Safety and Liveness in Branching Time

Panagiotis Manolios
Computer Sciences Department
University of Texas, Austin, TX, 78712, USA
pete@cs.utexas.edu

Richard Trefler
AT&T Labs-Research
180 Park Ave, P.O. Box 971, Room A013,
Florham Park, NJ, 07932, USA
trefler@research.att.com

Abstract

We extend the Alpern and Schneider linear time characterization of safety and liveness properties to branching time, where properties are sets of trees. We define two closure operators that give rise to the following four extremal types of properties: universally safe, existentially safe, universally live, and existentially live. The distinction between universal and existential properties captures the difference between the CTL path quantifiers A *(for all paths) and* E *(there is a path). We show that every branching time property is the intersection of an existentially safe property and an existentially live property, a universally safe property and a universally live property, and an existentially safe property and a universally live property. We also examine how our closure operators behave on linear time properties.*

We then focus on sets of finitely branching trees and show that our closure operators agree on linear time safety properties. Furthermore, if a set of trees is given implicitly as a Rabin tree automaton, $\mathcal{B}$*, we show that it is possible to compute the Rabin automata corresponding to the closures of the language of* $\mathcal{B}$*. This allows us to effectively compute* $\mathcal{B}_{safe}$ *and* $\mathcal{B}_{live}$ *such that the language of* $\mathcal{B}$ *is the intersection of the languages of* $\mathcal{B}_{safe}$ *and* $\mathcal{B}_{live}$*. As above,* $\mathcal{B}_{safe}$ *and* $\mathcal{B}_{live}$ *can be chosen so that their languages are existentially safe and existentially live, universally safe and universally live, or existentially safe and universally live.*

1 Introduction

Pnueli and Harel introduced the concept of a *reactive system*, a system whose behavior is characterized by nontermination and on-going interaction with an environment over which the system has little control [14]. Many safety critical systems, such as on-board controllers and network protocols, can be modeled as reactive systems and, therefore, the problem of specifying and verifying the correct behavior of reactive systems has become a very active area of research. Linear time properties of reactive systems have been grouped into three categories by Lamport [19]: safety properties, liveness properties, and properties which are neither. Informally, safety properties assert that nothing bad ever happens while liveness properties assert that something good happens eventually. This distinction plays an important role in the analysis of reactive systems since the proof methods employed to check safety properties differ from those used to check liveness properties. For example, proofs of liveness properties frequently require the construction of well-founded relations while safety properties are usually proven by induction on the transition relation. Furthermore, liveness properties often cannot be handled by the automatic proof techniques available for safety properties, *e.g.*, in some infinite state systems it is possible to automatically determine if a safety property can be violated, whereas the existence of a fair computation cannot be determined automatically [1].

In the linear time framework, where properties and the semantics of programs are sets of infinite strings, the distinction between safety and liveness is well understood. Alpern and Schneider [2] give a topological characterization in which safety properties are closed sets and liveness properties are dense sets. They also show that every linear time property can be given as the conjunction of a liveness property and a safety property. These results are well know and now appear in introductory textbooks on distributed systems. The topological characterization has been extended by various researchers, *e.g.*, Gumm has stated the notions of safety and liveness in the more abstract setting of Boolean algebras [13].

In the branching time framework —which includes process algebra and logics such as CTL [7] (which is used by many model checkers and is of great practical importance), CTL* [10], and the μ-calculus [21, 17, 9]— properties and the semantics of programs are sets of infinite trees. While there has been some work on characterizing safety and liveness for the branching time framework [6, 18], we present the first characterization that distinguishes between the CTL path quantifiers A and E, an essential distinction. In addition, we allow infinitely branching trees; such trees are closely related to considerations of fairness [5, 12, 4] and are useful for modeling input and programs with statements such as $x :=?$ (*i.e.*, non-deterministically assign a number to variable x). We define two closure operators which satisfy the conditions of Gumm [13]. Interestingly, we show that one of the operators defines a topology and the other does not. The closures give rise to four extremal types of properties: *universally safe*, *universally live*, *existentially safe*, and *existentially live*. Universally safe properties are those that correspond to linear time safety properties over all computations while existentially safe properties are those which guarantee at least one safe computation. In a similar manner, universally live and existentially live properties distinguish between linear time liveness properties over all and over some computations. For example, the CTL properties $\mathsf{AG}P$ —along every computation all states satisfy P— is a universally safe property, while $\mathsf{EG}P$ —there is a computation along which all states satisfy P— is an existentially safe property.

The paper is organized as follows: in the next section the basic notations and some preliminaries are given. Section 3 contains a review of the linear time results as well as the definitions of prefixes of trees, our closure operators, and safety and liveness in branching time. Section 3 also includes the results regarding the decomposition of properties into the extremal properties as well as some examples taken from Rem [22]. In Section 4 we consider finitely branching trees and show that for any linear time safety property h, $\mathsf{A}h$ and $\mathsf{E}h$ are both universally safe and existentially safe. In addition, for any linear time liveness property h, $\mathsf{A}h$ is universally live and $\mathsf{E}h$ is both universally live and existentially live. We further specialize our results to properties expressible as Rabin tree automata and show that if a set of trees is given implicitly as a Rabin tree automaton, $\mathcal{B}$, it is possible to effectively compute $\mathcal{B}_{safe}$ and $\mathcal{B}_{live}$ such that the language of $\mathcal{B}$ is the intersection of the languages of $\mathcal{B}_{safe}$ and $\mathcal{B}_{live}$, where $\mathcal{B}_{safe}$ and $\mathcal{B}_{live}$ can be chosen so that their languages are existentially safe and existentially live, universally safe and universally live, or existentially safe and universally live. Finally, Section 5 contains a brief conclusion and comparison with other work.

2 Preliminaries

$\mathbb{N}$ and ω both denote the natural numbers, *i.e.*, $\{0, 1, \ldots\}$. $[i..j]$ denotes the set $\{k \in \mathbb{N} \,:\, i \leq k \leq j\}$; $Dom.f$ denotes the domain of function f. Function application is sometimes denoted by an infix dot "." and is right associative. $\langle Qx : r : b\rangle$ denotes a quantified expression, where Q is the quantifier, x the bound variable, r the range of x ($\mathbf{true}$ if omitted), and b the body. $\mathcal{P}(S)$ denotes the powerset of S. For a relation R, we write $R|_S$ for R left-restricted to the set S, *i.e.*, $R|_S = \{\langle a, b\rangle : (\langle a, b\rangle \in R) \;\wedge\; (a \in S)\}$. S^* denotes the set of finite sequences over S; S^ω denotes the set of infinite sequences (functions from ω) over S; $S^\infty = S^* \cup S^\omega$. Suppose $s, t \in S^\infty$, $\#s$ denotes the length of s or, equivalently, the cardinality of $Dom.s$; s is a prefix of t ($s \preceq t$) iff $Dom.s \subseteq Dom.t$ and for all $i \in Dom.s$, $s.i = t.i$; s is a proper prefix of t ($s \prec t$) iff $s \preceq t$ and $s \neq t$. A set $U \subseteq S^\infty$ is prefix-closed iff for all $u \in U$ and for all $t \preceq u$, $t \in U$.

