A Simple Treatment of Property Preservation via Simulation

Ching-Tsun Chou
⟨chou@cs.ucla.edu⟩

Computer Science Department
University of California at Los Angeles
Los Angeles, CA 90024, U.S.A.

(Last revised: 21 March 1995)

Abstract

Let s_1 and s_2 be two reactive systems related by simulation. For what kind of properties Φ can one infer $s_1 \models \Phi$ (s_1 satisfies Φ) from $s_2 \models \Phi$, or vice versa? In this paper we present a simple treatment of this problem, which we call *property preservation via simulation*, that unifies and generalizes several results in the literature. Our treatment is based on two ideas. First, we introduce an *infinitary logic* $\mathcal{L}$ that is so expressive that preservation results for its fragments generalize existing preservation results for linear-time, branching-time, modal, and fixpoint logics. Thus $\mathcal{L}$ provides a unifying framework within which existing results can be derived and compared. Second, we formulate a set of *reduction lemmas*, one for each construct of $\mathcal{L}$, that allows the preservation proof of a compound property to be reduced to the preservation proofs of its constituent properties. Each reduction lemma is easy to prove by itself, but they together with the infinitary logic yield nontrivial results. The reduction lemmas show clearly where property preservation via simulation works and where it does not.

1 Introduction

A powerful technique for relating and reasoning about reactive systems is that of *simulation* [16]. Roughly speaking, a relation R between reactive systems is a simulation iff (if and only if) whenever s_1 and s_2 are related by R, if s_1 can perform an action a and become s_1', then s_2 can also perform a and become s_2' such that s_1' and s_2' are still related by R. The power of the simulation technique stems from the fact that if two reactive systems are related by simulation, then one can infer certain properties of one from those of the other. So a natural and important problem to ask is:

> Let s_1 and s_2 be two reactive systems related by simulation. For what kind of properties Φ can one infer $s_1 \models \Phi$ (s_1 satisfies Φ) from $s_2 \models \Phi$, or vice versa?

This problem, which we call *property preservation via simulation*, has been treated in the literature for linear-time logics [4], branching-time logics [6], modal logics [12], and fixpoint logics [2]. Those treatments, however, are ad hoc and (sometimes) unnecessarily complicated. In this paper we present a simple treatment that nonetheless unifies and generalizes the results cited above.

Our treatment is based on two ideas. First, we introduce an *infinitary logic* $\mathcal{L}$ that is so expressive that preservation results for its fragments generalize existing preservation results for linear-time, branching-time, modal, and fixpoint logics. Thus $\mathcal{L}$ provides a unifying framework within which existing results can be derived and compared. Second, we formulate a set of *reduction lemmas*, one for each construct of $\mathcal{L}$, that allows the preservation proof of a compound property to be reduced to the preservation proofs of its constituent properties. Each reduction lemma is easy to prove by itself, but they together with the infinitary logic yield nontrivial results. The reduction lemmas show clearly where property preservation via simulation works and where it does not.

Some computer scientists may feel uneasy about infinitary logics, for their formulas cannot be manipulated by a computer. But using infinitary logics in computer science is nothing new. For example, Hennessy-Milner Logic (as described in [17]) is an infinitary logic, and infinitary logics have been used extensively in finite model theory (see, e.g., [13]).

The rest of this paper is organized as follows. Section 2 defines the syntax and semantics of $\mathcal{L}$ and some of its fragments that will be used later. Section 3 proves the reduction lemmas and the preservation results for the

fragments of $\mathcal{L}$. Section 4 applies the preservation results for the fragments of $\mathcal{L}$ to linear-time, branching-time, modal, and fixpoint logics found in the literature. Section 5 is the conclusion.

The reader is referred to [18] for the set theory used in this paper, in particular transfinite recursion and induction, ordinals, and fixpoints. The symbol "$\triangleq$" means "equals by definition".

2 Infinitary Logic $\mathcal{L}$ and Its Fragments

The infinitary logic $\mathcal{L}$ is a branching-time logic whose formulas are built up from primitive state and path formulas using negation, *arbitrary* conjunction and disjunction, the *path quantifiers* $\forall$ and $\exists$, and the *next* operator $\bigcirc$. For ease in stating and proving preservation results, the formulas of $\mathcal{L}$ are given in *positive normal form*, i.e., negation appears only on primitive formulas.

Definition 1 [Syntax of $\mathcal{L}$]
Let $\mathcal{P}$ be a set of *primitive formulas* which is partitioned into a set $\mathcal{P}^\Sigma$ of *primitive state formulas* and a set $\mathcal{P}^\Pi$ of *primitive path formulas*. The infinitary logic $\mathcal{L}$ consists of a class[1] $\mathcal{L}^\Sigma$ of *state formulas* and a class $\mathcal{L}^\Pi$ of *path formulas*, which are inductively defined as the smallest classes of expressions such that:

$$\begin{aligned}
P \in \mathcal{P}^\Sigma &\Rightarrow P \in \mathcal{L}^\Sigma \text{ and } \neg P \in \mathcal{L}^\Sigma \\
\{\Phi_i \mid i \in I\} \subseteq \mathcal{L}^\Sigma &\Rightarrow \wedge_{i\in I}\Phi_i \in \mathcal{L}^\Sigma \text{ and } \vee_{i\in I}\Phi_i \in \mathcal{L}^\Sigma \\
\Psi \in \mathcal{L}^\Pi &\Rightarrow \forall(\Psi) \in \mathcal{L}^\Sigma \text{ and } \exists(\Psi) \in \mathcal{L}^\Sigma \\
Q \in \mathcal{P}^\Pi &\Rightarrow Q \in \mathcal{L}^\Pi \text{ and } \neg Q \in \mathcal{L}^\Pi \\
\{\Psi_i \mid i \in I\} \subseteq \mathcal{L}^\Pi &\Rightarrow \wedge_{i\in I}\Psi_i \in \mathcal{L}^\Pi \text{ and } \vee_{i\in I}\Psi_i \in \mathcal{L}^\Pi \\
\Psi \in \mathcal{L}^\Pi &\Rightarrow \bigcirc\Psi \in \mathcal{L}^\Pi \\
\Phi \in \mathcal{L}^\Sigma &\Rightarrow \Phi \in \mathcal{L}^\Pi
\end{aligned}$$

