The good, the bad, and the ugly, but how ugly is ugly?

Andreas Bauer, Martin Leucker, and Christian Schallhart

Institut für Informatik, Technische Universität München

Abstract. When monitoring a system w. r. t. a property defined in some temporal logic, e. g., LTL, a major concern is to settle with an adequate interpretation of observable system events; that is, models of temporal logic formulae are usually infinite streams of events, whereas at runtime only prefixes are available.
This work defines a four-valued semantics for LTL over finite traces, which extends the classical semantics, and allows to infere whether a system behaves (1) according to the monitored property, (2) violates the property, (3) will possibly violate the property in the future, or (4) will possibly conform to the property in the future, if the system has stabilised. Notably, (1) and (2) correspond to the classical semantics of LTL, whereas (3) and (4) are chosen whenever an observed system behaviour has not yet lead to a violation or acceptance of the monitored property.
This logic called FLTL seems to correspond with the semantics realised by the Temporal Rover and has, to the best of our knowledge, not been formally captured elsewhere. We further present a monitor construction for FLTL properties.

1 Introduction

While the syntax and semantics of LTL on infinite traces is well accepted in the literature, there is no consensus on defining LTL over finite strings. Several versions of a *two-valued* semantics for LTL on finite strings have been proposed. For instance, Eisner et al. give a good overview on the topic [EFH$^+$03]. Further, in [BLS06], a three-valued semantics is proposed which extends the classical semantics over finite traces in a natural manner: a property is $true$, respectively $false$, w. r. t. a finite observation, iff the observation is either a satisfying prefix, respectively violating prefix, of all possible infinite extensions; otherwise, the observation is said to be inconclusive, and the property assigned a ?. This scheme coincides well with the notion of *safety* (e. g., Gp—always p) and *co-safety* (e. g., Fp—eventually p) properties, since these are either finitely refutable or satisfiable.

However, monitoring a system w. r. t. a safety property that does, in fact, never exhibit violating behaviour, results in infinitely many inconclusive results from the monitor, likewise with co-safety properties. Further, when monitoring a *liveness* property [AS84] that is not co-safety, i. e., finitely satisfiable, then neither the violation nor the satisfaction of the property can be determined using a finite stream of observations, and not much is said about the possible future.

Contribution. In this work, we submit the idea that an inconclusive result of a monitor should be more detailed, allowing to draw conclusions what the future may hold for a system w. r. t. a trace seen so far and the type of property being monitored; that is, we define a four-valued semantics for LTL that not only results in either $true$, $false$, or ?, but yields *possibly true* and *possibly false* whenever the system's behaviour so far is not conclusive in the strictly Boolean sense. We call the resulting logic *Finite Linear Temporal Logic* (FLTL).

Further, we have defined a translation from formulae in FLTL to Mealy machines, which then form a suitable foundation for runtime verification, in that the output alphabet of the automata corresponds to the four truth values sketched above.

2 FLTL at a glance

As discussed in [MP95], the difficulty for an LTL semantics over finite strings lies in the next-state operator X. Given a finite string $u = a_0 \dots a_{n-1}$ of length n, it is unclear whether $u, n-1 \models X\varphi$ holds. Hence, a first axiom of a sound truncated path semantics would be to require that

– *$X\varphi$ means there exists a next state and this state satisfies φ* **(∃X)**

which we term the *existential-next view*, abbreviated by (∃**X**). Consequently, the above example is $false$, as there is no next state. A second axiom we consider essential is that negation, indeed, expresses that a formula's truth value is complemented, formulated as

– *a formula and its negation yield complementary truth values.* **(¬=C)**

Then, however, a negated next-state formula should be $true$. This, however, conflicts the classical equivalence $\neg X\varphi \equiv X\neg\varphi$, which can no longer hold on finite strings (unless $true$ equals $false$). It is, therefore, helpful to distinguish a *strong* or *existential* (denoted by X) and a *weak* or *universal* version (denoted by $\bar{X}$) of the next-state operator.