From highest to lowest binding power, we have: parentheses, function application, binary relations (*e.g.*, sBw), equality ($=$) and membership ($\in$), conjunction ($\wedge$) and disjunction ($\vee$), implication ($\Rightarrow$), and finally, binary equivalence ($\equiv$). Spacing is used to reinforce binding: more space indicates lower binding.

Throughout this paper Σ denotes a fixed *alphabet*, a nonempty set of symbols. An unlabeled tree is a prefix-closed subset of $\mathbb{N}^*$. A tree w is a pair $\langle \mathsf{W}, \mathrm{w}\rangle$ where W is an unlabeled tree and $\mathrm{w} : \mathsf{W} \to \Sigma$. A tree $\langle \mathsf{W}, \mathrm{w}\rangle$ is *total* if $\mathsf{W} \neq \emptyset$ and for all $\sigma \in \mathsf{W}$, there exists $\rho \in \mathsf{W}$ such that $\sigma \prec \rho$. A tree $\langle \mathsf{W}, \mathrm{w}\rangle$ is *finite-depth* if there exists $n \in \mathbb{N}$ such that for all $\sigma \in \mathsf{W}$, $\#\sigma \leq n$. By A^{tot}, A^{nt}, and A^f we denote the set of total, non-total, and finite-depth trees, respectively. The set of trees is denoted by A^{all}; note $A^{all} = A^{tot} \cup A^{nt}$ and $A^f \subset A^{nt}$. Let $t = \langle \mathsf{W}, \mathrm{w}\rangle$ be a tree. A $p \subseteq \mathsf{W}$ is a path in t iff p is a totally ordered (by $\preceq$), prefix-closed subset of W. Given a tree $\langle \mathsf{W}, \mathrm{w}\rangle$ and a node $\sigma \in \mathsf{W}$ we define the path $\overline{\sigma} = \{\sigma' \in \mathsf{W} : \sigma' \preceq \sigma\}$. We extend w to paths: given path $p = p_0 p_1 \cdots$, $\mathrm{w}(p) = (\mathrm{w}.p_0)(\mathrm{w}.p_1)\cdots$.

We briefly describe CTL, CTL*, and LTL [20] formulae (see [8] for complete details). LTL formula are formed from propositions, boolean connectives and the temporal operators X (next time), F (eventually), G (always) and U (until). LTL formulae define sets of infinite strings of (sets of) propositions. CTL* adds the universal and existential branching operators A and E to the LTL syntax. CTL is formed similarly with the restriction that each LTL temporal operator appear paired with its own path quantifier. CTL* and CTL formulae define sets of infinite depth trees labeled with (sets of) propositions.

3 Safety and Liveness

For the linear time framework, Alpern and Schneider define a closure operator on Σ^ω and show that it defines a topology [2]. Their closure operator, $lcl : \mathcal{P}(\Sigma^\omega) \to \mathcal{P}(\Sigma^\omega)$, is defined as follows: $lcl.T = \{t \in \Sigma^\omega : \langle \forall x : x \prec t : \langle \exists t' \in T :: x \preceq t'\rangle\rangle\}$. Properties are subsets of Σ^ω. Safety properties are defined to be the closed sets induced by lcl and liveness properties are defined to be the dense sets. It is shown that any property P is the intersection of $lcl.P$, a safety property, and $P \cup \neg(lcl.P)$, a liveness property. Gumm defines safety and liveness in the more abstract setting of Boolean algebras [13]. Given B_1 and B_2, two $\vee$-complete Boolean algebras, and $\varphi : B_1 \to B_2$, a $\vee$-preserving map, the *closure* $\overline{a}$ of $a \in B_1$ is defined as $\vee\{x \in B_1 : \varphi.a = \varphi.x\}$. For element $e \in B_1$, e is a *safety element* iff $\overline{e} = e$ and e is a *liveness element* iff $\overline{e} = 1$ (1 is the unit (top) element of B_1). It is proved that every element of B_1 is the conjunction of a safety element with a liveness element. We obtain the Alpern Schneider result by setting B_1 to $\langle \mathcal{P}(\Sigma^\omega), \Sigma^\omega, \emptyset, \cup, \cap, \neg \rangle$, B_2 to $\langle \mathcal{P}(\Sigma^*), \Sigma^*, \emptyset, \cup, \cap, \neg \rangle$, and $\varphi.T = \{x \in \Sigma^* : \langle \exists a \in T :: x \preceq a \rangle\}$.

To define safety and liveness properties for branching time, we start by defining what it means to concatenate trees and use this notion to define what it means for one tree to be a prefix of another. We then define two prefix operators corresponding to φ above. The closure, safety elements, and liveness elements are defined as above. We then explore the consequences and show that our characterization captures the intuitive notions of safety and liveness in the branching time framework.

3.1 A Partial Order for Trees

Given trees w and x, we define a preliminary notion of tree concatenation, denoted $w \cdot x$.

Definition 1 *Let* $w = \langle \mathsf{W}, \mathrm{w} \rangle$ *and* $x = \langle \mathsf{X}, \mathrm{x} \rangle$ *be trees.* $w \cdot x = \langle \mathsf{W} \cup \mathsf{X}, \mathrm{w} \cup (\mathrm{x}|_{\mathsf{X} \setminus \mathsf{W}}) \rangle$.

Note that $w \cdot x$ is a tree and that this notion of concatenation amounts to superimposing x on w. Unfortunately, the above notion of concatenation turns out not to be what we need. The problem is that it allows us to extend w at non-leaf nodes. Below, we define what it means to be a leaf and then introduce the notion of concatenation we require, where w concatenated with x is denoted by wx.

Definition 2 *Let* $w = \langle \mathsf{W}, \mathrm{w} \rangle$ *be a tree.* $\mathit{leaf}(z, w) \equiv z \in \mathsf{W} \;\wedge\; \neg\langle \exists y \in \mathsf{W} :: z \prec y \rangle$.

Definition 3 *Let* $w = \langle \mathsf{W}, \mathrm{w} \rangle$ *and* $x = \langle \mathsf{X}, \mathrm{x} \rangle$ *be trees. Let* $\mathsf{X}' = \{y \in \mathsf{X} : y \in \mathsf{W} \;\vee\; \langle \exists z : \mathit{leaf}(z, w) : z \prec y \rangle\}$. *Let* $x' = \langle \mathsf{X}', \mathrm{x}|_{\mathsf{X}'} \rangle$. $wx = w \cdot x'$.

Note that wx is a tree; the proof amounts to showing that x' is a tree. We now define what it means for one tree to be a prefix of another.