where I is an arbitrary index set. The above inductive definition can be summarized by the following "grammar":

$$\begin{aligned}
\Phi &::= P \mid \neg P \mid \wedge_{i\in I}\Phi_i \mid \vee_{i\in I}\Phi_i \mid \forall(\Psi) \mid \exists(\Psi) \\
\Psi &::= Q \mid \neg Q \mid \wedge_{i\in I}\Psi_i \mid \vee_{i\in I}\Psi_i \mid \bigcirc\Psi \mid \Phi
\end{aligned}$$

[1] Since the conjunction and disjunction of arbitrary sets of formulas are allowed, $\mathcal{L}^\Sigma$ and $\mathcal{L}^\Pi$ are too "big" to be sets and must be proper classes.

Finitary conjunctions of the form $\wedge_{i\in\{1,\ldots,n\}}\Phi_i$ (where $n \geq 2$) will often be written in infix form as $\Phi_1 \wedge \cdots \wedge \Phi_n$; similar remarks apply to disjunctions and path formulas. Note that we can define *truth* **tt** as $\wedge_{i\in\emptyset}\Phi_i$ and *falsity* **ff** as $\vee_{i\in\emptyset}\Phi_i$.

Formulas of $\mathcal{L}$ are interpreted on a labeled transition system, where a state formula is true or false for a state and a path formula is true or false for an (infinite) computation path, as follows.

Definition 2 [Labeled Transition Systems]
A *labeled transition system* (*LTS*) is a triple $T = (S, A, \rightarrow)$, where S is a set of *states*, A is a set of *actions*, and $\rightarrow \subseteq S \times A \times S$ is a *labeled transition relation*. For any $s, s' \in S$ and $a \in A$, $(s, a, s') \in \rightarrow$ will be abbreviated as $s \xrightarrow{a} s'$. A *path* in T is an infinite sequence of alternating states and actions, $\pi = s_0 a_0 s_1 a_1 s_2 \cdots$, such that $\forall\, n \geq 0 : s_n \xrightarrow{a_n} s_{n+1}$. For any $n \geq 0$, the n-th state, action, and *suffix* of π are $\boldsymbol{s}_n(\pi) \triangleq s_n$, $\boldsymbol{a}_n(\pi) \triangleq a_n$, and $\boldsymbol{f}_n(\pi) \triangleq s_n a_n s_{n+1} a_{n+1} s_{n+2} \cdots$, respectively. The set of paths in T is denoted by Π_T; for any $s \in S$, the set of paths in T starting from s is denoted by $\Pi_T(s)$. The subscripts of Π_T and $\Pi_T(\cdot)$ will be dropped whenever they are clear from context.

Definition 3 [Semantics of $\mathcal{L}$]
A *model* of $\mathcal{L}$ is a pair $M = (T, V)$, where $T = (S, A, \rightarrow)$ is a LTS and $V = (V^\Sigma, V^\Pi) \in (\mathcal{P}^\Sigma \rightarrow 2^S) \times (\mathcal{P}^\Pi \rightarrow 2^A)$ is a pair of *valuations* of primitive formulas of $\mathcal{L}$. The *satisfaction* of a state formula $\Phi \in \mathcal{L}^\Sigma$ by a state $s \in S$ (denoted $s \models^\Sigma_M \Phi$) and that of a path formula $\Psi \in \mathcal{L}^\Pi$ by a path $\pi \in \Pi_T$ (denoted $\pi \models^\Pi_M \Psi$) are defined recursively as follows:

$$
\begin{array}{lcl}
s \models^{\Sigma}_{M} P & \Leftrightarrow & s \in V^{\Sigma}(P) \\
s \models^{\Sigma}_{M} \neg P & \Leftrightarrow & s \notin V^{\Sigma}(P) \\
s \models^{\Sigma}_{M} \wedge_{i \in I} \Phi_i & \Leftrightarrow & \forall\, i \in I : s \models^{\Sigma}_{M} \Phi_i \\
s \models^{\Sigma}_{M} \vee_{i \in I} \Phi_i & \Leftrightarrow & \exists\, i \in I : s \models^{\Sigma}_{M} \Phi_i \\
s \models^{\Sigma}_{M} \forall(\Psi) & \Leftrightarrow & \forall\, \pi \in \Pi_T(s) : \pi \models^{\Pi}_{M} \Psi \\
s \models^{\Sigma}_{M} \exists(\Psi) & \Leftrightarrow & \exists\, \pi \in \Pi_T(s) : \pi \models^{\Pi}_{M} \Psi \\
\pi \models^{\Pi}_{M} Q & \Leftrightarrow & \boldsymbol{a}_0(\pi) \in V^{\Pi}(Q) \\
\pi \models^{\Pi}_{M} \neg Q & \Leftrightarrow & \boldsymbol{a}_0(\pi) \notin V^{\Pi}(Q) \\
\pi \models^{\Pi}_{M} \wedge_{i \in I} \Psi_i & \Leftrightarrow & \forall\, i \in I : \pi \models^{\Pi}_{M} \Psi_i \\
\pi \models^{\Pi}_{M} \vee_{i \in I} \Psi_i & \Leftrightarrow & \exists\, i \in I : \pi \models^{\Pi}_{M} \Psi_i \\
\pi \models^{\Pi}_{M} \bigcirc \Psi & \Leftrightarrow & \boldsymbol{f}_1(\pi) \models^{\Pi}_{M} \Psi \\
\pi \models^{\Pi}_{M} \Phi & \Leftrightarrow & \boldsymbol{s}_0(\pi) \models^{\Sigma}_{M} \Phi
\end{array}
$$

The subscripts and superscripts of $\models^{\Sigma}_{M}$ and $\models^{\Pi}_{M}$ will be dropped whenever they are clear from context.

