SIGACT News Logic Column 11

The Finite and the Infinite in Temporal Logic*

Riccardo Pucella
Cornell University
Ithaca, NY 14853 USA
riccardo@cs.cornell.edu

Note from the Editor: I am always looking for contributions. If you have any suggestion concerning the content of the Logic Column, or even better, if you would like to contribute by writing a survey or tutorial on your own work or topic related to your area of interest, feel free to get in touch with me.

At the last TACAS in Barcelona, already almost a year ago, Alur, Etessami, and Madhusudan [2004] introduced CaRet, a temporal logic framework for reasoning about programs with nested procedure calls and returns. The details of the logic were themselves interesting (I will return to them later), but a thought struck me during the presentation, whether an axiomatization might help understand the new temporal operators present in CaRet. Thinking a bit more about this question quickly led to further questions about the notion of finiteness and infinity in temporal logic as it is used in Computer Science. This examination of the properties of temporal logic operators under finite and infinite interpretations is the topic that I would like to discuss here. I will relate the discussion back to CaRet towards the end of the article, and derive a sound and complete axiomatization for an important fragment of the logic.

Temporal logic is commonly used in Computer Science to reason about temporal properties of state sequences [Pnueli 1977; Gabbay, Pnueli, Shelah, and Stavi 1980]. Generally, these state sequences are the states that arise during the execution of a program. Temporal logic lets one write down properties such as “an acquired lock is eventually released” or “it is never the case that the value of such variable is zero”. These kinds of properties become even more important in concurrent programs, where properties such as “every process eventually executes its critical section”, or “no two processes ever execute their critical section simultaneously” are, shall I say, critical. Many approaches have been developed for reasoning about programs using temporal logic. Most modern methods are based on model checking (see [Clarke, Grumberg, and Peled 1999], for instance), while other popular approaches are more proof-theoretic (see [Schneider 1997], for instance).

In the vast majority of cases, temporal logic is interpreted over infinite state sequences. Those infinite sequences arise naturally, for example, when modeling reactive systems, which are systems that maintain a permanent interaction with their environment, and hence are assumed to never

*© Riccardo Pucella, 2005. This version differs slightly from that published in SIGACT News 36(1); it corrects a number of typos in the semantics. Thanks to Claudia Zepeda for pointing them out.
terminate [Manna and Pnueli 1992]. Even when modeling systems that may terminate, it is often acceptable to assume that the final state of the system is simply infinitely repeated; this allows infinite state sequences to be used. Intuitively, this approach works as long as nothing of interest happens after the system has finished executing. What happens, however, when one wants to reason about explicitly finite state sequences? For instance, one may want to reason about a sequence of states embedded in a larger structure, where extending the sequence to an infinite sequence by repeating the final state is not necessarily a reasonable step to take. This is exactly what happens in CAReT, where some of the temporal operators are interpreted over the finite traces that make up procedure invocations, all in the context of a complete program execution.\(^1\)

In order to characterize the properties of the CAReT operators, we need to understand the properties of temporal operators in the presence of finite sequences. Accordingly, my first goal is to make clear the properties of temporal operators when interpreted over (1) finite state sequences, (2) infinite state sequences, and (3) both finite and infinite state sequences. To do this, I present a particularly simple axiomatization of temporal logic that is sound and complete over the class of finite and infinite state sequences. As expected, a sound and complete axiomatization for the logic interpreted over finite state sequences only can be derived by simply adding an axiom that says “there are no infinite state sequences”, and a sound and complete axiomatization for the logic interpreted over infinite state sequences only can be derived by simply adding an axiom that says “all state sequences are infinite”. Interestingly, there is a uniform elementary proof that covers all the cases. These results can be found in various forms in the literature, albeit often implicitly. The presentation I give is meant to emphasize the contribution of exclusively finite and exclusively infinite traces to the axiomatization of the temporal operators. The axiomatization will be used as the basis of the sound and complete axiomatization for a fragment of CAReT.

**Temporal Logic Over Infinite Sequences**

Let me first discuss finiteness and infinity in the context of the simpler framework of propositional linear temporal logic (LTL). The only temporal operators we consider are future time operators, meaning that at a given state one can only reason about the current and future states, and not past states. Furthermore, LTL embodies a linear notion of time: from any given state, there is a single sequence of states describing the future.\(^2\)

The language LTL is defined inductively by the following grammar, where \(p\) ranges over primitive propositions taken from a set \(\Phi_0\):

\[
\varphi, \psi ::= p \mid \neg \varphi \mid \varphi \land \psi \mid \varnothing \varphi \mid \varphi U \psi.
\]

Let \(\varphi \lor \psi\) stand for \(\neg (\neg \varphi \land \neg \psi)\), and \(\varphi \Rightarrow \psi\) stand for \(\neg \varphi \lor \psi\). Further, let \(\diamond \varphi\) stand for \(true U \varphi\), and \(\Box \varphi\) stand for \(\neg \diamond \neg \varphi\). Finally, define \(\varnothing \varphi\) as the dual of \(\Diamond\), namely, \(\neg \Diamond \neg \varphi\). The operator \(\Diamond\) is sometimes called “weak next”; \(\varnothing \varphi\) reads “if there is a next state, then \(\varphi\) holds there”. The operator

\(^1\)Another context where this occurs is in process logics [Pratt 1979], which lets one reason about finite segments of program executions within a larger and potentially infinite execution. I hope to revisit this topic in an upcoming column. Saake and Lipeck [1988] and Havelund and Roşu [2001] give additional motivation for considering temporal reasoning over finite sequences. Other uses of temporal logic, for instance in descriptive complexity theory, often assume an interpretation restricted to finite words [Straubing 1994].

\(^2\)This is in contrast to logics interpreted over branching time, where a state can possibly have multiple futures, and formulas can involve quantification over futures. See Emerson and Halpern [1986] for details on the relationship between linear and branching time temporal logics.
○ is sometimes called “strong next”; ○φ reads “there is a next state, and φ holds there”. The formula φ U ψ reads “φ holds until ψ is true”, while ○φ reads “φ will eventually be true” and □φ reads “φ is and always will be true”.

Temporal logic is interpreted over (linear) temporal structures. A temporal structure is a tuple $M = (S, σ, π)$ where $S$ is a set of states, $σ$ is a finite or infinite sequence of states in $S$, and $π$ is a valuation on the states, where $π(s)$ is the set of primitive propositions true at state $s$. Let $|σ|$ denote the length of $σ$, understood to be $∞$ if $σ$ is infinite. Infinity is assumed to behave in the standard way with respect to integers, for instance, $i < ∞$ for all integers $i$. A temporal structure $M = (S, σ, π)$ is finite if $σ$ is finite, and infinite otherwise. (Thus, finiteness of a structure depends on the finiteness of the sequence, not that of the state space.) If $σ = s_0s_1s_2\ldots$, I will sometimes use the notation $σ_i$ to refer to state $s_i$ in $σ$.

