Reasoning with Temporal Logic on Truncated Paths

Cindy Eisner[1], Dana Fisman[1,2]*, John Havlicek[3], Yoad Lustig[1], Anthony McIsaac[4], and David Van Campenhout[5]**

[1] IBM Haifa Research Lab [2] Weizmann Institute of Science
[3] Motorola, Inc. [4] STMicroelectronics, Ltd. [5] Verisity Design, Inc.

Abstract. We consider the problem of reasoning with linear temporal logic on *truncated* paths. A truncated path is a path that is finite, but not necessarily maximal. Truncated paths arise naturally in several areas, among which are incomplete verification methods (such as simulation or bounded model checking) and hardware resets. We present a formalism for reasoning about truncated paths, and analyze its characteristics.

1 Introduction

Traditional LTL semantics over finite paths [15] are defined for maximal paths in the model. That is, if we evaluate a formula over a finite path under traditional LTL finite semantics, it is because the last state of the path has no successor in the model. One of the consequences of extending LTL [16] to finite paths is that the *next* operator has to be split into a *strong* and a *weak* version [15]. The strong version, which we denote by $\mathsf{X!}\varphi$, does not hold at the last state of a finite path, while the weak version, which we denote by $\mathsf{X}\varphi$, does.

In this paper, we consider not only finite maximal paths, but finite *truncated* paths. A truncated path is a finite path that is not necessarily maximal. RV Truncated paths arise naturally in incomplete verification methods such as simulation or bounded model checking. There is also a connection to the problem of describing the behavior of hardware resets in temporal logic, since intuitively we tend to think of a reset as somehow cutting the path into two disjoint parts - a finite, truncated part up until the reset, and a possibly infinite, maximal part after the reset.

Methods of reasoning about finite maximal paths are insufficient for reasoning about truncated paths. When considering a truncated path, the user might want to reason about properties of the truncation as well as properties of the model. For instance, the user might want to specify that a simulation test goes on long enough to discharge all outstanding obligations, or, on the other hand, that an

* The work of this author was supported in part by the John Von Neumann Minerva Center for the Verification of Reactive Systems.

** E-mail addresses: eisner@il.ibm.com (C. Eisner), dana@wisdom.weizmann.ac.il (D. Fisman), john.havlicek@motorola.com (J. Havlicek), yoadl@il.ibm.com (Y. Lustig), anthony.mcisaac@st.com (A. McIsaac), dvc@verisity.com (D. Van Campenhout).

obligation need not be met if it "is the fault of the test" (that is, if the test is too short). The former approach is useful for a test designed (either manually or by other means) to continue until correct output can be confirmed. The latter approach is useful for a test which has no "opinion" on the correct length of a test - for instance, a monitor running concurrently with the main test to check for bus protocol errors.

At first glance, it seems that the strong operators ($\mathsf{X!}$ and U) can be used in the case that all outstanding obligations must be met, and the weak operators (X and W) in the case that they need not. However, we would like a specification to be independent of the verification method used. Thus, for instance, for a specification $[p \mathsf{U} q]$, we do not want the user to have to modify the formula to $[p \mathsf{W} q]$ just because she is running a simulation.

In such a situation, we need to define the semantics over a truncated path. In other words, at the end of the truncated path, the truth value must be decided. If the path was truncated after the evaluation of the formula completed, the truth value is already determined. The problem is to decide the truth value if the path was truncated before the evaluation of the formula completed, i.e., where there is *doubt* regarding what would have been the truth value if the path had not been truncated. For instance, consider the formula $\mathsf{F}p$ on a truncated path such that p does not hold for any state. Another example is the formula $\mathsf{G}q$ on a truncated path such that q holds for every state. In both cases we cannot be sure whether or not the formula holds on the original untruncated path.

We term a decision to return *true* when there is doubt the *weak view* and a decision to return *false* when there is doubt the *strong view.* Thus in the weak view the formula $\mathsf{F}p$ holds for any finite path, while $\mathsf{G}q$ holds only if q holds at every state on the path. And in the strong view the formula $\mathsf{F}p$ holds only if p holds at some state on the path, while the formula $\mathsf{G}q$ does not hold for any finite path. Alternatively, one can take the position that one should demand the maximum that can be reasonably expected from a finite path. For formulas of the form $\mathsf{F}p$, a prefix on which p holds for some state on the path is sufficient to show that the formula holds on the entire path, thus it is reasonable to demand that such a prefix exist. In the case of formulas of the form $\mathsf{G}q$, no finite prefix can serve as evidence that the formula holds on the entire path, thus requiring such evidence is not reasonable. Under this approach, then, the formula $\mathsf{F}p$ holds only if p holds at some state on the path, while the formula $\mathsf{G}q$ holds only if q holds at every state on the path. This is exactly the traditional LTL semantics over finite paths [15], which we term the *neutral view.*

In this paper, we present a semantics for LTL over truncated paths based on the weak, neutral, and strong views. We study properties of the *truncated semantics* for the resulting logic LTL^{trunc}, as well as its relation to the *informative prefixes* of [12]. We examine the relation between truncated paths and hardware resets, and show that our truncated semantics are mathematically equivalent to the *reset semantics* of [3].