A logic $\mathcal{F}$ is a *fragment* of another logic $\mathcal{G}$ iff every formula of $\mathcal{F}$ is a formula of $\mathcal{G}$. Usually, $\mathcal{F}$ is obtained from $\mathcal{G}$ by banning the use of certain constructs of $\mathcal{G}$. Some useful fragments of $\mathcal{L}$ are defined below.

Definition 4 [Fragments of $\mathcal{L}$]

1. $\forall\mathcal{L}$ is the fragment of $\mathcal{L}$ obtained by banning the use of $\exists$.
2. $\exists\mathcal{L}$ is the fragment of $\mathcal{L}$ obtained by banning the use of $\forall$.
3. $\boldsymbol{\ell}\mathcal{L}$ is the fragment of $\mathcal{L}$ obtained by banning the use of both $\forall$ and $\exists$.
4. $\mathcal{L}_+$, $\forall\mathcal{L}_+$, $\exists\mathcal{L}_+$, and $\boldsymbol{\ell}\mathcal{L}_+$ are the fragments of, respectively, $\mathcal{L}$, $\forall\mathcal{L}$, $\exists\mathcal{L}$, and $\boldsymbol{\ell}\mathcal{L}$ obtained by banning the use of $\neg$.

Fragments $\boldsymbol{\ell}\mathcal{L}$ and $\boldsymbol{\ell}\mathcal{L}_+$ are linear-time logics, in that their path formulas may contain state formulas but not vice versa; the other fragments are branching-time logics, in that their path and state formulas may contain each other. Note that the set $\mathcal{P}$ of primitive formulas is also a fragment of $\mathcal{L}$. For each fragment $\mathcal{F}$ of $\mathcal{L}$, $\mathcal{F}^{\Sigma}$ (respectively, $\mathcal{F}^{\Pi}$) denote the set of state (path) formulas of $\mathcal{F}$; note that both $\mathcal{F}^{\Sigma}$ and $\mathcal{F}^{\Pi}$ are also fragments of $\mathcal{L}$.

3 Property Preservation via Simulation

Let $T_1 = (S_1, A, \rightarrow_1)$ and $T_2 = (S_2, A, \rightarrow_2)$ be two LTS's which share a common set A of actions, $M_1 = (T_1, V_1)$ and $M_2 = (T_2, V_2)$ be two models of $\mathcal{L}$ based on T_1 and T_2 respectively, and $R \subseteq S_1 \times S_2$ be a relation from the states of T_1 to the states of T_2. The subscripts of $\rightarrow_1$ and $\rightarrow_2$ will be dropped whenever they are clear from context. The *inverse* of R, $R^{-1} \subseteq S_2 \times S_1$, is defined by: $(s_2, s_1) \in R^{-1} \Leftrightarrow (s_1, s_2) \in R$.

A state formula Φ is (bi-)preserved by R iff whenever s_1 and s_2 are related by R, if s_1 satisfies Φ then s_2 satisfies Φ (and vice versa). Preservation of path formulas is similarly defined, after R is suitably "lifted" to a relation $\Box R$ between paths in T_1 and paths in T_2. A fragment $\mathcal{F}$ of $\mathcal{L}$ is (bi-)preserved by R iff every formula of $\mathcal{F}$ is (bi-)preserved by R.

Definition 5 [Preservation and Bi-preservation]
For a state formula $\Phi \in \mathcal{L}^{\Sigma}$, R *preserves* Φ iff:

$$\forall (s_1, s_2) \in R : \; s_1 \models \Phi \;\Rightarrow\; s_2 \models \Phi$$

and R *bi-preserves* Φ iff:

$$\forall (s_1, s_2) \in R : \; s_1 \models \Phi \;\Leftrightarrow\; s_2 \models \Phi$$

For a path formula $\Psi \in \mathcal{L}^{\Pi}$, $\Box R$ *preserves* Ψ iff:

$$\forall (\pi_1, \pi_2) \in \Box R : \; \pi_1 \models \Psi \;\Rightarrow\; \pi_2 \models \Psi$$

and $\Box R$ *bi-preserves* Ψ iff:

$$\forall (\pi_1, \pi_2) \in \Box R : \; \pi_1 \models \Psi \;\Leftrightarrow\; \pi_2 \models \Psi$$

where $\Box R \subseteq \Pi_{T_1} \times \Pi_{T_2}$ is defined by:

$$(\pi_1, \pi_2) \in \Box R \;\Leftrightarrow\; \forall n \geq 0 : (\boldsymbol{s}_n(\pi_1), \boldsymbol{s}_n(\pi_2)) \in R \wedge (\boldsymbol{a}_n(\pi_1) = \boldsymbol{a}_n(\pi_2))$$

For a fragment $\mathcal{F}$ of $\mathcal{L}$, R (*bi-*)*preserves* $\mathcal{F}$ iff R (bi-)preserves each state formula Φ of $\mathcal{F}$ and $\Box R$ (bi-)preserves each path formula Ψ of $\mathcal{F}$.

Bi-preservation can be expressed in terms of preservation, as follows.

Lemma 6

(1) R bi-preserves $\Phi \;\Leftrightarrow\; R$ preserves $\Phi \wedge R^{-1}$ preserves Φ

(2) $\Box R$ bi-preserves $\Psi \;\Leftrightarrow\; \Box R$ preserves $\Psi \wedge \Box R^{-1}$ preserves Ψ

(3) R bi-preserves $\mathcal{F} \;\Leftrightarrow\; R$ preserves $\mathcal{F} \wedge R^{-1}$ preserves $\mathcal{F}$

The following proposition provides simple sufficient conditions for the preservation and bi-preservation of primitive path formulas.