Let $M$ be the set of finite and infinite temporal structures. Let $M^{inf}$ be the class of infinite structures, and $M^{fin}$ be the class of finite structures. Satisfiability of a formula can be defined in a number of equivalent ways. If $M = (S, σ, π)$, where $σ = s_0s_1\ldots$, possibly finite, define $(M, i) \models φ$, meaning that formula $φ$ is true in structure $M$ at position $i ∈ \{0, \ldots, |σ|\}$, inductively as follows:

1. $(M, i) \models p$ if $p ∈ π(s_i)$
2. $(M, i) \models ¬φ$ if $(M, i) \not\models φ$
3. $(M, i) \models φ ∧ ψ$ if $(M, i) \models φ$ and $(M, i) \models ψ$
4. $(M, i) \models ○φ$ if $i = |σ|$ or $(M, i + 1) \models φ$
5. $(M, i) \models φ U ψ$ if $∃j \in \{i, \ldots, |σ|\}$ such that $(M, j) \models ψ$ and $∀k \in \{i, \ldots, j − 1\}$, $(M, k) \models φ$.

Observe that ○φ is defined in such a way that if the sequence is finite, ○φ is true for all formulas φ at the final state of the sequence. More drastically, ○false is true at a state if and only if it is the final state in the sequence. A formula φ is valid, written $\models φ$, if $(M, i) \models φ$ for all structures $M$ and positions $i$.

The following axiomatization AX is well-known to be sound and complete for temporal logic, as interpreted over infinite structures [Gabbay, Pnueli, Shelah, and Stavi 1980; Fagin, Halpern, Moses, and Vardi 1995; Halpern, Meyden, and Vardi 2004]:

1. From φ and φ ⇒ ψ infer ψ.
2. From φ U ψ infer φ V (φ ∧ ○(φ U ψ)).
3. From φ infer ○φ.
4. From φ′ ⇒ ¬ψ ∧ ○φ′ infer φ′ ⇒ ¬(φ U ψ).

Prop. All instances of propositional tautologies in LTL.

Recall that an axiomatization is sound if every provable formula is valid, and complete if every valid formula is provable.
This axiomatization, by virtue of soundness and completeness, intrinsically characterizes infinite structures. In fact, it is not hard to see that the axiomatization is not sound for finite structures. More precisely, axioms T2 and T3 are not valid in finite structures. To see this, let $p$ be a primitive proposition, and consider the structure $M_1 = (\{s\}, s, \pi)$, that is, a finite structure with a single state $s$, a sequence consisting of that single state $s$, and where $\pi(s) = \{p\}$. It is easy to verify that

$$(M, 0) \not\models false \lor (p \land \Box(p U false)) \Rightarrow p U false,$$

which is an instance of T2, specifically, the $\Leftarrow$ implication of T2, and

$$(M, 0) \not\models \Box(false) \Rightarrow \neg \Box(true),$$

which is an instance of T3. Thus, in order to derive an axiomatization that is sound and complete for a class structure including finite ones, axioms T2 and T3 must somehow be weakened.

A General Axiomatization

There is an axiomatization that is sound and complete for the class of finite and infinite structures. Let $AX^{gen}$ be the following axiomatization, obtained from $AX$ by replacing axioms T2 and T3 by axioms T2' and T3':

Prop. All instances of propositional tautologies in LTL.

MP. From $\varphi$ and $\varphi \Rightarrow \psi$ infer $\psi$.

T1. $\Box\varphi \land \Box(\varphi \Rightarrow \psi) \Rightarrow \Box\psi$.

T2'. $\varphi U \psi \iff \psi \lor (\varphi \land \Box(\varphi U \psi))$.

T3'. $\Box\varphi \iff (\Box false \lor \Box\varphi)$.

RT1. From $\varphi$ infer $\Box\varphi$.

RT2. From $\varphi' \Rightarrow \neg \psi \land \Box\varphi'$ infer $\varphi' \Rightarrow \neg(\varphi U \psi)$.

Axiom T3' captures the following intuition for $\Box\varphi$: either the next time step does not exist, or $\varphi$ is true there. As I have already argued, the fact that the next time step does not exist is expressed by $\Box false$. The following variants of T1 are provable in $AX^{gen}$: $\Box\varphi \land \Box(\varphi \Rightarrow \psi) \Rightarrow \Box\psi$, and $\Box\varphi \land \Box(\varphi \Rightarrow \psi) \Rightarrow \Box\psi$. This axiomatization is a simplification of the axiomatization of the future fragment of the temporal logic of Lichtenstein, Pnueli, and Zuck [1985]. Roughly speaking, the inference rule RT2 subsumes their axioms relating $\Box$ and $\square$, using the fact that $\Box$ is expressible using $U$.

The following two axioms can be used to tailor the axiomatization to the case where the structures are infinite, and the case where the structures are finite. For infinite structures, an axiom is needed to capture the fact that there is no final state:

Inf. $\neg \Box false$.

To obtain an axiomatization for finite structures, an axiom is needed to capture the fact that every finite structure has a final state:
Let $\textbf{AX}^\text{inf}$ be the axiomatization $\textbf{AX}^\text{gen}$ augmented with axiom Inf, and let $\textbf{AX}^\text{fin}$ be the axiomatization $\textbf{AX}^\text{gen}$ augmented with axiom Fin. These axiomatizations completely characterize validity in the appropriate class of structures. More precisely, the following result holds.

**Theorem 1.** For formulas in the language LTL,

(a) $\textbf{AX}^\text{gen}$ is a sound and complete axiomatization with respect to $\mathcal{M}$,
(b) $\textbf{AX}^\text{inf}$ is a sound and complete axiomatization with respect to $\mathcal{M}^\text{inf}$,
(c) $\textbf{AX}^\text{fin}$ is a sound and complete axiomatization with respect to $\mathcal{M}^\text{fin}$.

The proof of this theorem is not difficult, and uses well-understood technology. The only difficulty, in some sense, is coming up with the proposed axiomatization. To illustrate where all the details are used, let me spell out the details of the proof. Soundness is straightforward to establish in all cases. Completeness is established by proving the following equivalent statement. Recall that a formula $\phi$ is ax-consistent, for an axiomatization $\text{ax}$, if $\neg \phi$ is not provable using the axioms and inference rules of $\text{ax}$. Completeness is equivalent to the fact that consistency implies satisfiability. Thus, it suffices to show that if $\phi$ is consistent with respect to one of the particular axiomatization, then it is satisfiable in a structure in the corresponding class, that is, it is possible to construct an appropriate structure such that $\phi$ is true in a state of the structure.

The construction is essentially independent of the axiomatization under consideration. Fix the formula $\phi$. The states of the model will be constructed from an extension of the set of subformulas of $\phi$. Let $\text{Cl}'(\phi)$ be the smallest set $S$ such that:

(a) $\phi \in S$,
(b) $\text{true} \cup \text{false} \in S$,
(c) if $\neg \psi \in S$ then $\psi \in S$,
(d) if $\psi_1 \land \psi_2 \in S$ then $\psi_1 \in S$ and $\psi_2 \in S$,
(e) if $\circ \psi \in S$ then $\psi \in S$,
(f) if $\circ \neg \psi \in S$ then $\neg \psi \in S$,
(g) if $\psi_1 \lor \psi_2 \in S$ then $\psi_1 \in S$, $\psi_2 \in S$, and $\circ(\psi_1 \cup \psi_2) \in S$.

