A Topological Characterization of Weakness*

Cindy Eisner
Formal Methods Group
IBM Haifa Research Lab
Haifa 31905, Israel
eisner@il.ibm.com

Dana Fisman
IBM Haifa Research Lab and
Weizmann Institute of Science
Rehovot 76100, Israel
dana.fisman@weizmann.ac.il

John Havlicek
Functional Design Verification
Freescale Semiconductor, Inc.
Austin, TX 78729, USA
john.havlicek@freescale.com

ABSTRACT

We are interested in the relation between weak and strong temporal operators. We would like to find a characterization that shows what it means for an operator to be the weak or strong version of another operator, or more generally for a formula to be a weak or strong version of another formula. We show that the weak version of a formula is not the same as Alpern and Schneider's *safety component*. By working over an extended alphabet, we show that their topological characterization of safety can be adapted to obtain a topological characterization of weakness. We study the resulting topology and the relations between weak and strong formulas. Finally, we apply the method to show the internal consistency of a logic containing both weak and strong versions of regular expressions.

Categories and Subject Description

F.4.1 [**Mathematical Logic and Formal Languages**]: Mathematical Logic - Temporal Logic

General Terms

Languages, Theory, Verification

Keywords

safety, liveness, safety component, weak and strong components, semantically weak and strong formulas, syntactically weak and strong formulas, temporal weakness and strength

1. INTRODUCTION

Some temporal logic operators come in *weak* and *strong* versions. The strong form requires that some condition eventually occur, while the weak form makes no such requirement. For instance, the LTL [15] formula $p \mathsf{U} q$ (read p *strong-until* q) states that p holds until q holds, and in addition, that q eventually holds. The weak version, $p \mathsf{W} q$ (read p *weak-until* q) states that p holds until q holds, but makes no statement about whether or not q eventually holds (if it does not, then p must hold forever).

We are interested in the relation between weak and strong temporal operators. We would like to find a characterization that shows what it means for an operator to be the weak or strong version of another operator, or more generally for a formula to be a weak or strong version of another formula. Our motivating example will be the pair of LTL operators *weak-until* (W) and *strong-until* (U). That is, we are looking for a characterization that will let us say that two formulas, φ and φ', are related in the same way as the pair $\psi = p \mathsf{W} q$ and $\psi' = p \mathsf{U} q$, and furthermore, that the weak version of any new temporal operator in some extension of LTL is related to its strong version in the same way that W is related to U. Although our work is theoretical, our motivation is practical: we would like to create a framework under which we can test whether the weak and strong operators of a new extension of LTL have a solid theoretical basis.

We begin by noting that the weak/strong dichotomy is related to another dichotomy in temporal logic, that of *safety* vs. *liveness* properties. The distinction between safety and liveness plays an important role in the design and analysis of reactive systems, and, as noted in [14], introductory textbooks on distributed computing now cover the issue. Loosely speaking, a safety property claims that "something bad" does not happen, while a liveness propery claims that "something good" eventually happens. Some properties are neither safety nor liveness. For instance, the property described by the formula $\mathsf{G} p$ is a safety formula, the property described by the formula $\mathsf{F} q$ is a liveness formula, and the property described by the formula $p \mathsf{U} q$, equivalent to $(p \mathsf{W} q) \wedge \mathsf{F} q$, is neither.

Alpern and Schneider [1] showed that any property can be decomposed into a safety property and a liveness property whose conjunction is the original. Their proof is based on the observation that the safety properties are the closed sets of the Cantor topology on the space of infinite words and the liveness properties are the dense sets in this topology. In [2] they refer to the safety property obtained by taking the closure in this topology as the *safety component*. Intuitively,

*The work of the first two authors was partially supported by the European Community project FP6-IST-507219 (PROSYD). The work of the second author was carried out at the John von Neumann Minerva Center for the Verification of Reactive Systems.

*A full version of this paper is available at `http://www.wisdom.weizmann.ac.il/~dana/podc05/full.ps`.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
PODC'05, July 17–20, 2005, Las Vegas, Nevada, USA.
Copyright 2005 ACM 1-58113-994-2/05/0007 ...$5.00.

their safety component should give a weak version of the original property. However, this does not hold in all cases. For example, $p \mathsf{W} \mathit{false}$ is the weak version of $p \mathsf{U} \mathit{false}$, but the safety component of the property described by $p \mathsf{U} \mathit{false}$ is the empty set.

As we discussed in [6], many verification methods can examine only a finite prefix of an infinite execution. Therefore we are interested in a characterization that includes finite as well as infinite words. We show that Alpern and Schneider's topological ideas can be generalized to the set of finite and infinite words, and that the desired *weak component* can be obtained by taking the closure in an analogous topology where the alphabet is extended with two special letters, $\top$ and $\bot$. The resulting framework allows us to show interesting relationships between weak and strong operators and formulas. For example, we show that the set of semantically weak formulas is exactly the set of "good" safety properties, thus strengthening the result in [11]. Finally, we apply the method to show the internal consistency of a logic containing both weak and strong versions of regular expressions.

Outline

The remainder of the paper is organized as follows. In Section 2 we provide an extension of LTL to both finite and infinite words. In Section 3 we extend the topological characterization of Alpern and Schneider to the space of finite and infinite words. Section 4 is the heart of the paper in which we characterize weakness and strength, present some basic results, and show the relation of our work to that of [1] and [11]. In Section 5 we apply our ideas to test the relation between weak and strong regular expressions. In Section 6 we discuss related work and in Section 7 we conclude. The propositions are given here without proofs. The proofs appear in the full version of the paper, which is available on http://www.wisdom.weizmann.ac.il/~dana/podc05/full.ps.

