Alternating the Temporal Picture for Safety

Zohar Manna and Henny B. Sipma *

Computer Science Department
Stanford University
Stanford, CA. 94305-9045
{manna,sipma}@cs.stanford.edu

Abstract. We use alternating automata on infinite words to reduce the verification of linear temporal logic (LTL) safety properties over infinite-state systems to the proof of first-order verification conditions. This method generalizes the traditional deductive verification approach of providing verification rules for particular classes of formulas, such as invariances, nested precedence formulas, etc. It facilitates the deductive verification of arbitrary safety properties without the need for explicit temporal reasoning.

1 Introduction

Temporal logic is a powerful language for specifying properties of reactive systems. However, a specification language should be accompanied by verification methods to be useful in practice. For finite-state systems model checking provides such a verification method: it is (largely) automatic and applicable to arbitrary temporal properties. For infinite-state systems the situation is different. Although complete proof systems have been proposed for CTL [GF96] and LTL [MP91], the proof system presented for LTL requires the formula to be in a canonical form for the rules to be applicable. Transforming a formula into canonical form is expensive, the formula may grow exponentially, and worse, it may result in a formula that is so different from the original that the user’s intuition proves useless for constructing the invariants and intermediate assertions necessary to complete the proof.

In this paper we present a verification rule safe that reduces the proof of LTL safety formulas to the proof of first-order validities. The rule safe constructs an alternating automaton [Var96,Var97] for the formula to be proven. This automaton may have to be strengthened by the user. First-order verification conditions are then generated based on the structure and labeling of this automaton.

The approach resembles that of verification diagrams [BMS95] and assertion graphs [BBM97], which also reduce the proof of temporal properties to first-order

* This research was supported in part by the National Science Foundation under grant CCR-98-04100 and CCR-99-00984 ARO under grants DAAH04-96-1-0122 and DAAG55-98-1-0471, ARO under MURI grant DAAH04-96-1-0341, by Army contract DABT63-96-C-0096 (DARPA), and by Air Force contract F33615-99-C-3014.
verification conditions. In principle verification diagrams can be generated automatically (apart from the strengthening and refinement necessary for fairness), using an algorithm similar to the tableau construction for LTL, thus reducing a temporal property to first-order verification conditions. However, verification diagrams are based on (nondeterministic) $\omega$-automata, and the size of the resulting diagram can, in the worst case, be exponential in the size of the formula, giving rise to a number of first-order verification conditions of the same order of magnitude, which clearly is undesirable. Alternating automata have an advantage over the regular $\omega$-automata that they are linear in the size of the formula, thus making the number of verification conditions generated also proportional to it.

The remainder of the paper is organized as follows. Section 2 provides the preliminaries: it presents our computational model of transition systems, and specification language of linear temporal logic (LTL). Alternating automata and their models are introduced in Section 3. In Section 4 we give an algorithm to construct an alternating automaton for future LTL formulas, and we prove that the language accepted by the constructed automaton is precisely the set of sequences of states that satisfy the formula. Section 5 proposes the verification rules B-SAFE and SAFE that reduce the verification of future safety formulas to first-order verification conditions, and it is shown that the special verification rules of [MP95] are subsumed by this rule. In Section 6 we give an algorithm to construct an alternating automaton for LTL formulas involving past operators, and we propose a verification rule for such formulas. Finally, in Section 7 we discuss some of the limitations of these rules and give some ideas on how they could be overcome.

2 Preliminaries

2.1 Computational Model: Fair Transition Systems

The computational model used for reactive systems is that of a transition system [MP95] (TS), $S = \langle V, \Theta_S, \mathcal{T} \rangle$, where $V$ is a finite set of variables, $\Theta_S$ is an initial condition, and $\mathcal{T}$ is a finite set of transitions. A state $s$ is an interpretation of $V$, and $\Sigma$ denotes the set of all states. A transition $\tau \in \mathcal{T}$ is a function $\tau : \Sigma \rightarrow 2^\Sigma$, and each state in $\tau(s)$ is called a $\tau$-successor of $s$. We say that a transition $\tau$ is enabled on $s$ if $\tau(s) \neq \emptyset$, otherwise $\tau$ is disabled on $s$. Each transition $\tau$ is represented by a transition relation $\rho_\tau(s, s')$, an assertion that expresses the relation between the values of $V$ in $s$ and the values of $V$ (referred to by $V'$) in any of its $\tau$-successors $s'$.

A run of $S$ is an infinite sequence of states such that the first state satisfies $\Theta_S$ and any two consecutive states satisfy a $\rho_\tau$ for some $\tau \in \mathcal{T}$. A state $s$ is called $S$-accessible if it appears in some run of $S$. The set of all runs of $S$ is denoted by $\mathcal{L}(S)$.

2.2 Specification Language: Linear Temporal Logic

The specification language studied in this paper is linear temporal logic. We assume an underlying assertion language which is a first-order language over