Let $\text{Cl}(\phi) = \text{Cl}'(\phi) \cup \{\neg \psi \mid \psi \in \text{Cl}'(\phi)\}$. It is easy to check that for any $\phi$, $\text{Cl}(\phi)$ is a finite set of formulas. Note that $\text{false}$ and $\neg \text{false}$ are always in $\text{Cl}(\phi)$.

Let $\text{ax}$ range over $\textbf{AX}^\text{gen}$, $\textbf{AX}^\text{inf}$, and $\textbf{AX}^\text{fin}$. An $\text{ax}$-atom of $\phi$ is a maximally $\text{ax}$-consistent subset of formulas in $\text{Cl}(\phi)$. It is easy to see that $\text{ax}$-atoms are finite. Let $\text{At}^\text{ax}(\phi)$ be the set of $\text{ax}$-atoms of $\phi$; we use $V, W, \ldots$ to denote $\text{ax}$-atoms. Associate with every $\text{ax}$-atom $V$ a formula $\hat{V}$, the conjunction of all the formulas in $V$, that is, $\hat{V} = \bigwedge_{\psi \in V} \psi$. It is straightforward to check that for every formula $\psi \in \text{Cl}(\phi)$ and every $\text{ax}$-atom $V$ of $\phi$, either $\psi$ or $\neg \psi$ is in $V$. (If not, then $V$ is not maximally $\text{ax}$-consistent.) Using axiom Prop, it is easy to show that any formula $\psi \in \text{Cl}(\phi)$ is provably equivalent to the disjunction $\bigvee_{V \in \text{At}^\text{ax}|\psi \in V} \hat{V}$, and $\text{true}$ is provably equivalent to the disjunction $\bigvee_{V \in \text{At}^\text{ax}} \hat{V}$.
For ax-atoms $V$ and $W$, define $V \xrightarrow{ax} W$ if $\widehat{V} \land \widehat{W}$ is ax-consistent. Let $V \xrightarrow{ax}$ be the set $\{W \mid V \xrightarrow{ax} W\}$. A chain of ax-atoms is a finite or infinite sequence $V_0, V_1, \ldots$ of ax-atoms with the property that $V_i \xrightarrow{ax} V_{i+1}$, for all $i$. A chain $V_0, V_1, \ldots$ of ax-atoms is acceptable if for all $i$, whenever $\psi_1 \cup \psi_2 \in V_i$, then there exists $j \geq i$ such that $\psi_2 \in V_j$ and $\psi_1 \in V_i, \ldots, V_{j-1}$. The following lemma isolates all the properties needed to prove the completeness results.

**Lemma 2.**

(a) For all $\bigcirc \psi \in \text{Cl}(\varphi)$ and ax-atoms $V$, $\bigcirc \psi \in V$ if and only if for all $W \in V \xrightarrow{ax}$, $\psi \in W$.

(b) For all $\bigcirc \psi \in \text{Cl}(\varphi)$ and ax-atoms $V$, $\bigcirc \psi \in V$ if and only if there exists $W \in V \xrightarrow{ax}$ such that $\psi \in W$.

(c) For all $\psi_1 \cup \psi_2 \in \text{Cl}(\varphi)$ and ax-atoms $V_0$, $\psi_1 \cup \psi_2 \in V_0$ if and only if there exists a finite chain $V_0, V_1, \ldots, V_k$ such that $\psi_1 \in V_0, \ldots, V_{k-1}$ and $\psi_2 \in V_k$.

(d) For all ax-atoms $V$, $\bigcirc \text{false} \in V$ if and only if $V \xrightarrow{ax} = \emptyset$.

(e) For all $AX^{\text{fin}}$-atoms $V_0$, there exists a finite chain $V_0, \ldots, V_k$ such that $\bigcirc \text{false} \in V_k$.

(f) Every finite chain of $AX^{\text{gen}}$-atoms is extensible to an acceptable chain (finite or infinite).

(g) Every finite chain of $AX^{\text{inf}}$-atoms is extensible to an infinite acceptable chain.

(h) Every finite chain of $AX^{\text{fin}}$-atoms is extensible to a finite acceptable chain.

**Proof.** The proof technique is adapted from that of Halpern, van der Meyden, and Vardi [2004].

(a) Assume that $\bigcirc \psi \in V$, and let $W \in V \xrightarrow{ax}$. By way of contradiction, assume that $\psi \notin W$. Then, $\neg \psi \in W$, that is, $\vdash \widehat{W} \Rightarrow \neg \psi$. By Prop and RT1, $\vdash (\psi \Rightarrow \neg \widehat{W})$. By assumption, $\bigcirc \psi \in V$, that is, $\vdash \widehat{V} \Rightarrow \bigcirc \psi$. By MP and T1, $\vdash \widehat{V} \Rightarrow \bigcirc \neg \widehat{W}$. But $V \xrightarrow{ax} W$ means that $\widehat{V} \land \widehat{W}$ is consistent, so that $\not\vdash \bigcirc \widehat{V} \Rightarrow \bigcirc \neg \widehat{W}$, a contradiction. So $\psi \in W$.

Conversely, assume that for all $W \in V \xrightarrow{ax}$, $\psi \notin W$. By way of contradiction, assume that $\bigcirc \psi \notin V$, so that $\neg \bigcirc \psi \in V$, and thus $\vdash \widehat{V} \Rightarrow \neg \bigcirc \psi$. For any $W$ such that $\psi \notin W$, it must be the case that $W \notin V \xrightarrow{ax}$, and thus $\widehat{V} \land \widehat{W}$ is inconsistent. Thus, $\widehat{V} \land \widehat{W}$ is inconsistent for all $W$ such that $\psi \notin W$, and $\vdash (\bigcirc \psi \notin V)$, and $\vdash (\psi \notin W)$, is inconsistent, that is, $\widehat{V} \land \bigcirc \psi$ is inconsistent, and $\vdash \widehat{V} \Rightarrow \bigcirc \neg \psi$, or $\vdash \widehat{V} \Rightarrow \bigcirc \psi$. By assumption, $\vdash \widehat{V} \Rightarrow \bigcirc \psi$, so that $\vdash \widehat{V} \Rightarrow \bigcirc \psi$, that is, $\vdash \neg \widehat{V}$, which contradicts the fact that $V$ is a consistent set of formulas. Thus, $\bigcirc \psi \in V$, as desired.

(b) Assume that $\bigcirc \psi \in V$. If $\bigcirc \psi \in V$, then $\bigcirc \psi \in V$, and $\bigcirc \psi \in \text{Cl}(\varphi)$ by closure rule (4). Hence, by part (a), all $W \in V \xrightarrow{ax}$ are such that $\psi \in W$. It suffices to show then there is a $W$ such that $V \xrightarrow{ax} W$. Assume not. Then $\vdash (\bigcirc \psi \notin V)$, and hence $\vdash (\psi \land \text{true})$, and $\vdash \widehat{V} \Rightarrow \bigcirc \text{false}$. Because $\bigcirc \psi \in V$, then $\vdash \widehat{V} \Rightarrow \neg \bigcirc \psi$. So $\vdash \widehat{V} \Rightarrow \bigcirc \psi$, that is, $\vdash \neg \widehat{V}$, contradicting $V$ being consistent. So there must be a $W \in V \xrightarrow{ax}$.