Definition 4 $x \sqsubseteq y \quad \equiv \quad \langle \exists z :: xz = y \rangle$

Notice that when restricted to sequences, $\sqsubseteq$ agrees with the usual notion of prefix.

Lemma 1 $x \sqsubseteq y \quad \Rightarrow \quad wx \sqsubseteq wy$

Lemma 2 $\sqsubseteq$ *is a partial order.*

Note that, due to space restrictions, some of the proofs are omitted.

Elements of $\mathcal{P}(A^{tot})$ are the branching time properties. Note that $\langle \mathcal{P}(A^{tot}), A^{tot}, \emptyset, \cup, \cap, \neg \rangle$ and $\langle \mathcal{P}(A^{all}), A^{all}, \emptyset, \cup, \cap, \neg \rangle$ are Boolean algebras.

3.2 Prefixes and Closures

We define the non-total and finite-depth prefix operators, *npref* and *fpref*, functions from $\mathcal{P}(A^{tot})$ to $\mathcal{P}(A^{all})$, as follows.

Definition 5 $\mathit{npref}.p = \{x \in A^{nt} : \langle \exists y \in p :: x \sqsubseteq y \rangle\}$

Definition 6 $\mathit{fpref}.p = \{x \in A^{f} : \langle \exists y \in p :: x \sqsubseteq y \rangle\}$

The prefix operators correspond to φ, the $\vee$-preserving map described above. The induced closure functions, from $\mathcal{P}(A^{tot})$ to $\mathcal{P}(A^{tot})$, are:

Definition 7 $\mathit{ncl}.p \;=\; \cup\{q \subseteq A^{tot} : \mathit{npref}.q = \mathit{npref}.p\}$

Definition 8 $\mathit{fcl}.p \;=\; \cup\{q \subseteq A^{tot} : \mathit{fpref}.q = \mathit{fpref}.p\}$

The closure functions have the following properties.

Lemma 3 $\mathit{ncl}.p = \{\, y \in A^{tot} \;:\; \langle \forall x \in A^{nt} : x \sqsubseteq y : x \in \mathit{npref}.p \rangle \,\}$

Lemma 4 $\mathit{fcl}.p = \{\, y \in A^{tot} \;:\; \langle \forall x \in A^{f} : x \sqsubseteq y : x \in \mathit{fpref}.p \rangle \,\}$

After expanding the definitions of the prefix operators in the above two lemmas, notice that the characterizations of ncl and fcl are very similar to the definition of lcl.

Lemma 5 $p \subseteq \mathit{ncl}.p$ *and* $p \subseteq \mathit{fcl}.p$

Lemma 6 $\mathit{ncl}.\mathit{ncl}.p = \mathit{ncl}.p$ *and* $\mathit{fcl}.\mathit{fcl}.p = \mathit{fcl}.p$

3.3 Safety

We say that a property is a *safety property* if the property is equal to its closure. Since we have two types of closures, we have two types of safety properties: existentially safe (ES) and universally safe (US). The intuition is that the existentially safe properties guarantee at least one computation along which nothing bad happens. The universally safe properties guarantee that nothing bad happens during any computation. This type of distinction is made with the CTL operators E, which existentially quantifies over paths, and A, which universally quantifies over paths. In the sequel, we implicitly extend functions on sets to functions on formulae, be applying the functions to the sets of trees or strings which the formulae define.

Definition 9 *(Existentially Safe)* $p \in ES \quad \equiv \quad p = ncl.p$

Definition 10 *(Universally Safe)* $p \in US \quad \equiv \quad p = fcl.p$

Lemma 7 $ncl.p \subseteq fcl.p$

Proof The domain of the quantifier in the definition of fcl, A^f, is a subset of A^{nt}, the domain of the quantifier in the definition of ncl. $\Box$

Lemma 8 $p \in US \quad \Rightarrow \quad p \in ES$

Proof $p \in US \quad \equiv \quad p = fcl.p$, but $p \subseteq ncl.p$ and $ncl.p \subseteq fcl.p$, so $p = ncl.p$, *i.e.*, $p \in ES$. $\Box$

Lemma 9 $ncl.fcl.p = fcl.p$

Proof $fcl.p \subseteq ncl.fcl.p \subseteq fcl.fcl.p = fcl.p$ $\Box$

We note that $fcl.ncl.p = ncl.p$ does not hold, for example, when $p = \mathsf{EG}a$ (there exists a path such that every node in the path is labeled by an a), we will see that $ncl.p = p$, but $fcl.p \neq p$.

Lemma 10 $p \subseteq q \quad \Rightarrow \quad ncl.p \subseteq ncl.q \ \wedge \ fcl.p \subseteq fcl.q$

Recall that an operator $c : \mathcal{P}(X) \rightarrow \mathcal{P}(X)$ defines a topology on X with closed sets $\{a \subseteq X : c.a = a\}$ iff the following four conditions hold [15]:

- $c.\emptyset = \emptyset$
- $a \subseteq c.a$
- $c.c.a = c.a$
- $c(a \cup b) \ = \ c.a \cup c.b$

Therefore, the following lemma shows that fcl defines a topology.

Lemma 11 $fcl.p \cup fcl.q \quad = \quad fcl.(p \cup q)$

Since $ncl.(p \cup q) \ \subseteq \ ncl.p \cup ncl.q$ is not a theorem, ncl does not define a topology. This does not cause us any technical difficulties, but it is interesting because lcl, the closure operator in the linear time case, does define a topology. We have the following, however.

Lemma 12 $ncl.p \cup ncl.q \quad \subseteq \quad ncl.(p \cup q)$

3.4 Liveness

We will now define what it means for a property to be a liveness property. A liveness property is one whose closure is the set of all trees. Given our two notions of closure, we have two notions of liveness.

Definition 11 *(Existentially Live)* $p \in EL \quad \equiv \quad ncl.p = A^{tot}$

Definition 12 *(Universally Live)* $p \in UL \quad \equiv \quad fcl.p = A^{tot}$

Lemma 13 $p \in EL \quad \Rightarrow \quad p \in UL$

Proof $p \in EL \quad \equiv \quad ncl.p = A^{tot}$, but since $ncl.p \subseteq fcl.p$, $fcl.p = A^{tot}$, *i.e.*, $p \in UL$ $\Box$

Lemma 14 $US \cap UL \quad = \quad \{\ A^{tot}\ \}$

Proof $p \in (US \cap UL) \quad \equiv \quad p = fcl.p \ \wedge \ fcl.p = A^{tot} \quad \equiv \quad p = A^{tot}$ $\Box$

Lemma 15 $ES \cap EL \quad = \quad \{\ A^{tot}\ \}$

Lemma 16 $US \cap EL \quad = \quad \{\ A^{tot}\ \}$

Proof $p \in (US \cap EL) \quad \equiv \quad p = fcl.p \ \wedge \ ncl.p = A^{tot} \quad \equiv \quad p = ncl.p \ \wedge \ ncl.p = A^{tot} \quad \equiv \quad p = A^{tot}$ $\Box$

Note that $ES \cap UL \quad = \quad \{\ A^{tot}\ \}$ does not hold, *e.g.*, ($\mathsf{AF}a$ means along all futures a eventually holds) let $p = ncl.\mathsf{AF}a$, then $p = ncl.p$ and $fcl.p = A^{tot}$, but $p \neq A^{tot}$.

