The Expressive Power of Temporal Logic of Actions

ARKADY ESTRIN, diCarta Inc., 600 Allerton Street, Redwood City, CA 94063, USA.
E-mail: aestrin@dicarta.com

MICHAEL KAMINSKI, Department of Computer Science, Technion – Israel Institute of Technology, Haifa 32000, Israel.
E-mail: kaminski@cs.technion.ac.il

Abstract

It is shown that a stutter-invariant property is expressible in Temporal Logic of Actions if and only if it is expressible in Second-order Temporal Logic. In particular, validity questions can be translated from one logic to the other. The proof is based on equivalence transformations between the formulas of Temporal Logic of Actions and Second-order Temporal Logic. The translation from Second-order Temporal Logic into Temporal Logic of Actions is linear and the translation from Temporal Logic of Actions into Second-order Temporal Logic is quadratic.

Keywords: Temporal interpretations, invariance under stuttering.

1 Introduction

A concurrent algorithm is usually specified by a program and correctness of the algorithm means that the program satisfies some desired properties. In other words, in a standard program verification, given a model (as a program, etc.) and a specification (as a logical formula or an automaton), a formal verification is a proof that the model satisfies the specification. A number of methods for reasoning about concurrent programs and hardware devices are based on proof systems for temporal logic. A few years ago Lamport [7] suggested an approach in which both the algorithm and the property were specified by formulas in a single logic — Temporal Logic of Actions (TLA). In TLA correctness of an algorithm means that the formula specifying the algorithm implies the formula specifying the property, where ‘implies’ is an ordinary logical implication. That is, in TLA correctness of an algorithm means that the corresponding implication is semantically valid. Thus, correctness of an algorithm means validity of a single TLA formula.

The main feature of TLA is that satisfiability of its formulas is invariant under stuttering. This property, apart from being interesting in its own right, has applications in reasoning about hierarchically constructed concurrent systems [3] and model checking via partial order reduction [4]. Also, as argued in [7], TLA is very convenient for program specification, because it allows modularization and refinement (that is a natural feature of stuttering).

TLA combines two logics: a logic of actions and a standard temporal logic. An action is a formula without temporal operators containing variables, primed variables, and constant symbols. In general, an action represents a relation between the current state and the next state, where the unprimed variables refer to the current state and the primed ones refer to the
next state. For example, \( x' = y + 1 \), where \( x \) and \( y \) are variables, is an action stating that the value of \( x \) in the next state equals the value of \( y \) in the current state plus 1. Elementary formulas of TLA are those not containing primed variables or temporal operators and formulas of the form \( [A]_t \), where \( A \) is an action. The formula \( [A]_t \) states that either \( A \) holds between the current and the next states or the value of the term \( t \) does not change when passing to the next state. In this way stuttering steps which leave all variables unchanged are allowed. General TLA formulas are obtained from the elementary ones using, under certain restrictions, boolean connectives, TLA quantification, and the unary temporal operator \( \square \) (always).

Whereas propositional versions of TLA and stuttering have been studied in the literature [2, 9–12] nothing is known about the original Lamport’s TLA. Thus, its relationship to known temporal logics is of interest.

In addition to allowing modularization and refinement, TLA was invented to reduce the expressive power of temporal logic. As Lamport pointed out in [7], TLA is less powerful than temporal logic, because TLA can only express invariant under stuttering formulas, whereas temporal logic can express non-invariant ones as well.

In this paper we show that, to some extent, TLA and Second-order Temporal Logic (SOTL) can be simulated by each other. Namely, we present an equivalence translation of TLA formulas into SOTL formulas\(^1\) and an equivalence translation of SOTL formulas whose satisfiability is invariant under stuttering into TLA formulas, cf. [9]. In particular, loosely speaking, we show that SOTL can be embedded into TLA ‘modulo stuttering’. As an immediate corollary we obtain an equivalence translation of SOTL formulas without free flexible variables into TLA formulas.

Of course, there is no equivalence embedding of SOTL into TLA, because sets of temporal interpretations definable in TLA are closed under stuttering, whereas sets of temporal interpretations definable in SOTL are not. Thus, in a sense, our embedding is ‘the best possible interpretation of SOTL in TLA’.

Contrary to Lamport’s claim in [7], our translations are relatively simple: the translation from SOTL into TLA is linear and the translation from TLA into SOTL is quadratic. In particular, as an immediate corollary to an equivalence translation from TLA into SOTL presented in this paper, we obtain a quadratic time translation of the TLA validity questions into the SOTL validity. Also, the proofs of the correctness of the translations explicitly show the values of hidden variables in the corresponding temporal interpretations, which gives a new insight into TLA by establishing a relationship between structures with the same ‘unstuttered behavior’, cf. [3].

Finally, it should be pointed out that a translation from SOTL into TLA is quite surprising in view of the expressive power of SOTL that is as strong as Second-order Arithmetic.

The paper is organized as follows. In the next section we recall the notion of temporal interpretation. In Section 3 we introduce an ‘intermediate’ Extended Raw Temporal Logic of Actions (ERTLA) needed for intertranslations between TLA and SOTL and define the syntax and semantics of TLA. In Section 4 we recall the syntax and semantics of SOTL and present embeddings of SOTL and ERTLA into each other. These embeddings are standard and are presented for the sake of completeness only. In Sections 5 and 6 we present the translations of TLA into ERTLA and of ERTLA into TLA, respectively. The embeddings of SOTL and TLA

---

\(1\) Even though the syntaxes of TLA and SOTL are different, both logics are interpreted over the same structures. Thus, if we consider a logic whose syntax contains both syntaxes of TLA and SOTL, we can talk about the usual semantical equivalence. That is, a temporal interpretation satisfies a TLA/SOTL formula if and only if it satisfies its translation.
duced in [7] and is used as preliminary steps in the definition of TLA. Also, we shall use Temporal Logic of Actions (RTLA) and Simple Temporal Logic of Actions (STLA) into ERTLA. ERTLA is a second-order temporal logic that contains as fragments both the Raw First we introduce a new temporal logic called Extended Raw Temporal Logic of Actions 3 Temporal Logics of Actions
2 Temporal interpretations
Semantics of all temporal logics considered in this paper is based on the notion of temporal interpretations defined below. First we recall the definition of interpretations of the ordinary first-order logic that is referred to as Predicate Calculus (PC).
A PC interpretation (or just an interpretation) \( M \) consists of a non-empty domain \( D_M \), an assignment to each \( n \)-place function symbol \( f \) of the underlying language of an \( n \)-place function \( f^M : D^n_M \to D_M \) (we treat the language constants as 0-place function symbols), and an assignment to each \( n \)-place predicate symbol \( P \) of the underlying language of an \( n \)-place relation \( P^M \) on \( D_M \), such that \( =^M \) is the identity relation on \( D_M \). That is, we assume that interpretations are normal, see [8, p. 78] for details.
Temporal logics considered in this paper have two types of variables: one — ordinary variables whose (temporal) interpretation is flexible, i.e. may have different values in different states. The latter will be referred to as flexible variables and will be denoted by bold-face letters \( \mathbf{x}, \mathbf{y}, \mathbf{z} \), etc.
A temporal interpretation consists of a PC interpretation \( M \), an assignment to a finite set of rigid variables of elements of \( D_M \), and an assignment to a finite set of flexible variables of functions from \( \mathbb{N} \) into \( D_M \), where \( \mathbb{N} \) denotes the set of non-negative integers. The finite set of variables with assigned values will always be a set of all free variables (rigid and flexible) of the formula under consideration. We denote the set of all functions from \( \mathbb{N} \) into \( D_M \) by \( D^n_M \).
Let \( d \in D^n_M \) be an assignment to a flexible variable \( \mathbf{x} \). Then, for a non-negative integer \( i \), \( d(i) \) is the ‘value’ of \( \mathbf{x} \) in the \( i \)th state (or in the \( i \)th moment of time). That is, (interpretations of) flexible variables may change in time.
One can think of a temporal interpretation as of a sequence of states \( s_0, s_1, \ldots \) which are PC interpretations over the same domain and differ only in assignments to flexible variables in a given formula. Namely, for an assignment \( d \) to a flexible variable \( \mathbf{x} \), \( d(i) \) is the assignment to \( \mathbf{x} \) in \( s_i \).
Finally, for a function \( d \in D^n_M \) and a non-negative integer \( i \) we denote by \( d^{+i} \) the element of \( D^n_M \) such that \( d^{+i}(j) = d(j + i), j = 0, 1, \ldots \). That is, if \( d \) is ‘a behavior of a flexible variable from time 0’, then \( d^{+i} \) is ‘a behavior of that variable from time \( i \)’. In other words, if the assignments \( d_1, \ldots, d_n \) correspond to the sequence of states \( s_0, s_1, \ldots \), then the assignments \( d_1^{+i}, \ldots, d_n^{+i} \) correspond to the sequence of states \( s_i, s_{i+1}, \ldots \).
3 Temporal Logics of Actions
First we introduce a new temporal logic called Extended Raw Temporal Logic of Actions (ERTLA). ERTLA is a second-order temporal logic that contains as fragments both the Raw Temporal Logic of Actions (RTLA) and Simple Temporal Logic of Actions (STLA) introduced in [7] and is used as preliminary steps in the definition of TLA. Also, we shall use

---

2 In fact, Lamport in [7] assumes that the interpretation domain is always infinite.
ERTLA as an intermediate station between the intertranslations of SOTL and TLA. Namely, we first translate SOTL and TLA into ERTLA and then we translate ERTLA into TLA and SOTL, respectively.