Conversely, assume that there exists $W \in V \xrightarrow{ax}$ and $\psi \in W$. Since $\widehat{V} \land \widehat{W}$ is consistent, so is $\vdash (\psi \land \bigcirc \psi)$, and $\vdash \neg \bigcirc \psi$. Assume by way of contradiction that $\psi \notin V$, so that $\neg \bigcirc \psi \notin V$. Then $\vdash \widehat{V} \Rightarrow \neg \bigcirc \psi$, a contradiction. Therefore, $\bigcirc \psi \in V$.

(c) Assume that $\psi_1 \cup \psi_2 \in V_0$. Suppose by way of contradiction that no suitable chain exists. Let $T$ be the smallest set $S$ of ax-atoms of $\varphi$ such that $V_0 \in S$, and if $W \in V \xrightarrow{ax}$ (for some $V$ in $S$) and $\widehat{W} \land \psi_1$, then $W \in S$. If $T$ is a set of ax-atoms, let $\widehat{T} = \bigvee_{W \in T} \widehat{W}$. Clearly, $\neg \psi_2 \in W$ for
all $W$ in $T$, and thus, $\vdash \widehat{T} \Rightarrow \neg \psi_2$. Moreover, for every $V$ in $T$ and $W \in V^{ax}$, either $W \in T$, or $\neg \psi_2 \in W$ and $\neg \psi_2 \in W$. This yields $\vdash \widehat{T} \Rightarrow \circ (T \lor (\neg \psi_1 \land \neg \psi_2)))$. It follows easily from T1, T2', RT1, RT2 that $\vdash \widehat{T} \Rightarrow \neg (\psi_1 U \psi_2)$. In particular, $\vdash \widehat{V}_0 \Rightarrow \neg (\psi_1 U \psi_2)$, contradicting $\psi_1 U \psi_2 \in V_0$.

Conversely, by induction on $k$, if there exists $V_1 \in V_0^{ax}, \ldots, V_k \in V_{k-1}^{ax}$, $\psi_1 \in V_i$ for $i \in \{0, \ldots, k-1\}$, and $\psi_2 \in V_k$, then $\psi_1 U \psi_2 \in V_0$. If $k = 0$, the result follows immediately by an application of T2' and T3'. For a general $k$, assume by way of contradiction that $\psi_1 U \psi_2 \notin V_0$ (so that $\neg \psi_1 U \psi_2 \in V_0$), and consider the subchain $V_1, \ldots, V_k$, such that $V_2 \in V_1^{ax}, \ldots, V_k \in V_{k-1}^{ax}$, $\psi_1 \in V_1, \ldots, V_{k-1}$, and $\psi_2 \in V_k$. By the induction hypothesis, $\psi_1 U \psi_2 \in V_1$, that is, $\vdash \widehat{V}_1 \Rightarrow \psi_1 U \psi_2$. Since $V_1 \in V_0^{ax}$, $\widehat{V}_0 \land \circ \widehat{V}_1$ consistent, and by an application of RT1 and a $\circ$-variant of T1, $\widehat{V}_0 \land \psi_1 U \psi_2$ is consistent. Since $\psi_1 \in V_0$, $\vdash \widehat{V}_0 \Rightarrow \psi_1$, and thus $\widehat{V}_0 \land \psi_1 \land \circ \psi_1 U \psi_2$ is also consistent. By T2', $\widehat{V}_0 \land \psi_1 U \psi_2$ is consistent, that is, $\neg \widehat{V}_0 \Rightarrow \neg \psi_1 U \psi_2$, contradicting the assumption that $\neg \psi_1 U \psi_2 \in V_0$. Thus, $\psi_1 U \psi_2$ must be in $V_0$, as desired.

(d) Assume that $\vdash \widehat{V} \Rightarrow \circ false$. By way of contradiction, assume there is a $W \in V^{ax}$. By Prop, $false \Rightarrow \neg \widehat{W}$, and by RT1, $\vdash \circ (false \Rightarrow \neg \widehat{W})$. By propositional reasoning and T1, $\vdash \widehat{V} \Rightarrow \circ \neg \widehat{W}$, which is equivalent to $\vdash \widehat{V} \Rightarrow \neg \circ \widehat{W}$, that is, $\vdash \neg (\circ \widehat{V})$, contradicting the assumption that $W \in V^{ax}$, that is, that $\widehat{V} \land \circ \widehat{W}$ is consistent.

Conversely, assume that there is no $W \in V^{ax}$. Therefore, for all $W$, $\widehat{V} \land \circ \widehat{W}$ is inconsistent, and thus, $\lor \widehat{W} (\widehat{V} \land \circ \widehat{W})$ is inconsistent. By propositional reasoning, $\widehat{V} \land \circ \lor \widehat{W}$, and thus $\widehat{V} \land \circ \lor true$ is inconsistent. By propositional reasoning and definition of $\circ$, this simply means that $\vdash \widehat{V} \Rightarrow \circ false$.

(e) Let $V_0$ be an $AX^{fin}$-atom. Suppose by way of contradiction that no suitable chain exists. Let $T$ be the smallest set $S$ of $AX^{fin}$-atoms of $\varphi$ such that $V_0 \in S$, and if $W \in V^{AX^{fin}}$ (for some $V$ in $S$) then $W \in S$. Clearly, $\neg \circ false \in W$ for all $W$ in $T$ (otherwise, it could be used to construct a finite chain assumed not to exist), and thus, $\vdash \widehat{T} \Rightarrow \neg \circ false$. Moreover, for every $V$ in $T$ and $W \in V^{AX^{fin}}$, $W \in T$. Therefore, it is possible to derive $\vdash \widehat{T} \Rightarrow \circ \widehat{T}$, which implies that $\vdash \widehat{T} \Rightarrow \circ (\widehat{T} \lor (\neg true \land \neg \circ false))$. It follows easily from T1, T2', RT1, RT2 that $\vdash \widehat{T} \Rightarrow \neg (true \land \circ false)$. In particular, $\vdash \widehat{V}_0 \Rightarrow \neg (true \land \circ false)$, contradicting $true \land \circ false \in V_0$, by virtue of axiom Fin.

(f) Let $V_0, \ldots, V_n$ be a finite chain of $AX^{gen}$-atoms. Consider a formula $\psi_1 U \psi_2 \in V_0$. It follows, from T2' and parts (a) and (b), either that $\psi_2 \in V_j$ for some $j \in \{0, \ldots, n\}$ and $\psi_1 \in V_i$ for $l \in \{0, \ldots, j-1\}$, or that $\psi_1 \in V_j$ for all $j \in \{0, \ldots, n\}$, and $\psi_1 U \psi_2 \in V_n$. In the latter case, by part (c), there exists a chain $V_n, \ldots, V_n'$ such that $\psi_1 \in V_k$ for $k \in \{n, \ldots, n'-1\}$ and $\psi_2 \in V_{n'}$. This gives a finite extension of the original chain that satisfies the obligation of acceptability for $\psi_1 U \psi_2$ at $V_0$. Applying this argument to the remaining $U$-formulas in $V_0$ produces a finite chain that satisfies all the obligations at $V_0$. Apply the same procedure to $V_1$, and so on. In the limit, this produces an acceptable chain extending the original chain. This chain can be either finite, or infinite.