Proposition 7

(4) $(\forall Q \in \mathcal{P}^\Pi : V_1^\Pi(Q) \subseteq V_2^\Pi(Q)) \Rightarrow R$ preserves $\mathcal{P}^\Pi$

(5) $(\forall Q \in \mathcal{P}^\Pi : V_1^\Pi(Q) = V_2^\Pi(Q)) \Rightarrow R$ bi-preserves $\mathcal{P}^\Pi$

The following reduction lemmas take care of all constructs of $\mathcal{L}$ except the path quantifiers $\forall$ and $\exists$.

Lemma 8 [Reduction Lemmas—Part 1a]

(6) R^{-1} preserves $P \Rightarrow R$ preserves $\neg P$

(7) $(\forall i \in I : R$ preserves $\Phi_i) \Rightarrow R$ preserves $\wedge_{i \in I} \Phi_i$

(8) $(\forall i \in I : R$ preserves $\Phi_i) \Rightarrow R$ preserves $\vee_{i \in I} \Phi_i$

(9) $\Box R^{-1}$ preserves $Q \Rightarrow \Box R$ preserves $\neg Q$

(10) $(\forall i \in I : \Box R$ preserves $\Psi_i) \Rightarrow \Box R$ preserves $\wedge_{i \in I} \Psi_i$

(11) $(\forall i \in I : \Box R$ preserves $\Psi_i) \Rightarrow \Box R$ preserves $\vee_{i \in I} \Psi_i$

(12) $\Box R$ preserves $\Psi \Rightarrow \Box R$ preserves $\bigcirc \Psi$

(13) R preserves $\Phi \Rightarrow \Box R$ preserves Φ

Proof: We prove only (12); the other cases are all straightforward. By definition, to show that $\Box R$ preserves $\bigcirc \Psi$, it suffices to show that:

$$\forall (\pi_1, \pi_2) \in \Box R : (\boldsymbol{f}_1(\pi_1) \models \Psi \Rightarrow \boldsymbol{f}_1(\pi_2) \models \Psi)$$

It is easy to see that $(\pi_1, \pi_2) \in \Box R \Rightarrow (\boldsymbol{f}_1(\pi_1), \boldsymbol{f}_1(\pi_2)) \in \Box R$, from which the above goal follows using the assumption that $\Box R$ preserves Ψ. ∎

Lemma 9 [Reduction Lemmas—Part 1b]

(14) R bi-preserves $P \Rightarrow R$ bi-preserves $\neg P$

(15) $(\forall i \in I : R$ bi-preserves $\Phi_i) \Rightarrow R$ bi-preserves $\wedge_{i \in I} \Phi_i$

(16) $(\forall i \in I : R$ bi-preserves $\Phi_i) \Rightarrow R$ bi-preserves $\vee_{i \in I} \Phi_i$

(17) $\Box R$ bi-preserves $Q \Rightarrow \Box R$ bi-preserves $\neg Q$

(18) $(\forall i \in I : \Box R$ bi-preserves $\Psi_i) \Rightarrow \Box R$ bi-preserves $\wedge_{i \in I} \Psi_i$

(19) $(\forall i \in I : \Box R$ bi-preserves $\Psi_i) \Rightarrow \Box R$ bi-preserves $\vee_{i \in I} \Psi_i$

(20) $\Box R$ bi-preserves $\Psi \Rightarrow \Box R$ bi-preserves $\bigcirc \Psi$

(21) R bi-preserves $\Phi \Rightarrow \Box R$ bi-preserves Φ

Proof: By Lemmas 6 and 8. ∎

Note that, up to this point, no assumption whatsoever has been made of R, which can be *any* relation from S_1 to S_2. Only the reduction lemmas for the path quantifiers $\forall$ and $\exists$ need R to be a simulation.

Definition 10 [Simulation and Bisimulation]
R is a *simulation* from T_1 to T_2 iff:

$$\forall\,(s_1, s_2) \in R :\ \forall\, a \in A :\ \forall\, s_1' : s_1 \xrightarrow{a} s_1' \ \Rightarrow \\ \exists\, s_2' : s_2 \xrightarrow{a} s_2' \ \wedge\ (s_1', s_2') \in R$$

R is a *bisimulation* between T_1 and T_2 iff both R is a simulation from T_1 to T_2 and R^{-1} is a simulation from T_2 to T_1.

Lemma 11 [Reduction Lemmas—Part 2a]

(22) R is a simulation from T_1 to T_2 $\Rightarrow$
$\Box R$ preserves Ψ $\Rightarrow$ R preserves $\exists(\Psi)$

(23) R^{-1} is a simulation from T_2 to T_1 $\Rightarrow$
$\Box R$ preserves Ψ $\Rightarrow$ R preserves $\forall(\Psi)$

Proof: We prove only (22); the proof of (23) is similar. By definition, to show that R preserves $\exists(\Psi)$, it suffices to show that:

$$\forall\,(s_1, s_2) \in R :\ (\,\exists\,\pi_1 \in \Pi(s_1) : \pi_1 \models \Psi\,) \ \Rightarrow\ (\,\exists\,\pi_2 \in \Pi(s_2) : \pi_2 \models \Psi\,)$$

An induction argument shows that the assumption that R is a simulation from T_1 to T_2 implies that:

$$\forall\,(s_1, s_2) \in R :\ \forall\,\pi_1 \in \Pi(s_1) :\ \exists\,\pi_2 \in \Pi(s_2) :\ (\pi_1, \pi_2) \in \Box R$$

from which the above goal follows easily using the assumption that $\Box R$ preserves Ψ. ∎

Lemma 12 [Reduction Lemmas—Part 2b]

(24) R is a bisimulation between T_1 and T_2 $\Rightarrow$
$\Box R$ bi-preserves Ψ $\Rightarrow$ R bi-preserves $\exists(\Psi)$

(25) R is a bisimulation between T_1 and T_2 $\Rightarrow$
$\Box R$ bi-preserves Ψ $\Rightarrow$ R bi-preserves $\forall(\Psi)$

Proof: By Lemmas 6 and 11. ∎

Finally, we are ready to state and prove preservation theorems for the fragments of $\mathcal{L}$ introduced in Definition 4.