2. PRELIMINARIES

We denote a letter from a given alphabet Γ by ℓ, and an empty, finite, or infinite word from Γ by u, v, or w. The *concatenation* of u and v is denoted by uv. If u is infinite, then $uv = u$. The empty word is denoted by ϵ, so that $w\epsilon = \epsilon w = w$. If $w = uv$, we define $w/u = v$ and we say that u is a *prefix* of w, denoted $u \preceq w$, that v is a *suffix* of w, and that w is an *extension* of u, denoted $w \succeq u$.

We denote the *length* of word v as $|v|$. The empty word ϵ has length 0, a finite word $v = (\ell_0 \ell_1 \cdots \ell_n)$ has length $n+1$, and an infinite word has length ω.[1] We use i, j, and k to denote non-negative integers. For $i < |v|$ we use v^i to denote the $(i+1)^{st}$ letter of v (since counting of letters starts at zero). We denote by $v^{i\cdot\cdot}$ the suffix of v starting at v^i. We denote by ℓ^k the word of length k, each letter of which is ℓ, and by ℓ^ω the infinite-length word, each letter of which is ℓ.

We denote a set of finite/infinite words by U, V or W and refer to them as *properties*. The *concatenation* of U and V, denoted UV, is the set $\{uv \mid u \in U,\ v \in V\}$. Define $V^0 = \{\epsilon\}$ and $V^k = VV^{k-1}$ for $k \geq 1$. The *Kleene closure* of V, denoted V^*, is the set $V^* = \bigcup_{k<\omega} V^k$. The notation V^+ is used for the set $\bigcup_{0<k<\omega} V^k$. The infinite concatenation of V to itself is denoted V^ω. The union $V^* \cup V^\omega$ is denoted V^∞. For a letter ℓ we use ℓ^*, ℓ^+, and ℓ^∞ as abbreviations of $\{\ell\}^*$, $\{\ell\}^+$ and $\{\ell\}^\infty$, respectively.

[1] Here ω denotes the first transfinite ordinal number.

2.1 LTL over Finite and Infinite Paths

We use a simple extension of LTL over finite as well as infinite and empty as well as non-empty words, which we denote ELTL. To do so, the syntax of LTL is extended to include two *next-time* operators, one weak (X) and one strong ($\mathsf{X!}$) [13, pp. 272-273]. The semantics distinguish between the weak and strong versions only on the last letter of a word: $\mathsf{X}\ \varphi$ holds on the last letter of any word for any φ, and $\mathsf{X!}\ \varphi$ does not. Similarly, since we interpret the logic over the empty word as well, we define weak and strong boolean expressions, so that the strong boolean expression is satisfied over a word if the first letter satisfies the boolean expression, and the weak boolean expression is satisfied also if there is no first letter, i.e. if the word is empty.

We define ELTL formulas with respect to a non-empty set of atomic propositions P, and a set of boolean expressions B over P, which we identify with 2^{2^P}, as follows:

DEFINITION 2.1 (ELTL FORMULAS). *Let b be boolean expression. The set of* ELTL formulas *is recursively defined as follows:*

$$\varphi ::= b! \mid b \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathsf{X!}\ \varphi \mid \mathsf{X}\ \varphi \mid \varphi\ \mathsf{U}\ \varphi \mid \varphi\ \mathsf{W}\ \varphi$$

Additional syntactic operators are defined as follows:

- $\mathsf{F}\ \varphi \stackrel{\text{def}}{=} \mathit{true}\ \mathsf{U}\ \varphi$
- $\mathsf{G}\ \varphi \stackrel{\text{def}}{=} \varphi\ \mathsf{W}\ \mathit{false}$

Here *true* and *false* denote the elements 2^P and $\emptyset$ of B, respectively.

The semantics of ELTL is defined inductively with respect to the alphabet $\Sigma = 2^P$. For a boolean expression $b \in B = 2^{2^P}$ and a letter $\ell \in \Sigma = 2^P$ we define the boolean satisfaction relation $\Vdash$ by $\ell \Vdash b$ iff $\ell \in b$.

DEFINITION 2.2. *The relation $\models$ is defined as follows, where v denotes a word, b denotes a boolean expression, and φ and ψ denote* ELTL *formulas.*

1. $v \models b! \iff |v| > 0$ *and* $v^0 \Vdash b$
   $v \models b \iff$ *either* $v \models b!$ *or* $|v| = 0$
2. $v \models \neg\varphi \iff v \not\models \varphi$
3. $v \models \varphi \vee \psi \iff v \models \varphi$ *or* $v \models \psi$
   $v \models \varphi \wedge \psi \iff v \models \varphi$ *and* $v \models \psi$
4. $v \models \mathsf{X!}\ \varphi \iff |v| > 1$ *and* $v^{1\cdot\cdot} \models \varphi$
   $v \models \mathsf{X}\ \varphi \iff$ *either* $v \models \mathsf{X!}\ \varphi$ *or* $|v| \leq 1$
5. $v \models \varphi\ \mathsf{U}\ \psi \iff \exists k < |v| : v^{k\cdot\cdot} \models \psi$ *and* $\forall j < k : v^{j\cdot\cdot} \models \varphi$
   $v \models \varphi\ \mathsf{W}\ \psi \iff$ *either* $v \models \varphi\ \mathsf{U}\ \psi$ *or* $\forall k < |v| : v^{k\cdot\cdot} \models \varphi$

3. SAFETY AND LIVENESS

In this section we show an extension of the topological characterization of Alpern and Schneider [1] to the set of finite and infinite words. We further show a number of topological ideas that come out of this framework. We assume that the reader has some familiarity with elementary topology and refer to [10] for background.

Intuitively, a formula φ is said to define a *safety* property iff any word violating φ contains a finite prefix all of whose extensions violate φ. A formula φ is said to define a *liveness* property iff any arbitrary finite word has an extension satisfying φ. The formal definitions for safety and liveness in the literature correspond to reasoning over infinite words alone. Our definitions modify those of [13] to reason about finite words as well.