(g) Let $V_0, \ldots, V_n$ be a finite chain of $AX^{inf}$-atoms. Just as in part (f), it is possible to construct an acceptable chain extending this chain that satisfies all the obligations of the $U$-formulas. If this process results in a finite acceptable chain $V_0, \ldots, V_n$, this chain can be extended to an infinite acceptable chain as follows. Given the final state $V_n'$ of the chain, there exists a state $V_{n'+1} \in V_n^{AX^{inf}}$. Otherwise, by part (d), $\vdash AX^{inf} \widehat{V}_{n'} \Rightarrow \circ false$. However, by Inf, $\vdash AX^{inf} \neg \circ false$, and thus by MP, $\vdash AX^{inf} \neg \widehat{V}_{n'}$, contradicting the fact that $V_{n'}$ is $AX^{inf}$-consistent. Thus, there
must exist $V_{n+1}' \in V_{n}'^{AX}$. Let $V_0, \ldots, V_{n+1}'$ be the new chain formed in this way. This chain can be once again extended to an acceptable chain, by ensuring that all the obligations of the $\nu$-formulas are satisfied. In the limit, this new procedure produces an infinite acceptable chain.

(h) Let $V_0, \ldots, V_n$ be a finite chain of $AX^{fin}$-atoms, $V_0, \ldots, V_k$. By part (e), there exists a finite chain $V_k, \ldots, V_n$ such that $\circ false \in V_n$. By part (d), this means that there $V_n^{AX^{fin}} = \emptyset$. It remains to show that the chain is acceptable, that is, for every $\psi_1 U \psi_2$ in $V_0, \ldots, V_n$, the obligations are met. Let $\psi_1 U \psi_2 \in V_i$. Just as in part (f), it follows, from T2' and parts (a) and (b), either that $\psi_2 \in V_j$ for some $i \leq j \leq n$ and $\psi_1 \in V_l$ for $i \leq l < j$, or that $\psi_1 \in V_j$ for all $i \leq j \leq n$, and both $\neg \psi_2 \in V_j$ and $\psi_1 U \psi_2 \in V_j$. In the former case, the obligations for $\psi_1 U \psi_2$ are met. The latter case cannot arise. Indeed, if $\psi_1 U \psi_2 \in V_i$, then $\Vdash V_i \Rightarrow \psi_1 U \psi_2$, so that $\Vdash V_i \Rightarrow \psi_2 \lor (\psi_1 \land \circ(\psi_1 U \psi_2))$. Since $\neg \psi_2 \in V_n$, $\Vdash V_n \Rightarrow \neg \psi_2$, so that $\Vdash V_n \Rightarrow \psi_1 \land \circ(\psi_1 U \psi_2)$. Therefore, $\circ(\psi_1 U \psi_2)$ must be in $V_n$. By part (b), there must exist $W \in V_n^{AX^{fin}} \cap \psi_1 U \psi_2 \in W$, which contradicts that fact that $V_n^{AX^{fin}} = \emptyset$.

The completeness results of Theorem 1 follow easily from Lemma 2. Consider the axiomatization $AX^{gen}$. Assume that $\varphi$ is $AX^{gen}$-consistent. Since $\varphi \in Cl(\varphi)$, $\varphi \in V^\varphi$ for some $AX^{gen}$-atom $V^\varphi$ of $\varphi$. Construct the structure $M = (S, \sigma, \pi)$ by taking the set of states $S$ to be the set $Art_{AX^{gen}}(\varphi)$ of $AX^{gen}$-atoms of $\varphi$. Define the interpretation $\pi$ by $\pi(V) = \{ p \mid p \in V \}$. All that remains now is to extract a sequence $\sigma$ in $S$ that satisfies $\varphi$. By Lemma 2(f), $V^\varphi$, a one-element finite chain of $AX^{gen}$-atoms, is extensible to an acceptable chain $\sigma = V_0 V_1 \ldots$. It is easy to check, by induction on the structure of $\varphi$, that $(M, i) \models \varphi$ if and only if $\varphi \in V_i$. Since $\varphi \in V^\varphi = V_0$, then $(M, 0) \models \varphi$. A similar argument holds for $AX^{inf}$ and $AX^{fin}$, invoking Lemma 2(g) and Lemma 2(h), respectively, to construct an acceptable chain $\sigma$.

The Linear Temporal Logic of Calls and Returns

While the above discussion is still fresh, let me now talk about the CaRet logic. CaRet was designed for reasoning about programs, in the form of state sequences, each sequence corresponding to an execution of the program. It was especially designed for reasoning about nonregular properties of programs. The classical example of such a property is the correctness of procedures with respect to pre and post conditions, that is, verifying that if $\varphi$ holds before every call to a procedure, then $\psi$ holds after the procedure returns. The nonregularity of this property is due to the fact that finding the state where the procedure returns requires matching the number of calls and returns within the body of the procedure. CaRet provides operators for doing just that. While frameworks for verifying procedure with respect to pre and post conditions go back to the seminal work of Hoare [1969], the main contribution of CaRet is a decidable model-checking procedure for programs expressed as recursive state machines (equivalently, pushdown systems) [Alur, Etessami, and Yannakakis 2001; Benedikt, Godefroid, and Reps 2001]. To achieve this, CaRet assumes that every state is tagged, indicating whether it is a call state (meaning it is a state that performs a procedure call), a return state (meaning it is a state that corresponds to having returned from a procedure), or an internal state (everything else). I will not discuss the model-checking algorithm here, but instead examine the properties of the new operators that CaRet introduces.

The language CaRet of linear propositional temporal logic with calls and returns is defined inductively by the following grammar, where $p$ ranges over primitive propositions taken from a set

---

*ACM SIGACT News* 8 Vol. —, No. —
\( \Phi_0 \), which includes call, ret, and int:\(^4\)

\[
\varphi, \psi ::= p \mid \neg \varphi \mid \varphi \land \psi \mid \bigcirc \varphi \mid \varphi \mathcal{U} \psi \mid \bigcirc^a \varphi \mid \varphi \mathcal{U}^a \psi.
\]

As before, define the usual abbreviations. Let \( \varphi \lor \psi \) stand for \( \neg (\neg \varphi \land \neg \psi) \), and \( \varphi \rightarrow \psi \) stand for \( \neg \varphi \lor \psi \). Define, as in LTL, \( \Diamond \varphi \) to stand for true \( \forall \varphi \), \( \square \varphi \) to stand for \( \neg \Diamond \neg \varphi \), and \( \bigcirc \varphi \) to stand for \( \neg \circ \neg \varphi \). Define \( \bigcirc^a \varphi \), \( \square^a \varphi \), and \( \bigcirc^a \varphi \) in a similar way.