Theorem 13 [Preservation Theorems for $\mathcal{L}$]

(26) R is a bisimulation between T_1 and T_2 $\Rightarrow$
R bi-preserves $\mathcal{P}$ $\Rightarrow$ R bi-preserves $\mathcal{L}$

(27) R is a bisimulation between T_1 and T_2 $\Rightarrow$
R preserves $\mathcal{P}$ $\Rightarrow$ R preserves $\mathcal{L}_+$

(28) R is a simulation from T_1 to T_2 $\Rightarrow$
R bi-preserves $\mathcal{P}$ $\Rightarrow$ R preserves $\exists\mathcal{L}$

(29) R is a simulation from T_1 to T_2 $\Rightarrow$
R preserves $\mathcal{P}$ $\Rightarrow$ R preserves $\exists\mathcal{L}_+$

(30) R^{-1} is a simulation from T_2 to T_1 $\Rightarrow$
R bi-preserves $\mathcal{P}$ $\Rightarrow$ R preserves $\forall\mathcal{L}$

(31) R^{-1} is a simulation from T_2 to T_1 $\Rightarrow$
R preserves $\mathcal{P}$ $\Rightarrow$ R preserves $\forall\mathcal{L}_+$

(32) R bi-preserves $\mathcal{P}$ $\Rightarrow$ R bi-preserves $\ell\mathcal{L}$

(33) R preserves $\mathcal{P}$ $\Rightarrow$ R preserves $\ell\mathcal{L}_+$

Proof: By structural induction on formulas and the reduction lemmas. ∎

4 Applications

4.1 Linear-Time Logics

Chou [4] proved reduction lemmas for the linear-time temporal operators $\Box$ ("always") and $\Diamond$ ("sometime") defined by:

$$\pi \models \Box\Psi \quad \Leftrightarrow \quad \forall\, n \geq 0 : \boldsymbol{f}_n(\pi) \models \Psi$$
$$\pi \models \Diamond\Psi \quad \Leftrightarrow \quad \exists\, n \geq 0 : \boldsymbol{f}_n(\pi) \models \Psi$$

It is easy to see that $\Box$ and $\Diamond$ can be expressed in $\mathcal{L}$ as follows:

$$\Box\Psi \quad \equiv \quad \wedge_{n\geq 0} \bigcirc^n \Psi$$
$$\Diamond\Psi \quad \equiv \quad \vee_{n\geq 0} \bigcirc^n \Psi$$

So Theorem 13 immediately implies the reduction lemmas in [4].

4.2 Branching-Time Logics

CTL* [5] is a finitary branching-time logic whose syntax in positive normal form is specified by the following grammar [6]:

$$\begin{array}{rcl} \Phi & ::= & P \mid \neg P \mid \mathbf{tt} \mid \mathbf{ff} \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid \forall(\Psi) \mid \exists(\Psi) \\ \Psi & ::= & \Phi \mid \Psi_1 \wedge \Psi_2 \mid \Psi_1 \vee \Psi_2 \mid \bigcirc\Psi \mid \Psi_1 \,\mathbf{U}\, \Psi_2 \mid \Psi_1 \,\mathbf{V}\, \Psi_2 \end{array}$$

where Φ and Φ_i's (respectively, Ψ and Ψ_i's) denote state (path) formulas, P denotes a primitive state formula, $\neg$, $\mathbf{tt}$, $\mathbf{ff}$, $\wedge$, $\vee$, $\forall$, $\exists$, and $\bigcirc$ have the same meanings as in $\mathcal{L}$, and $\mathbf{U}$ ("until") and $\mathbf{V}$ ("releases") are defined by:

$$\begin{array}{rcl} \pi \models \Psi_1 \,\mathbf{U}\, \Psi_2 & \Leftrightarrow & \exists\, n \geq 0 : \boldsymbol{f}_n(\pi) \models \Psi_2 \wedge \forall\, m < n : \boldsymbol{f}_m(\pi) \models \Psi_1 \\ \pi \models \Psi_1 \,\mathbf{V}\, \Psi_2 & \Leftrightarrow & \forall\, n \geq 0 : \boldsymbol{f}_n(\pi) \models \Psi_2 \vee \exists\, m < n : \boldsymbol{f}_m(\pi) \models \Psi_1 \end{array}$$

Let $\mathcal{P}_{\mathrm{CTL}^*}$ denote the set of primitive state formulas of CTL*; CTL* has no primitive path formula.

CTL* can be viewed as a fragment of $\mathcal{L}$, since $\mathbf{U}$ and $\mathbf{V}$ can be expressed in $\mathcal{L}$ as follows:

$$\begin{array}{rcl} \Psi_1 \,\mathbf{U}\, \Psi_2 & \equiv & \vee_{n \geq 0} \bigcirc^n \Psi_2 \wedge \wedge_{m < n} \bigcirc^m \Psi_1 \\ \Psi_1 \,\mathbf{V}\, \Psi_2 & \equiv & \wedge_{n \geq 0} \bigcirc^n \Psi_2 \vee \vee_{m < n} \bigcirc^m \Psi_1 \end{array}$$

where $\bigcirc^k$ denotes the k-fold iteration of $\bigcirc$: $\bigcirc^0 \Psi \triangleq \Psi$ and $\bigcirc^{k+1} \Psi \triangleq \bigcirc(\bigcirc^k \Psi)$ for all $k \geq 0$. Consequently, the preservation results for the fragments of $\mathcal{L}$ (Theorem 13) immediately imply the following preservation results for the fragments of CTL*, where $\forall$CTL*, $\exists$CTL*, ℓCTL*, ... are defined in terms of CTL* in the same way as $\forall\mathcal{L}$, $\exists\mathcal{L}$, $\ell\mathcal{L}$, ... are defined in terms of $\mathcal{L}$ (Definition 4).