DEFINITION 3.1 (SAFETY,LIVENESS). *Let $W \subseteq \Gamma^\infty$.*

- *W is a* safety *property if for all $w \in \Gamma^\infty - W$ there exists a finite prefix $u \preceq w$ such that for all $v \succeq u$, $v \in \Gamma^\infty - W$.*
- *W is a* liveness *property if for all finite u, there exists $v \succeq u$ such that $v \in W$.*

A formula is said to be a *safety (liveness)* formula iff its language is a safety (liveness) property. Some formulas are neither safety nor liveness. For instance, $\mathsf{G}\ p$ is a safety formula, $\mathsf{F}\ q$ is a liveness formula, and $p\ \mathsf{U}\ q$, equivalent to $(p\ \mathsf{W}\ q) \wedge \mathsf{F}\ q$, is neither.

Alpern and Schneider showed in [1] that any property on infinite words can be decomposed into a safety property and a liveness property whose conjunction is the original. Their proof is based on the observation that the safety properties are the closed sets of the Cantor topology on Γ^ω and the liveness properties are the dense sets in this topology. In [2] they refer to the safety property obtained by taking the closure in such a topology, where Γ is set to $\Sigma = 2^P$, as the *safety component*. The following proposition asserts that the same decomposition exists when considering the space of finite and infinite words, with the above definitions of safety and liveness.

PROPOSITION 3.2. *The safety properties are the closed sets of a topology on Γ^∞ and the liveness properties are the dense sets in this topology. Furthermore, every property on Γ^∞ can be decomposed into a safety and a liveness property whose conjunction is the original.*

We understand Γ^∞ to be topologized in this way. From now on, whenever we say "the topology of Γ^∞" we refer to the topology given in Proposition 3.2. Note that we may use arbitrary alphabets for Γ.

3.1 Topological Characteristics

In the following we define the closure and interior of a set of finite and infinite words over a given alphabet Γ.

DEFINITION 3.3. *Let V be a property of Γ^∞. Define*

- $cl(V) = \{w \in \Gamma^\infty \mid \forall \text{ finite } u \preceq w : \exists v \succeq u : v \in V\}$
- $int(V) = \{w \in \Gamma^\infty \mid \exists \text{ finite } u \preceq w : \forall v \succeq u : v \in V\}$

The following proposition asserts that the above definition characterizes the closure and interior operators in the topological space Γ^∞.

PROPOSITION 3.4.

- *$cl(\cdot)$ is the* closure *operator in the topology of Γ^∞*
- *$int(\cdot)$ is the* interior *operator in the topology of Γ^∞*

Thus, by Propositions 3.2 and 3.4 a property U is safety iff $cl(U) = U$ and it is liveness iff $cl(U) = \Gamma^\infty$ or equivalently, $int(\Gamma^\infty - U) = \emptyset$.

We characterize the topology on the space of finite and infinite words in terms of prefix closed sets and extension closed sets.

DEFINITION 3.5 (PREFIX/EXTENSION CLOSED SETS). *Let $V \subseteq \Gamma^\infty$.*

- *V is* prefix closed *if $u \preceq v \in V$ implies $u \in V$.*
- *V is* extension closed *if $w \succeq v \in V$ implies $w \in V$.*

Since we are interested in both finite as well as infinite words, we need the following notions as well.

DEFINITION 3.6 (LIMIT CLOSED, FINITE WITNESSES). *Let $V \subseteq \Gamma^\infty$.*

- *V is* limit closed *if: forall $w \in \Gamma^\infty$, if all finite prefixes of w are in V then $w \in V$.*
- *V has* finite witnesses *if: forall $w \in V$, there exists finite $u \preceq w$ such that $u \in V$.*

We observe that prefixed closed sets and extension closed sets are dual to each other. That is, V is prefix closed iff $\Gamma^\infty - V$ is extension closed. Similarly, V is limit closed iff $\Gamma^\infty - V$ has finite witnesses.

The notions of prefix/extension closed are stronger than the notions of limit closed and finite witnesses in the following sense. If V is prefix closed then V has finite witnesses. And, if V is extension closed then V is limit closed.

The following proposition shows that closed and open sets can be defined alternatively using the above concepts.

PROPOSITION 3.7. *Let $V \subseteq \Gamma^\infty$.*

1. *V is (topologically) closed iff V is both prefix closed and limit closed.*
2. *V is open iff V is extension closed and has finite witnesses.*

3.2 Examples

We consider the following set of properties, which are slight modifications of Rem's examples [16] (see also [14]). In these examples, P denotes the set of atomic propositions, $\Sigma = 2^P$ is the alphabet, $B = 2^{2^P}$ is the set of boolean expressions, and a,b,c denote particular boolean expressions in $B - \{true, false\}$.

P_0: The empty set ($\emptyset$)
P_1: The empty word ($\{\epsilon\}$)
P_2: The set of non-empty finite words (Σ^+)
P_3: The set of finite words (Σ^*)
P_4: The set of infinite words (Σ^ω)
P_5: The set of finite and infinite words (Σ^∞)
P_6: There are at most 3 letters. The first (resp., second, third) letter, if it exists, satisfies a (resp., b, c). ($abc \cup ab \cup a \cup \{\epsilon\}$)
P_7: There are at least 3 letters. The first and third letters satisfy b and the second satisfies c ($bcb\Sigma^\infty$)
P_8: The first letter satisfies a and there exists a letter that does not satisfy a ($a\Sigma^*(\Sigma - a)\Sigma^\infty$)
P_9: The number of letters satisfying a is finite ($\Sigma^*(\Sigma - a)^\infty$)
P_{10}: The number of letters satisfying a is infinite ($(\Sigma^* a)^\omega$)
P_{11}: There is a letter satisfying b and every prior letter satisfies a ($a^* b\Sigma^\infty$)

For each property P_i above we give an ELTL formula φ_i such that $P_i = \{w \in \Sigma^\infty \mid w \models \varphi_i\}$.