This view is meaningful in a setting, in which we are faced with only *maximal* traces. In runtime verification, however, we are given a *prefix* of an infinite trace. Therefore, it is clear that there will be a next state, but not known *what* it will be.

It can also be argued (see, e. g., [HR02]) that the finally operator F is of existential nature, as some property should finally hold, while the globally operator G is of a universal character, in a sense that something should hold in every position of a path. Accordingly, one can argue that $F\varphi$ should evaluate to $false$ if φ does not hold in the currently known prefix, while $G\varphi$ should be $true$, if φ is not violated in the currently known prefix, and in both cases nothing is known about the successor states.

In LTL, it holds that $F\varphi \equiv \varphi \vee XF\varphi$, as well as, $G\varphi \equiv \varphi \wedge XG\varphi$. Consequently, $XF\varphi$ should be $false$, if no subsequent state exists, while $XG\varphi$ should be $true$ in the same situation. This contradiction is elegantly solved by having the existential as well as the universal version of the next-state operator, opening the possibility of having the equivalence $F\varphi \equiv \varphi \vee XF\varphi$ and $G\varphi \equiv \varphi \wedge \bar{X}G\varphi$.

Therefore, for runtime verification, we postulate two further axioms. The first can be stated as

– *say true or false only, if the future does not matter.* **(Sound)**

The string a (of length 1) clearly satisfies the proposition p iff $p \in a$. While, understanding a as a prefix of an infinite string, the value of $X\varphi$ is of less certainty, as the successor state of a is not known. Choosing either true or false (depending on whether to understand X strongly or weakly) would diminish the qualitative difference of the knowledge on p and $X\varphi$ based on the string a. Therefore, we require a semantics to yield four values: *true*, *possibly true*, *possibly false*, and *false*. Roughly, $true$ and $false$ are used for Boolean combinations of propositions and *possibly true* and *possibly false* for statements for which the future is important.

Actually, this view seems to be already adopted in the runtime verification tool Temporal Rover [Dru00]. However, no formal semantics of Rover's employed LTL is available. Note that identifying *possible true* and $true$ as $true$, respectively *possible false* and $false$ as $false$, yields the finite trace semantics as proposed in [MP95].

So far, we have disregarded a further issue that we consider important. When considering $X\varphi$ in the last state of a finite string u, there is no reason to go for false (or possibly false), if every possible continuation of u satisfies φ. A trivial example would be $X\,true$. While every single letter extension of u would make $X\,true$ true in u's last position, the semantics discussed so far would come up with false or possibly false. Therefore, we require that

– *if the future does not matter, say true or false.* **(Precise)**

FLTL captures the ideas formulated as **(∃X)**, **(¬=C)**, **(Sound)**, and **(Precise)**, and can be efficiently translated into Mealy machines, whose output alphabet corresponds to the four truth values.

References

[AS84] Bowen Alpern and Fred B. Schneider. Defining liveness. Technical report, Ithaca, NY, USA, 1984.

[BLS06] Andreas Bauer, Martin Leucker, and Christian Schallhart. Monitoring of real-time properties. In *Proceedings of the 26th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 4337 of *LNCS*. Springer, December 2006.

[Dru00] Doron Drusinsky. The temporal rover and the atg rover. In Klaus Havelund, John Penix, and Willem Visser, editors, *SPIN*, volume 1885 of *Lecture Notes in Computer Science*, pages 323–330. Springer, 2000.

[EFH+03] Cindy Eisner, Dana Fisman, John Havlicek, Yoad Lustig, Anthony McIsaac, and David Van Campenhout. Reasoning with temporal logic on truncated paths. In *CAV03*, volume 2725 of *LNCS*, pages 27–39, Boulder, CO, USA, July 2003. Springer.

[HR02] Klaus Havelund and Grigore Rosu. Synthesizing Monitors for Safety Properties. In *Tools and Algorithms for Construction and Analysis of Systems*, pages 342–356, 2002.

[MP95] Zohar Manna and Amir Pnueli. *Temporal Verification of Reactive Systems: Safety*. Springer, New York, 1995.