Journal of Logical and Algebraic Methods in Programming 96 (2018) 65–80

Contents lists available at ScienceDirect

Journal of Logical and Algebraic Methods in Programming

www.elsevier.com/locate/jlamp

Verification of finite-state machines: A distributed approach

Roberto Gorrieri

Dipartimento di Informatica — Scienza e Ingegneria Università di Bologna, Mura A. Zamboni, 7, 40127 Bologna, Italy

ARTICLE INFO

Article history:
Received 6 September 2017
Received in revised form 8 November 2017
Accepted 20 November 2017
Available online 31 January 2018

ABSTRACT

Finite-state machines, a simple class of finite Petri nets, are equipped with a truly concurrent, bisimulation-based, behavioral equivalence, called *team equivalence*, which conservatively extends classic bisimulation equivalence over labeled transition systems and which is checked in a distributed manner, without necessarily building a global model of the overall behavior. An associated distributed modal logic, called *basic team modal logic* (BTML, for short), is presented and shown to be coherent with team equivalence: two markings are team equivalent if and only if they satisfy the same BTML formulae.

1. Introduction

A finite-state machine (FSM, for short) is a simple type of finite Petri net [15,27,31] whose transitions have singleton pre-set and singleton, or empty, post-set; therefore, they are very similar to finite-state, labeled transition systems (LTSs, for short) [20], a class of models that are suitable for describing sequential, nondeterministic systems, and are also widely used as a semantic model for process algebras (see, e.g., [14]). On this class of models, there is widespread agreement that a very natural and convenient equivalence relation is bisimulation equivalence [26,23], an equivalence relation that can be verified efficiently for finite-state LTSs; more precisely, if m is the number of transitions and n is the number of states of the LTS, checking whether two states are bisimilar can be done in $O(m \log n)$ time [29].

Even if FSMs are the simplest distributed model of computation, the equivalence checking problem may be not easy. For instance, if we want to check if two markings m_1 and m_2 are *interleaving bisimilar*, (see, e.g., [15]), we have first to build two LTSs, one rooted in m_1 and the other rooted in m_2, usually called the *interleaving marking graphs*, and then to check whether these two rooted LTSs are bisimilar. However, such LTSs have a number of states that can grow exponentially with the size of the marked net, in particular w.r.t. the size of the involved markings, so that the equivalence checking problem is exponential, in general. This problem is shared by essentially all the equivalences that have been proposed in the literature for FSMs (see, e.g., [7,28,13,15]), because all these equivalences are defined directly over the markings of the net.

Our main goal is to define a new equivalence relation that can be computed in a distributed manner, without resorting to a global model of the overall behavior of the analyzed marked net. The initial observation is that a place in an FSM represents a sequential process type and the number of tokens in that place represents the number of currently available instances of that sequential process type. Since an FSM is so similar to an LTS, we propose to define bisimulation equivalence [26,23] directly over the set of places of the *unmarked* net. The advantage is that bisimulation equivalence is a relation on places, rather than on markings, and so much more easily computable; more precisely, if m is the number of net transitions and n is the number of places, checking whether two places are bisimilar can be done in $O(m \log (n+1))$ time, by adapting

E-mail address: roberto.gorrieri@unibo.it.

https://doi.org/10.1016/j.jlamp.2017.11.005
2352-2208/

the algorithm in [29]. Moreover, the resulting notion of bisimilarity enjoys the same properties of bisimulation over LTSs, i.e., it is coinductive and equipped with a fixed-point characterization [23,32,14].

After the bisimulation equivalence over the set of places has been computed once and for all, we can define, in a purely structural way, that two markings m_1 and m_2 are *team equivalent* if they have the same cardinality, say $|m_1| = k = |m_2|$, and there is a bisimulation-preserving, bijective mapping between the two markings, so that each of the k pairs of places (s_1, s_2), with $s_1 \in m_1$ and $s_2 \in m_2$, is such that s_1 and s_2 are bisimilar. Team equivalence is a truly concurrent behavioral equivalence as it is sensitive to the size of the distributed state; as a matter of fact, it relates markings of the same size, only. Therefore, a sequential finite-state machine, i.e., an FSM with a singleton initial marking, can never be equated to a concurrent finite-state machine, i.e., an FSM with a non-singleton initial marking. The name *team equivalence* reminds us that two distributed systems, composed of a team of non-cooperating, sequential processes, are equivalent if it is possible to match each sequential component of the first system with one bisimulation-equivalent, sequential component of the other one, as in any sports where two competing (distributed) teams have the same number of (sequential) players. Once bisimilarity has been computed, checking whether two markings of size k are team equivalent can be computed in $O(k^2)$ time.