φ_0: *false*!
φ_1: *false*
φ_2: $\mathsf{F}\ \mathsf{X}\ \mathit{false}$
φ_3: $\mathit{false} \vee \mathsf{F}\ \mathsf{X}\ \mathit{false}$
φ_4: $\mathit{true}! \wedge \mathsf{G}\ \mathsf{X}!\ \mathit{true}$
φ_5: *true*
φ_6: $a \wedge \mathsf{X}\ (b \wedge \mathsf{X}\ (c \wedge \mathsf{X}\ \mathit{false}))$
φ_7: $b! \wedge \mathsf{X}!\ c \wedge \mathsf{X}!\mathsf{X}!\ b$
φ_8: $a! \wedge \mathsf{F}\ \neg a$
φ_9: $\varphi_3 \vee \mathsf{F}\ \mathsf{G}\ \neg a$
φ_{10}: $\varphi_4 \wedge \mathsf{G}\ \mathsf{F}\ a$
φ_{11}: $a\ \mathsf{U}\ b$

For each of these properties we state whether it is prefix closed, extension closed, limit closed, has finite witnesses. Properties P_0 and P_5 have all 4 qualities. Properties P_3 and P_9 are prefix closed and (so) have finite witnesses but are neither extension closed nor limit closed. Property P_2 is limit closed and has finite witnesses but is neither prefix closed nor extension closed. Properties P_4 and P_{10} are extension closed and (so) limit closed but are neither prefix closed nor have finite witnesses. Properties P_7, P_8 and P_{11} have all qualities but being prefix closed. Finally, properties P_1 and P_6 have all qualities but being extension closed.

Thus by Proposition 3.7, P_0, P_1, P_5 and P_6 are closed and P_0, P_5, P_7, P_8 and P_{11} are open. It follows that φ_0, φ_1, φ_5, φ_6 are safety. The closure of properties P_2, P_3, P_4, P_5, P_9, and P_{10} is Σ^∞ and thus φ_2, φ_3, φ_4, φ_5, φ_9 and φ_{10} are liveness. Properties φ_7, φ_8 and φ_{11} are neither safety nor liveness (the closure of P_7 is $\{\epsilon\} \cup b \cup bc \cup P_7$, the closure of P_8 is $\{\epsilon\} \cup a\Sigma^\infty$, and the closure of P_{11} is $a^\infty \cup P_{11}$). Note that P_0 and P_5 are both open and closed, and that φ_5 is both safety and liveness.

4. WEAKNESS AND STRENGTH

4.1 The Characterization

Our goal is to define the *weak component* of an ELTL formula φ, such that $p\ \mathsf{W}\ q$ is the weak component of $p\ \mathsf{U}\ q$, $\mathsf{X}\ p$ is the weak component of $\mathsf{X}!\ p$, and such that the weak component of an ELTL formula ψ is related to ψ in the same way that $p\ \mathsf{W}\ q$ is related to $p\ \mathsf{U}\ q$.

First we note that Alpern and Schneider's safety component does not achieve our goal. Their safety component is the closure in the topology of Γ^ω, where Γ is set to $\Sigma = 2^P$. Thus, the property described by $p\ \mathsf{W}\ q$ is not the safety component of $p\ \mathsf{U}\ q$ if q is set to *false*. Eliminating boolean expressions equivalent to *false* does not solve the problem. For instance, the safety component of $(\mathit{true}\ \mathsf{U}\ \mathsf{G}p) \wedge (\mathit{true}\ \mathsf{U}\ \mathsf{G}\neg p) \equiv \mathit{false}!$ is the empty set, while we would like the weak component to be the property described by $(\mathit{true}\ \mathsf{W}\ \mathsf{G}p) \wedge (\mathit{true}\ \mathsf{W}\ \mathsf{G}\neg p) \equiv \mathit{true}$.[2]

Furthermore we note that it is not possible to define the weak component of a *property* in a way that meets our goal. The reason is that $p\ \mathsf{U}\ \mathit{false} \equiv \mathsf{X}!\ \mathit{false}$ (that is, both formulas define the same property), but our goal dictates that their weak components be different. Nonetheless, we would like to obtain a semantic rather than syntactic characterization, so that it will capture the essence of the relation between weak and strong formulas and so that it can be applied to any new temporal operator.

On the one hand, we want a characterization that distinguishes between two formulas which are syntactically distinct, but semantically equivalent. On the other hand, we want a characterization that is semantic, rather than syntactic. It seems that we have set ourselves an impossible task. Our approach is to enhance the set of models over which formulas are interpreted such that previously equivalent formulas for which our goal dictates different weak components are no longer equivalent, in such a way that such formulas remain equivalent when restricted to the original set of models.

Our solution is based on noticing that the "problem" with Alpern and Schneider's safety component is that the property described by $p\ \mathsf{U}\ \mathit{false}$ is the empty one. We therefore augment the set of models of a formula in such a way that no formula describes the empty property, while ensuring that the satisfiability over the original set of models ($(2^P)^\infty$) is not compromised. We accomplish this by augmenting the alphabet with two special letters $\top$ and $\bot$, such that $\top^\omega$ models every formula and $\bot^\omega$ models no formula.

Formally, we define the semantics of ELTL over an enhanced alphabet $\widehat{\Sigma} = 2^P \cup \{\top, \bot\}$. We use $\overline{w}$ to denote the word obtained by replacing every $\top$ with a $\bot$ and vice versa. For a property W, we use $\overline{W}$ to denote the property $\{\overline{w} \mid w \in W\}$. [3] We call $\overline{w}$ ($\overline{W}$) the *dual word (property)* of w (W). For a property W, we denote by W^c the property $\widehat{\Sigma}^\infty - \overline{W}$. We refer to W^c as the *complement* property of W.

We augment the boolean satisfaction relation $\models$ to all of $\widehat{\Sigma}$ by defining $\top \models b$ and $\bot \not\models b$ for any boolean expression b.

