Theoretical Computer Science 537 (2014) 29–41

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Topology, monitorable properties and runtime verification

Volker Diekert [a], Martin Leucker [b,*]

[a] FMI, Universität Stuttgart, Universitätsstr. 38, D-70569 Stuttgart, Germany
[b] Institute for Software Engineering and Programming Languages, Universität zu Lübeck, Ratzeburger Allee 160, D-23562 Lübeck, Germany

ARTICLE INFO

Article history:
Received 3 April 2012
Received in revised form 28 December 2013
Accepted 21 February 2014

Keywords:
Omega-regular language
Monitorable property
Runtime verification
Topology

ABSTRACT

We review concepts like safety, liveness, and monitorability from a rigorous topological viewpoint. Thus, monitorability of an ω-language means that the boundary in the Cantor topology has an empty interior. We show that all ω-regular languages which are deterministic and co-deterministic are monitorable, but certain deterministic liveness properties like "infinitely many a's" cannot be written as a countable union of monitorable languages. We briefly discuss model checking with LTL, its three-valued variant LTL_3 and monitor constructions based upon LTL_3.

0. Introduction

This paper is based on a one-day-tutorial held during ICTAC 2011 at the wonderful Mabalingwe Nature Reserve, South Africa. The idea of the tutorial was to present concepts developed in finite state verification from a rigorous topological viewpoint and, more importantly, to see how possibly new results can be obtained by this approach. A topological view is not original, but experience shows that researchers in formal verification sometimes miss the topological background in order to take advantage out of such an abstract setting.

Today, automata theoretical verification is a success story with large scale industrial applications. A real-life system is modeled by some (finite-state) transition system. The runs through the system are given as infinite words over some finite alphabet. The model-checking problem asks whether all runs satisfy a given specification. If the specification is written in some logical formalism like monadic second-order logic, first-order logic or LTL, all runs obeying the specification can be expressed effectively by some Büchi automaton. This is a key fact, because the verification problem becomes an inclusion problem on ω-regular languages which in turn leads to a reachability problem in finite graphs.

A main problem is however that precise knowledge of runs in the system might not be known in advance or it is extremely complex. Therefore instead of model checking a *monitor* is used, which is used to check the underlying system at runtime. The question arises, which properties can be checked in such a way, or, in other words, which properties are *monitorable*. The formal definition of monitorable properties has been given first in [13] by Pnueli and Zaks. It generalizes the notion of a *safety property*. If φ is a safety property (say specified as an ω-regular set), then a deterministic finite automaton can raise an alarm $\bot$ by observing finite prefixes, once the property is violated. A monitorable property gives also a positive feedback $\top$, if all prolongations of a finite prefix obey the specification. A language is monitorable, if such a monitor is useful on all inputs. This means whenever we have seen a finite prefix then there is the possibility to reach at least one of the special signals $\bot$ or $\top$, otherwise we can stop monitoring. Monitors are usually easy to implement and

* Corresponding author.
E-mail addresses: diekert@fmi.uni-stuttgart.de (V. Diekert), leucker@isp.uni-luebeck.de (M. Leucker).

http://dx.doi.org/10.1016/j.tcs.2014.02.052
0304-3975/

have a wide range of applications. Extensions and applications for stochastic automata have been proposed in Sistla et al., see [8,14].

A topological interpretation of monitorability follows from the widely known fact that safety properties correspond to closed sets in the Cantor topology. More precisely, we will see that monitorable languages correspond exactly to those languages where the boundary has no interior. Corollary 3 states that all languages in the Borel class $G_\delta \cap F_\sigma$ are monitorable.

The outline of the paper is as follows. In Section 3.1 we give a brief account to the topology restricted to very basic concepts. Readers familiar with topology may skip this section. For newcomers Section 3.1 contains a *self-test* (with hints how to solve the problems) at the end. Even with the very limited material presented in the brief introduction, the self-test should be easy.

Section 3.2 reviews the well-known topological interpretations for safety and liveness. The main issue of this paper is reflected in Section 4. We give a purely topological definition of *monitorability* and we recover the notion in Corollary 2 as the property that the boundary is nowhere dense, i.e., the boundary has an empty interior as boundaries are always closed. This leads to Corollary 3 that all sets in the Borel class $G_\delta \cap F_\sigma$ are monitorable in complete metric spaces. Actually we prove a stronger result in Theorem 1 which also shows that a liveness property like "infinitely many a's" cannot be written as a countable union of monitorable sets. This is a sort of non-approximation result for certain liveness properties and it is discussed in Section 5.3, see Proposition 8.