The remainder of this paper is structured as follows. Section 2 presents our *truncated* semantics. Section 3 studies properties of our logic as well as its relation to the *informative prefixes* of [12]. Section 4 shows the relation to hardware resets. Section 5 discusses related work. Section 6 concludes.

2 The Truncated Semantics

Recall that LTL is the logic with the following syntax:

Definition 1 (LTL formulas).

- *Every atomic proposition is an* LTL *formula.*
- *If* φ *and* ψ *are* LTL *formulas then the following are* LTL *formulas:*
  - $\neg\varphi$ • $\varphi \wedge \psi$ • $\mathsf{X!}\ \varphi$ • $[\varphi\ \mathsf{U}\ \psi]$

Additional operators are defined as syntactic sugaring of the above operators:

- $\varphi \vee \psi \overset{\text{def}}{=} \neg(\neg\varphi \wedge \neg\psi)$ • $\varphi \rightarrow \psi \overset{\text{def}}{=} \neg\varphi \vee \psi$ • $\mathsf{X}\ \varphi \overset{\text{def}}{=} \neg(\mathsf{X!}\ \neg\varphi)$
- $\mathsf{F}\ \varphi \overset{\text{def}}{=} [\mathit{true}\ \mathsf{U}\ \varphi]$ • $\mathsf{G}\ \varphi \overset{\text{def}}{=} \neg\mathsf{F}\ \neg\varphi$ • $[\varphi\ \mathsf{W}\ \psi] \overset{\text{def}}{=} [\varphi\ \mathsf{U}\ \psi] \vee \mathsf{G}\varphi$

According to our motivation presented above, the formula φ holds on a truncated path in the weak view if up to the point where the path ends, "nothing has yet gone wrong" with φ. It holds on a truncated path in the neutral view according to the standard LTL semantics for finite paths. In the strong view, φ holds on a truncated path if everything that needs to happen to convince us that φ holds on the original untruncated path has already occurred. Intuitively then, our truncated semantics are related to those of standard LTL on finite paths as follows: the weak view weakens all operators (e.g. U acts like W, $\mathsf{X!}$ like X), the neutral view leaves them unchanged, and the strong view strengthens them (e.g. W acts like U, X like $\mathsf{X!}$).

We define the truncated semantics of LTL formulas over words[1] from the alphabet 2^P. A letter is a subset of the set of atomic propositions P such that *true* belongs to the subset and *false* does not. We will denote a letter from 2^P by ℓ and an empty, finite, or infinite word from 2^P by w. We denote the length of word w as $|w|$. An empty word $w = \epsilon$ has length 0, a finite word $w = (\ell_0\ell_1\ell_2\cdots\ell_n)$ has length $n+1$, and an infinite word has length ∞. We denote the i^{th} letter of w by w^{i-1} (since counting of letters starts at zero). We denote by $w^{i..}$ the suffix of w starting at w^i. That is, $w^{i..} = (w^i w^{i+1}\cdots w^n)$ or $w^{i..} = (w^i w^{i+1}\cdots)$. We denote by $w^{i..j}$ the finite sequence of letters starting from w^i and ending in w^j. That is, $w^{i..j} = (w^i w^{i+1}\cdots w^j)$.

We make use of an "overflow" and "underflow" for the indices of w. That is, $w^{j..} = \epsilon$ if $j \geq |w|$, and $w^{j..k} = \epsilon$ if $j \geq |w|$ or $k < j$. For example, in the semantics of $[\varphi\ \mathsf{U}\ \psi]$ under weak context, when we say "$\exists k$", k is not required to be less than $|w|$.

The truncated semantics of an LTL formula are defined with respect to finite or infinite words and a context indicating the *strength*, which can be either

[1] Relating the semantics over words to semantics over models is done in the standard way. Due to lack of space, we omit the details.

weak, neutral or strong. Under the neutral context only non-empty words are evaluated; under weak/strong contexts, empty words are evaluated as well. We use $w \models^{S} \varphi$ to denote that φ is satisfied under the model (w, S), where S is "$-$" if the context is weak, null if it is neutral, and "$+$" if it is strong. We use w to denote an empty, finite, or infinite word, φ and ψ to denote LTL formulas, p to denote an atomic proposition, and j and k to denote natural numbers.

holds weakly: For w such that $|w| \geq 0$,

1. $w \models^{-} p \iff |w| = 0$ or $p \in w^0$
2. $w \models^{-} \neg\varphi \iff w \not\models^{+} \varphi$
3. $w \models^{-} \varphi \wedge \psi \iff w \models^{-} \varphi$ and $w \models^{-} \psi$
4. $w \models^{-} \mathsf{X!}\ \varphi \iff w^{1..} \models^{-} \varphi$
5. $w \models^{-} [\varphi \mathsf{U} \psi] \iff \exists k$ such that $w^{k..} \models^{-} \psi$, and for every $j < k$, $w^{j..} \models^{-} \varphi$

holds neutrally: For w such that $|w| > 0$,

1. $w \models p \iff p \in w^0$
2. $w \models \neg\varphi \iff w \not\models \varphi$
3. $w \models \varphi \wedge \psi \iff w \models \varphi$ and $w \models \psi$
4. $w \models \mathsf{X!}\ \varphi \iff |w| > 1$ and $w^{1..} \models \varphi$
5. $w \models [\varphi \mathsf{U} \psi] \iff \exists k < |w|$ such that $w^{k..} \models \psi$, and for every $j < k$, $w^{j..} \models \varphi$