On account of Lemma 10, we have the following two properties:

Lemma 17 $p \subseteq q \ \wedge \ p \in EL \quad \Rightarrow \quad q \in EL$

Lemma 18 $p \subseteq q \ \wedge \ p \in UL \quad \Rightarrow \quad q \in UL$

Lemma 19 $(p \cup \neg ncl.p) \in EL$

Proof $ncl.(p \cup \neg ncl.p) \quad \supseteq \quad ncl.p \cup ncl.(\neg ncl.p) \quad \supseteq \quad ncl.p \cup \neg ncl.p \ = \ A^{tot}$ $\Box$

Lemma 20 $(p \cup \neg fcl.p) \in UL$

Theorem 1 *Every property is the intersection of: (1) an existentially safe and an existentially live property, (2) a universally safe and a universally live property, and (3) an existentially safe and a universally live property.*

Proof (1). $ncl.p \in ES$ and $(p \cup \neg ncl.p) \in EL$; their intersection yields $ncl.p \cap (p \cup \neg ncl.p) = ncl.p \cap p = p$. (2). $fcl.p \in US$ and $(p \cup \neg fcl.p) \in UL$; their intersection yields $fcl.p \cap (p \cup \neg fcl.p) = fcl.p \cap p = p$. (3). $ncl.p \in ES$ and $(p \cup \neg fcl.p) \in UL$; their intersection yields $ncl.p \cap (p \cup \neg fcl.p) = (ncl.p \cap p) \cup (ncl.p \cap \neg fcl.p) = p \cup (ncl.p \cap \neg fcl.p) = p$, since $ncl.p \subseteq fcl.p$. $\Box$

The next theorem shows that certain properties do not correspond to the intersection of a universally safe and an existentially live property.

Theorem 2 *[16] Let Q be a subset of A^{tot} such that $fcl.Q = A^{tot}$ and $ncl.Q \neq A^{tot}$. There do not exists sets $S, L \subseteq A^{tot}$ such that $fcl.S = S$, $ncl.L = A^{tot}$, and $S \cap L = Q$.*

Proof Suppose $S \cap L = Q$, $fcl.S = S$, and $ncl.L = A^{tot}$, then $Q \subseteq S$, which gives $A^{tot} = fcl.Q \subseteq fcl.S$ and hence $S = A^{tot}$. Since $Q = S \cap L$, then $L = Q$ which implies that $ncl.Q = A^{tot}$. $\Box$

We will see shortly that the set of trees satisfying the CTL formula $\mathsf{AF}p$ satisfies the preconditions on the previous theorem.

Our decomposition of a property into a safety property and a liveness property is extreme in the following sense.

Lemma 21 *If $(q \in ES \ \vee \ q \in US)$ and $p = (q \cap r)$, then $ncl.p \subseteq q$ and $r \subseteq (p \cup \neg ncl.p)$.*

Proof For the first part note that $p = (q \cap r) \Rightarrow p \subseteq q \Rightarrow ncl.p \subseteq ncl.q \ \wedge \ ncl.q \subseteq fcl.q \Rightarrow ncl.p \subseteq q$, as by assumption $q = ncl.q \ \vee \ q = fcl.q$.

For the other part, we have $(q \cap r) = p$, which by $ncl.p \subseteq q$ (the first part) implies $(ncl.p \cap r) \subseteq p$, which, if we union $\neg ncl.p$ to both sides and simplify the left, implies $(\neg ncl.p \cup r) \subseteq (p \cup \neg ncl.p)$, which implies $r \subseteq (p \cup \neg ncl.p)$ $\Box$

Theorem 3 *Let h be an LTL formula which is a safety property, then $fcl.\mathsf{A}h = ncl.\mathsf{A}h = \mathsf{A}h$ and $ncl.\mathsf{E}h = \mathsf{E}h$.*

Proof Suppose $t \in A^{tot}$, $t = \langle T, \tau \rangle$, $t \in fcl.\mathsf{A}h$, and $t \notin \mathsf{A}h$. Then there is some path x in t such that $\tau(x) \notin h$. Since h is a safety property, $lcl.h = h$, this implies that for some $i \in \mathbb{N}$, $\tau(x_0 \cdots x_i)$ cannot be extended to a string in h. Hence for any $u \in A^f$ such that $u \sqsubseteq t$ and includes $x_0 \cdots x_i$, u cannot be extended into a tree v such that $v \in \mathsf{A}h$. Hence there is no such t and $fcl.\mathsf{A}h = \mathsf{A}h$. We also have $\mathsf{A}h \subseteq ncl.\mathsf{A}h \subseteq fcl.\mathsf{A}h = \mathsf{A}h$, so $ncl.\mathsf{A}h = \mathsf{A}h$.

Suppose $t \in A^{tot}$, $t = \langle T, \tau \rangle$, and $t \in ncl.\mathsf{E}h$ and $t \notin \mathsf{E}h$. Since $t \notin \mathsf{E}h$ then for no full path, y, in t is $\tau(y) \in h$. Let x be a path whose prefix $\tau(x_0 \cdots x_i)$ cannot be extended to a string in h. Let $u \in A^{nt}$ be the tree obtained from t by making x_i a leaf (*i.e.*, removing all its descendants). Then u cannot be extended into a tree v such that $v \in \mathsf{E}h$ and hence $t \notin ncl.\mathsf{E}h$. Therefore $ncl.\mathsf{E}h = \mathsf{E}h$. $\Box$

The following property shows that *fcl* is not an appropriate closure operator for existentially quantified safety properties of paths. That is, a safety property to which existential quantification is added is not necessarily closed under fcl.

Lemma 22 $fcl.\mathsf{EG}P \neq \mathsf{EG}P$

Proof Consider a total tree whose root has an infinite number of children, but every other node has exactly one child. Furthermore, the path through the first child is labeled by $a(\neg a)^\omega$. The path through the second child is labeled by $aa(\neg a)^\omega$ and so on. No path in the tree satisfies $\mathsf{G}a$, so the tree is not in $\mathsf{EG}a$, but any finite depth prefix of the tree can be extended to a tree in $\mathsf{EG}a$. $\Box$

Theorem 4 *Let h be an LTL formula which is a liveness property, then $fcl.\mathsf{A}h = A^{tot}$ and $ncl.\mathsf{E}h = fcl.\mathsf{E}h = A^{tot}$.*

Proof h is a liveness property implies that $lcl.h = \{\sigma \in \Sigma^\omega\}$.