The language of ERTLA is obtained from the language of PC by

- dividing its variables into two sorts: one — ordinary variables, whose temporal interpretation is rigid, and the other — variables whose temporal interpretation is flexible. The latter are referred to as flexible variables and are denoted by bold-face letters \( x, y, z, \) etc.
- adding to the language of PC a unary function symbol \( ' — \) prime and a unary temporal operator \( □ — \) always. As usual, we use \( ◯ \) as an abbreviation for \( □¬\).

The formula formation rules of ERTLA are the ordinary ones with the exception that \( ' \) can be applied to unprimed flexible variables only: the meaning of \( x' \) is the value of \( x \) in the next state.

Next we define the semantics of ERTLA.

Let \( t(\bar{x}, \bar{y}) \) be a term whose rigid and flexible free variables are among \( \bar{x} = x_1, \ldots, x_m \) and \( \bar{y} = y_1, \ldots, y_n \), respectively, and let \( M \) be a temporal interpretation. Let \( \bar{d} = d_1, \ldots, d_m \in D_M \) and \( \bar{d} = d_1, \ldots, d_n \in D_M^n \). We define the value of \( t(\bar{x}, \bar{y}) \) at \( \bar{d}, \bar{d} \) in the \( i \)th state of \( M \), denoted \( t^M,i(\bar{d}, \bar{d}) \), by induction, as follows:

- if \( t \) is the rigid variable \( x_k \), then \( t^M,i(\bar{d}, \bar{d}) = d_k \);
- if \( t \) is the flexible variable \( x_j \), then \( t^M,i(\bar{d}, \bar{d}) = d_j(i) \);
- if \( t \) is of the form \( x'_j \), then \( t^M,i(\bar{d}, \bar{d}) = d_j(i+1) \); and
- if \( t \) is of the form \( f(t_1, \ldots, t_k) \), where \( f \) is a \( k \)-place function symbol, then \( t^M,i(\bar{d}, \bar{d}) = f^M(t_1^M,i(\bar{d}, \bar{d}), \ldots, t_k^M,i(\bar{d}, \bar{d})) \).

Recall that we treat constants as 0-place function symbols.

That is, \( t^M,i(\bar{d}, \bar{d}) \) is the value of \( t(\bar{x}, \bar{y}) \) under the assignment of \( d_j \) to \( x_j, j = 1, \ldots, m \) and \( d_k \) to \( x_k, k = 1, \ldots, n \).

**Definition 3.1**

Let \( \varphi(\bar{x}, \bar{y}) \) be an ERTLA formula whose free rigid and flexible variables are among \( \bar{x} = x_1, \ldots, x_m \) and \( \bar{y} = y_1, \ldots, y_n \), respectively, and let \( M \) be a PC interpretation. Let \( \bar{d} = d_1, \ldots, d_m \in D_M \) and \( \bar{d} = d_1, \ldots, d_n \in D_M^n \). We say that \( M \) satisfies \( \varphi(\bar{x}, \bar{y}) \) at \( \bar{d}, \bar{d} \), denoted \( M |= \varphi(\bar{d}, \bar{d}) \), if the following holds:

- if \( \varphi \) is an atomic formula \( P(t_1(\bar{x}, \bar{y}), \ldots, t_k(\bar{x}, \bar{y})) \), then \( M |= \varphi \) if and only if \( (t_1^M,i(\bar{d}, \bar{d}), \ldots, t_k^M,i(\bar{d}, \bar{d})) \in P^M \);
- \( M |= \neg \varphi(\bar{d}, \bar{d}) \) if and only if \( M \not|= \varphi(\bar{d}, \bar{d}) \);
- \( M |= \varphi(\bar{d}, \bar{d}) \Rightarrow \psi(\bar{d}, \bar{d}) \) if and only if \( M |= \varphi(\bar{d}, \bar{d}) \) implies \( M |= \psi(\bar{d}, \bar{d}) \);
- \( M |= \exists x \varphi(x, \bar{d}, \bar{d}) \) if and only if for some \( d \in D_M, M |= \varphi(d, \bar{d}, \bar{d}) \);
- \( M |= \exists x \varphi(\bar{d}, x, \bar{d}) \) if and only if for some \( d \in D_M^n, M |= \varphi(\bar{d}, d, \bar{d}) \); and
- \( M |= □ \varphi(\bar{d}, \bar{d}) \) if and only if for each \( i = 0, 1, \ldots, M |= \varphi(\bar{d}, \bar{d}^{-i}) \), where \( \bar{d}^{-i} = d_1^{i+1}, \ldots, d_n^{i+1} \).

That is, \( M \) satisfies \( \varphi(\bar{x}, \bar{y}) \) under the assignment of \( d_j \) to \( x_j, j = 1, \ldots, m \) and \( d_k \) to \( x_k, k = 1, \ldots, n \), or, in other words, the temporal interpretation \( (M, \bar{d}, \bar{d}) \) satisfies \( \varphi(\bar{x}, \bar{y}) \).

A formula \( \varphi(\bar{x}, \bar{y}) \) is satisfiable (valid, denoted \( |= \varphi(\bar{x}, \bar{y}) \)), if for some (any) PC interpretation \( M \) and some (any) \( \bar{d} \in D_M^m \) and \( \bar{d} \in (D_M^n)^n \), \( M |= \varphi(\bar{d}, \bar{d}) \).
DEFINITION 3.2
The Raw Temporal Logic of Actions (RTLA) is the fragment of ERTLA consisting of all ERTLA formulas without quantified flexible variables which do not contain $\Box$ in the scope of (rigid variable) quantifiers.

RTLA formulas not containing $\Box$ are called actions. Thus, an action can be thought of as a relation between two consecutive states of a temporal interpretation.

Next we define the Simple Temporal Logic of Actions (STLA) that will be extended to TLA by allowing, under certain restrictions, quantification over both rigid and flexible variables outside of $\Box$. We shall need the following notation. For an expression $E$ not containing $'$, we denote by $E'$ the expression that is obtained from $E$ by replacing all occurrences of every flexible variable $x$ in $E$ with $x'$.

Let $A$ be an action and $t$ be a term not containing $'$. We shall denote formulas $A \lor t = t'$ and $A \land t \neq t'$ by $[A]_t$ and $[A]_t$, respectively.

Now we define STLA as the fragment of RTLA (see Definition 3.2) where each action $A$ containing $'$ appears in the form $\Box [A]$. Since $\neg \Box [A]$ is equivalent to $\Diamond \langle A \rangle$, the latter is an STLA formula as well. We refer the reader to [7] for a comprehensive study of STLA.

Finally, for two tuples of terms $(u_1, u_2, \ldots, u_n)$ and $(v_1, v_2, \ldots, v_n)$, denoted $\pi$ and $\tau$, respectively, we abbreviate the formula $\bigwedge_{i=1}^n u_i = v_i$ by $\pi = \tau$. It can be easily verified that $\Box [A]_\pi$ is equivalent to $\bigwedge_{i=1}^n \Box [A]_{u_i}$. Thus, both $\Box [A]_\pi$ and $\Diamond \langle A \rangle_\pi$ are STLA formulas.

As has been pointed out in [7], satisfiability of an STLA formula by a temporal interpretation is not affected if we add to (or remove from) it stuttering steps. In other words, satisfiability of STLA formulas is invariant under stuttering. To be more specific, we need the following notation.

Let $M$ be a PC interpretation and let $\varphi(\pi, \pi')$ be an ERTLA formula whose free rigid and flexible variables are among $\pi = x_1, \ldots, x_n$, and $\pi' = x_1, \ldots, x_n$. Let $d = d_1, \ldots, d_n \in D_M$ and $c = c_1, \ldots, c_n \in D^n_M$. For a non-negative integer $i \in \mathbb{N}$ we denote by $\mu(i)$ the maximal integer $j \geq i$ such that for all $i \leq k \leq j$, $c(k) = c(i)$, where $\overrightarrow{d}(k) = a_1(k), \ldots, a_n(k)$. That is, $\mu(i)$ is the maximal integer $j$ such that all the computation steps between $i$ and $j$ are stuttering. Note that if $\overrightarrow{d}(k) = \overrightarrow{c}(i)$ for all $k \geq i$, i.e. the computation is halted, then $\mu(i)$ is undefined.

Next we define a function $\zeta : \mathbb{N} \to \mathbb{N}$ by

\[
\zeta(0) = \begin{cases} 
\mu(0) & \text{if } \mu(0) \text{ is defined}, \\
0 & \text{otherwise}
\end{cases}
\]

and

\[
\zeta(i + 1) = \begin{cases} 
\mu(\zeta(i) + 1) & \text{if } \mu(\zeta(i) + 1) \text{ is defined}, \\
\zeta(i) + 1 & \text{otherwise}
\end{cases}
\]

Finally, for each $d = d_1, \ldots, d_n$ we define a function $\overrightarrow{d} \in D^n_M$ by $\overrightarrow{d}(\zeta(i)) = d(\zeta(i))$, $i \in \mathbb{N}$, and we define $\overrightarrow{d}$ by $\overrightarrow{d} = \overrightarrow{d_1}, \ldots, \overrightarrow{d_n}$.