holds strongly: For w such that $|w| \geq 0$,

1. $w \models^{+} p \iff |w| > 0$ and $p \in w^0$
2. $w \models^{+} \neg\varphi \iff w \not\models^{-} \varphi$
3. $w \models^{+} \varphi \wedge \psi \iff w \models^{+} \varphi$ and $w \models^{+} \psi$
4. $w \models^{+} \mathsf{X!}\ \varphi \iff w^{1..} \models^{+} \varphi$
5. $w \models^{+} [\varphi \mathsf{U} \psi] \iff \exists k$ such that $w^{k..} \models^{+} \psi$, and for every $j < k$, $w^{j..} \models^{+} \varphi$

Our goal was to give a semantics to LTL formulas for truncated paths, but we have actually ended up with two parallel semantics: the neutral semantics, and the weak/strong semantics. The weak/strong semantics form a coupled dual pair because the negation operator switches between them. Before analyzing these semantics, we first unify them by augmenting LTL with truncate operators that connect the neutral semantics to the weak/strong semantics. Intuitively, trunc_w truncates a path using the weak view, while trunc_s truncates using the strong view. Formally, LTL^{trunc} is the following logic, where we use the term *boolean expression* to refer to any application of the standard boolean operators to atomic propositions, and we associate satisfaction of a boolean expression over a letter w^i with satisfaction of the boolean expression over the word $w^{i..i}$.

Definition 2 **(LTL^{trunc} formulas).**

– *Every atomic proposition is an LTL^{trunc} formula.*

– *If φ and ψ are* LTL^{trunc} *formulas and b is a boolean expression, then the following are* LTL^{trunc} *formulas:*
  - $\neg\varphi$ • $\varphi \wedge \psi$ • $\mathsf{X!}\ \varphi$ • $[\varphi\ \mathsf{U}\ \psi]$ • $\varphi\ \mathsf{trunc\_w}\ b$

We also add the dual of the trunc_w operator as syntactic sugar as follows:

$$\varphi\ \mathsf{trunc\_s}\ b \overset{\text{def}}{=} \neg(\neg\varphi\ \mathsf{trunc\_w}\ b)$$

The semantics of the standard LTL operators are as presented above. The semantics of the truncate operator are as follows:

– $w \overset{-}{\models} \varphi\ \mathsf{trunc\_w}\ b \iff w \overset{-}{\models} \varphi$ or $\exists k < |w|$ s.t. $w^k \models b$ and $w^{0..k-1} \overset{-}{\models} \varphi$
– $w \models \varphi\ \mathsf{trunc\_w}\ b \iff w \models \varphi$ or $\exists k < |w|$ s.t. $w^k \models b$ and $w^{0..k-1} \overset{-}{\models} \varphi$
– $w \overset{+}{\models} \varphi\ \mathsf{trunc\_w}\ b \iff w \overset{+}{\models} \varphi$ or $\exists k < |w|$ s.t. $w^k \models b$ and $w^{0..k-1} \overset{-}{\models} \varphi$

Thus, trunc_w performs a truncation and takes us to the weak view, and, as we show below, trunc_s performs a truncation and takes us to the strong view. There is no way to get from the weak/strong views back to the neutral view. This corresponds with our intuition that once a path has been truncated, there is no way to "untruncate" it.

3 Characteristics of the Truncated Semantics

In this section, we study properties of the truncated semantics as well as its relation to the *informative prefixes* of [12]. All theorems are given here without proof; the proofs can be found in the full version of the paper. We first examine relations between the views. The first theorem assures that the strong context is indeed stronger than the neutral, while the neutral is stronger than the weak.

Theorem 3 (Strength relation theorem). *Let w be a non-empty word.*

1. $w \overset{+}{\models} \varphi \Longrightarrow w \models \varphi$
2. $w \models \varphi \Longrightarrow w \overset{-}{\models} \varphi$

The proof, obtained by induction on the structure of the formula, relies on the following lemma.

Lemma 4 *Let φ be a formula in* LTL^{trunc}*. Then both $\epsilon \overset{-}{\models} \varphi$ and $\epsilon \overset{+}{\not\models} \varphi$.*

The following corollary to Theorem 3 states that for infinite paths, the weak/neutral/strong views are the same. Recall that the neutral view without the trunc_w operator is that of standard LTL over finite and infinite paths. Thus, for LTL^{trunc} formulas with no truncation operators (that is, for LTL formulas), Corollary 5 implies that all three views are equivalent over infinite paths to standard LTL semantics.

Corollary 5 *If w is infinite, then $w \overset{-}{\models} \varphi$ iff $w \models \varphi$ iff $w \overset{+}{\models} \varphi$.*

Intuitively, a truncated path w satisfies φ in the weak view if w "carries no evidence against" φ. It should then follow that any prefix of w "carries no evidence against" φ. Similarly, w satisfies φ in the strong view if it "supplies all the evidence needed" to conclude that φ holds on the original untruncated path. Hence any extension of w should also "supply all evidence needed" for this conclusion. The following theorem confirms these intuitive expectations. We first formalize the notions of prefix and extension.