Theorem 14 [Preservation Theorems for CTL*]

(34) R is a bisimulation $\wedge$ R bi-preserves $\mathcal{P}_{\mathrm{CTL}^*}$ $\Rightarrow$ R bi-preserves CTL*

(35) R is a bisimulation $\wedge$ R preserves $\mathcal{P}_{\mathrm{CTL}^*}$ $\Rightarrow$ R preserves CTL^*_+

(36) R is a simulation $\wedge$ R bi-preserves $\mathcal{P}_{\mathrm{CTL}^*}$ $\Rightarrow$ R preserves $\exists$CTL*

(37) R is a simulation $\wedge$ R preserves $\mathcal{P}_{\mathrm{CTL}^*}$ $\Rightarrow$ R preserves $\exists\mathrm{CTL}^*_+$

(38) R^{-1} is a simulation $\wedge$ R bi-preserves $\mathcal{P}_{\mathrm{CTL}^*}$ $\Rightarrow$ R preserves $\forall$CTL*

(39) R^{-1} is a simulation $\wedge$ R preserves $\mathcal{P}_{\mathrm{CTL}^*}$ $\Rightarrow$ R preserves $\forall\mathrm{CTL}^*_+$

(40) R bi-preserves $\mathcal{P}_{\mathrm{CTL}^*}$ $\Rightarrow$ R bi-preserves ℓCTL*

(41) R preserves $\mathcal{P}_{\mathrm{CTL}^*}$ $\Rightarrow$ R preserves $\ell\mathrm{CTL}^*_+$

Clarke, Grumberg, and Long [6] proved (34) and (38) in the special case that R is induced by an abstraction mapping h: $R(s_1, s_2) \Leftrightarrow (s_1 = h(s_2))$. It should be pointed out that, though [6] does not speak directly of simulation and bisimulation, the notions of approximation and exact approximation of transition systems in [6] imply the former notions.

4.3 Modal Logics

The Generalized Hennessy-Milner Logic (HML*) is an infinitary modal logic whose syntax in positive normal form is specified by the following grammar:

$$\Phi \quad ::= \quad P \mid \neg P \mid \wedge_{i \in I} \Phi_i \mid \vee_{i \in I} \Phi_i \mid [a]\,\Phi \mid \langle a \rangle\,\Phi$$

where Φ and Φ_i's denote state formulas, P denotes a primitive state formula, $\neg$, $\wedge_{i \in I}$, and $\vee_{i \in I}$ have the same meanings as in $\mathcal{L}$, and, for each action a, $[a]$ ("necessity") and $\langle a \rangle$ ("possibility") are defined by:

$$s \models [a]\,\Phi \quad \Leftrightarrow \quad \forall\, s' : s \xrightarrow{a} s' \;\Rightarrow\; s' \models \Phi$$
$$s \models \langle a \rangle\,\Phi \quad \Leftrightarrow \quad \exists\, s' : s \xrightarrow{a} s' \;\wedge\; s' \models \Phi$$

Note that HML* has no path formulas. Let $\mathcal{P}_{\text{HML}^*}$ denote the set of primitive formulas of HML*. The original Hennessy-Milner Logic [12] does not allow for primitive formulas; we have added them to obtain more general results which will be used in the next subsection.

Lemma 15 [Reduction Lemmas for Modal Operators]

(42) R is a simulation from T_1 to T_2 $\Rightarrow$
R preserves Φ $\Rightarrow$ R preserves $\langle a \rangle\,\Phi$

(43) R^{-1} is a simulation from T_2 to T_1 $\Rightarrow$
R preserves Φ $\Rightarrow$ R preserves $[a]\,\Phi$

(44) R is a bisimulation between T_1 and T_2 $\Rightarrow$
R bi-preserves Φ $\Rightarrow$ R bi-preserves $\langle a \rangle\,\Phi$

(45) R is a bisimulation between T_1 and T_2 $\Rightarrow$
R bi-preserves Φ $\Rightarrow$ R bi-preserves $[a]\,\Phi$

Proof: Similar to (in fact simpler than) the proofs of Lemmas 11 and 12. Alternatively, this lemma can be derived as a corollary of Lemmas 11 and 12 by noting that $[a]$ and $\langle a \rangle$ can be expressed in $\mathcal{L}$ as follows:

$$[a]\,\Phi \quad \equiv \quad \forall(\neg Q_a \vee \bigcirc \Phi)$$
$$\langle a \rangle\,\Phi \quad \equiv \quad \exists(Q_a \wedge \bigcirc \Phi)$$

where Q_a is a primitive path formula with the fixed interpretation that $V^{\Pi}(Q_a) = \{a\}$, *assuming* that the underlying labeled transition relation $\rightarrow$ is *total* in the sense that $\forall\, s \in S : \exists\, a, s' : s \xrightarrow{a} s'$. The last assumption can be satisfied by adding a new, dummy action ι and extending $\rightarrow$ such that $\forall\, s, s' \in S : s \xrightarrow{\iota} s' \Leftrightarrow (s = s')$ in the underlying LTS. ∎

In the following, $[\cdot]$HML* (respectively, $\langle\cdot\rangle$HML*) is the fragment of HML* obtained by banning the use of the $\langle a\rangle$ ($[a]$) operators, and HML$^*_+$, $[\cdot]$HML$^*_+$, and $\langle\cdot\rangle$HML$^*_+$ are the fragments of, respectively, HML*, $[\cdot]$HML*, and $\langle\cdot\rangle$HML* obtained by banning the use of $\neg$.