DEFINITION 4.1. *The semantics of* ELTL *over the alphabet* $\widehat{\Sigma}$ *are exactly as in Definition 2.2, but using the augmented boolean satisfaction relation and using the dual word in the semantics of negation as follows:*

$$v \models \neg\varphi \iff \overline{v} \not\models \varphi$$

.

[2] $\varphi \equiv \psi$ denotes that $\{w \mid w \models \varphi\} = \{w \mid w \models \psi\}$.
[3] Note that the notation $\overline{W}$ should not be confused with the topological closure of W, which we denote $cl(W)$.

We denote the set $\{w \in \widehat{\Sigma}^\infty \mid w \models \varphi\}$ by $[\![\varphi]\!]$.

Note that in particular we get that $\top \models \mathit{false}$ and $\bot \not\models \mathit{true}$. The inductive definition guarantees that $\top^\omega$ satisfies every formula and $\bot^\omega$ satisfies no formula.

PROPOSITION 4.2. *Let φ be an* ELTL *formula. Then $\top^\omega \models \varphi$ and $\bot^\omega \not\models \varphi$.*

Moreover, if a word w satisfies a formula φ then the word obtained from w by replacing some of w's letters with $\top$ will also satisfy φ. Similarly, if w does not satisfy φ then by replacing some of w's letters with $\bot$ we get a word that does not satisfy φ. We say that word w is *finitely consistent with* φ if for every finite prefix $u \preceq w$, $u\top^\omega$ satisfies φ. For finite w, it follows that w is finitely consistent with φ iff $w\top^\omega \models \varphi$. For example, bc is finitely consistent with $\varphi_7 : b! \wedge \mathsf{X}! \; c \wedge \mathsf{X}!\mathsf{X}! \; b$, and both aaa and a^ω are finitely consistent with $\varphi_{11} : a \; \mathsf{U} \; b$. Similarly we say that word w *finitely establishes* φ if w has a finite prefix u such that u and all its extensions, including $u\bot^\omega$, satisfy φ. For example bcb and $bcbaa$ finitely establish φ_7, and aab finitely establishes φ_{11}.

We are now ready to define the weak and strong components of a formula. We do so directly, and then show the relation to [1]. In analogy with the notions of weak and strong satisfaction in [6], we let the weak component of a formula φ consist of all words that are finitely consistent with φ, and we let the strong component consist of all words that finitely establish φ. Formally,

DEFINITION 4.3 (WEAK AND STRONG COMPONENTS). *Let φ be an* ELTL *formula. Its weak and strong components, denoted $weak(\varphi)$ and $strong(\varphi)$ respectively, are defined as follows.*

- $weak(\varphi) = \{w \in \widehat{\Sigma}^\infty \mid \forall \textit{ finite } u \preceq w : u\top^\omega \in [\![\varphi]\!]\}$
- $strong(\varphi) = \{w \in \widehat{\Sigma}^\infty \mid \exists \textit{ finite } u \preceq w : u\bot^\omega \in [\![\varphi]\!]\}$

The safety component of Alpern and Schneider is obtained by taking the closure in the Cantor topology on the set of infinite words over the alphabet $\Sigma = 2^P$, where the closed sets are the safety properties. The following proposition states that our weak component is obtained by taking the closure in a topology on both finite and infinite words over the augmented alphabet $\widehat{\Sigma} = \Sigma \cup \{\top, \bot\}$, and that dually, the strong component is obtained by taking the interior in the same topology.

PROPOSITION 4.4. *Let φ be an* ELTL *formula. Then,*

- $cl([\![\varphi]\!]) = weak(\varphi)$
- $int([\![\varphi]\!]) = strong(\varphi)$

From basic results in topology we conclude:

COROLLARY 4.5. *Let φ be an* ELTL *formula.*

$$strong(\varphi) \subseteq [\![\varphi]\!] \subseteq weak(\varphi)$$

We say that a formula is *semantically weak* iff it is equivalent to its weak component. We say that a formula is *semantically strong* iff it is equivalent to its strong component.

DEFINITION 4.6 (SEMANTICALLY WEAK/STRONG). *Let φ be an* ELTL *formula.*

- *φ is* semantically weak *iff $weak(\varphi) = [\![\varphi]\!]$*
- *φ is* semantically strong *iff $strong(\varphi) = [\![\varphi]\!]$*

The following proposition asserts that semantically weak and strong formulas are dual to each other.

PROPOSITION 4.7. *Let φ be an* ELTL *formula. Then, φ is semantically strong iff $\neg\varphi$ is semantically weak.*

To see that the above definitions capture our intuition, we would like to state that the weak operators are the weak components of their strong counterparts. In other words that what is syntactically weak (strong) is also semantically weak (strong). We will first give a formal definition of syntactically weak/strong formulas similar to that of [17], then state the desired result.

DEFINITION 4.8 (SYNTACTICALLY WEAK/STRONG). *Let b be a boolean expression.*

- *The set of* syntactically weak ELTL *formulas is recursively defined as follows:*

$$\varphi \; ::= b \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathsf{X} \, \varphi \mid \varphi \; \mathsf{W} \; \varphi$$

- *The set of* syntactically strong ELTL *formulas is recursively defined as follows:*

$$\varphi \; ::= b! \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathsf{X}! \, \varphi \mid \varphi \; \mathsf{U} \; \varphi$$

The following proposition shows that for each formula in positive normal form, weakening/strenthening each of the operators results in the weak/strong component, respectively.

PROPOSITION 4.9. *Let φ be an* ELTL *formula in positive normal form. Let φ_w be the formula obtained by weakening all operators (replacing $\mathsf{X}!$ with X, U with W, and $b!$ with b for a boolean expression b). Let φ_s be the formula obtained by strengthening all operators. Then,*

- $[\![\varphi_w]\!] = weak(\varphi)$
- $[\![\varphi_s]\!] = strong(\varphi)$.