Let $t \in A^{tot}$, $t = \langle T, \tau \rangle$, and $u \sqsubseteq t$ such that $u \in A^f$. Consider any full path x in u. $\tau(x)$ is a prefix of some $\sigma \in lcl.h$, hence, x can be extended to a path y such that $\tau(y) \in h$. Therefore $u \in fcl.\mathsf{A}h$. Hence $fcl.\mathsf{A}h = A^{tot}$.

Let $t \in A^{tot}$, $t = \langle T, \tau \rangle$, and $u \sqsubseteq t$, such that $u \in A^{nt}$. If u contains a path σ such that $\tau(\sigma) \in h$ then $t \in ncl.\mathsf{E}h$. Else, consider any full finite path x in u. As in the first proof above, x can be extended to an infinite path $y \in h$ and hence $t \in ncl.\mathsf{E}h$. Therefore $ncl.\mathsf{E}h = A^{tot}$. We also have $A^{tot} = ncl.\mathsf{E}h \subseteq fcl.\mathsf{E}h$, hence, $fcl.\mathsf{E}h = A^{tot}$. $\Box$

The following property shows that *ncl* is not an appropriate closure operator for universally quantified liveness properties of paths. That is, given a liveness property of paths, adding universal quantification and taking the ncl closure does not necessarily result in the set of all trees.

Lemma 23 $ncl.\mathsf{AF}P \neq A^{tot}$

3.5 Examples

We now take a moment to consider the ramifications of our approach by comparison with Martin Rem's [22] example properties, listed below. Rem's examples are formulated as predicates on t, an infinite (Σ) sequence.

$p0$: **false** (corresponds to $\emptyset$);
$p1$: the first symbol of t is a;
$p2$: the first symbol of t differs from a;
$p3$: the first symbol of t is a, and t contains a symbol that differs from a;
$p4$: the number of a's in t is finite;
$p5$: the number of a's in t is infinite;
$p6$: **true** (corresponds to Σ^ω).

If we are dealing with sequences, $p0$, $p1$, $p2$, and $p6$ are safety properties. The (linear) closure of $p3$ is $p1$, so $p3$ is not a safety property. The closures of $p4$ and $p5$ are both Σ^ω; so they are not safety properties, but they are liveness properties.

Note that if we restrict t to infinite sequences, then both fcl and ncl agree with lcl. In order to examine the above properties in a branching time framework, we will write them down in LTL [20] and CTL*. Note that in translating the above examples to properties over trees there is some ambiguity. In particular, we have translated $p4$ into both $q4a$ and $q4b$ and in fact neither of these translations captures the notion that there are only a finite number of a's in a tree but rather that there are a finite number of a's on a path (on all paths) in the tree.

$q0$:	**false**	**false** (corresponds to $\emptyset$);		
$q1$:	a	a;		
$q2$:	$\neg a$	$\neg a$;		
$q3a$:	$a \wedge \mathsf{F}\neg a$	$\mathsf{A}(a \wedge \mathsf{F}\neg a)$	$\equiv$	$a \wedge \mathsf{AF}\neg a$;
$q3b$:		$\mathsf{E}(a \wedge \mathsf{F}\neg a)$	$\equiv$	$a \wedge \mathsf{EF}\neg a$;
$q4a$:	$\mathsf{FG}\ \neg a$	$\mathsf{A}(\mathsf{FG}\ \neg a)$;		
$q4b$:		$\mathsf{E}(\mathsf{FG}\ \neg a)$;		
$q5a$:	$\mathsf{GF}\ a$	$\mathsf{A}(\mathsf{GF}\ a)$;		
$q5b$:		$\mathsf{E}(\mathsf{GF}\ a)$;		
$q6$:	**true**	**true** (corresponds to A^{tot}).		

Below we give an informal translation of the above CTL* sentences. $q1$ is true of any tree whose root is labeled with a; similarly for $q2$. $q3a$ is true of the trees whose root is labeled with a and along each path have a node labeled with $\neg a$. $q3b$ is true of the trees whose root is labeled with a and along some path have a node labeled with $\neg a$. $q4a$ is true of the trees where along each path, eventually all nodes are labeled with $\neg a$. $q4b$ is true of the trees where along some path, eventually all nodes are labeled with $\neg a$. $q5a$ is true of the trees where along each path, infinitely many nodes are labeled with a. $q5b$ is true of the trees where along some path, infinitely many nodes are labeled with a.

It is not difficult to show that $q0$, $q1$, $q2$, and $q6$ are universally safe (and hence existentially safe).

$fcl.q3a = q1$, as before, but $ncl.q3a \neq q1$ (consider a tree that has at least two paths such that along one of the paths a always holds; this tree is not in $ncl.q3a$). $ncl.q3a \neq q3a$ (trees can be sequences, so $\{y : y \in \Sigma^\omega\} \subseteq ncl.q3a$). $ncl.q3b = q1$ and $fcl.q3b = q1$.

$fcl.q4a = A^{tot}$, as before, but $ncl.q4a \neq A^{tot}$ (consider a tree that has at least two paths such that along one of the paths a always holds; this tree is not in $ncl.q4a$). $ncl.q4a \neq q4a$ (trees can be sequences, so $\{y : y \in \Sigma^\omega\} \subseteq ncl.q4a$). $ncl.q4b = A^{tot}$, so $fcl.q4b = A^{tot}$.

$fcl.q5a = A^{tot}$, as before, but $ncl.q5a \neq A^{tot}$ (consider a tree that has at least two paths such that along one of the paths $\neg a$ always holds; this tree is not in $ncl.q5a$). $ncl.q5a \neq q5a$ (trees can be sequences, so $\{y : y \in \Sigma^\omega\} \subseteq ncl.q5a$). $ncl.q5b = A^{tot}$, so $fcl.q5b = A^{tot}$.

4 Finite Branching Trees

In the previous sections we studied sets of trees that included infinitely branching trees. However, many systems do not have such trees and it is interesting to see what benefits are obtainable when considering only bounded branching structures.

Let $k \in \mathbb{N}$. A tree $\langle \mathsf{W}, \mathsf{w}\rangle$ is a k-branching tree iff for all $\sigma \in \mathsf{W}$ there exists exactly k unique elements of $\mathbb{N}$, $a_0, \ldots, a_{k-1}$, such that $\sigma a_0, \ldots, \sigma a_{k-1} \in \mathsf{W}$. In what follows we consider sets of trees which are k-branching. By $A^{k,tot}$ and $A^{k,f}$ we denote, respectively, the set of k-branching trees and the set of finite trees whose non-leaf nodes have exactly k successors. We carry over the definitions of ncl and fcl from the previous sections, restricted now to k-branching trees over finite alphabets. Below we show that ncl and fcl agree on linear time safety properties (recall that $ncl.p \subseteq fcl.p$).

