (A. Wijs).

http://dx.doi.org/10.1016/j.scico.2014.04.004
0167-6423/© 2014 Elsevier B.V. All rights reserved.

One way to improve the performance of model checking is to check the property at a higher level of abstraction; by abstracting parts of the system behavior away from the specification, its corresponding state space will be smaller, thereby easier to check. This can either be done globally, i.e., before verifying the property, or on-the-fly, i.e., during verification. However, one needs to be careful not to abstract away any details crucial for the outcome of the check, i.e., relevant for the property. This is known as *action abstraction* in action-based formalisms, where state spaces are represented by *Labeled Transition Systems* (Ltss), specifications are written using some flavor of *process algebra* [3], and temporal properties are described using an action-based temporal logic, such as the modal μ*-calculus* (L_μ) [4,5]. Abstracted behavior is then represented by some predefined action, denoted τ in process algebras. In the past, the main focus in this area has been on devising L_μ fragments adequate w.r.t. specific relations, such as Actl\X [6], which is adequate w.r.t. divergence-sensitive branching bisimilarity [7,8],[1] or weak L_μ [5], which is adequate w.r.t. weak bisimilarity [9]. For such fragments, the minimization of an Lts modulo the specific relation preserves the truth value of all formulas written in the adequate L_μ fragment. Other works focused on devising reductions targeted to specific formulas, such as those written in the *selective* L_μ [10]. For each selective L_μ formula, it is possible to hide all actions not occurring in the formula, and subsequently minimize the Lts modulo $\tau^*.a$ bisimilarity [11] before verifying the formula.

In this article, we propose two enhancements with respect to existing work. Firstly, starting from an arbitrary L_μ formula, we determine automatically the maximal set of actions which can be hidden in an Lts without affecting the truth value of the formula on the Lts. This yields the maximum potential for reduction, and therefore for improving the performance of model checking. After hiding, the Lts can be minimized, e.g., modulo strong bisimilarity without disturbing the outcome of the verification of the formula. This method is not intrusive, in the sense that it does not force the specifier to write formulas in a certain way (as it is the case, e.g., for the selective L_μ).

Secondly, to achieve further reduction of the Lts, we study the relationship between L_μ formulas and weak equivalence relations, which take into account the presence of transitions labeled by the invisible action τ. More precisely, we consider divergence-sensitive branching bisimilarity ($\approx_{br}^{ds}$), a weak equivalence relation that preserves the branching structure and the divergences (cycles of invisible transitions) of Ltss, and still can yield substantial reductions in practice. We identify a fragment of L_μ, called L_μ^{dsbr}, and prove its adequacy with $\approx_{br}^{ds}$, meaning that two Ltss are equivalent modulo this relation if and only if they satisfy the same set of L_μ^{dsbr} formulas (in our previous work [1], only the compatibility of L_μ^{dsbr} with $\approx_{br}^{ds}$ was shown, i.e., the fact that a L_μ^{dsbr} formula has the same truth value on two Ltss equivalent modulo $\approx_{br}^{ds}$). This enables us to reduce an Lts modulo $\approx_{br}^{ds}$ after applying maximal hiding and before checking a L_μ^{dsbr} formula, and thus to improve the performance of the overall verification process.

Finally, we show that L_μ^{dsbr} is equally expressive to μ-Actl\X [12], the extension of Actl\X with fixed point operators, and it subsumes the weak L_μ as well as (a relevant fragment of) the selective L_μ. Compared to these μ-calculi, which require that action formulas contain only names of visible actions, our L_μ^{dsbr} fragment also accepts the presence of the τ action, therefore providing additional flexibility in the specification of properties. Moreover, our adequacy result of L_μ^{dsbr} w.r.t. $\approx_{br}^{ds}$ also provides a proof of the adequacy of μ-Actl\X w.r.t. $\approx_{br}^{ds}$, which completes the previously known adequacy of this logic w.r.t. strong bisimulation [13,14].

We illustrate the reduction approach for L_μ^{dsbr} within the Cadp[2] verification toolbox [15]. The model checking of a L_μ^{dsbr} formula can be optimized generally in two ways: *globally*, by generating the Lts, then hiding the maximal set of actions according to the formula, and minimizing the Lts modulo strong or divergence-sensitive branching bisimilarity before checking the formula; and *locally* (or on-the-fly), by applying maximal hiding and reduction modulo divergence-sensitive τ-confluence simultaneously with the verification. The experiments we carried out on several examples of protocols and distributed systems, including a recent industrial case-study, show that these optimizations can yield significant performance improvements.

The rest of the article is organized as follows. Section 2 defines the formalisms and equivalence relations considered. Section 3 studies the maximal hiding of actions in an Lts w.r.t. a given L_μ formula. Section 4 introduces the L_μ^{dsbr} fragment, shows its adequacy with divergence-sensitive branching bisimilarity, and compares its expressiveness with other logics. Section 5 illustrates experimentally the model checking optimizations obtained by applying maximal hiding and reductions for L_μ^{dsbr} formulas. Section 6 gives concluding remarks and directions for future work. The proofs of all lemmas and propositions are given in Appendix A.

2. Background

Labeled transition system. We consider as interpretation model the classical Lts, which underlies process algebras and related action-based description languages. An Lts is a tuple $\langle S, A, T, s_0\rangle$, where S is the set of states, A is the set of actions (including the invisible action τ), $T \subseteq S \times A \times S$ is the transition relation, and $s_0 \in S$ is the initial state. The visible actions

[1] In fact, a distinction can be made between *divergence-sensitive branching bisimilarity* [6] and *branching bisimilarity with explicit divergence* [7,8]. Contrary to the former, the latter distinguishes deadlocks and livelocks, and the latter is the coarsest congruence contained in the former. In this paper, we use the latter, but refer to it with the (more classical) name of the former.

[2] See http://cadp.inria.fr.

Download English Version:

https://daneshyari.com/en/article/434134

Download Persian Version:

https://daneshyari.com/article/434134

Daneshyari.com