The \( \bigcirc \) and \( \mathcal{U} \) operators, the global-time operators, are the standard operators from LTL, interpreted over whole sequences of states.\(^5\) Thus, \( \bigcirc \varphi \) means that \( \varphi \) holds at the next state whether or not the next state is a state in an invoked procedure, or the next state follows from returning from a procedure. The \( \bigcirc^a \) and \( \mathcal{U}^a \) operators, the abstract-time operators, do not consider all states in the sequence, but only the states in the current procedure context. Thus, \( \bigcirc^a \varphi \) means that \( \varphi \) holds at the abstract next state of the procedure—if the current state is a procedure call, then the abstract next state is in fact the matching return state; if the current state is the last state of a procedure invocation, there is no abstract next state; similarly, if the current state is a procedure call that never returns (say, it enters an infinite loop), there is no abstract next state. Correspondingly, \( \varphi \mathcal{U}^a \psi \) means that the abstract path from the current state (i.e., the path formed by successive abstract successors) satisfies \( \varphi \mathcal{U} \psi \).

To formalize these intuitions, CARET is interpreted over structured (linear) temporal structures. An structured temporal structure is a tuple \( M = (S, \sigma, \pi) \) where \( S \) is a set of states, \( \sigma \) is an infinite sequence of structured states in \( S \times \{\text{call, ret, int}\} \), and \( \pi \) is a valuation on the states, where \( \pi(s) \) is the set of primitive propositions true at state \( s \). For a sequence \( \sigma \), define an abstract successor function \( \text{succ}^a_{\sigma} \) giving, for every index \( i \) into \( \sigma \), the index of the next abstract state for \( \sigma_i \). Formally, the abstract successor is defined as follows. First, for a sequence of structured states \( \sigma \), define the partial map \( R_{\sigma}(i) \), which maps any \( i \) to the first unmatched return after \( i \), that is, the first return that does not correspond to a procedure call performed after \( i \): \( R_{\sigma}(i) = j \), where \( j \) is the smallest \( j' \) such that \( j' > i \), \( \sigma_{j'} \) is a return state, and the number of calls and returns in \( \sigma_{i+1}, \ldots, \sigma_{j'-1} \) are equal; \( R_{\sigma}(i) = \bot \) if there is no such \( j' \). (Intuitively, \( \bot \) represents the value “undefined”.) The abstract successor functions can now be defined:

\[
\text{succ}^a_{\sigma}(i) \triangleq \begin{cases} 
R_{\sigma}(i) & \text{if } \sigma_i = (-, \text{call}) \\
\bot & \text{if } \sigma_i \neq (-, \text{call}) \text{ and } \sigma_{i+1} = (-, \text{ret}) \\
i + 1 & \text{otherwise.}
\end{cases}
\]

Let \( \mathcal{M}^\sigma \) be the set of structured temporal structures. Satisfiability of a formula is defined as follows. If \( M = (S, \sigma, \pi) \), where \( \sigma = ((s_0, t_0), (s_1, t_1), \ldots) \), define \( (M, i) \models \varphi \), meaning that formula \( \varphi \) is true in structure \( M \) at position \( i \geq 0 \) inductively as follows:

\[
(M, i) \models p \text{ if } p \in \pi(s_i) \text{ or } p = t_i
\]

\[
(M, i) \models \neg \varphi \text{ if } (M, i) \not\models \varphi
\]

\[
(M, i) \models \varphi \land \psi \text{ if } (M, i) \models \varphi \text{ and } (M, i) \models \psi
\]

\(^4\)In fact, this is just a fragment of CARET. The full logic includes past-time temporal operators that walk back the call chain of a procedure. I believe that the development in this section extends in a straightforward way to the full language, but I have not checked the details.

\(^5\)Alur, Etessami, and Madhusudan use \( \bigcirc^a \varphi \) and \( \varphi \mathcal{U}^a \psi \) for the global-time operators.
\((M, i) \models \Diamond \varphi\) if \((M, i + 1) \models \varphi\)

\((M, i) \models \varphi \mathcal{U} \psi\) if \(\exists j \geq i\) such that \((M, j) \models \psi\) and \(\forall k \in \{i, \ldots, j - 1\}, (M, k) \models \varphi\)

\((M, i) \models \Diamond^a \varphi\) if \(\text{succ}^a(\sigma(i)) = \bot\) or \((M, \text{succ}^a(\sigma(i))) \models \varphi\)

\((M, i) \models \varphi \mathcal{U}^a \psi\) if \(\exists i_0, i_1, \ldots, i_k\) (with \(i_0 = i\)) such that \(\text{succ}^a(i_j) = i_{j+1} \neq \bot\) (for \(j = 0, \ldots, k - 1\)), \((M, i_k) \models \psi\), and \((M, i_j) \models \varphi\) (for \(j = 0, \ldots, k - 1\)).

(The semantics above uses a “weak” semantics for \(\Diamond^a \varphi\), while the original description of CaRet uses a “strong” semantics. In other words, the interpretation of \(\Diamond^a \varphi\) in the original CaRet is the same as that of \(\Diamond \varphi\) here. I made this choice for consistency with the usual reading of \(\Diamond\) as a weak next and to reuse the development of last section. Clearly, there is no loss of expressiveness from this change.)

What about an axiomatization for this logic? The following axioms account for the fact that the \(\Diamond/\mathcal{U}\) fragment of CaRet is essentially LTL interpreted over infinite sequences.

Prop. All instances of propositional tautologies in CaRet.

MP. From \(\varphi\) and \(\varphi \Rightarrow \psi\) infer \(\psi\).

G1. \(\Diamond \varphi \land \Diamond(\varphi \Rightarrow \psi) \Rightarrow \Diamond \psi\).

G2. \(\varphi \mathcal{U} \psi \Leftrightarrow \psi \lor (\varphi \land \Diamond(\varphi \mathcal{U} \psi))\).

G3. \(\Diamond \varphi \Leftrightarrow (\Diamond false \lor \Diamond \varphi)\).

G4. \(\neg \Diamond false\).

RG1. From \(\varphi\) infer \(\Diamond \varphi\).

RG2. From \(\varphi' \Rightarrow \neg \psi \land \Diamond \varphi'\) infer \(\varphi' \Rightarrow \neg(\varphi \mathcal{U} \psi)\).

The operators \(\Diamond^a\) and \(\mathcal{U}^a\) behave like the standard temporal operators, except they are interpreted over possibly finite sequences.

A1. \(\Diamond^a \varphi \land \Diamond^a(\varphi \Rightarrow \psi) \Rightarrow \Diamond^a \psi\).

A2. \(\varphi \mathcal{U}^a \psi \Leftrightarrow \psi \lor (\varphi \land \Diamond^a(\varphi \mathcal{U}^a \psi))\).

A3. \(\Diamond^a \varphi \Leftrightarrow (\Diamond^a false \lor \Diamond^a \varphi)\).

RA1. From \(\varphi\) infer \(\Diamond^a \varphi\).

RA2. From \(\varphi' \Rightarrow \neg \psi \land \Diamond^a \varphi'\) infer \(\varphi' \Rightarrow \neg(\varphi \mathcal{U}^a \psi)\).

The remaining axioms capture the relationship between the kind of states (call states, return states, internal states), and the behavior of the various next-time operators. Roughly, this amounts to capturing the properties of the \(\text{succ}^a\) function, when it is defined, and when it is not. The following axiom says that there is exactly one of call, ret, int that holds at any state.