By definition, $\overrightarrow{d}$ is obtained from $\overrightarrow{c}$ by removing all stuttering steps (except for the halted part of the computation), i.e. all states $s_i$ such that no flexible variable changes its value when passing to state $s_{i+1}$, or, in other words, by removing all states $s_i$ such that $\overrightarrow{d}(i) = \overrightarrow{d}(i + 1)$.

\[\text{Note that the semantics of TLA flexible quantifiers differs from the ERTLA (usual) semantics of those quantifiers.}\]
Consider a TLA-like formula.

In the visual form, the assignment to denoted of Lamport [7, Equation (48)].

It can be readily seen that the above definition of ‘unstuttering’ is equivalent to the definition of Lamport [7, Equation (48)].

The precise statement of invariance under stuttering is as follows.

**Definition 3.3**

Let \( \varphi(\overline{x}, \overline{x}) \) be an ERTLA formula whose free rigid and flexible variables are among \( \overline{x} = x_1, \ldots, x_m \), and \( \overline{x} = x_1, \ldots, x_n \), respectively. We say that satisfiability of a formula \( \varphi \) is **invariant under stuttering** if for all PC interpretations \( M \), all \( \overline{a} = a_1, \ldots, a_n \in D^M_{\overline{a}} \), all \( \overline{a} = a_1, \ldots, a_n \in D^M_{\overline{a}} \) and all \( \overline{b} = b_1, \ldots, b_n \in D^M_{\overline{b}} \) such that \( \mathcal{M} \models \varphi(\overline{a}, \overline{x}) \) if and only if \( M \models \varphi(\overline{a}, \overline{x}) \).

Now we are about to define the syntax and semantics of TLA. TLA formulas are built from STLA ones by the means of propositional connectives and rigid and non-standard flexible variable quantifiers. We precede Example 3.5 with the following definition.

**Definition 3.4**

An ERTLA formula \( \varphi \) is called **TLA-like** if

- \( \varphi \) is an STLA formula;
- \( \varphi \) is of the form \( \neg \psi \) or \( \psi \supset \theta \), where \( \psi \) and \( \theta \) are TLA-like formulas;
- \( \varphi \) is of the form \( \exists \psi \) or \( \forall \psi \), where \( \psi \) is a TLA-like formula.

The TLA formulas will be obtained from the TLA-like ones by replacing each appearance of \( \exists \) applied to a flexible variable with \( \exists_l \), where \( \exists_{x} \) with \( \exists_{l,x} \), where \( \exists_{l} \) is Lamport’s TLA flexible variable quantifier, see Definition 3.8 later in this section.

**Example 3.5**

Consider a TLA-like formula

\[
\exists ! chg \ (chg = 0 \land \\
\square [x \neq x' \supset \ y = y'] \land \\
\square [x \neq x' \supset \ chg' = 1] \land \\
\square [y \neq y' \supset (chg = 1 \land chg' = 0)] ,
\]

denoted \( F(x, y) \). This formula states that \( x \) changes between every two changes of \( y \) (in particular, before the first change of \( y \)) and that \( x \) and \( y \) never change simultaneously. Namely, the flexible hidden variable \( chg \) is a flag indicating whether \( y \) may change. The conjuncts \( chg = 0 \) and \( \square [y \neq y' \supset (chg = 1 \land chg' = 0)] \) imply that \( y \) may not change in the initial state, the conjunct \( \square [x \neq x' \equiv y = y'] \) states that \( x \) and \( y \) may not change simultaneously, the conjunct \( \square [x \neq x' \supset chg' = 1] \) states that after (i.e. in the next state to each change of \( x \), the flag \( chg \) becomes 1 (consequently, \( y \) may change), and the conjunct \( \square [y \neq y' \supset (chg = 1 \land chg' = 0)] \) states that when \( y \) changes, the flag \( chg \) is 1 and after that it becomes 0 (consequently, \( y \) may not change one more time before \( x \) does).

Let \( M \) be a PC interpretation such that there are two distinct elements \( a \) and \( b \) in \( D^M \) and let the assignment \( b \in D^M_{\overline{b}} \) to \( y \) be defined by

\[
b(i) = \begin{cases} 
a & i = 0, 1 
b & i = 2, 3, \ldots 
\end{cases}
\]

In the visual form, the assignment to \( y \) is as follows.
Then \( M \models \exists x F(x, b) \), because \( M \models F(a, b) \), where

\[
a(i) = \begin{cases} 
  a & i = 0 \\
  b & i = 1, 2, \ldots
\end{cases}
\]

Let \( d \in D^N_M \) be defined by

\[
d(i) = \begin{cases} 
  a & i = 0 \\
  b & i = 1, 2, \ldots
\end{cases}
\]

That is, \( d = a \). Obviously, \( \exists b = \exists d \). However, \( M \not\models \exists x F(x, d) \), because \( d \) already changes in the initial state. Thus, satisfiability of \( \exists x F(x, y) \) is not invariant under stuttering.

Proposition 3.6 below reflects the intuitive expectation that invariance under stuttering depends only on the flexible variable quantifiers.

**Proposition 3.6**

If satisfiability of formulas \( \varphi \) and \( \psi \) is invariant under stuttering, then so is that of \( \neg \varphi \), \( \varphi \supset \psi \), \( \exists x \varphi \), and \( \Box \varphi \).

The proof of Proposition 3.6 follows from Definition 3.3 by a straightforward induction on the formula complexity and is left to the reader.

To obtain invariance under stuttering Lamport in [7] introduced a new semantics of (TLA) flexible variable quantifiers. In this paper, to distinguish between the TLA and the (ordinary) ERTLA quantifiers, we shall refer to the Lamport TLA quantifiers by \( \exists_L \) and \( \forall_L \). To define the semantics of \( \exists_L \), we extend Definition 3.1 with the following clause.

\[
M \models \exists_L x \varphi(\bar{d}, x, \bar{d}) \text{ if and only if there exist } c_1, \ldots, c_n \in D^N_M \text{ such that } \bar{c} = \bar{d} \text{ and for some } d \in D^N_M, M \models \varphi(\bar{d}, d, \bar{c}).
\]

**Remark 3.7**

Obviously, \( \exists x \varphi \supset \exists_L x \varphi \) is a valid formula (over the ERTLA language extended with \( \exists_L \)).

At last we have arrived at the definition of TLA.

**Definition 3.8**

The language of TLA results from the language of ERTLA by replacing the flexible variable quantifier \( \exists \) with \( \exists_L \) and TLA formulas are defined, by induction, as follows.

\[
\begin{align*}
\text{• Each STLA formula is an atomic TLA formula;} \\
\text{• if } \varphi \text{ and } \psi \text{ are TLA formulas then } \neg \varphi \text{ and } \varphi \supset \psi \text{ are TLA formulas;} \text{ and} \\
\text{• if } \varphi \text{ is a TLA formula, } x \text{ is a rigid variable and } x \text{ is a flexible variable, then } \exists x \varphi \text{ and } \exists_L x \varphi \text{ are TLA formulas.}
\end{align*}
\]
This example deals with the ERTLA formula

$$\exists c \text{chg}( \text{chg} = 0 \land$$

$$\square [x \neq x' \equiv y = y']_{(\text{chg}, x, y)} \land$$

$$\square [x \neq x' \supset \text{chg}' = 1]_x \land$$

$$\square [y \neq y' \supset (\text{chg} = 1 \land \text{chg}' = 0)]_y,$$

denoted $F_1(x, y)$. This formula is obtained from formula $F(x, y)$ in Example 3.5 by replacing the unique appearance of the flexible variable quantifier $\exists$ with $\exists_L$. Let $M, a, b,$ and $d$ be as in Example 3.5. Then $M \models \exists_L x F_1(x, d)$, because $\exists d = \exists b$ and $M \models F(a, b)$. In general, satisfiability of TLA formulas is invariant under stuttering, see [7].

We conclude this section with two additional examples which illustrate the difference between the semantics of $\exists$ (the ordinary semantics) and $\exists_L$ (Lamport’s semantics).

**Example 3.10**
Consider a TLA formula

$$\exists c \text{chg}\exists c, n(\text{chg} = 0 \land n = 1 \land$$

$$\square [x \neq x' \equiv y = y']_{(\text{chg}, n, x, y)} \land$$

$$\square [x \neq x' \supset \text{chg}' = \text{chg} + 1 \land n' = n]_x \land$$

$$\square [y \neq y' \supset (\text{chg} \geq n \land \text{chg}' = 0 \land n' = n + 1)]_y).$$

This formula refines formula $F_1(x, y)$ from the previous example (Example 3.9) by stating that between the $(n - 1)\text{th}$ and the $n\text{th}$ changes of $y$, $x$ must change at least $n$ times. Thus, there is no upper bound on the number of stuttering states to be added between successive states of a temporal interpretation to check whether it satisfies a TLA formula containing quantified flexible variables.