It follows that if an ELTL formula φ is syntactically weak, then it is semantically weak. And similarly if φ is syntactically strong, then φ is semantically strong.

The ELTL operators $\mathsf{X}!, \mathsf{X}$ and U, W are not only strong/weak versions of one another, they are also duals of one another. That is, $\mathsf{X}! \; \varphi \equiv \neg(\mathsf{X} \; \neg\varphi)$ and $\varphi \; \mathsf{U} \; \psi \equiv \neg[\neg\psi \; \mathsf{W} \; (\neg\varphi \wedge \neg\psi)]$. The following proposition shows that this result generalizes to any weak/strong components of a formula.

PROPOSITION 4.10. *Let φ be an* ELTL *formula. Then,*

- $weak(\neg\varphi) = (strong(\varphi))^c$
- $strong(\neg\varphi) = (weak(\varphi))^c$

4.2 The Relation to Classification of Safety Formulas

The usual definitions of safety and liveness (e.g., [12]) are typically used with an alphabet such that some LTL formulas describe the empty property. Previously we have shown that in our augmented alphabet, $\top^\omega$ satisfies every formula.

As a result, some formulas that are safety in the unaugmented alphabet are no longer safety in the new alphabet. For instance, the formula $p \mathsf{U} \mathit{false}$ is a safety formula for the alphabet $\Sigma = 2^P$, but is not a safety formula for the alphabet $\widehat{\Sigma} = \Sigma \cup \{\top, \bot\}$. In this section, we examine the relation of our work to the classification of safety formulas proposed in [11], and we will use the term *safety formula* to denote formulas that are safety under the original, unaugmented alphabet Σ.

It follows from Propositions 4.4 and 3.7 that $\mathit{weak}(\varphi)$ is both prefix closed and limit closed and $\mathit{strong}(\varphi)$ is both extension closed and has finite witnesses. The finite witnesses quality of $\mathit{strong}(\varphi)$ can be understood in terms of the definitive prefix of [6] and the informative bad prefixes of [11]. The definitive prefix of a word w with respect to a formula φ is the shortest finite prefix $u \preceq w$ such that either u and all its extensions satisfy φ or u and all its extensions fail to satisfy φ. Such a finite prefix need not exist. The minimal finite witness of a word $w \in \mathit{strong}(\varphi)$ is the definitive prefix of w with respect to φ.

The concept of an informative prefix is defined syntactically in [11]. In [6] we gave a semantic definition, which, as shown in [8], can be restated in terms of the alphabet $\widehat{\Sigma}$ as follows:

Definition 4.11 (Informative Bad Prefix). *Let v be a finite word and φ an* ELTL *formula. Then, v is an informative bad prefix for φ iff $v\bot^\omega \models \neg\varphi$.*

It follows that the finite words in $\mathit{strong}(\neg\varphi)$ are exactly the informative bad prefixes for φ. It is shown in [6] that the definitive prefix of w with respect to φ (if it exists) is the shortest informative bad prefix for either φ or $\neg\varphi$.

The relation to informative prefixes is interesting since it helps us show that semantically weak formulas are not only safety formulas, but are also of a "good kind". Kupferman and Vardi [11] classify safety formulas according to whether all, some, or none of their bad prefixes are informative, and name them intentionally safe, accidentally safe and pathologically safe, respectively. It is shown in [11] that *non-pathologically* safe formulas (the set of formulas that are intentionally or accidentally safe) are easier to verify than pathologically safe formulas and that a syntactically safe formula is non-pathologically safe. The proposition below strengthens this result by showing that the same and the converse hold for semantically weak formulas.

Proposition 4.12. *Let φ be an* ELTL *formula.*

1. *φ is semantically weak iff φ is non-pathologically safe.*
2. *φ is semantically strong iff $\neg\varphi$ is non-pathologically safe.*

Thus the notion of weakness captures semantically exactly the set of "good" safety properties − that is, those that are computationally easy to verify.

5. AN APPLICATION

One of our motivations was to provide a characterization of weakness that would allow us to check whether a proposed weak version of some new temporal operator is related to its strong version in a way that maintains the internal consistency of the logic. We illustrate the use of our characterization on the logic RELTL, which adds weak and strong regular expressions to ELTL. We will define the semantics of weak and strong regular expressions directly, then show that the direct definitions achieve our goal.

We first define regular expressions (REs) and the syntax of RELTL.

Definition 5.1 (Regular Expressions (RE s)). *Let b be a boolean expression. The set of* regular expressions (REs) *is recursively defined as follows:*

$$r ::= b \mid r{\cdot}r \mid r \cup r \mid r^*$$

Definition 5.2 (RELTL Formulas). *Let r be a regular expression. The set of* RELTL formulas *is recursively defined as follows:*

$$\varphi ::= r! \mid r \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid X! \varphi \mid X \varphi \mid \varphi \mathsf{U} \varphi \mid \varphi \mathsf{W} \varphi$$

Note that weak and strong boolean expressions are included in the logic by means of weak and strong regular expressions.

We first define the semantics of regular expressions, then those of RELTL formulas. The semantics of regular expressions come in plain and weak versions, and are given below. The plain version $\mathcal{L}(\cdot)$ is the obvious extension of the usual semantics of regular expressions (e.g. as in [9]) to a syntax in which the atoms are boolean expressions. The weak version $\mathcal{L}_{weak}(\cdot)$ is reminiscent of Harel and Sherman's $\mathit{loop}(\cdot)$ [7]. Intuitively, a word v that is not in the language of a regular expression r is in the weak language of r iff v does not "contradict" r, and either v is finite but "too short" to be in the language of r, or v is infinite and "stuck in a starred sub-expression" of r.

In the following, b denotes a boolean expression and r, r_1, r_2 denote regular expressions.