Theorem 16 [Preservation Theorems for HML*]

(46) R is a bisimulation $\wedge$ R bi-preserves $\mathcal{P}_{\text{HML}^*}$ $\Rightarrow$ R bi-preserves HML*

(47) R is a bisimulation $\wedge$ R preserves $\mathcal{P}_{\text{HML}^*}$ $\Rightarrow$ R preserves HML$^*_+$

(48) R is a simulation $\wedge$ R bi-preserves $\mathcal{P}_{\text{HML}^*}$ $\Rightarrow$ R preserves $\langle\cdot\rangle$HML*

(49) R is a simulation $\wedge$ R preserves $\mathcal{P}_{\text{HML}^*}$ $\Rightarrow$ R preserves $\langle\cdot\rangle$HML$^*_+$

(50) R^{-1} is a simulation $\wedge$ R bi-preserves $\mathcal{P}_{\text{HML}^*}$ $\Rightarrow$ R preserves $[\cdot]$HML*

(51) R^{-1} is a simulation $\wedge$ R preserves $\mathcal{P}_{\text{HML}^*}$ $\Rightarrow$ R preserves $[\cdot]$HML$^*_+$

Proof: Same as the proof of Theorem 13, except that Lemmas 11 and 12 are replaced by Lemma 15. ∎

Hennessy and Milner [12] proved (46) and (48) for the original Hennessy-Milner Logic (i.e., for the special case that $\mathcal{P}_{\text{HML}^*} = \emptyset$).

4.4 Fixpoint Logics

The Modal μ-Calculus (MμC) [14] is a finitary modal logic whose syntax in positive normal form is specified by the following grammar:

$$\begin{aligned}\Phi \;::=\;& P \mid \neg P \mid \mathbf{tt} \mid \mathbf{ff} \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid [a]\,\Phi \mid \langle a\rangle\,\Phi \mid \\ & X \mid \mu X : \Phi \mid \nu X : \Phi\end{aligned}$$

where Φ and Φ_i's denote state formulas, P denotes a primitive state formula, $\neg$, **tt**, **ff**, $\wedge$, $\vee$, $[a]$, and $\langle a\rangle$ have the same meanings as in HML*, X denotes a (propositional) *variable*, and μ and ν are the *least* and *greatest fixpoint operators*, both of which are variable binders (like quantifiers). Let $\mathcal{P}_{\text{M}\mu\text{C}}$ denote the set of primitive formulas and variables of MμC; note that free variables in a MμC formula can be viewed as primitive formulas.

MμC can be viewed as a fragment of HML*, since the least and greatest fixpoints can be expressed in HML* as follows [14]:

$$\begin{aligned} \mu X : \Phi &\equiv \Phi^{\mu}_{\lambda} \\ \nu X : \Phi &\equiv \Phi^{\nu}_{\lambda} \end{aligned}$$

where λ is a "big enough" ordinal (more about this later) and we define recursively for each ordinal α:

$$\begin{aligned} \Phi^{\mu}_{\alpha} &\triangleq \bigvee_{\beta<\alpha} \Phi[\Phi^{\mu}_{\beta}/X] \\ \Phi^{\nu}_{\alpha} &\triangleq \bigwedge_{\beta<\alpha} \Phi[\Phi^{\nu}_{\beta}/X] \end{aligned}$$

where $\Phi[\Theta/X]$ denotes the formula obtained by substituting Θ for all free occurrences of X in Φ. How "big" λ should be depends on the LTS's involved: if κ is the cardinality of the largest of the state sets of those LTS's, then letting $\lambda = 2^{\kappa}$ is certainly sufficient. Consequently, preservation results for the fragments of HML* (Theorem 16) immediately imply the following preservation results for the fragments of MμC, where $[\cdot]$MμC, $\langle\cdot\rangle$MμC, ... are defined in terms of MμC in the same way as $[\cdot]$HML*, $\langle\cdot\rangle$HML*, ... are defined in terms of HML*.

Theorem 17 [Preservation Theorems for MμC]

(52) R is a bisimulation $\wedge$ R bi-preserves $\mathcal{P}_{\mathrm{M}\mu\mathrm{C}}$ $\Rightarrow$ R bi-preserves MμC

(53) R is a bisimulation $\wedge$ R preserves $\mathcal{P}_{\mathrm{M}\mu\mathrm{C}}$ $\Rightarrow$ R preserves MμC$_+$

(54) R is a simulation $\wedge$ R bi-preserves $\mathcal{P}_{\mathrm{M}\mu\mathrm{C}}$ $\Rightarrow$ R preserves $\langle\cdot\rangle$MμC

(55) R is a simulation $\wedge$ R preserves $\mathcal{P}_{\mathrm{M}\mu\mathrm{C}}$ $\Rightarrow$ R preserves $\langle\cdot\rangle$MμC$_+$

(56) R^{-1} is a simulation $\wedge$ R bi-preserves $\mathcal{P}_{\mathrm{M}\mu\mathrm{C}}$ $\Rightarrow$ R preserves $[\cdot]$MμC

(57) R^{-1} is a simulation $\wedge$ R preserves $\mathcal{P}_{\mathrm{M}\mu\mathrm{C}}$ $\Rightarrow$ R preserves $[\cdot]$MμC$_+$

Bensalem, Bouajjani, Loiseaux, and Sifakis [2] proved essentially the same theorem using an algebraic formulation in terms of Galois connections. A *Galois connection* [19] between two posets P and Q is a pair of functions, $\phi : P \to Q$ and $\psi : Q \to P$, such that for any $p \in P$ and $q \in Q$:

(58) $\phi(p) \leq_Q q \quad \Leftrightarrow \quad p \leq_P \psi(q)$

It can be shown [7][2] that for any Galois connection (ϕ, ψ) between two power sets 2^X and 2^Y (ordered by set inclusion), there must exist a relation

[2]Note that the definition of Galois connections in [7] has the order $\leq_Q$ reversed.