**Example 3.11**
This example deals with the ERTLA formula $\exists x \forall y \square [x \neq x']_y$ stating that some $x$ changes at least as fast as any other $y$, i.e. in each computation step where $y$ changes, $x$ changes as well. This formula is valid. Indeed, let $M$ be a temporal interpretation. If $D_M$ consists of only one element, obviously, $M \models \exists x \forall y [x \neq x']_y$. If there are two distinct elements $a$ and $b$ in $D_M$, then for $d \in D_M$ defined by

$$d(i) = \begin{cases} a & i \text{ is even} \\ b & i \text{ is odd} \end{cases}$$

we have $M \models \forall y[d \neq d']_y$, and $M \models \exists x \forall y [x \neq x']_y$ follows from the definition of $\models$.

On the other hand, the negation $\forall x \exists y \square [x = x']_y$ of $\exists x \forall y [x \neq x']_y$ (that is the TLA counterpart of $\exists x \forall y \square [x \neq x']_y$) is satisfied by all temporal interpretations whose domain contains two or more distinct elements. That is, $\exists x \forall y \square [x \neq x']_y$ is ‘almost a TLA-contradiction’. Indeed, let $M$ be a temporal interpretation such that there are two distinct elements $a$ and $b$ in $D_M$. It suffices to show that for any $a \in D_M$ there is a 'stretching' $b$ of $a$, i.e., $\exists b = \exists a$, such that, in some moment, some $d \in D_M$ changes before $b$. We define $b$ by ‘doubling the first state of $(M, a)$’, i.e.

$$b(i) = \begin{cases} a(0) & i = 0 \\ a(i - 1) & i = 1, 2, \ldots \end{cases}$$
and define \( d \) by
\[
d(i) = \begin{cases} 
  a & i = 0 \\
  b & i = 1, 2, \ldots 
\end{cases}
\]
Then \( M \models b = b' \land d \neq d' \), i.e. \( M \models \langle b = b'_d \rangle \). Therefore, by the definition of \( \models \), \( M \models \Diamond (b = b')_d \). Hence, \( M \models \forall x \exists y (x = x'_y) \) follows.

In Section 5 we show that TLA can be simulated by ERTLA and the combination of the translations in Sections 4 and 6 shows that the fragment of ERTLA consisting of all formulas whose satisfiability is invariant under stuttering can be simulated by TLA.

### 4 Second-order Temporal Logic

In this section we recall syntax and semantics of Second-order Temporal Logic (SOTL) and embed SOTL and ERTLA into each other.

The language of SOTL is obtained from the language of PC by
- dividing its variables in two sorts: one — ordinary variables, whose temporal interpretation is rigid, and the other — variables whose temporal interpretation is flexible.
- adding to the language of PC two unary temporal operators \( \square – \text{next} \) and \( \Box – \text{always} \), and one binary temporal operator \( \mathcal{U} – \text{until} \).

To define the semantics of \( \square \) and \( \mathcal{U} \) we extend Definition 3.1 with the following two clauses.
- \( M \models \square \varphi(d, \bar{d}) \) if and only if \( M \models \varphi(d, \bar{d}^{+1}) \); and
- \( M \models \varphi(d, \bar{d}) \mathcal{U} \psi(d, \bar{d}) \) if and only if there is an \( i \geq 0 \) such that \( M \models \psi(d, \bar{d}^{+i}) \) and for each \( j = 0, 1, \ldots, i - 1, M \models \varphi(d, \bar{d}^{+j}) \).

Next we present an equivalence translation of ERTLA formulas into SOTL formulas.

**Remark 4.1**

Even though the syntaxes of ERTLA and SOTL are different, both logics are interpreted over the same structures. Thus, if we consider a logic whose syntax contains both syntaxes of ERTLA and SOTL, we can talk about the usual semantical equivalence, cf. footnote 1.

Let \( y_1, \ldots, y_\ell \) be all primed flexible variables that appear in an ERTLA formula \( \varphi \). For each \( j = 1, \ldots, \ell \) we pick a ‘new’ rigid variable, denoted \( y_j^\sigma \) and for a term \( t \) that appears in \( \varphi \) we denote by \( t^\sigma \) the result of simultaneous replacement of each \( y_j^\sigma \) in \( t \) with \( y_j^\sigma \), \( j = 1, \ldots, \ell \). Now we define the translation of \( \varphi \) into SOTL, denoted \( \varphi^{\text{SOTL}} \), by induction as follows.
- If \( \varphi \) is an atomic formula \( P(t_1, \ldots, t_k) \), then \( \varphi^{\text{SOTL}} \) is
  \[
  \forall y_1^\sigma \cdots \forall y_\ell^\sigma ((\bigwedge_{j=1}^{\ell} \square(y_j^\sigma = y_j^\sigma)) \supset P(t_1^\sigma, \ldots, t_k^\sigma)).
  \]
- \( (\neg \varphi)^{\text{SOTL}}, (\varphi \supset \psi)^{\text{SOTL}}, (\exists x \varphi)^{\text{SOTL}}, (\exists x \varphi)^{\text{SOTL}}, \) and \( (\Box \varphi)^{\text{SOTL}}, (\Box \varphi)^{\text{SOTL}} \) are \( \neg \varphi^{\text{SOTL}}, \varphi^{\text{SOTL}} \supset \psi^{\text{SOTL}}, \exists x \varphi^{\text{SOTL}}, \exists x \varphi^{\text{SOTL}}, \) and \( \Box \varphi^{\text{SOTL}}, \) respectively.
Theorem 4.2
Let \( \varphi(\overline{\mathbf{x}}, \overline{\mathbf{y}}, \overline{\mathbf{y}}) \) be an ERTLA formula all whose free rigid, flexible, and primed flexible variables are among \( \overline{\mathbf{x}} = x_1, \ldots, x_m, \overline{\mathbf{y}} = x_1, \ldots, x_n, \) and \( \overline{\mathbf{y}} = y_1, \ldots, y_k, \) respectively, and let \( M \) be a PC interpretation. Then for all assignments \( \overline{d} = d_1, \ldots, d_m \in D_M, \overline{d} = d_1, \ldots, d_m \in D^R_M, \) and \( \overline{p} = p_1, \ldots, p_\ell \in D^F_M, \) to \( \overline{\mathbf{x}}, \overline{\mathbf{y}}, \) and \( \overline{\mathbf{y}}, \) respectively, \( M \models \varphi(\overline{d}, \overline{d}, \overline{p}) \) if and only if \( M \models \varphi^{opt}(\overline{d}, \overline{d}, \overline{p}). \)

The corollary to Theorem 4.2 below immediately follows from the definition of validity.

Corollary 4.3
\[ \models \forall x_1 \ldots \forall x_m \forall x_n \forall y_1 \ldots \forall y_\ell (\varphi(\overline{\mathbf{x}}, \overline{\mathbf{y}}, \overline{\mathbf{y}}) \equiv \varphi^{opt}(\overline{\mathbf{x}}, \overline{\mathbf{y}}, \overline{\mathbf{y}})). \]

Proof. [Theorem 4.2] The proof is by induction on the complexity of \( \varphi. \)

Basis: Let \( \varphi \) be an atomic formula \( P(t_1, \ldots, t_k). \) First we observe the following. Let \( t(\overline{\mathbf{x}}, \overline{\mathbf{y}}, \overline{\mathbf{y}}) \) be a term that appears in \( P, \) let \( i \geq 0, \) and let \( \overline{p} = p_1, \ldots, p_\ell \in D^F_M \) be such that \( M \models t_{M,i}(\overline{d}, \overline{d}, \overline{p}) \) for all \( j = 0 \) to \( \ell. \) Then

\[ t_{M,i}(\overline{d}, \overline{d}, \overline{p}) = t_{M,i}(\overline{d}, \overline{d}, \overline{p}). \] (4.1)

Indeed, by definition, \( O(p_j, \overline{p}) \) is equivalent to \( p_j^{i+1} = p_j, j = 1, \ldots, \ell, \) and the proof of (4.1) follows from the definition of function \( t_{M,i} \) by a straightforward induction on the complexity of \( t(\overline{\mathbf{x}}, \overline{\mathbf{y}}, \overline{\mathbf{y}}). \) We leave it to the reader.

Now, let \( M \models P(t(\overline{d}, \overline{d}, \overline{p}), \ldots, t_k(\overline{d}, \overline{d}, \overline{p})), \) i.e.

\[ (t_{M,0}(\overline{d}, \overline{d}, \overline{p}), \ldots, t_{M,0}(\overline{d}, \overline{d}, \overline{p})) \in P^M. \] (4.2)

We shall prove that

\[ M \models \forall \overline{y}^\circ \ldots \forall \overline{y}^\circ ((\bigwedge_{j=1}^\ell (\bigwedge_{j=1}^\ell (O(p_j = y_j) \supset P(t_1(\overline{d}, \overline{d}, \overline{y})) \ldots, t_k(\overline{d}, \overline{d}, \overline{y}))). \]

So, let \( \overline{p} = p_1, \ldots, p_\ell \in D^F_M \) be such that \( M \models O(p_j = p_j), j = 1, \ldots, \ell. \) We have to show that \( M \models P(t_1(\overline{d}, \overline{d}, \overline{p}), \ldots, t_k(\overline{d}, \overline{d}, \overline{p})), \) i.e. by definition,

\[ (t_{M,0}(\overline{d}, \overline{d}, \overline{p}), \ldots, t_{M,0}(\overline{d}, \overline{d}, \overline{p})) \in P^M. \]

Since, by (4.1) with \( i = 0, t_{M,0}(\overline{d}, \overline{d}, \overline{p}) = t_{M,0}(\overline{d}, \overline{d}, \overline{p}), j = 1, \ldots, k, \) the desired containment follows from (4.2).