Theorem 5 *Suppose h is a safety property over Σ^ω then $fcl.\mathsf{E}h = ncl.\mathsf{E}h = \mathsf{E}h$ and $fcl.\mathsf{A}h = ncl.\mathsf{A}h = \mathsf{A}h$.*

Proof We have that $\mathsf{E}h \subseteq fcl.\mathsf{E}h$. So suppose $t = \langle T, \tau\rangle \in fcl.\mathsf{E}h$, this means that for all $u \in A^{k,f}$, $u \sqsubseteq t$ implies there is a $t' \in A^{k,tot}$ such that $u \sqsubseteq t'$ and $t' \in \mathsf{E}h$.

We will show that t contains a path p such that $\tau(p) \models h$. Consider the tree $v = \langle V, \phi\rangle \in A^{all}$ defined as follows: $V = \{\sigma \in T : \langle \exists y \in \Sigma^\omega :: \tau(\overline{\sigma})y \models h\rangle\}$ and $\phi.\sigma = \tau.\sigma$. V has an infinite number of nodes as any prefix of t can be extended to a tree in $\mathsf{E}h$. V is also finitely branching, thus, by König's lemma, has an infinite path. For any such infinite path p, $\tau(p) \models h$ because h is a safety property. Since $v \sqsubseteq t$, p is a path in T, hence, $t \in \mathsf{E}h$.

The rest of the proof is along the lines of the proof of Theorem 3. $\square$

For linear time liveness properties, however, fcl and ncl do not agree, *e.g.*, $fcl.\mathsf{AF}P = A^{k,tot}$ whereas $ncl.\mathsf{AF}P \neq A^{k,tot}$. We do have the following.

Lemma 24 *Suppose h is a liveness property over Σ^ω then $fcl.\mathsf{E}h = ncl.\mathsf{E}h = A^{k,tot}$ and $fcl.\mathsf{A}h = A^{k,tot}$.*

The close relationship between properties of programs and automata has been well documented [24, 8]. In particular, given a finite state Büchi automaton, $\mathcal{B}$, over infinite strings (recall that Büchi automata recognize regular languages of ω-strings), it is possible to decompose $\mathcal{B}$ into automata $\mathcal{B}_S$ and $\mathcal{B}_L$ such that the set of strings accepted by $\mathcal{B}_S$ is a safety property and the set of strings accepted by $\mathcal{B}_L$ is a liveness property [3]. Furthermore, the set of strings accepted by $\mathcal{B}$ is equal to the intersection of the set of strings accepted by $\mathcal{B}_S$ and $\mathcal{B}_L$. We show that a similar result for Rabin tree automata is possible to achieve (recall that Rabin automata recognize regular languages of ω-trees). That is, we show that any set of trees recognizable by a Rabin tree automaton is decomposable into the intersection of: a universally safe set and a universally live set, an existentially safe set and an existentially live set, and an existentially safe set and a universally live set, all of which are Rabin tree automata definable.

A Rabin tree automaton $\mathcal{B} = (\Sigma, Q, q_0, \delta, \Phi)$ on k-ary infinite trees is defined as follows: Σ is a finite alphabet, Q is a finite set of states, $q_0 \in Q$ is the start state, $\delta : Q \times \Sigma \to \mathcal{P}(Q^k)$ is the transition relation, and Φ is the accepting condition.

Let $t = \langle \mathsf{W}, \mathrm{w}\rangle \in A^{k,tot}$. A run of $\mathcal{B}$ on t is a Q labeled tree $r = \langle \mathsf{W}, \rho\rangle \in A_Q^{k,tot}$ such that $\rho.\lambda = q_0$ and for all $\sigma \in \mathsf{W}$ and successors $\sigma a_0, \ldots, \sigma a_{k-1} \in \mathsf{W}$, $(\rho.\sigma a_0, \ldots, \rho.\sigma a_{k-1}) \in \delta(\rho.\sigma, \mathrm{w}.\sigma)$. Run r is accepting iff for all infinite paths p in W, $\rho(p) \models \Phi$. $\mathcal{L}(\mathcal{B}) = \{t \in A^{k,tot}$: there is an accepting run of $\mathcal{B}$ on $t\}$ is the language of $\mathcal{B}$.

The accepting condition, Φ, is given by specifying pairs of sets $(green_i, red_i) \in (\mathcal{P}(Q))^2$ for $i \in [0..m]$, for some m. Φ holds on a path if for some i, some green state is visited infinitely often and all red states are visited finitely often, *i.e.*, $\Phi = \vee_{i\in[1..m]}[(\vee_{g\in green_i}\mathsf{GF}g)\wedge(\wedge_{r\in red_i}\mathsf{FG}\neg r)]$.

For notational convenience, given a Rabin automaton $\mathcal{B} = (\Sigma, Q, q_0, \delta, \Phi)$ we will refer to $\mathcal{B}(q)$, $q \in Q$, as the automaton given by $(\Sigma, Q, q, \delta, \Phi)$. Given automaton $\mathcal{B} = (\Sigma, Q, q_0, \delta, \Phi)$ such that $\mathcal{L}.\mathcal{B} \neq \emptyset$, note that $\mathcal{L}.\mathcal{B} = \mathcal{L}.(\Sigma, Q', q_0, \delta', \Phi)$ where $Q' = \{q \in Q : \mathcal{L}(\mathcal{B}(q)) \neq \emptyset\}$ and δ' is δ restricted to Q'. We define the finite depth closure, *rfcl*, of an automaton as follows: if $\mathcal{L}.\mathcal{B} = \emptyset$, $rfcl.\mathcal{B} = \mathcal{B}$; otherwise, $rfcl.\mathcal{B} = (\Sigma, Q', q_0, \delta', \Phi')$ where $\Phi' = \vee_{q\in Q'}\mathsf{GF}q$ is a condition that holds along all paths and is generated from $\{(Q', \emptyset)\}$.

Lemma 25 $\mathcal{L}(rfcl.\mathcal{B}) = fcl.\mathcal{L}(\mathcal{B})$.

Proof If $\mathcal{L}.\mathcal{B} = \emptyset$, then $\mathcal{L}(rfcl.\mathcal{B}) = fcl(\mathcal{L}.\mathcal{B}) = \emptyset$, so we assume $\mathcal{L} \neq \emptyset$.

Suppose $t = \langle \mathsf{W}, \mathrm{w}\rangle \in \mathcal{L}(rfcl.\mathcal{B})$ then there is an accepting run $r = \langle \mathsf{W}, \rho\rangle$ of $rfcl.\mathcal{B}$ on t. Consider any $u = \langle \mathsf{U}, v\rangle \in A^{k,f}$ such that $u \sqsubseteq t$; $r' = \langle \mathsf{U}, \rho|_{\mathsf{U}}\rangle$ is a partial run of $rfcl.\mathcal{B}$ on u. By the construction of $rfcl.\mathcal{B}$, each leaf node $\sigma \in \mathsf{U}$ of r' is labeled by a node $\rho.\sigma \in Q'$. This means, however, that $\mathcal{L}.\mathcal{B}(\rho.\sigma) \neq \emptyset$ and therefore that each leaf node is extendible into some tree which is accepted by the automaton node labeling the leaf. This implies that for some tree $t' \in \mathcal{L}.\mathcal{B}$, $u \sqsubseteq t'$ and hence $t \in fcl(\mathcal{L}.\mathcal{B})$.