C1. \((\text{call} \land \neg \text{ret} \land \neg \text{int}) \lor (\neg \text{call} \land \text{ret} \land \neg \text{int}) \lor (\neg \text{call} \land \neg \text{ret} \land \text{int})\).
If the current state is not a call state, then the properties of the abstract next state operator depend on whether the next global state is a return state.

C2. \( \neg call \land \bigcirc ( \neg ret) \Rightarrow ( \bigcirc \varphi \iff \bigcirc^a \varphi) \).

C3. \( \neg call \land \bigcirc ( ret) \Rightarrow \bigcirc^a false \).

C4. \( \bigcirc^a \varphi \Rightarrow \bigdiamond \varphi \).

Already, it is possible to derive from these axioms that if a state is not a call state and there is no abstract next state, then the global next state must be a return state; in other words, the only case where there is no abstract next state (unless a procedure call is performed) is at the end of a procedure invocation. Here is a formal derivation of \( \neg call \land \bigcirc^a false \Rightarrow \bigcirc ret \):

1. \( \vdash \neg call \land \bigcirc ret \Rightarrow ( \bigcirc true \iff \neg \bigcirc^a false) \) (C2)
2. \( \vdash \neg call \Rightarrow ( \bigcirc false \Rightarrow ( \bigcirc true \Rightarrow \neg \bigcirc^a false)) \) (1, Taut, MP)
3. \( \vdash ( \bigcirc false \Rightarrow ( \bigcirc true \Rightarrow \neg \bigcirc^a false)) \Rightarrow ( \bigcirc true \Rightarrow ( \bigcirc^a false \Rightarrow \neg \bigcirc false)) \) (Taut)
4. \( \vdash \neg call \Rightarrow ( \bigcirc true \Rightarrow ( \bigcirc^a false \Rightarrow \neg \bigcirc false)) \) (2, 3, MP)
5. \( \vdash \bigcirc true \Rightarrow (\neg call \Rightarrow ( \bigcirc^a false \Rightarrow \neg \bigcirc false)) \) (4, Taut, MP)
6. \( \vdash \bigcirc true \) (Taut, RG1)
7. \( \vdash \neg call \Rightarrow ( \bigcirc false \Rightarrow \neg \bigcirc false) \) (5, 6, MP)
8. \( \vdash \neg call \land \bigcirc^a false \Rightarrow \neg \bigcirc false \) (7, Taut, MP)
9. \( \vdash \bigcirc ret \Rightarrow ( \bigcirc false \lor \neg \bigcirc false) \) (G3)
10. \( \vdash ( \bigcirc ret \iff ( \bigcirc false \lor \neg \bigcirc ret)) \Rightarrow (\neg \bigcirc false \Rightarrow ( \neg \bigcirc false \Rightarrow \bigcirc ret)) \) (Taut)
11. \( \vdash \neg \bigcirc false \Rightarrow ( \neg \bigcirc ret \Rightarrow \bigcirc ret) \) (10, MP)
12. \( \vdash \neg \bigcirc false \) (G4)
13. \( \vdash \neg \bigcirc \neg ret \Rightarrow \bigcirc ret \) (11, 12, MP)
14. \( \vdash \neg call \land \bigcirc^a false \Rightarrow \bigcirc ret \) (8, 13, MP).

If the current position is a call, then the abstract successor exists or not, depending on whether or not there is a balanced number of calls and returns before the return matching the call. This turns out to be painful to capture. Intuitively, the logic cannot count—there is no way to say (directly) that “there are exactly the same number of call states as there are return states before the matching return”. The best one can do is basically enumerate all possibilities. Define the class of formulas \( \text{CR}^c_{m,n}(\varphi) \), one for every \( c, m, n \geq 0 \) such that \( c + m \geq n \). Intuitively, \( \text{CR}^c_{m,n}(\varphi) \) says that between the current state and the first state where \( \varphi \) holds, there are exactly \( m \) call states and \( n \) return states, and moreover there are never more then \( c \) return states more than the number of call states.

\[
\text{CR}^c_{m,n}(\varphi) \triangleq \begin{cases} 
\text{int} \mathcal{U} \varphi & \text{if } m = 0, n = 0 \\
\text{int} \mathcal{U} (\text{call} \land \bigcirc \text{CR}^{c+1}_{m-1,n}) & \text{if } m > 0, n = 0 \\
\text{int} \mathcal{U} (\text{ret} \land \bigcirc \text{CR}^{c+1}_{m,n-1}) & \text{if } m = 0, n > 0 \\
\text{int} \mathcal{U} (\text{call} \land \bigcirc \text{CR}^{c+1}_{m-1,n}) & \text{if } m > 0, n > 0, c = 0 \\
\text{int} \mathcal{U} (\text{call} \land \bigcirc \text{CR}^{c+1}_{m-1,n}(\varphi)) & \text{if } m > 0, n > 0, c > 0 \\
\lor \text{int} \mathcal{U} (\text{ret} \land \bigcirc \text{CR}^{c+1}_{m,n-1}(\varphi)) & \text{if } m > 0, n > 0, c > 0.
\end{cases}
\]
With this, it is possible to define a countable number of axioms that essentially say that if the current state is a call state, and the number of calls and returns following the current state match before there is a return where \( \varphi \) holds, then there is an abstract next state and \( \varphi \) holds there.

C5. \( \text{call} \land \circ CR_{m,n}^0 (\text{ret} \land \varphi) \Rightarrow \circ^n \varphi \) (for \( n \geq 0 \)).

Of course, C5 is a family of axioms, one for each \( n \geq 0 \).

Similarly, if the number of calls exceeds the number of returns after a call state, then there cannot be an abstract next state.

C6. \( \text{call} \land \circ CR_{m,n}^0 (\square \text{ret}) \Rightarrow \circ^m \text{false} \) (for \( m > n \geq 0 \)).

Again, C6 is a family of axioms, one for each \( m > n \geq 0 \).

**Theorem 3.** \( AX^\varphi \) is a sound and complete axiomatization for \( \text{CARet} \) with respect to structured temporal structures.

Soundness is straightforward. The strategy for proving completeness is, unsurprisingly, analogous to that of the proof of Theorem 3: assuming \( \varphi \) is consistent, completeness requires showing that \( \varphi \) is satisfiable; to construct a model of \( \varphi \), take an atom of \( \varphi \) containing \( \varphi \), and extend it to an acceptable infinite sequence of atoms. I leave it to the reader to generalize \( \hat{\varphi} \) qualification, as there is a single axiomatization to consider. The relation \( \hat{\varphi} \) is again defined to hold if \( \hat{\varphi} \) is consistent. A chain of atoms is a finite or infinite sequence \( V_0, V_1, \ldots \) of atoms with the property that \( V_i \rightarrow V_{i+1} \), for all \( i \). It remains to show how to extend a finite chain of atoms into a suitably defined acceptable chain. This is where things vary from the proof of Theorem 1, to account for the structured states.