Conversely, let

\[ M \models \forall \overline{y}^\circ \ldots \forall \overline{y}^\circ ((\bigwedge_{j=1}^\ell (O(p_j = y_j) \supset P(t_1(\overline{d}, \overline{d}, \overline{y})) \ldots, t_k(\overline{d}, \overline{d}, \overline{y}))) \]

and let \( \overline{p} = p_1, \ldots, p_\ell \in D^F_M \) be such that \( M \models O(p_j = p_j), j = 1, \ldots, \ell. \) Then \( M \models P(t_1(\overline{d}, \overline{d}, \overline{p}), \ldots, t_k(\overline{d}, \overline{d}, \overline{p})), \) which, by the definition of \( \models, \) means

\[ (t_{M,0}(\overline{d}, \overline{d}, \overline{p}), \ldots, t_{M,0}(\overline{d}, \overline{d}, \overline{p})) \in P^M. \]
Since, by (4.1) with $i = 0$, $t_j^M(\mathcal{I}, \mathcal{A}, \mathcal{P}) = t_j^{M,0}(\mathcal{I}, \mathcal{A}, \mathcal{P})$, $j = 1, \ldots, k$,

$$(t_1^{M,0}(\mathcal{I}, \mathcal{A}, \mathcal{P}), \ldots, t_k^{M,0}(\mathcal{I}, \mathcal{A}, \mathcal{P})) \in P^M$$

as well. That is, $M$ satisfies $P(t_1(\mathcal{I}, \mathcal{A}, \mathcal{P}), \ldots, t_k(\mathcal{I}, \mathcal{A}, \mathcal{P}))$.

The proof of the induction step immediately follows from the definition of $|-$ and the induction hypothesis and is left to the reader.

Next we embed SOTL into ERTLA. The embedding is based on the notion of time variable $t$ needed for translations of both SOTL and TLA into ERTLA. The time variable $t$ refers to the ERTLA time. That is, if (the assignment to) $t$ equals to (the assignment to) $u$ in a given state, then the time in that state is (the assignment to) $u$. The definition is similar to that of the time predicate in [6]: the desired properties of $t$ are reflected by the formula below

$$\forall x \Box[t \neq t]_x \land \forall x \forall y \Box((t = x \land \diamond (t = y \land \diamond (t = x))) \supset x = y),$$

denoted $T(t)$, cf. formula $\exists x \forall y \Box[x \neq x']_y$ in Example 3.11.

Note that $T(t)$ is a TLA-like formula, see Definition 3.4 in Section 3. Its first conjunct states that $t$ changes in each state and the second conjunct of $T(t)$ states that the values of $t$ in different states are different. One might think of replacing conjunct $\forall x \Box[t \neq t]_x$ with a simpler formula $\Box([t \neq t]_x)$, but, for combining the translation of SOTL into ERTLA with that of ERTLA into TLA we need SOTL formulas to be translated into TLA-like ones.

Next we define a binary relation $<$ on the values of $t$ — the ‘time order’ — by $x < y$ if and only if

$$x \neq y \land \diamond (T(x) \land \diamond T(y)).$$

That is, $x < y$ if and only if both $x$ and $y$ are values of $t$ in some (different) moments and value $x$ appears before value $y$. Thus, $<$ is isomorphic to the order of $\mathbb{N}$. Finally, we define a binary relation $\text{NEXT}$ that relates two consecutive moments of time by $\text{NEXT}(x, y)$ if and only if

$$\Box[t = x \supset t' = y]_t.$$

That is, $\text{NEXT}(x, y)$ asserts that $y$ follows $x$ in the order imposed by $<$. Again, both $x < y$ and $\text{NEXT}(x, y)$ are TLA-like formulas.

Note that the ‘time axiom’ $T(t)$ cannot be satisfied by PC interpretations over finite domains. Thus, in the rest of this section and in Section 5 we assume that all temporal interpretations under consideration are over infinite domains and in Section 7 we show how finite domains can be simulated by infinite ones.

Now, for a SOTL formula $\varphi$, a rigid variable $u$ — the present moment of ERTLA — that does not appear in $\varphi$, and a flexible variable $t$ — the time of ERTLA — we define an ERTLA formula $\mathcal{F}(\varphi, t, u)$ whose intended meaning is that $\varphi$ holds in the moment $u$ of the time $t$. The formula $\mathcal{F}(\varphi, t, u)$ is defined by induction as follows.

- If $\varphi$ is an atomic formula $P(t_1, \ldots, t_k)$, then $\mathcal{F}(\varphi, t, u)$ is
  $$\Box(t = u \supset P(t_1, \ldots, t_k));$$
- $\mathcal{F}(\neg \varphi, t, u)$ is $\neg \mathcal{F}(\varphi, t, u);$ 
- $\mathcal{F}(\varphi \supset \psi, t, u)$ is $\mathcal{F}(\varphi, t, u) \supset \mathcal{F}(\psi, t, u);$ 

4 In fact, a more involved construction than that we use in this paper shows that for embedding SOTL into ERTLA ‘the infinite domain assumption’ is not required, but it seems to be crucial for embedding TLA into ERTLA.
The Expressive Power of Temporal Logic of Actions

- \( \mathcal{F}(\exists x\varphi, t, u) \) is \( \exists x\mathcal{F}(\varphi, t, u) \);
- \( \mathcal{F}(\exists x\varphi, t, u) \equiv \exists x\mathcal{F}(\varphi, t, u) \);
- \( \mathcal{F}(\bigvee \varphi, t, u) \equiv \forall v(\text{NEXT}(u, v) \supset \mathcal{F}(\varphi, t, v)) \);
- \( \mathcal{F}(\bigvee \varphi, t, u) \equiv \forall v(v \geq u \supset \mathcal{F}(\varphi, t, v)) \);
- \( \mathcal{F}(\bigvee \varphi, t, u) \equiv \exists v((v \geq u) \land \mathcal{F}(\psi, t, v) \land \forall w(u \leq w < v \supset \mathcal{F}(\varphi, t, w))) \).

**Theorem 4.4**

Let \( \varphi(x, \overline{x}) \) be a SOTL formula all whose free rigid and flexible variables are among \( x = x_1, \ldots, x_m \) and \( \overline{x} = \overline{x}_1, \ldots, \overline{x}_n \), respectively, and let \( M \) be a PC interpretation. Then for all assignments \( \overline{d} = d_1, \ldots, d_m \in D_M \) and \( \overline{d} = d_1, \ldots, d_n \in D_M \) and \( \overline{x}, \overline{\overline{x}} \) respectively, \( M \models \varphi(\overline{d}, \overline{d}) \) if and only if \( M \models \exists \exists u(\text{NEXT}(u) \land (t = u) \land \mathcal{F}(\varphi(\overline{d}, \overline{d}), t, u)) \).

**Corollary 4.5**

\( \models \forall \overline{x}_1 \ldots \forall \overline{x}_m \forall \overline{x}_1 \ldots \forall \overline{x}_n(\varphi(\overline{x}, \overline{\overline{x}}) \equiv \exists t \exists u(\text{NEXT}(u) \land (t = u) \land \mathcal{F}(\varphi(\overline{d}, \overline{d}), t, u)). \)

**Remark 4.6**

Note that, since \( T(t), u \leq v \), and \( \text{NEXT}(u, v) \) are TLA-like formulas, \( T(t) \land (t = u) \land \mathcal{F}(\varphi, t, u) \) is a TLA-like formula as well.

**Proof.** [Theorem 4.4] We start with the proof of the 'only if' part of the theorem. Since \( D_M \) is infinite, there exists a sequence \( \tau_0, \tau_1, \ldots \) of elements of \( D_M \) such that \( M \models \text{NEXT}(\tau_i, \tau_{i+1}) \), \( i = 0, 1, \ldots \). Let \( \tau \in D_M \) be defined by \( \tau(i) = \tau_i, i = 0, 1, \ldots \). It suffices to show that \( M \models \varphi(\overline{d}, \overline{d}) \) if and only if \( M \models T(\tau) \land (\tau = \tau_0) \land \mathcal{F}(\varphi(\overline{d}, \overline{d}), \tau, \tau_0). \)

Since, by definition, \( M \models T(\tau) \land (\tau = \tau_0) \), we have to prove that \( M \models \varphi(\overline{d}, \overline{d}) \) if and only if \( M \models \mathcal{F}(\varphi(\overline{d}, \overline{d}), \tau, \tau_0) \). We shall prove by induction on the complexity of \( \varphi \) that \( M \models \varphi(\overline{d}, \overline{d})^{i+1} \) if and only if \( M \models \mathcal{F}(\varphi(\overline{d}, \overline{d}), \tau, \tau_i) \), \( i = 0, 1, \ldots \).

**Basis:** If \( \varphi(\overline{x}, \overline{\overline{x}}) \) is an atomic formula, then \( \mathcal{F}(\varphi(\overline{d}, \overline{d}), \tau, \tau_i) \equiv \square(\tau = \tau_i \supset \varphi(\overline{d}, \overline{d})). \) If \( M \models \square(\tau = \tau_i \supset \varphi(\overline{d}, \overline{d})) \), then, in particular, \( M \models \tau^{i+1} = \tau_{i+1} \supset \varphi(\overline{d}, \overline{d})^{i+1} \). Since, by definition, \( \tau(i) = \tau_i \), we obtain \( M \models \varphi(\overline{d}, \overline{d})^{i+1} \).