$R \subseteq X \times Y$ such that for any $p \subseteq X$ and $q \subseteq Y$:

$$\begin{aligned} \phi(p) &= \{y \in Y \mid \exists\, x \in X : (x, y) \in R \wedge x \in p\} \\ \psi(q) &= \{x \in X \mid \forall\, y \in Y : (x, y) \in R \Rightarrow y \in q\} \end{aligned}$$

and either side of (58) is equivalent to:

(59) $\forall\, (x, y) \in R : x \in p \Rightarrow y \in q$

In other words, any Galois connection between two power sets (which is the only kind of Galois connections used in [2]) is induced by a relation between the base sets of the power sets. Essentially, we have used (59) as the basis of our definition of property preservation (cf. Definition 5). We feel that dispensing with Galois connections greatly simplified our treatment; the reader is invited to compare [2] with this paper. Also, [2] proves the preservation results only for finitely branching transition systems[3], while our proofs work for arbitrary labeled transition systems.

5 Conclusion

The power of our treatment comes mainly from the use of the infinitary logic $\mathcal{L}$, which is both simple enough to allow straightforward and modular preservation proofs (viz., the reduction lemmas) and expressive enough to support the unification and generalization of several existing preservation results. Indeed, all future-tense propositional logics surveyed in [8, 20] appear to be fragments of $\mathcal{L}$, so our main result (Theorem 13) is very general.

This work can be extended in at least two directions. First, in order to deal properly with reactive systems that hide internal details from external environments, external and internal actions need to be distinguished. Second, in order to support modular development of reactive systems, the interaction of property preservation and program composition needs to be investigated. (It should pointed out that works in this direction already exist; see [10, 11]). We hope to address these two issues in future work.

[3] It is stated in [2] (p. 266) that:

> In the sequel we consider only finite branching transition systems, i.e., transition systems where any state has a finite number of successors. This condition guaranties that the formulas can be interpreted as continuous functions on sets of states.

This assertion is actually false; see Example 4.3.2 of [20] (p. 533). Hence the proofs of Theorems 14 and 15 in [2] are problematic.

Acknowledgements.

The author is grateful to Professor Yiannis Moschovakis for enlightening discussions about infinitary logics, fixpoints, and ordinals.

References

[1] S. Abramsky, D.M. Gabbay, and T.S.E. Maibaum (Ed.), *Handbook of Logic in Computer Science, Vol. 2*, Oxford University Press, 1992.

[2] S. Bensalem, A. Bouajjani, C. Loiseaux, and J. Sifakis, "Property Preserving Simulations", in [3], pp. 260–273.

[3] G.v. Bochmann and D.K. Probst (Ed.), *CAV'92: Computer-Aided Verification, 4th International Workshop*, LNCS 663, Springer-Verlag, 1992.

[4] Ching-Tsun Chou, "Mechanical Verification of Distributed Algorithms in Higher-Order Logic", to appear in *The Computer Journal* in 1995.

[5] E.M. Clarke, E.A. Emerson, and A.P. Sistla, "Automatic Verification of Finite-State Concurrent Systems Using Temporal Logic Specifications", *ACM Trans. on Programming Languages and Systems*, Vol. 8, No. 2, pp. 244–263, Apr. 1986.

[6] E.M. Clarke, O. Grumberg, and D.E. Long, "Model Checking and Abstraction", *ACM Trans. on Programming Languages and Systems*, Vol. 16, No. 5, pp. 1512–1542, Sep. 1994.

[7] B.A. Davey and H.A. Priestley, *Introduction to Lattices and Order*, Cambridge University Press, 1990.

[8] E.A. Emerson, "Temporal and Modal Logic", Chapter 16 of [15], pp. 995–1072.

[9] M.-C. Gaudel and J.-P. Jouannaud (Ed.), *TAPSOFT'93: Theory and Practice of Software Development, 4th Int. Joint Conf. CAAP/FASE*, LNCS 668, Springer-Verlag, 1993.

[10] S. Graf and C. Loiseaux, "Property Preserving Abstractions under Parallel Composition", in [9], pp. 644–657.

[11] O. Grumberg, and D.E. Long, "Model Checking and Modular Verification", *ACM Trans. on Programming Languages and Systems*, Vol. 16, No. 3, pp. 843–871, May. 1994.

[12] M.C. Hennessy and R. Milner, "Algebraic Laws for Nondeterminism and Concurrency", *Journal of ACM*, Vol. 32, No. 1, pp. 137–161, 1985.

[13] Ph.G. Kolaitis and M.Y. Vardi, "Infinitary Logics and 0-1 Laws", *Information and Computation*, Vol. 98, pp. 258–294, 1992.

[14] Dexter Kozen, "Results on the Propositional μ-Calculus", *Theoretical Computer Science*, Vol. 27, pp. 333–354, 1983.

[15] J. van Leeuwen (Ed.), *Handbook of Theoretical Computer Science, Vol. B: Formal Models and Semantics*, The MIT Press/Elsevier, 1990.

[16] Robin Milner, "An Algebraic Definition of Simulation between Programs", *Proc. of the 2nd Int. Joint Conf. on Artificial Intelligence*, pp. 481–489, 1971.

[17] Robin Milner, *Communication and Concurrency*, Prentice Hall, 1989.

[18] Yiannis N. Moschovakis, *Notes on Set Theory*, Springer-Verlag, 1994.

[19] Oystein Ore, "Galois Connexions", *Trans. Amer. Math. Soc.*, Vol. 55, pp. 493–513, 1944.

[20] C. Stirling, "Modal and Temporal Logics", in [1], pp. 477–563.