Suppose $t \in fcl(\mathcal{L}.\mathcal{B})$, then for all $t^i \sqsubseteq t$, where t^i denotes the subtree of t up to level i, there exists u_i such that $t^i \sqsubseteq u_i$ and $u_i \in \mathcal{L}.\mathcal{B}$, hence, there exists run r_i of $\mathcal{B}$ on u_i. We now define a run, r, of $rfcl.\mathcal{B}$ on t. r^i, the subtree of r up to level i, is defined recursively as follows. For the base case, r^0 labels the root by q_0 and $R_0 = \omega$. For the recursive case, choose r^{i+1} so that for infinitely many $j \in R_i, r_j$ labels t^{i+1} by r^{i+1} and $R_{i+1} = R_i \setminus \{j \in R_i : r_j^{i+1} \neq r^{i+1}\}$. Note that for all i, R_i is an infinite set such that for all $j \in R_i, r_j^i = r^i$. This is true for R_0 as all runs label the root q_0. Assuming it is true for R_i, then by definition, if R_{i+1} contains j, $r_j^{i+1} = r^{i+1}$. Since the number of possible labelings of t^{i+1} is finite, by the pigeon-hole principle, an infinite subset of R_i indexes runs that assign the same labeling to t^i. Since the acceptance condition for $rfcl.\mathcal{B}$ is trivially satisfied, we have show that r is a run of $rfcl.\mathcal{B}$ on t. $\Box$

The consequence of this is the following:

Theorem 6 *For any Rabin tree automaton, $\mathcal{B}$, there exist effectively derivable Rabin automata $\mathcal{B}_{safe}$ and $\mathcal{B}_{live}$ such that $\mathcal{L}.\mathcal{B} = \mathcal{L}.\mathcal{B}_{safe} \cap \mathcal{L}.\mathcal{B}_{live}$ and $\mathcal{L}.\mathcal{B}_{safe}$ is universally safe while $\mathcal{L}.\mathcal{B}_{live}$ is universally live.*

Proof Recall that non-emptiness of Rabin tree automata is decidable and Rabin automata are effectively closed under complementation and union [24]. Thus, $\mathcal{B}_{safe} = rfcl.\mathcal{B}$ and $\mathcal{B}_{live} = \mathcal{B} \cup (A^{k,tot} \setminus rfcl.\mathcal{B})$ can be effectively derived from $\mathcal{B}$. That $\mathcal{B}_{safe}$ is safe follows from the above lemma and $\mathcal{B}_{live}$ is live because for any property P, $P \cup (A^{k,tot} \setminus rfcl.P)$ is live. $\Box$

Similarly, it is possible to define the non-total closure of a Rabin automaton, which gives rise to the following theorem.

Theorem 7 *For any Rabin tree automaton, $\mathcal{B}$, there exist effectively derivable Rabin automata $\mathcal{B}_{safe}$ and $\mathcal{B}_{live}$ such that $\mathcal{L}.\mathcal{B} = \mathcal{L}.\mathcal{B}_{safe} \cap \mathcal{L}.\mathcal{B}_{live}$ and $\mathcal{L}.\mathcal{B}_{safe}$ is existentially safe (existentially safe) while $\mathcal{L}.\mathcal{B}_{live}$ is existentially live (universally live).*

5 Conclusion

We have given a computation-tree based semantic characterization of the intuitive notions of safety and liveness. Our characterizations are given in terms of the closures of sets of trees in a manner analogous to Gumm's [13] generalization of the work of Alpern and Schneider for linear time [2]. In fact, our results when restricted to sets of strings are identical since in that case $ncl.p = fcl.p = lcl.p$. Our approach and examples draw heavily on Rem's very readable presentation [22] of the Alpern and Schneider results.

Decomposing branching time properties into four extremal classes, *viz.*, universally safe, universally live, existentially safe, and existentially live, has allowed a characterization of safety and liveness properties which respects the branching time temporal operators A and E of CTL. In contrast, the work of Bouajjani *et al.* [6] is restricted to the regular trees[1] and does not distinguish between existentially and universally safe. They consider only a single closure operator and choosing either *fcl* or *ncl* as that operator results in either $\mathsf{EG}P$ not being a safety property or $\mathsf{AF}P$ not being a liveness property. In particular, it is possible to show that $\mathsf{EG}P$ is not definable by the class of *safety recognizers* (a restricted class of Rabin tree automata) —see the appendix for a proof— and therefore is not classified as a safety property as defined by Bouajjani *et al.* even under the restriction of finitely branching regular trees.

Possible directions for future work include defining subclasses of safety and liveness formulae and syntactically characterizing them, as has been done in the linear time framework by Sistla [23]. Another question is whether there are efficient model checking algorithms for branching time safety properties (see Kupferman and Vardi [18]).

Acknowledgments

We thank Nils Klarlund for many interesting discussions regarding this work. We also thank an anonymous referee for suggesting, among other things, the adverbs "universally" and "existentially" to distinguish between the two types of safety and liveness properties.

References

[1] P. A. Abdulla, K. Cerans, B. Jonsson, and T. Yih-Kuen. Algorithmic analysis of programs with well quasi-ordered domains. *Information and Computation*. To appear.

[2] B. Alpern and F. B. Schneider. Defining liveness. *Information Processing Letters*, 21(4):181–185, Oct. 1985.

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

[4] K. R. Apt and E.-R. Olderog. *Verification of Sequential and Concurrent Programs*. Springer-Verlag, New York, 1991.

[5] K. R. Apt and G. D. Plotkin. Countable nondeterminism and random assignment. *Journal of the ACM*, 33(4):724–767, Oct. 1986.

[1]Regular trees are a strict subset of the set of trees [24].

[6] A. Bouajjani, J. C. Fernandez, S. Graf, C. Rodriguez, and J. Sifakis. Safety for branching time semantics. In *18th ICALP, Automata, Languages and Programming*, volume 510 of *LNCS*. Springer-Verlag, 1991.

[7] E. M. Clarke and E. A. Emerson. Synthesis of synchronization skeletons for branching time temporal logic. In *Logic of Programs: Workshop*, volume 131 of *LNCS*, pages 52–71. Springer-Verlag, May 1981.

[8] E. A. Emerson. Temporal and modal logic. In van Leeuwen [25], pages 995–1072.

[9] E. A. Emerson and E. M. Clarke. Characterizing correctness properties of parallel programs as fixpoints. In *Proceedings 7th International Colloquium on Automata, Languages, and Programming*, volume 85 of *LNCS*. Springer-Verlag, 1981.