Definition 6 (Prefix, extension).

u is a prefix *of v, denoted $u \preceq v$, if there exists a word u' such that $uu' = v$.*
w is an extension *of v, denoted $w \succeq v$, if there exists a word v' such that $vv' = w$.*

Theorem 7 (Prefix/extension theorem).

1. $v \overset{-}{\models} \varphi \iff \forall u \preceq v,\ u \overset{-}{\models} \varphi$
2. $v \overset{+}{\models} \varphi \iff \forall w \succeq v,\ w \overset{+}{\models} \varphi$

We now examine our intuitions regarding some derived operators. Since the trunc_w operator takes us to the weak view, we expect the trunc_s operator to take us to the strong view. The following observation confirms our intuition by capturing directly the semantics of the trunc_s operator.

Observation 8

– $w \overset{-}{\models} \varphi$ *trunc_s* $b \iff w \overset{-}{\models} \varphi$ *and* $\forall k < |w|$ *if* $w^k \models b$ *then* $w^{0..k-1} \overset{+}{\models} \varphi$
– $w \models \varphi$ *trunc_s* $b \iff w \models \varphi$ *and* $\forall k < |w|$ *if* $w^k \models b$ *then* $w^{0..k-1} \overset{+}{\models} \varphi$
– $w \overset{+}{\models} \varphi$ *trunc_s* $b \iff w \overset{+}{\models} \varphi$ *and* $\forall k < |w|$ *if* $w^k \models b$ *then* $w^{0..k-1} \overset{+}{\models} \varphi$

The following observation shows that our intuitions regarding F and G on truncated paths hold. In particular, Fφ holds for any formula φ in weak context on a truncated path, and Gφ does not hold for any formula φ in strong context on a truncated path.

Observation 9

- $w \overset{-}{\models} F\varphi \iff \exists k\ s.t.\ w^{k..} \overset{-}{\models} \varphi$
- $w \models F\varphi \iff \exists k < |w|\ s.t.\ w^{k..} \models \varphi$
- $w \overset{+}{\models} F\varphi \iff \exists k\ s.t.\ w^{k..} \overset{+}{\models} \varphi$
- $w \overset{-}{\models} G\varphi \iff \forall k,\ w^{k..} \overset{-}{\models} \varphi$
- $w \models G\varphi \iff \forall k < |w|,\ w^{k..} \models \varphi$
- $w \overset{+}{\models} G\varphi \iff \forall k,\ w^{k..} \overset{+}{\models} \varphi$

Note that for $k \geq |w|$, $w^{k..} = \epsilon$ and by Lemma 4, $\epsilon \overset{-}{\models} \varphi$ and $\epsilon \overset{+}{\not\models} \varphi$ for every φ. Thus Observation 9 shows that for every formula φ and for every finite word w, $w \overset{-}{\models} \mathsf{F}\varphi$ and $w \overset{+}{\not\models} \mathsf{G}\varphi$.

We have already seen that for infinite words, the semantics of the weak/neutral/strong contexts are equivalent and, in the absence of truncation operators, are the same as those of standard LTL. The following observations show that for finite words, the strength of an operator matters only in the neutral context since in a weak context every operator is weak (U acts like W and X! acts like X) and in a strong context every operator is strong (W acts like U and X acts like X!).

Observation 10 *Let w be a finite word.*

- $w \models X\varphi \iff w \models \neg(X!\, \neg\varphi)$ • $w \models [\varphi U \psi] \iff w \models \neg[\neg\psi W(\neg\varphi \wedge \neg\psi)]$
- $w \overset{+}{\models} X\varphi \iff w \overset{+}{\models} X!\, \varphi$ • $w \overset{+}{\models} [\varphi U \psi] \iff w \overset{+}{\models} [\varphi W \psi]$
- $w \overset{-}{\models} X\varphi \iff w \overset{-}{\models} X!\, \varphi$ • $w \overset{-}{\models} [\varphi U \psi] \iff w \overset{-}{\models} [\varphi W \psi]$

A consequence of this is that under weak context it might be the case that both φ and $\neg\varphi$ hold, while under strong context it might be the case that neither φ nor $\neg\varphi$ holds. It follows immediately that $\varphi \wedge \neg\varphi$ may hold in the weak context, while $\varphi \vee \neg\varphi$ does not necessarily hold in the strong context. For example, let $\varphi = XXp$. Then on a path w of length 1, $w \overset{-}{\models} \varphi \wedge \neg\varphi$, and $w \overset{+}{\not\models} \varphi \vee \neg\varphi$. This property of the truncated semantics is reminiscent of a similar property in intuitionistic logic [6], in which $\varphi \vee \neg\varphi$ does not necessarily hold.

We now argue that the truncated semantics formalizes the intuition behind the weak, neutral and strong views. Recall that one of the motivating intuitions for the truncated semantics is that if a path is truncated before evaluation of φ "completes", then the truncated path satisfies φ weakly but does not satisfy φ strongly. If the evaluation of φ "completes" before the path is truncated, then the truth value on the truncated path is the result of the evaluation. Thus, in order to claim that we capture the intuition we need to define when the evaluation of a formula completes. In other words, given a word w and a formula φ we would like to detect the shortest prefix of w which suffices to conclude that φ holds or does not hold on w. We call such a prefix the *definitive prefix* of φ with respect to w.