Conversely, let \( M \models \varphi(\overline{d}, \overline{d})^{i+1} \). Since, by the definition of \( \tau \), \( M \models \tau(j) = \tau_i \) if and only if \( j = i \), we obtain \( M \models \square(\tau = \tau_i \supset \varphi(\overline{d}, \overline{d})). \)

**Induction step:** The cases of the logical connectives \( \text{and} \) and \( \supset \) and the quantifiers over rigid and flexible variables immediately follow from the definition of the translation and the induction hypothesis.

Let \( \varphi \) be of the form \( \bigvee \psi \). Then \( \mathcal{F}(\varphi(\overline{d}, \overline{d}), t, u) \equiv \forall v(\text{NEXT}(u, v) \supset \mathcal{F}(\psi(\overline{d}, \overline{d}), t, v)) \) and

- \( M \models \bigvee \psi(\overline{d}, \overline{d})^{i+1} \) if and only if \( M \models \psi(\overline{d}, \overline{d})^{i+1} \) if and only if \( M \models \mathcal{F}(\psi(\overline{d}, \overline{d}), \tau, \tau_{i+1}) \) if and only if \( M \models \forall v(\text{NEXT}(\tau_{i+1}, v) \supset \mathcal{F}(\psi(\overline{d}, \overline{d}), t, v)) \),

where the first equivalence follows from the definition of semantics of \( \bigvee \), the second equivalence follows from the induction hypothesis, and the last equivalence follows from the observation that, by the definitions of \( \text{NEXT} \) and \( \tau \), \( M \models \text{NEXT}(\tau_i, \tau) \) if and only if \( \tau = \tau_{i+1} \).

Let \( \varphi \) be of the form \( \square \psi \). Then \( \mathcal{F}(\varphi(\overline{d}, \overline{d}), t, u) \equiv \forall v(v \geq u \supset \mathcal{F}(\psi(\overline{d}, \overline{d}), t, v)) \) and
Suppose, for the sake of argument, that there is an STLA formula into (the TLA-like fragment of) ERTLA. The idea lying behind our translation is illustrated and let the induction step.

(\exists u ((v \geq u) \land \mathcal{F}(\theta(\overline{\alpha}, \overline{\beta}), t, v) \land \forall w (u \leq w < v \supset \mathcal{F}(\psi(\overline{\alpha}, \overline{\beta}), t, w)))

Finally, let \( \varphi \) be of the form \( \psi(\theta) \). Then \( \mathcal{F}(\psi(\overline{\alpha}, \overline{\beta}), \theta(\overline{\alpha}, \overline{\beta}), t, u) \) is

\[
M \models \exists u ((v \geq u) \land \mathcal{F}(\theta(\overline{\alpha}, \overline{\beta}), t, v) \land \forall w (u \leq w < v \supset \mathcal{F}(\psi(\overline{\alpha}, \overline{\beta}), t, w))
\]

and

\[
M \models \psi(\overline{\alpha}, \overline{\alpha}^{i+1}) \land \theta(\overline{\alpha}, \overline{\beta})^{i+1} \land \theta(\overline{\alpha}, \overline{\beta})^{i}
\]

\[
\begin{align*}
M &\models \psi(\overline{\alpha}, \overline{\alpha}^{i}) \land \theta(\overline{\alpha}, \overline{\beta})^{i}, & \text{if and only if} \\
M &\models \psi(\overline{\alpha}, \overline{\alpha}^{k}), & \text{if and only if}
\end{align*}
\]

where the first equivalence follows from the definition of semantics of \( \mathcal{U} \), the second equivalence follows from the induction hypothesis, and the last equivalence follows from the observations that, by the definition of \( \tau \), \( M \models \tau \geq \tau_i \) if and only if \( \tau = \tau_j, j = i, i + 1, \ldots \). This proves the induction step.

Now we pass to the proof of the ‘if’ part of the theorem. Let \( \phi(\overline{\alpha}, \overline{\beta}) \) be a SOTL formula and let \( M \) be a temporal interpretation such that for some assignment \( \overline{\alpha}, \overline{\beta}, \tau \) to \( \overline{\alpha}, \overline{\beta}, \tau \), and \( u \), respectively, \( M \models \mathcal{T}(\tau) \land (\tau = \tau) \land \mathcal{F}(\psi(\overline{\alpha}, \overline{\beta}), \tau, \tau) \). In particular, \( \tau = \tau_0, \tau_1, \ldots \), where \( \tau_0 = \tau \) and \( M \models \mathcal{N}(\tau_i, \tau_{i+1}), i = 0, 1, \ldots \). Thus, \( M \models \phi(\overline{\alpha}, \overline{\beta}) \) follows from the proof of the ‘only if’ part of the theorem.

5 Embedding TLA into ERTLA

In this section, using the time variable, we present an equivalence translation of TLA formulas into (the TLA-like fragment of) ERTLA. The idea lying behind our translation is illustrated by Example 5.1 below.

Example 5.1

Suppose, for the sake of argument, that there is an STLA formula \( \mathcal{F}(\overline{x}, \overline{y}) \) stating that always \( \overline{x} \) changes before \( \overline{y} \) and \( \overline{x} \) and \( \overline{y} \) never change simultaneously.\(^5\)

Let \( \overline{M} \) and \( \overline{d} \) be as in Example 3.5. That is, there are two distinct elements \( a \) and \( b \) in \( D_{\overline{M}} \) and \( \overline{d} \) is defined by

\[
\overline{d}(i) = \begin{cases} 
  a & i = 0 \\
  b & i = 1, 2, \ldots
\end{cases}
\]

\(^5\)Alternatively, the reader can modify the example for formula \( F(\overline{x}, \overline{y}) \) from Example 3.5.
Then, exactly like in Example 3.5, one can show that $M \models \exists x \bar{F}(x, d)$, but $M \not\models \exists x \bar{F}(x, d)$. To simulate $\exists x \bar{F}(x, d)$ by an ERTLA formula we first replace the free flexible variable $y$ in $\exists x \bar{F}(x, y)$ with a new free flexible variable $\mathbf{u}_y$ that behaves like $y$ after adding/removing stuttering steps, and then find an assignment to $x$ such that $M$ satisfies $\bar{F}(x, \mathbf{u}_y)$ under that assignment.

For example, we can assign to $\mathbf{u}_y$ the function $b \in \mathbb{D}^\mathbb{N}_M$ defined by

$$b(i) = \begin{cases} a & i = 0, 1 \\ b & i = 2, 3, \ldots \end{cases}$$

and assign to $x$ the function $\mathbf{a} \in \mathbb{D}^\mathbb{N}_M$ defined by

$$\mathbf{a}(i) = \begin{cases} a & i = 0 \\ b & i = 1, 2, \ldots \end{cases}$$

More specifically, the free variable $\mathbf{u}_y$ is introduced in three stages.

1. First, we ‘add to $(M, d)$’ a new time variable $t$ (whose assignment is $\tau$). The intuitive meaning of $t$ is that it reflects the fastest time. That is, $\tau$ changes in each state.
2. Then we add an additional time variable $t_{\mathbf{a}}$ (whose assignment is $\tau_{\mathbf{a}}$). The intuitive meaning of $t_{\mathbf{a}}$ is that it simulates time after adding/removing stuttering steps to/from $(M, d, \tau)$.
3. Finally, we introduce $\mathbf{u}_y$ (whose assignment is $b$) so that it respects the new time $t_{\mathbf{a}}$. That is, $\mathbf{u}_y$ may change only if $t_{\mathbf{a}}$ changes.

After all that we choose an appropriate assignment $\mathbf{a}$ to $\mathbf{x}$ in $M$. For example, this can result in the following sequence of states (temporal interpretation) that satisfies $\bar{F}(x, \mathbf{u}_y)$:

$$\langle x = a, y = a, t = \tau_0, t_{\mathbf{a}} = \tau_0, \mathbf{u}_y = a \rangle$$
$$\langle x = b, y = b, t = \tau_1, t_{\mathbf{a}} = \tau_0, \mathbf{u}_y = a \rangle$$
$$\langle x = b, y = b, t = \tau_2, t_{\mathbf{a}} = \tau_1, \mathbf{u}_y = b \rangle$$

In view of Example 5.1 above, we shall need a ‘stretched’ time variable $t_{\mathbf{a}}$ in order to simulate $\exists x \bar{F}(x, \mathbf{u}_y)$ in ERTLA. It is introduced by the formula

$$t = t_{\mathbf{a}} \land \forall x \forall y (x \leq y) \equiv \bigcirc(t_{\mathbf{a}} = x \land \bigcirc(t_{\mathbf{a}} = y)),$$

denoted $S(t, t_{\mathbf{a}})$, whose meaning is that $t_{\mathbf{a}}$ simulates time $t$ after adding or removing stuttering steps. Namely, $S(t, t_{\mathbf{a}})$ will always appear in conjunction with $T(t)$. In this case, it states that $t_{\mathbf{a}}$ refers to time, ‘modulo adding/removing stuttering steps’. Thus, the whole range of $t_{\mathbf{a}}$ coincides with that of $t$, even though the latter may change faster than the former.