[10] E. A. Emerson and J. Y. Halpern. "Sometimes" and "not never" revisited: on branching versus linear time temporal logic. *JACM*, 33(1):151–178, Jan. 1986.

[11] W. Feijen, editor. *Beauty is Our Business*. Springer-Verlag, 1990.

[12] N. Francez. *Fairness*. Springer-Verlag, Berlin, 1986.

[13] H. P. Gumm. Another glance at the alpern-schneider characterization of safety and liveness in concurrent executions. *Information Processing Letters*, 47(6):291–294, Oct. 1993.

[14] D. Harel and A. Pnueli. On the development of reactive systems. In K. R. Apt, editor, *Logics and Models of Concurrent Systems*, volume F-13 of *NATO ASI Series*, pages 477–498. Springer-Verlag, 1985.

[15] J. L. Kelly. *General Topology*. D. Van Nostrand, 1955.

[16] N. Klarlund. personal communication.

[17] D. Kozen. Results on the propositional Mu-Calculus. *Theoretical Computer Science*, pages 334–354, December 1983.

[18] O. Kupferman and M. Y. Vardi. Model checking of safety properties. In N. Halbwachs and D. Peled, editors, *Computer-Aided Verification–CAV '99*, volume 1633 of *LNCS*, pages 172–183. Springer-Verlag, 1999.

[19] L. Lamport. Proving the correctness of multiprocess programs. *IEEE Transactions on Software Engineering SE-3*, 2:125–143, Mar. 1997.

[20] A. Pnueli. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science*, pages 46–57, Providence, Rhode Island, 31 Oct.–2 Nov. 1977. IEEE.

[21] V. R. Pratt. A decidable mu-calculus: Preliminary report. In *22nd Annual Symposium on Foundations of Computer Science*, pages 421–427, Nashville, Tennessee, Oct. 1981. IEEE.

[22] M. Rem. A personal perspective of the alpern-schneider characterization of safety and liveness. In Feijen [11], pages 365–372.

[23] A. P. Sistla. Safety, liveness and fairness in temporal logic. *Formal Aspects of Computing*, 6:495–511, 1994.

[24] W. Thomas. Automata on infinite objects. In van Leeuwen [25], pages 135–192.

[25] J. van Leeuwen, editor. *Handbook of Theoretical Computer Science: Volume B: Formal Models and Semantics*. Elsevier, Amsterdam, 1990.

A Appendix

All definitions and terminology in the appendix are taken from Bouajjani *et al.* [6]

Definition 13 *A Kripke tree is a tuple $K = (Q, \Sigma, q_0, R, \pi)$ where Q is a countable set of states, q_0 is the initial state, $R \subseteq Q \times Q$ is the transition relation (having no cycles and enforcing finite branching), Σ is a finite alphabet and $\pi : Q \to \Sigma$ is the labeling function.*

Definition 14 *A safety recognizer is a tuple $S = (\Sigma, W, w_0, \rho)$ where W is a finite set of states, w_0 is the initial state and $\rho \subseteq W \times \Sigma \times \mathcal{P}(W)$ is the transition relation.*

Definition 15 *A safety recognizer $S = (\Sigma, W, w_0, \rho)$ accepts Kripke tree $K = (Q, \Sigma, q_0, R, \pi)$ iff there exists $\lambda : Q \to W$ such that $\lambda(q_0) = w_0$ and for all $q \in Q$, there exists $\Gamma \subseteq W$ such that $(\lambda(q), \pi(q), \Gamma) \in \rho$ and $\{\lambda(q') : (q, q') \in R\} \subseteq \Gamma$.*

Lemma 26 *There is no safety recognizer S such that, S accepts K iff K satisfies $\mathsf{EG}P$.*

Proof Assume, to the contrary, that there is such an $S = (\Sigma, W, w_0, \rho)$.

Let $K = (Q, \Sigma, q_0, R, \pi)$ be defined as follows. $\{q : (q_0, q) \in R\} = \{q_1, q_2\}$. Furthermore, $\pi(q_0) = \pi(q_2) = P$ and $\pi(q_1) = \neg P$. Also, suppose K is a total tree and K has one full path (through q_2) which satisfies $\mathsf{G}P$. Since K satisfies $\mathsf{EG}P$ then by assumption K is accepted by S and there exits $\lambda : Q \to W$ such that $\lambda(q_0) = w_0$ and for all $q \in Q$, there exists $\Gamma \subseteq W$ such that $(\lambda(q), \pi(q), \Gamma) \in \rho$ and $\{\lambda(q') : (q, q') \in R\} \subseteq \Gamma$.

Consider $K' = (Q', \Sigma, q_0, R', \pi')$ defined as follows. Intuitively, K' consists of the root of K, the subtree of K rooted at q_1 plus another copy of the subtree rooted at q_1 in place of the subtree rooted at q_2 which has been completely excised. Formally, let $Q_1 = \{q \in Q : q$ is a descendant of q_1 in $K\}$. Then $Q' = \{q_0\} \cup Q_1 \cup \{q' : q \in Q_1\}$. $(a, b) \in R'$ iff

- $a = q_0$ and $b = q_1$ or q_1' or
- $a, b \in Q_1$ and $(a, b) \in R$ or
- $q, r \in Q_1$, $(q, r) \in R$, $a = q'$, and $b = r'$.

π' is defined as follows: $\pi'(q_0) = \pi(q_0)$; for $q \in Q_1$, $\pi'(q) = \pi(q)$; for $q \in Q_1, \pi'(q') = \pi(q)$. Clearly, K' does not satisfy $\mathsf{EG}P$.

Consider $\lambda' : Q' \to W$ defined as follows. $\lambda'(q_0) = \lambda(q_0)$. For all $q \in Q_1$, $\lambda'(q) = \lambda(q)$ and $\lambda'(q') = \lambda(q)$. Then $\lambda'(q_0) = \lambda(q_0) = w_0$. Also, $(\lambda(q_0), \pi(q_0), \Gamma) \in \rho$ and $\{\lambda(q_1), \lambda(q_2)\} \subseteq \Gamma$, for some Γ. Hence, $\{\lambda'(q_1), \lambda'(q_1')\} \subseteq \Gamma$ and $(\lambda'(q_0), \pi'(q_0), \Gamma) \in \rho$.

Suppose $q \in Q_1$. $(\lambda(q), \pi(q), \Gamma_q) \in \rho$ for some Γ_q and that $\{\lambda(r) : (q, r) \in R\} \subseteq \Gamma_q$. This implies that $(\lambda'(q), \pi'(q), \Gamma_q) \in \rho$ and that $\{\lambda'(r) : (q, r) \in R'\} \subseteq \Gamma_q$. Furthermore $(\lambda'(q'), \pi'(q'), \Gamma_q) \in \rho$ and that $\{\lambda'(r') : (q, r) \in R'\} \subseteq \Gamma_q\}$. Hence, S accepts K', contradicting the assumption that S accepts only those trees satisfying $\mathsf{EG}P$. $\square$