Definition 11 (Definitive prefix). *Let w be a non-empty path and φ a formula. The* definitive prefix *of w with respect to φ, denoted $dp(w, \varphi)$, is the shortest finite prefix $u \preceq w$ such that*

$$u \overset{-}{\models} \varphi \iff u \models \varphi \iff u \overset{+}{\models} \varphi$$

if such u exists and $\top$ otherwise.

Intuitively, if w is finite and $dp(w, \varphi) = \top$, then even after examination of all of w, our decision procedure leaves doubt about the dispositions of both φ and $\neg\varphi$ on w. Therefore, both are satisfied weakly on w, neither is satisfied strongly on w, and all of w is needed to determine which one is satisfied neutrally on w. If $dp(w, \varphi) \neq \top$, then for finite or infinite w, examination of $dp(w, \varphi)$ is exactly enough for our decision procedure to resolve without doubt the truth value of φ over any prefix v of w such that $v \succeq dp(w, \varphi)$. Therefore, any proper prefix of $dp(w, \varphi)$ satisfies weakly both φ and $\neg\varphi$, while $dp(w, \varphi)$ satisfies strongly exactly one of φ or $\neg\varphi$, as do all of its extensions. The following theorem states this formally:

Theorem 12 (Definitive prefix theorem). *Let v be a non-empty word and φ an* LTL^{trunc} *formula.*

– *If $dp(v, \varphi) \neq \top$ then*
  - $u \prec dp(v, \varphi) \Longrightarrow u \overset{-}{\models} \varphi$ *and* $u \overset{-}{\models} \neg\varphi$

  - $u \succeq dp(v,\varphi) \Longrightarrow u \models^{\!+} \varphi$ *or* $u \models^{\!+} \neg\varphi$
- *Otherwise*
  - *for every finite* $u \preceq v$, ($u \models^{\!-} \varphi$ *and* $u \models^{\!-} \neg\varphi$) *and* ($u \not\models^{\!+} \varphi$ *and* $u \not\models^{\!+} \neg\varphi$)

Plainly, $dp(w,\varphi) = dp(w,\neg\varphi)$. If u is the definitive prefix of w with respect to φ, then it is its own definitive prefix with respect to φ. That is:

Proposition 13 *Let w be a non-empty word and φ an* LTL^{trunc} *formula. Then*

$$dp(w,\varphi) \neq \top \Longrightarrow dp(w,\varphi) = dp(dp(w,\varphi),\varphi)$$

The definitive prefix of the truncated semantics is closely related to the concept of informative prefix in [12]. That work examines the problem of model checking safety formulas for standard LTL over maximal paths. Let a *safety formula* be a formula φ such that any path w violating φ contains a prefix $w^{0..k}$ all of whose infinite extensions violate φ [15]. Such a prefix is termed a *bad prefix* by [12]. Our intuitive notion of a bad prefix says that it should be enough to fully explain the failure of a safety formula. However, [12] showed that for LTL over maximal paths, there are safety formulas for which this does not hold. For instance, consider the formula $\varphi = (\mathsf{G}(q \vee \mathsf{FG}p) \wedge \mathsf{G}(r \vee \mathsf{FG}\neg p)) \vee \mathsf{G}q \vee \mathsf{G}r$. In standard LTL semantics, φ is equivalent to $\mathsf{G}q \vee \mathsf{G}r$, and the bad prefixes are exactly the finite words satisfying $\neg(\mathsf{G}q \vee \mathsf{G}r)$. However, we somehow feel that such a prefix is too short to "tell the whole story" of formula φ on path w, because it does not explain that $(\mathsf{FG}p) \wedge (\mathsf{FG}\neg p)$ is unsatisfiable.

The concept of a prefix which tells the whole story regarding the failure of formula φ on path w is formalized by [12] as an *informative prefix.* The precise definition in [12] is inductive over the finite path and the structure of $\neg\varphi$, which is assumed to be in positive normal form. The definition accomplishes an accounting of the discharge of the various sub-formulas of $\neg\varphi$ and is omitted due to lack of space. From the intuitive description, if u is an informative prefix for φ, then we should have that $u \models^{\!+} \neg\varphi$, or equivalently, $u \not\models^{\!-} \varphi$. The following theorem confirms this expectation and its converse.

Theorem 14 (Informative prefix theorem). *Let w be a non-empty finite word and φ an* LTL *formula.*

$$w \not\models^{\!-} \varphi \iff w \textit{ is informative for } \varphi$$

Notice that Theorem 14 shows that the notion of informative prefix for φ, defined in terms of syntactic structure, is captured semantically by the weak/strong truncated semantics. Furthermore, the definitive prefix does not require formulas to be in positive normal form, as does the informative prefix, and is symmetric in φ and $\neg\varphi$, as opposed to the informative prefix, which is defined only for formulas that do not hold. The precise relation of definitive prefixes to informative prefixes is given by the following corollary.