Definition 5.3 (The Language of REs). *The* language of a regular expression *is recursively defined as follows:*

1. $\mathcal{L}(b) = \{\ell \in \widehat{\Sigma} \mid \ell \Vdash b\ \}$
2. $\mathcal{L}(r_1{\cdot}r_2) = \mathcal{L}(r_1)\mathcal{L}(r_2)$
3. $\mathcal{L}(r_1 \cup r_2) = \mathcal{L}(r_1) \cup \mathcal{L}(r_2)$
4. $\mathcal{L}(r^*) = \mathcal{L}(r)^*$

Definition 5.4 (The Weak Language of REs). *The* weak language of a regular expression *is recursively defined as follows:*

1. $\mathcal{L}_{weak}(b) = \{\epsilon\} \cup \mathcal{L}(b)$
2. $\mathcal{L}_{weak}(r_1{\cdot}r_2) = \mathcal{L}_{weak}(r_1) \cup (\mathcal{L}(r_1)\mathcal{L}_{weak}(r_2))$
3. $\mathcal{L}_{weak}(r_1 \cup r_2) = \mathcal{L}_{weak}(r_1) \cup \mathcal{L}_{weak}(r_2)$
4. $\mathcal{L}_{weak}(r^*) = (\mathcal{L}(r)^*\mathcal{L}_{weak}(r)) \cup \mathcal{L}(r)^\omega$

In RELTL we have regular expressions instead of boolean expressions at the base of the inductive definition. In ELTL we defined that $\top$ satisfies every boolean expression and $\bot$ satisfies no boolean expression. The following proposition generalizes this for REs.

Proposition 5.5. *Let r be an* RE. *Then*

- $\exists k > 0$ *such that* $\top^k \in \mathcal{L}(r)$
- $\forall k > 0$, $\bot^k \notin \mathcal{L}(r)$

We are now ready to define the semantics of RELTL (over the extended alphabet $\widehat{\Sigma}$). The semantics of the operators common to ELTL are as given in Definition 4.1. The semantics of the new operators, $r!$ and r, are given as follows:

DEFINITION 5.6. *Let r be an* RE*, and v a word over $\widehat{\Sigma}$.*

- $v \models r! \iff \exists$ *non-empty* $u \preceq v : u \in \mathcal{L}(r)$
- $v \models r \iff$ *either* $v \models r!$ *or* $v \in \mathcal{L}_{weak}(r)$

Intuitively, r is a weak version of $r!$. As a sanity check, it is easy to see that $p^*q!$ is equivalent to $p \mathsf{U} q$, and that p^*q is equivalent to $p \mathsf{W} q$. The following proposition states the relation between r and $r!$ formally:

PROPOSITION 5.7. *Let r be an* RE*. Then,*

- $[\![r]\!] = weak(r!)$
- $[\![r!]\!] = strong(r)$

The proof of this proposition makes use of the following characterization of $\mathcal{L}_{weak}(\cdot)$: A word w is in $\mathcal{L}_{weak}(r)$ if and only if for every finite prefix $u \preceq w$, u has an extension (over the alphabet $\widehat{\Sigma}$) in $\mathcal{L}(r)$. The following proposition states this formally using the notion of closure given in Definition 3.3.

PROPOSITION 5.8. *Let r be an* RE*. Then*

$$\mathcal{L}_{weak}(r) = cl(\mathcal{L}(r))$$

The following proposition states that all the results of Section 4.1 for ELTL carry over into RELTL.

PROPOSITION 5.9. *If the definition of syntactically weak/strong formulas is extended to include the formulas r and $r!$ in the obvious way, and if the logic* ELTL *is replaced by the logic* RELTL *in the statement of Propositions 4.2, 4.4, 4.7, 4.9, 4.10, the propositions continue to hold.*

If we include a regular expression intersection operator, Proposition 5.5 does not hold: the RE $a \cap (a \cdot a)$, for example, is not satisfied by $\top^k$ for any k. As a result, Propositions 4.2, 5.7, 5.8 and 5.9 do not hold for such a logic. We believe this can be fixed by defining $\mathcal{L}(b)$ from Definition 5.3 as $\{\ell \in \widehat{\Sigma} \mid \ell \Vdash b\ \} \cup \top^+$, and that with this new definition all the propositions (in particular 5.5, 4.2, 5.7, 5.8 and 5.9) hold as well for the logic with an intersection operator on REs.

6. RELATED WORK

At the level of automata, Alpern and Schneider's safety component can be achieved by considering all states of a Büchi automaton to be accepting [2]. Vardi and Wolper's looping acceptance condition [18] does exactly this, and it is observed that an automaton with looping acceptance defines a safety property. However, the particular safety property that results from applying looping acceptance depends on the form of the automaton, not just on its original language, in the same way that our weak component depends on the form of the LTL formula, not just on the property (in the original alphabet $\Sigma = 2^P$) that it represents. For instance, consider the Büchi automaton consisting of a single (not final) state and no transitions. Obviously, its language is empty. Now add a single transition on the letter a. Since the single state is not final, we have not changed the language of the automaton, which remains empty. However, applying the looping acceptance condition of [18] to the original and the modified automata will result in different languages. Thus our weak component can be made equivalent to the looping acceptance condition for logics which associate a specific Büchi automaton with each temporal operator, that is, for logics whose semantics are presented in the form of automata construction. Our topological approach eliminates the need to specify the particular automaton associated with each operator, and thus allows the characterization of weakness to be applied to logics without regard to the form in which their semantics are presented, as we have done above for the denotational semantics of ELTL and RELTL.

By considering the augmented alphabet $\widehat{\Sigma}$, our weak component distinguishes between formulas such as $p \mathsf{U}$ *false* and *false*!, which are equivalent under the unaugmented alphabet Σ, while the safety component of [2] does not. Thus the difference between Alpern and Schneider's safety component and our weak component is reminiscent of the difference between the abort semantics and the reset semantics examined by Armoni et al in [3]. Their goal was a complexity analysis, whereas ours is a characterization of weakness.