Note that to check whether two markings are team equivalent we need not construct an LTS describing the global behavior of the whole system, but only find a suitable, bisimulation-preserving match among the local, sequential states (i.e., the elements of the markings); in other words, we consider a collection of LTSs for the local, sequential states only, and try to match them through bisimilarity. Nonetheless, we will prove that team equivalence is coherent with the global behavior of the net. More precisely, we will show that team equivalence is finer than interleaving bisimilarity (so it respects the token game), actually it coincides with *strong place bisimilarity* [4,5] (and so it respects the causal semantics of nets). Since we need not to construct the global behavior of the net under scrutiny, if we need to check whether other two markings of the same net, say m'_1 and m'_2, are team equivalent, we can reuse the already computed bisimulation equivalence over places, and so such a verification will take only $O(k^2)$ time, if k is the size of m'_1 and m'_2.

The second part of the paper approaches the problem of finding a modal characterization of team equivalence over FSMs, in line of what Hennessy and Milner proved for standard bisimulation equivalence over LTSs [18]. The basic modal logic we start with is Hennessy–Milner Logic (HML) [18,3], which is here slightly extended in order to distinguish between successful and unsuccessful termination; the resulting modal logic is called HMT. We prove a *basic coherence theorem* comparing model checking and equivalence checking: two places of an FSM are bisimilar if and only if they satisfy the same HMT formulae. Basic team modal logic (BTML, for short) is a proper, conservative extension of HMT, with an additional operator of parallel composition $\_ \otimes \_$ of formulae, to be used at the top level only. Also in this case, we prove a *full coherence theorem*: two markings are team equivalent if and only if they satisfy the same BTML formulae.

The paper is organized as follows. Section 2 introduces the basic definitions about finite-state machines and two behavioral equivalences: interleaving bisimilarity and strong place bisimilarity [4,5]; the latter is quite interesting, as we will prove that team equivalence coincides with strong place bisimilarity for FSMs. Section 3 copes with the equivalence checking problem; first, bisimulation over places of an unmarked net is defined, showing that the classic results of bisimulation over LTSs also hold in this case; then, team equivalence is introduced and some examples discussing its pros and cons are presented; moreover, the minimization of an FSM w.r.t. bisimilarity is defined. Section 4 describes first HMT (the new variant of HML), its syntax and semantics, and shows the basic coherence theorem. Then, the new modal logic BTML is introduced and the full coherence theorem is proved. Finally, Section 5 discusses related literature, some future research and open problems.

2. Basic definitions and behavioral equivalences

Definition 1. (Multiset) Let $\mathbb{N}$ be the set of natural numbers. Given a finite set S, a *multiset* over S is a function $m : S \to \mathbb{N}$. The *support* set $dom(m)$ of a marking m is the set $\{s \in S \mid m(s) \neq 0\}$. The set of all multisets over S, denoted by $\mathscr{M}(S)$, is ranged over by m, possibly indexed. We write $s \in m$ if $m(s) > 0$. The *multiplicity* of s in m is given by the number $m(s)$. The *cardinality* of m, denoted by $|m|$, is the number $\sum_{s \in S} m(s)$, i.e., the total number of its elements. A multiset m such that $dom(m) = \emptyset$ is called *empty* and is denoted by θ. We write $m \subseteq m'$ if $m(s) \leq m'(s)$ for all $s \in S$.

Multiset union $\_ \oplus \_$ is defined as follows: $(m \oplus m')(s) = m(s) + m'(s)$; the operation $\oplus$ is commutative, associative and has θ as neutral element. If $m_2 \subseteq m_1$, then we can define *multiset difference* $\_ \ominus \_$ as follows: $(m_1 \ominus m_2)(s) = m_1(s) - m_2(s)$. The *scalar product* of a number j with m is the multiset $j \cdot m$ defined as $(j \cdot m)(s) = j \cdot (m(s))$.

A multiset m over $S = \{s_1, \ldots, s_n\}$ can be represented as $k_1 \cdot s_1 \oplus k_2 \cdot s_2 \oplus \ldots \oplus k_n \cdot s_n$, where $k_j = m(s_j) \geq 0$ for $j = 1, \ldots, n$. □

Definition 2. (Finite-state machine) A labeled *finite-state machine* (FSM, for short) is a tuple $N = (S, A, T)$, where

- S is the finite set of *places*, ranged over by s (possibly indexed),
- A is the finite set of *labels*, ranged over by ℓ (possibly indexed), and
- $T \subseteq S \times A \times (S \cup \{\theta\})$ is the finite set of *transitions*, ranged over by t (possibly indexed), such that, for each $\ell \in A$, there exists a transition $t \in T$ of the form (s, ℓ, m).