Corollary 15 *Let w be a non-empty path and let φ be an* LTL *formula.*
If $dp(w,\varphi) = \top$, then w has no informative prefix for either φ or $\neg\varphi$.
Otherwise, $dp(w,\varphi)$ is the shortest *informative prefix of w for either φ or $\neg\varphi$.*

4 Relation to Hardware Resets

There is an intimate relation between the problem of hardware resets and that of truncated vs. maximal paths: a hardware reset can be viewed as truncating the path and canceling future obligations; thus it corresponds to the weak view of truncated paths. We now consider the relation between the semantics given to the hardware reset operators of ForSpec [3] (termed the *reset semantics* by [2]) and of Sugar2.0 [8] (termed the *abort semantics* by [2]) and the truncated semantics we have presented above. We show that the truncated semantics are equivalent to the reset semantics, thus by [2], different from the abort semantics.

Reset Semantics The reset semantics are defined as follows, where [3] uses accept_on as the name of the trunc_w operator. Let a and r be mutually exclusive boolean expressions, where a is the condition for truncating a path and accepting the formula, and r is the condition for rejection. Let w be a non-empty word[2]. As before, we use φ and ψ to denote LTL^{trunc} formulas, p to denote an atomic proposition, and j and k to denote natural numbers. The reset semantics are defined in terms of a four-way relation between words, contexts a and r, and formulas, denoted $\models_{\mathcal{R}}$. The definition of the reset semantics makes use of a two-way relation between letters and boolean expressions which is defined in the obvious manner.

1. $\langle w, a, r\rangle \models_{\mathcal{R}} p \iff w^0 \models_{\mathcal{R}} a \vee (p \wedge \neg r)$
2. $\langle w, a, r\rangle \models_{\mathcal{R}} \neg\varphi \iff \langle w, r, a\rangle \not\models_{\mathcal{R}} \varphi$
3. $\langle w, a, r\rangle \models_{\mathcal{R}} \varphi \wedge \psi \iff \langle w, a, r\rangle \models_{\mathcal{R}} \varphi$ and $\langle w, a, r\rangle \models_{\mathcal{R}} \psi$
4. $\langle w, a, r\rangle \models_{\mathcal{R}} \mathsf{X!}\ \varphi \iff w^0 \models_{\mathcal{R}} a$ or ($w^0 \not\models_{\mathcal{R}} r$ and $|w| > 1$ and $\langle w^{1..}, a, r\rangle \models_{\mathcal{R}} \varphi$)
5. $\langle w, a, r\rangle \models_{\mathcal{R}} [\varphi\ \mathsf{U}\ \psi] \iff$ there exists $k < |w|$ such that $\langle w^{k..}, a, r\rangle \models_{\mathcal{R}} \psi$, and for every $j < k$, $\langle w^{j..}, a, r\rangle \models_{\mathcal{R}} \varphi$
6. $\langle w, a, r\rangle \models_{\mathcal{R}} \varphi$ trunc_w $b \iff \langle w, a \vee (b \wedge \neg r), r\rangle \models_{\mathcal{R}} \varphi$

Abort Semantics The abort semantics are defined in [8] as the traditional LTL semantics over finite and infinite (non-empty) paths, with the addition of a truncate operator (termed there abort), as follows, where we use $\models_{\mathcal{A}}$ to denote satisfaction under these semantics:

$$w \models_{\mathcal{A}} \varphi \ \mathsf{trunc\_w}\ b \iff \text{either } w \models_{\mathcal{A}} \varphi \text{ or there exist } j < |w| \text{ and word } w' \text{ such that } w^j \models_{\mathcal{A}} b \text{ and } w^{0..j-1}w' \models_{\mathcal{A}} \varphi$$

Intuitively, the reset and abort semantics are very similar. They both specify that the path up to the point of reset must be "well behaved", without regard to

[2] In [3], the reset semantics are defined over infinite words. We present a straightforward extension to (non-empty) finite as well as infinite words.

the future behavior of the path. The difference is in the way future obligations are treated, and is illustrated by the following formulas:

$$(\mathsf{G}(p \rightarrow \mathsf{F}(\varphi \wedge \neg\varphi))) \ \mathsf{trunc\_w} \ b \tag{1}$$

$$(\mathsf{G}\neg p) \ \mathsf{trunc\_w} \ b \tag{2}$$

Formulas 1 and 2 are equivalent in the abort semantics, because the future obligation $\varphi \wedge \neg\varphi$ is not satisfiable. They are not equivalent in the reset semantics, because the reset semantics "do not care" that $\varphi \wedge \neg\varphi$ is not satisfiable. Thus there exist values of w, a, and r such that Formula 1 holds under the reset semantics, while Formula 2 does not. For example, consider a word w such that p holds on w^5 and for no other letter and b holds on w^6 and on no other letter. If $a = r = \mathit{false}$, then Formula 1 holds on word w in the reset semantics under contexts a and r, while Formula 2 does not.

As shown in [2], the difference between the reset and the abort semantics causes a difference in complexity. While the complexity of model checking the reset semantics is PSPACE-complete, the complexity of model checking the abort semantics is SPACE$(\exp(k, n))$-complete where n is the length of the formula and k is the nesting depth of $\mathsf{trunc\_w}$.

Unlike the abort semantics, the truncated and reset semantics make no existential requirements of a path after truncation. The truncated semantics discard the remainder of the path after truncation, while the reset semantics accumulate the truncate conditions for later use. Theorem 16 states that they are the same.