Now we define the desired translation of TLA formulas. For a TLA formula $\varphi$ and a new flexible variable $t$ (the fastest time variable) we define a (TLA-like) formula $\mathcal{G}(\varphi, t)$, by induction, as follows.

- If $\varphi$ is an STLA formula, then $\mathcal{G}(\varphi, t)$ is $\varphi$ itself;
- $\mathcal{G}(\neg \varphi, t)$ is $\neg \mathcal{G}(\varphi, t)$;
- $\mathcal{G}(\varphi \lor \psi, t)$ is $\mathcal{G}(\varphi, t) \lor \mathcal{G}(\psi, t)$;
The Expressive Power of Temporal Logic of Actions 853

- $G(\exists x \varphi, t)$ is $\exists x G(\varphi, t)$; and
- finally, let all of free flexible variables of $\varphi$ be among $x, \overline{x}$, where $\overline{x} = x_1, \ldots, x_n$.

Then $G(\exists x \varphi(x, x_1, \ldots, x_n), t)$ is

$$
\forall t \forall x \bigwedge_{i=1}^{n} (\varphi(t = t \land x_i = x) \equiv \square(t = t \supset u_i = x))\wedge
G(\varphi(x, u_1, \ldots, u_n), t).
$$

In the last clause of the definition of $G$, the quantifier part $\exists t \forall \exists u_1 \ldots \exists u_n$ reflects adding or removing stuttering steps to the original temporal interpretation and $\exists u$ states that there is an appropriate assignment to $u$ in the temporal interpretation ‘after adding or removing stuttering steps’; the conjunct $S(t, t_m)$ assures that (subject to satisfiability of $T(t)$) $t_m$ simulates $t$ in the temporal interpretation whose time is $t_m$ after adding or removing stuttering steps; the conjunct

$$
\forall t \forall x \bigwedge_{i=1}^{n} (\varphi(t = t \land u_i = x) \equiv \square(t = t \supset u_i = x))
$$

assures that the new flexible variables $u_i$ simulate $x_i$ in the temporal interpretation after adding or removing stuttering steps; and the last conjunct $G(\varphi(x, u_1, \ldots, u_n), t)$ is the recursive call of the translation of $\varphi(x, u_1, \ldots, u_n)$. We shall prove that a TLA formula $\varphi$ is satisfiable (by a temporal interpretation over an infinite domain) if and only if formula $T(t) \wedge G(\varphi, t)$ is satisfiable. We precede the proof of the above equivalence with Example 5.2 below that continues Example 5.1.

**EXAMPLE 5.2**

Let $\overline{F}(x, y), M, a, b, d, \tau$, and $\tau_m$ be as in Example 5.1. Obviously, $M \models T(\tau)$ and

$$
M \models S(\tau, \tau_m) \wedge \\
\forall t \forall x (\varphi(t = t \land d = x) \equiv \square(\tau_m = t \supset b = x)) \wedge \\
\overline{F}(a, b).
$$

Thus, $M \models T(\tau) \wedge G(\exists x \overline{F}(x, d), \tau)$.

**THEOREM 5.3**

Let $\varphi(\overline{x}, \overline{x})$ be a TLA formula all whose free rigid and flexible variables are among $\overline{x} = x_1, \ldots, x_n$ and $\overline{\overline{x}} = x_1, \ldots, x_n$, respectively, and let $M$ be a PC interpretation. Then for all assignments $\overline{d} = d_1, \ldots, d_\mathcal{M} \in D_M$ and $\overline{\overline{d}} = d_1, \ldots, d_\mathcal{M} \in D_M^{\overline{\overline{x}}}$ to $\overline{x}$ and $\overline{x}$, respectively, $M \models \varphi(\overline{d}, \overline{\overline{d}})$ if and only if $M \models \exists t(T(t) \wedge G(\varphi(\overline{d}, \overline{\overline{d}}), t))$.

**COROLLARY 5.4**

$$
\models \forall x_1 \ldots \forall x_n \forall x_1 \ldots \forall x_n (\varphi(\overline{x}, \overline{x}) \equiv \exists t(T(t) \wedge G(\varphi(\overline{d}, \overline{\overline{d}}), t)).
$$

**PROOF.** [Theorem 5.3] Since $D_M$ is infinite, there exists a sequence $\tau_0, \tau_1, \ldots$ of elements of $D_M$ such that $M \models \text{NEXT}(\tau_i) = \tau_{i+1}, i = 0, 1, \ldots$. Let $\tau_i \in D_M^{\overline{x}}$ be defined by $\tau(i) = \tau_i, i = 0, 1, \ldots$. It suffices to show that $M \models \varphi(\overline{d}, \overline{\overline{d}})$ if and only if $M \models T(\tau) \wedge G(\varphi(\overline{d}, \overline{\overline{d}}), \tau)$. Since, by definition, $M \models T(\tau)$, we have to prove that $M \models \varphi(\overline{d}, \overline{\overline{d}})$ if and only if $M \models G(\varphi(\overline{d}, \overline{\overline{d}}), \tau)$. The proof is by induction on the complexity of $\varphi$. If $\varphi$ is an STL formula, then $G(\varphi, t)$ is $\varphi$ itself and the induction basis follows.

For the induction step, the cases of of the propositional connectives $\neg$ and $\supset$ and the rigid variable quantifier immediately follow from the definition and the induction hypothesis. So,
In this section we embed a fragment of ERTLA into TLA. The fragment under consideration

Let \( \psi(\overline{x}, \overline{d}) \) be of the form \( \exists \overline{x} \psi(\overline{x}, \overline{d}) \) and assume that \( M \models \exists \overline{x} \psi(\overline{d}, \overline{x}, \overline{d}) \). That is, there exist \( \tau_\alpha \in D^*_M \) and \( \overline{d}_\alpha = d_{\alpha,1}, \ldots, d_{\alpha,n} \in D^*_M \) such that

\[
\xi(\tau_\alpha, \overline{d}_\alpha) = \xi(\tau, \overline{d})(= (\tau, \overline{d}))
\]

and for some \( c \in D^*_M \), \( M \models \psi(\overline{d}, c, \overline{d}_\alpha) \), because we can introduce \( t \) into the formula by the neutral conjunct \( t = t \), say. Therefore, by the induction hypothesis,

\[
M \models \mathcal{G}(\overline{d}, c, \overline{d}_\alpha), \tau).
\]

Also (5.1) together with \( M \models T(\tau) \) imply

\[
M \models S(\tau, \tau_\alpha)
\]

and

\[
M \models \forall t \forall x \bigwedge_{i=1}^{n} (\Diamond (\tau = t \land d_i = x) \equiv \Box (\tau_\alpha = t \lor d_{\alpha,i} = x)).
\]

Thus, it follows from (5.3), (5.4), and (5.2) that

\[
M \models S(\tau, \tau_\alpha) \land \\
\forall t \forall x \bigwedge_{i=1}^{n} (\Diamond (\tau = t \land d_i = x) \equiv \Box (\tau_\alpha = t \lor d_{\alpha,i} = x)) \land \\
\mathcal{G}(\overline{d}, c, \overline{d}_\alpha), \tau)
\]

that, in turn, implies

\[
M \models \exists \overline{x}_0 \exists u_1 \ldots \exists u_n \exists x (S(\tau, t_\alpha) \land \\
\forall t \forall x \bigwedge_{i=1}^{n} (\Diamond (\tau = t \land d_i = x) \equiv \Box (t_\alpha = t \lor u_i = x)) \land \\
\mathcal{G}(\psi(\overline{d}, \overline{d}, \overline{x}, u_0, \ldots, u_n), \tau)).
\]

Therefore, by the definition of \( \mathcal{G} \), \( M \models \mathcal{G}(\exists \overline{x}_0 \psi(\overline{d}, \overline{x}, \overline{d}), \tau) \). That is, \( M \models \mathcal{G}(\psi(\overline{d}, \overline{d}), \tau) \).

Reading the above proof 'backward', shows that \( M \models \mathcal{G}(\exists \overline{x}_0 \psi(\overline{d}, \overline{x}, \overline{d}), \tau) \) implies \( M \models \exists \overline{x}_0 \psi(\overline{d}, \overline{x}, \overline{d}) \), i.e. the converse implication of the \( \exists \overline{x}_0 \) case of the induction step. ■

6 Embedding a fragment of ERTLA into TLA

In this section we embed a fragment of ERTLA into TLA. The fragment under consideration consists of all TLA-like formulas, see Definition 3.4 in Section 3.

At first glance it might appear that the TLA-like fragment of ERTLA is very restrictive. In fact, it is not restrictive at all, because, as shown in Section 4, SOTL formulas are simulated in ERTLA by TLA-like formulas. Also, combining the embeddings of ERTLA into SOTL and SOTL into ERTLA in Section 4 with the embedding of the TLA-like fragment of ERTLA into TLA presented in this section, we obtain an embedding of ERTLA into TLA.