The ideas of weakness and strength examined in this paper are very closely related to the ideas of weak and strong satisfaction that we first examined in [6]. In that paper, we were concerned with the problem of reasoning on truncated paths, that is, over paths that are finite, but not necessarily maximal. In this paper, we are concerned with providing a semantic characterization of weakness for the purposes of verifying the internal consistency of a logic with weak and strong operators.

In [12] Maier gives a characterization of intuitionistic safety and liveness properties for LTL interpreted over finite as well as infinite words. He works with prefix closed sets of non-empty words. His safety properties are the subsets of $\Sigma^+ \cup \Sigma^\omega$ that are closed in the Scott topology defined from the prefix partial order. The Scott topology coincides with our topology on Σ^∞, and so his characterization of safety for properties over Σ is obtained from ours by eliminating the empty word. Since his properties are all prefix closed, his notion of liveness is distinct from ours. He does not consider the relationship of weakness to safety, and his framework of prefix closed properties is not suited to discussion of the dual notion of strength.

In [14] Manolios and Trefler gave an elegant characterization of safety and liveness using lattices, which unifies previous results on safety and liveness, including the results for the linear time and the branching time frameworks and for ω-regular string and tree languages. Our Proposition 3.2 can be proved by showing that $cl(\cdot)$ from Definition 3.3 is a lattice closure on $\langle \Sigma^\infty, \cap, \cup \rangle$. Thus, our extension of Alpern and Schneider's characterization to finite as well as infinite words conforms to the uniform characterization.

The particular extension of LTL with regular expressions that we present in Section 5 is intended only to illustrate the utility of our characterization of weakness. Previous works [19, 18, 4, 5] have examined the complexity and expressivity issues of extending LTL with regular expressions in various manners.

7. CONCLUSIONS

We have been concerned with finding a semantic characterization that relates weak and strong linear temporal operators. We have shown that the weak version of a strong temporal operator is not the same as Alpern and Schneider's safety component. We have extended Alpern and Schneider's topological characterization to the space of finite as well as infinite words, and shown that the desired weak component can be obtained by considering an alphabet augmented with two special letters $\top$ and $\bot$ in a topology on the set of finite and infinite words.

The resulting framework establishes a convenient way to study relations between weak and strong operators, and syntactically and semantically weak/strong formulas. E.g., we have shown that semantically weak formulas characterize exactly the set of "good" safety properties, thus strengthening the result in [11].

We have shown the utility of our characterization on the new pair of operators r and $r!$.

ACKNOWLEDGEMENTS

We would like to thank an anonymous referee for insightful comments that greatly improved the quality of this paper. The second author would like to thank her advisor, Amir Pnueli, for insightful questions and inspiring discussions.

8. REFERENCES

[1] B. Alpern and F. B. Schneider. Defining liveness. *Information Processing Letters*, 21:181–185, 1985.

[2] B. Alpern and F. B. Schneider. Recognizing safety and liveness. *Distributed Computing*, 2(3):117–126, 1987.

[3] R. Armoni, D. Bustan, O. Kupferman, and M. Y. Vardi. Aborts vs resets in linear temporal logic. In *Proc. 9th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, 2003.

[4] 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 *Proc. 8th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, LNCS 2280, pages 296–211. Springer, 2002.

[5] D. Bustan, D. Fisman, and J. Havlicek. Automata construction for PSL. Technical Report MCS05-04, The Weizmann Institute of Science, May 2005.

[6] C. Eisner, D. Fisman, J. Havlicek, Y. Lustig, A. McIsaac, and D. Van Campenhout. Reasoning with temporal logic on truncated paths. In *The 15th International Conference on Computer Aided Verification (CAV)*, LNCS 2725, pages 27–40. Springer-Verlag, July 2003.

[7] D. Harel and R. Sherman. Looping vs. repeating in dynamic logic. *Information and Control* 55, pages 175–192, 1982.

[8] J. Havlicek, D. Fisman, and C. Eisner. Basic results on the semantics of Accellera PSL 1.1 foundation language. Technical Report 2004.02, Accellera, May 2004.

[9] J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.

[10] J. Kelley. *General Topology*. Van Nostrand Reinhold, 1955. Springer, 1975.

[11] O. Kupferman and M. Y. Vardi. Model checking of safety properties. In *Proc.* 11^{th} *International Conference on Computer Aided Verification (CAV)*, LNCS 1633, pages 172–183. Springer-Verlag, 1999.

[12] P. Maier. Intuitionistic LTL and a new characterization of safety and liveness. In J. Marcinkowski and A. Tarlecki, editors, *Computer Science Logic, 18th International Workshop, CSL 2004, 13th Annual Conference of the EACSL; Karpacz, Poland, September 20-24, 2004; Proceedings*, volume 3210 of *Lecture Notes in Computer Science*, pages 295–309. Springer, 2004.

[13] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer-Verlag, New York, 1992.

[14] P. Manolios and R. Trefler. A lattice-theoretic characterization of safety and liveness. In *PODC '03: Proceedings of the Twenty-Second Annual Symposium on Principles of Distributed Computing*, pages 325–333. ACM Press, 2003.

[15] A. Pnueli. A temporal logic of concurrent programs. *Theoretical Computer Science*, 13:45–60, 1981.

[16] M. Rem. A personal perspective of Alpern-Schneider characterization of safety and liveness. In W. H. J. Feijen, editor, *Beauty is our business: a birthday salute to Edsger W. Dijkstra*, pages 365–372. Springer-Verlag, 1990.

[17] A. P. Sistla. Safety, liveness and fairness in temporal logic. *Formal Asp. Comput.*, 6(5):495–512, 1994.

[18] M. Y. Vardi and P. Wolper. Reasoning about infinite computations. *Information and Computation*, 115(1):1–37, 15 1994.

[19] P. Wolper. Temporal logic can be more expressive. *Information and Control*, 56(1/2):72–99, 1983.