Theorem 16 (Equivalence theorem). *Let φ be a formula of* LTL^{trunc}*, a and r mutually exclusive boolean expressions, and w a non-empty word. Then,*

$$\langle w, a, r \rangle \models_{\mathcal{R}} \varphi \iff w \models (\varphi \ \mathit{trunc\_w} \ a) \ \mathit{trunc\_s} \ r$$

In particular, for an infinite w, $\langle w, \mathit{false}, \mathit{false} \rangle \models_{\mathcal{R}} \varphi$ in the reset semantics if and only if $w \models \varphi$ in the truncated semantics. It follows easily that the truncated semantics are not more expressive or harder to decide than the reset semantics, which were shown in [2] to have the same expressiveness and complexity as LTL.

5 Related Work

Semantics for LTL is typically defined over infinite paths. Often finite paths are dealt with by infinitely repeating the last state (see e.g. [7]). Lichtenstein et al. [14] were the first to extend the semantics of LTL to finite paths. In particular, they introduced the *strong next* operator (see also [13],[15, pages 272-273]). However, they consider only finite maximal paths, and the issue of truncated paths is not considered.

The issue of using temporal logic specifications in simulation is addressed by [1]. They consider only a special class of safety formulas [4] which can be translated into formulas of the form $\mathsf{G}p$, and do not distinguish between maximal and truncated paths.

The idea that an obligation need not be met in the weak view if it "is the fault of the test" is directly related to the idea of weak clocks in [8], in which obligations need not be met if it "is the fault of the clock". The weak/strong clocked semantics of [8] were the starting point for investigations that have led to [9], which proposes a clocked semantics in which the clock is strengthless, and to the current work, which preserves much of the intuition of the weak/strong clocked semantics in a simpler, unclocked setting.

The work described here is the result of discussions in the LRM sub-committee of the Accellera Formal Verification Technical Committee. Three of the languages (Sugar2.0, ForSpec, CBV [10]) examined by the committee enhance temporal logic with operators intended to support hardware resets. We have discussed the reset and abort semantics of ForSpec and Sugar2.0 in detail. The operator of CBV, while termed abort, has semantics similar to those of ForSpec's accept_on/reject_on operators. As we have shown, our truncated semantics are mathematically equivalent to the reset semantics of ForSpec. However, the reset semantics take the operational view in that they tell us in a fairly direct manner how to construct an alternating automaton for a formula. Our approach takes the denotational view and thus tells us more directly the effect of truncation on the formula. This makes it easy to reason about the semantics in a way that is intuitively clear, because we can reason explicitly about three constant contexts (weak/neutral/strong) which are implicit in the operational view.

Bounded model checking [5] considers the problem of searching for counterexamples of finite length to a given LTL formula. The method is to solve the existential model checking problem for $\psi = \neg\varphi$, where φ is an LTL formula to be checked. An infinite path π of a model M that shows that $M \models \mathsf{E}\psi$ is called a *witness* for ψ. The existence of a witness has to be demonstrated by exhibiting an appropriate path π_k of finite length k. In some cases, this can be done by finding a path π_k with a loop (two identical states); this can be expanded to an infinite path in which the loop is repeated infinitely often. But a finite path π_k can also demonstrate the existence of a witness, even if it is not known to have a loop. This can be understood in the framework of the truncated semantics as follows: The *bounded semantics without a loop* of [5] is the strong semantics for LTL formulas in positive normal form. If π_k satisfies ψ in this semantics, then by Theorems 7 and 3, every extension of π_k satisfies ψ in the neutral semantics. Assuming there is at least one transition from every state in M, there is an infinite extension π of π_k that is a computation path of M. Then π is a witness for ψ. Conversely, if there is no path of length k satisfying ψ in the bounded semantics without a loop, then every path of length k weakly satisfies φ. As noted in [5], the bounded semantics without a loop break the duality between strong and weak operators. The truncated semantics provide the missing dual weak semantics, and therefore render unnecessary the restriction of [5] to positive normal form.

The *completeness threshold* ($\mathcal{CT}$) of [11] is reminiscent of our definitive prefix. However, $\mathcal{CT}$ is defined with respect to a *model* and a formula while the definitive prefix is defined with respect to a *word* and a formula. Even if we try to compare the definitions by taking for a word w a model M_w that accepts w alone, the definitions do not coincide: the definitive prefix for any word with respect to the formula $\mathsf{G}p$ is $\top$ but there exists a model M_w accepting $\mathsf{G}p$ with a bounded $\mathcal{CT}$.

In [17] the problem of determining the value of a formula over finite paths in simulation is also considered. Their semantics can be formulated using the notion of bad/good prefixes by defining a 3-valued satisfaction that returns *true* if a *good prefix* [12] is seen, *false* if a *bad prefix* is seen and *pending* otherwise. The resulting semantics is different than the truncated semantics and is quite similar to the abort semantics.

6 Conclusion and Future Work

We have considered the problem of reasoning in temporal logic over truncated as well as maximal paths, and have presented an elegant semantics for LTL augmented with a truncate operator over truncated and maximal paths. The semantics are defined relative to three views regarding the truth value of a formula when the truncation occurs before the evaluation of the formula completes. The *weak view* is consistent with a preference for false positives, the *strong view* with a preference for false negatives, and the *neutral view* with the desire to see as much evidence as can reasonably be expected from a finite path.