We give a direct proof for Theorem 1, but one could also deduce it from the classical result in general topology stating that open sets in complete metric spaces are *Baire spaces*, i.e, every countable intersection of open dense sets is dense. In particular, Corollary 3 can be viewed as folklore in topological set theory.

In Section 5 we restrict the topological interpretations to the more usual setting of ω-regular languages. In contrast to the topological part, we assume some familiarity with Büchi automata. By [11] the ω-regular languages in the Borel class $G_\delta \cap F_\sigma$ are exactly those languages which are deterministic and co-deterministic. Thus, we obtain Corollary 4 saying that all languages which are deterministic and co-deterministic are monitorable. The result itself is not deep, but it is a nice property of monitorability. To the best of our knowledge it has not been stated elsewhere. Moreover, Corollary 4 is optimal in the sense that inside the Borel hierarchy we cannot go beyond $G_\delta \cap F_\sigma$. There are non-monitorable languages in $G_\delta \setminus F_\sigma$, see Example 3.

In runtime verification, the formal specification is checked based on a finite observation of the system. In our setting, we are interested in checking LTL specifications formulated for infinite words but now checked on finite words. In [2], LTL_3 was proposed as an LTL logic with a semantics that a finite word is a prefix of a so-far unknown infinite run. In Section 6, we recall the semantics of LTL_3 and explain its underlying idea from a topological point of view. This allows us to re-use the monitor construction introduced in Section 5.2 also for LTL_3.

In Section 7 we give some outlook to some other work which underlines the importance to develop concepts like safety, liveness or monitorability relative to changing topologies.

1. Notation

If X is a set, then its *powerset* is the family of all subsets over X. A subset of the powerset of X is called a *Boolean algebra*, if it is closed under finite union and complementation. This includes the empty union and, as a consequence, the empty set $\emptyset$ and the whole set X belong to every Boolean algebra inside the powerset of X. If $L \subseteq X$ is a subset, then L^c denotes its complement, $L^c = X \setminus L$.

By $\mathbb{N}$, $\mathbb{Q}$, and $\mathbb{R}$ we denote the natural, rational, and real numbers respectively. We let Σ be a non-empty finite *alphabet*. Elements of Σ are abstract symbols, called *letters*. Frequently in applications, Σ is a powerset of a non-empty set of atomic propositions. Thus, we assume $|\Sigma| \geqslant 2$ whenever convenient. If we refer to letters $a, b, c \in \Sigma$, then we assume that they are pairwise different. By Σ^* (Σ^+ resp., Σ^ω resp., Σ^∞ resp.) we mean the set of finite (finite and non-empty resp., infinite resp., finite or infinite resp.) words over Σ. The length of a word $w \in \Sigma^\infty$ is denoted by $|w|$, it is a natural number or ω. The prefix relation between words $p, q \in \Sigma^\infty$ is denoted by $p \leqslant q$. We write $p < q$, if p is a proper prefix of q, i.e., $p \leqslant q$ and $p \neq q$. We say that $u \in \Sigma^*$ is a *factor* of $w \in \Sigma^\infty$, if we can write $w = xuy$ for some $x \in \Sigma^*$ and $y \in \Sigma^\infty$.

A *language* is a set of words. Words are used to represent *runs* of a system, frequently they are non-terminating. Hence our primary interest is in infinite words. Note that if a class of languages is specified by some logical formalism then this class is a Boolean algebra as soon the formalism can express disjunction, negation, and the truth value *false*.

2. Finite automata

A *non-deterministic finite automaton* (*NFA* for short) is given as a tuple $\mathcal{A} = (Q, \Sigma, \delta, I, R)$, where Q is a finite set of *states* and δ is a *transition relation*: $\delta \subseteq Q \times \Sigma \times Q$. The set $I \subseteq Q$ is called the set of *initial states*, the set $R \subseteq Q$ consists of *repeated* (or *final*) states.

If δ is a partially defined function from $Q \times \Sigma$ to Q and if, in addition, $|I| \leqslant 1$, then the automaton is called *deterministic* and we speak about a *DFA*. Thus, every DFA is an NFA. The NFA is called *complete*, if for every $(p, a) \in Q \times \Sigma$ there is some $q \in Q$ such that $(p, a, q) \in \delta$. It is called *reduced*, if each state is reachable from some initial state. In a complete DFA, the transition relation becomes a totally defined function $\delta : Q \times \Sigma \to Q$.