First, given a (possibly finite) chain \( V = V_0, V_1, V_2, \ldots \), define \( R_T \) and \( \text{succ}^a_V \) just as they are defined for sequences of states, but instead of checking that an element at index \( i \) is a call state (resp., a return state or an internal state) by checking that it is of the form \( (\text{call}) \) (resp., \( (\text{ret}) \) or \( (\text{int}) \)), do so by checking that \( \text{call} \in V_i \) (resp., \( \text{ret} \in V_i \) or \( \text{int} \in V_i \)). This is well defined, thanks to axiom C1.

An infinite chain \( \hat{V} = V_0, V_1, \ldots \) of atoms is acceptable if for all \( i \),

- whenever \( \psi_1 U \psi_2 \in V_i \), then there exists \( i \leq j \leq |\sigma| \) such that \( \psi_2 \in V_j \) and \( \psi_1 \in V_i, \ldots, V_{j-1} \);
- whenever \( \circ^a \psi \in V_i \), then \( \psi \in V_{\text{succ}^a_V(i)} \);
- whenever \( \psi_1 U^a \psi_2 \in V_i \), then there exists \( i_0, i_1, \ldots, i_k \) (with \( i_0 = i \)) such that \( \text{succ}^a_V(i_j) = i_{j+1} \neq \perp \) (for \( j = 0, \ldots, k-1 \)), \( \psi_2 \in V_{i_k} \), and \( \psi_1 \in V_{i_j} \) (for \( j = 0, \ldots, k-1 \)).

The following lemma isolates the properties needed to finish the proof of completeness.

**Lemma 4.**

(a) For all \( \circ \psi \in \text{Cl}(\varphi) \) and atoms \( V \), \( \circ \psi \in V \) if and only if for all \( W \in V \rightarrow, \psi \in W \).
(b) For all $\circ \psi \in Cl(\varphi)$ and atoms $V$, $\circ \psi \in V$ if and only if there exists $W \in V \rightarrow$ such that $\psi \in W$.

(c) For all $\psi_1 \cup \psi_2 \in Cl(\varphi)$ and atoms $V_0, \psi_1 \cup \psi_2 \in V_0$ if and only if there exists a finite chain $V_0, V_1, \ldots, V_k$ such that $\psi_1 \in V_0, \ldots, V_{k-1} \text{ and } \psi_2 \in V_k$.

(d) For all $\circ^a \psi \in Cl(\varphi)$ and atoms $V_0, \circ^a \psi \in V_0$ if and only if for all finite chains $V_0, V_1, \ldots, V_k$ such that $\psi \in V_k$.

(e) For all $\psi_1 \cup^x \psi_2 \in Cl(\varphi)$ and atoms $V_0, \psi_1 \cup^x \psi_2 \in V_0$ if and only if there exists a finite chain $V_0, V_1, \ldots, V_k$ and indices $i_0, \ldots, i_j < k$ such that $\text{succ}^a_{V}(i_l) = i_{l+1} \text{ (for } l = 0, \ldots, j - 1)$, $\text{succ}^x_{V}(i_j) = k$, $\psi_1 \in V_{i_0}, \ldots, V_{i_j}$ and $\psi_2 \in V_k$.

(f) Every finite chain of atoms is extensible to an acceptable chain.

I leave the proof as an exercise; it follows the structure of the proof of Lemma 2 quite closely, despite requiring a slightly more involved argument for parts (d) and (e), as one would expect.

So there we are: a sound and complete axiomatization for (an important fragment of) CaRet. We get the usual benefits from it, namely, the possibility of reasoning purely deductively about structured temporal structures, and this gives an alternative to model-checking for proving properties of programs. I do not know, at this point, whether the decision problem for the logic is decidable, and so reasoning deductively may not be feasible. One problem with the axiomatization $\textbf{AX}^{cr}$ is that it is not very nice. In fact, axioms C5 and C6 are downright ugly. I believe this is difficult to avoid. Since CaRet does not have operators for counting, the axioms must keep count the hard way—listing all possibilities—to match returns to calls. It may still be possible, however, to develop alternate axiomatizations more suited to using CaRet as a program logic. That remains to be seen.

References

Alur, R., K. Etessami, and P. Madhusudan (2004). A temporal logic of nested calls and returns. In Proc. 10th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS’04), Volume 2988 of Lecture Notes in Computer Science, pp. 467–481. Springer-Verlag.

Alur, R., K. Etessami, and M. Yannakakis (2001). Analysis of recursive state machines. In Proc. 13th Conference on Computer Aided Verification (CAV’01), Volume 2102 of Lecture Notes in Computer Science, pp. 207–220. Springer-Verlag.

Benedikt, M., P. Godefroid, and T. Reps (2001). Model checking of unrestricted hierarchical state machines. In Proc. 28th Colloquium on Automata, Languages, and Programming (ICALP’01), Volume 2076 of Lecture Notes in Computer Science, pp. 652–666. Springer-Verlag.

Clarke, E. M., O. Grumberg, and D. Peled (1999). Model Checking. MIT Press.

Emerson, E. A. and J. Y. Halpern (1986). “Sometimes” and “not never” revisited: On branching versus linear time temporal logic. Journal of the ACM 33(1), 151–178.

Fagin, R., J. Y. Halpern, Y. Moses, and M. Y. Vardi (1995). Reasoning about Knowledge. MIT Press.

ACM SIGACT News 13 Vol. —, No. —
Gabbay, D., A. Pnueli, S. Shelah, and J. Stavi (1980). On the temporal analysis of fairness. In Proc. 7th Annual ACM Symposium on Principles of Programming Languages (POPL’80), pp. 163–173.

Halpern, J. Y., R. van der Meyden, and M. Y. Vardi (2004). Complete axiomatizations for reasoning about knowledge and time. SIAM Journal on Computing 33(3), 674–703.

Havelund, K. and G. Roșu (2001). Testing linear temporal logic formulae on finite execution traces. RIACS Technical Report TR 01-08, Research Institute for Advanced Computer Science.

Hoare, C. A. R. (1969). An axiomatic basis for computer programming. Communications of the ACM 12, 576–580, 583.

Lichtenstein, O., A. Pnueli, and L. Zuck (1985). The glory of the past. In Proc. Conference on Logics of Programs, Number 193 in Lecture Notes in Computer Science, pp. 196–218. Springer-Verlag.

Manna, Z. and A. Pnueli (1992). The Temporal Logic of Reactive and Concurrent Systems. Springer-Verlag.

Pnueli, A. (1977). The temporal logic of programs. In Proc. 18th IEEE Symposium on the Foundations of Computer Science (FOCS’77), pp. 46–57.

Pratt, V. (1979). Process logic. In Proc. 6th Annual ACM Symposium on Principles of Programming Languages (POPL’79), pp. 93–100. ACM Press.

Saake, G. and U. W. Lipeck (1988). Using finite-linear temporal logic for specifying database dynamics. In Proc. 2nd Workshop on Computer Science Logic (CSL’88), Volume 385 of Lecture Notes in Computer Science, pp. 288–300. Springer-Verlag.

Schneider, F. B. (1997). On Concurrent Programming. Springer-Verlag.

Straubing, H. (1994). Finite Automata, Formal Logic, and Circuit Complexity. Birkhauser.