We have studied properties of the *truncated semantics* for the resulting logic LTL^{trunc}, as well as its relation to the *informative prefixes* of [12]. We have examined the relation between truncated paths and hardware resets, and have shown that our truncated semantics are mathematically equivalent to the *reset semantics* of [3].

Future work is to investigate how the weak/neutral/strong paradigm can be generalized: in particular, whether there are useful correspondences between alternative weak/neutral/strong semantics and other decision procedures for LTL, analogous to that between the truncated semantics and the classical tableau construction. Having a generalized framework, we might be able to find a logic that has the acceptable complexity of the truncated semantics, while allowing rewrite rules such as $(\varphi \wedge \neg\varphi \stackrel{\text{def}}{=} \mathit{false})$, which are prohibited in the truncated semantics.

In addition, we would like to combine the truncated semantics with those of $\text{LTL}^{@}$ [9], to provide an integrated logic which supports both hardware clocks and hardware resets for both complete and incomplete verification methods.

Acknowledgements

The IBM authors would like to thank Ishai Rabinovitz for many interesting and fruitful discussions, and Shoham Ben-David, Avigail Orni, and Sitvanit Ruah for close review and important comments on an earlier version of this work.

References

1. Y. Abarbanel, I. Beer, L. Gluhovsky, S. Keidar, and Y. Wolfsthal. FoCs - automatic generation of simulation checkers from formal specifications. In *Proc. 12th International Conference on Computer Aided Verification (CAV)*, LNCS 1855, 2000.

2. R. Armoni, D. Bustan, O. Kupferman, and M. Y. Vardi. Aborts vs resets in linear temporal logic. In *TACAS'03*, 2003. To appear.
3. R. Armoni, L. Fix, A. Flaisher, R. Gerth, B. Ginsburg, T. Kanza, A. Landver, S. Mador-Haim, E. Singerman, A. Tiemeyer, M. Y. Vardi, and Y. Zbar. The ForSpec temporal logic: A new temporal property-specification language. In *TACAS'02*, volume 2280 of *LNCS*. Springer, 2002.
4. I. Beer, S. Ben-David, and A. Landver. On-the-fly model checking of RCTL formulas. In *Proc.* 10th *International Conference on Computer Aided Verification (CAV)*, LNCS 1427, pages 184–194. Springer-Verlag, 1998.
5. A. Biere, A. Cimatti, E. Clarke, and Y. Zhu. Symbolic model checking without BDDs. In *Proc.* 5th *International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, LNCS 1579. Springer-Verlag, 1999.
6. L. Brouwer. *On the Foundations of Mathematics.* PhD thesis, Amsterdam, 1907. English translation in A. Heyting, Ed. L. E. J. Brouwer: Collected Works 1: Philosophy and Foundations of Mathematics, Amsterdam: North Holland / New York: American Elsevier (1975): 11-101.
7. E. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith. Progress on the state explosion problem in model checking. In *Informatics - 10 Years Back. 10 Years Ahead (Informatics'01)*, 2001.
8. C. Eisner and D. Fisman. Sugar 2.0 proposal presented to the Accellera Formal Verification Technical Committee, March 2002. At http://www.haifa.il.ibm.com/projects/verification/sugar/Sugar_2.0_Accellera.ps.
9. C. Eisner, D. Fisman, J. Havlicek, A. McIsaac, and D. Van Campenhout. The definition of a temporal clock operator. 13th International Colloquium on Automata, Languages and Programming (ICALP), June 2003. To appear.
10. J. Havlicek, N. Levi, H. Miller, and K. Shultz. Extended CBV statement semantics, partial proposal presented to the Accellera Formal Verification Technical Committee, April 2002. At http://www.eda.org/vfv/hm/att-0772/01-ecbv_statement_semantics.ps.gz.
11. D. Kroening and O. Strichman. Efficient computation of recurrence diameters. In L. Zuck, P. Attie, A. Cortesi, and S. Mukhopadhyay, editors, *4th International Conference on Verification, Model Checking, and Abstract Interpretation*, volume 2575 of *LNCS*, pages 298–309. Springer Verlag, January 2003.
12. O. Kupferman and M. Y. Vardi. Model checking of safety properties. In *Proc.* 11th *International Conference on Computer Aided Verification (CAV)*, LNCS 1633, 1999.
13. O. Lichtenstein. *Decidability, Completeness, and Extensions of Linear Time Temporal Logic.* PhD thesis, Weizmann Institute of Science, 1990.
14. O. Lichtenstein, A. Pnueli, and L. Zuck. The glory of the past. In *Proc. Conf. on Logics of Programs*, LNCS 193, pages 196–218. Springer-Verlag, 1985.
15. Z. Manna and A. Pnueli. *Temporal Verification of Reactive Systems: Safety.* Springer-Verlag, New York, 1995.
16. A. Pnueli. A temporal logic of concurrent programs. *Theoretical Computer Science*, 13:45–60, 1981.
17. J. Ruf, D. Hoffmann, T. Kropf, and W. Rosenstiel. Simulation-guided property checking based on a multi-valued AR-automata. In *Proceedings of the DATE 2001 on Design, Automation and Test in Europe*, pages 742–748, March 2001.