The embedding of the TLA-like fragment of ERTLA into TLA is based on the notion of state variable \( s \) that is intended to indicate the change of the system state. Namely, let \( \phi \) be an ERTLA formula, \( \overline{x} \) be a free flexible variable of \( \phi \), and let \( s_0, s_1, \ldots \) be a sequence of states of a temporal interpretation. If in some \( s_i \) the value of (the assignment to) \( \overline{x} \) differs from the value of (the assignment to) \( \overline{x} \) in state \( s_{i+1} \), then (the assignment to) state variable \( s \) in
state \( s_i \) differs from that in state \( s_{i+1} \). In particular, the time variable introduced in Section 4 can serve as a state variable, but we shall use a weaker definition that does not presume the infinite domain requirement.

To define state variable \( s \) we shall need the following notation. We denote the set of all free flexible variables of \( \varphi \) by \( \mathbf{V}_\varphi \). Now, for an ERTLA formula \( \varphi \) we define the 'state variable axiom of \( \varphi \)', denoted \( S_\varphi(s) \), as

\[
\Box[s \neq s']_{\mathbf{V}_\varphi}.
\]

Note that \( S_\varphi \) is an STLA formula stating that \( s \) changes at least as fast as any of the free flexible variables of \( \varphi \).

The translation \( \mathcal{H}(\varphi, s) \) of a TLA-like formula \( \varphi \) is defined, by induction, as follows.

- If \( \varphi \) is an STLA formula, then \( \mathcal{H}(\varphi, s) \) is \( \varphi \) itself;
- \( \mathcal{H}(\neg \varphi, s) \) is \( \neg \mathcal{H}(\varphi, s) \);
- \( \mathcal{H}(\varphi \lor \psi, s) \) is \( \mathcal{H}(\varphi, s) \lor \mathcal{H}(\psi, s) \);
- \( \mathcal{H}(\exists x \varphi, s) \) is \( \exists x \mathcal{H}(\varphi, s) \); and
- \( \mathcal{H}(\exists x \varphi, s) \) is \( \exists x(\Diamond[s \neq s']_x \land \mathcal{H}(\varphi, s)) \).

The relativization of the Lamport existential quantifier over the flexible variable \( x \) to \( \Diamond[s \neq s']_x \) in the last clause of the definition of \( \mathcal{H} \) is intended to neutralize the influence of adding or removing stuttering steps before choosing an appropriate assignment to \( x \) when passing to satisfiability by PC interpretations satisfying \( S_\varphi(s) \). Namely, the assignment to \( x \) cannot change faster than the assignment to \( s \) that does not change in the added stuttering steps.

The relationship between formulas \( \varphi \) and \( \mathcal{H}(\varphi, s) \) is as follows.

**Theorem 6.1**

Let \( \varphi(x, \mathbf{x}) \) be a TLA-like formula all whose free rigid and flexible variables are among \( \mathbf{x} = x_1, \ldots, x_m \) and \( \mathbf{x} = x_1, \ldots, x_n \), respectively, let \( M \) be a PC interpretation, and let \( \mathbf{d} = d_1, \ldots, d_m \in D_M \) and \( \mathbf{d} = d_1, \ldots, d_n \in D_M^n \) be assignments to \( x \) and \( \mathbf{x} \), respectively.

(a) If \( M \models \varphi(\mathbf{d}, \mathbf{d}) \), then \( M \models \exists s(S_\varphi(\mathbf{d}, \mathbf{d})(s) \land \mathcal{H}(\varphi(\mathbf{d}, \mathbf{d}), s)) \).

(b) If \( M \models \exists s(S_\varphi(\mathbf{d}, \mathbf{d})(s) \land \mathcal{H}(\varphi(\mathbf{d}, \mathbf{d}), s)) \), then for some \( \mathbf{e} \) such that \( \mathbf{e} = \mathbf{d} \), \( M \models \varphi(\mathbf{d}, \mathbf{e}) \).

**Corollary 6.2**

Let \( \varphi(x, \mathbf{x}) \) be a TLA-like formula whose satisfiability is invariant under stuttering and all whose rigid and flexible variables are among \( \mathbf{x} = x_1, \ldots, x_m \) and \( \mathbf{x} = x_1, \ldots, x_n \), respectively. Then

\[
\models \forall x_1 \ldots \forall x_m \forall x_1 \ldots \forall x_m (\varphi(x, \mathbf{x}) \equiv \exists s(S_\varphi(\mathbf{x}, \mathbf{x})(s) \land \mathcal{H}(\varphi(x, \mathbf{x}), s))).
\]

**Proof.** Let \( M \) be a temporal interpretation, and let \( \mathbf{d} = d_1, \ldots, d_m \in D_M \) and \( \mathbf{d} = d_1, \ldots, d_n \in D_M^n \) be assignments to \( x \) and \( \mathbf{x} \), respectively. We have to show that \( M \models \varphi(\mathbf{d}, \mathbf{d}) \) if and only if \( M \models \exists s(S_\varphi(\mathbf{d}, \mathbf{d})(s) \land \mathcal{H}(\varphi(\mathbf{d}, \mathbf{d}), s)) \). The 'only if' part of the corollary follows from part (a) of Theorem 6.1. For the proof of the 'if' part, let \( M \models \exists s(S_\varphi(\mathbf{d}, \mathbf{d})(s) \land \mathcal{H}(\varphi(\mathbf{d}, \mathbf{d}), s)) \). Then, by part (b) of Theorem 6.1, for some \( \mathbf{e} \) such that \( \mathbf{e} = \mathbf{d} \), \( M \models \varphi(\mathbf{d}, \mathbf{e}) \). Since satisfiability of \( \varphi(x, \mathbf{x}) \) is invariant under stuttering, \( M \models \varphi(\mathbf{d}, \mathbf{d}) \) as well. \( \blacksquare \)
COROLLARY 6.3
Let \( \varphi(x) \) be a TLA-like formula not containing a free flexible variable all whose free rigid variables are among \( x_1, \ldots, x_m \). Then \( \models \forall x_1 \ldots \forall x_m(\varphi(x)) \iff \exists s_0, s_1(S_{\varphi(x)}(s_0) \land H(\varphi(x), s)) \).

PROOF. By definition (Definition 3.3), satisfiability of formulas without free flexible variables is invariant under stuttering. Thus, the result follows from Corollary 6.2.

Theorem 6.1 is, in a sense, ‘the best possible interpretation of ERTLA in TLA’, because sets of temporal interpretations definable in TLA are closed under stuttering, whereas sets of temporal interpretations definable in SOTL are not. We precede the proof of Theorem 6.1 with an example that illustrates the influence of the conjunct \( \square[s \neq s']_\omega \) on the assignment to \( z \) in temporal interpretations which satisfy the formula translation.

EXAMPLE 6.4
Let \( \tilde{F}(x, y) \), \( M \), and \( d \) be as in Example 5.1. That is, \( \tilde{F}(x, y) \) states that always \( x \) changes before \( y \) and \( x \) and \( y \) never change simultaneously, there are two distinct elements \( a \) and \( b \) in \( D_M \), and \( d \) is defined by

\[
d(i) = \begin{cases} 
a & i = 0 
b & i = 1, 2, \ldots 
\end{cases}
\]

Then \( M \models \not\exists x \tilde{F}(x, d) \), but \( M \models \exists x \tilde{F}(x, d) \). We shall prove that for all \( \sigma \in D_M^n \), \( M \models S_{\exists x \tilde{F}(x, d)}(\sigma) \land \exists x(\square[s \neq s']_\omega \land \tilde{F}(x, d)) \). That is, \( M \) does not satisfy the translation of \( \exists x \tilde{F}(x, d) \). Assume to the contrary that for some \( \sigma \in D_M^n \):

\[
M \models S_{\exists x \tilde{F}(x, d)}(\sigma) \land \exists x(\square[s \neq s']_\omega \land \tilde{F}(x, d)).
\]

Therefore, by the definition \( \models \), for some \( (\sigma_1, d_1) \) such that \( \models (\sigma_1, d_1) = \models (\sigma, d) \) and some assignment \( c \) to \( x \):

\[
M \models S_{\exists x \tilde{F}(x, d)}(\sigma_1) \land \square[s \neq s']_\omega \land \tilde{F}(c, d_1).
\]

Since \( \sigma \) is a state variable with respect to \( d \) and the latter changes already in the first state, \( \models (\sigma_1, d_1) = \models (\sigma, d) \) implies that the first change of \( \sigma_1 \) cannot occur before the first change of \( d_1 \). Thus, were (6.1) be true, the first change of the assignment \( c \) to \( x \) that occurs before the first change of \( d_1 \), would also occur before the first change of \( \sigma_1 \). However, this is impossible, because (6.1) implies \( M \models \square[s \neq s']_c \).

PROOF. [Theorem 6.1] The theorem is, trivially true, if \( D_M \) consists of one element only. So we assume that \( D_M \) contains at least two different elements. We start with the proof of the first part of the theorem. Since \( D_M \) contains at least two different elements, there exists a sequence \( \sigma_0, \sigma_1, \ldots \) of elements of \( D_M \) such that \( M \models \sigma_i \neq \sigma_{i+1}, i = 0, 1, \ldots \). Let \( \sigma \in D_M \) be defined by \( \sigma(i) = \sigma_i, i = 0, 1, \ldots \). By Remark 3.7, it suffices to show that \( M \models F(\sigma) \) if and only if \( M \models S_{\phi}(\sigma) \land H(\phi(\sigma), \sigma) \). Since, by the definition of \( \phi \), \( M \models S_{\phi}(\sigma) \), we have to prove that \( M \models F(\sigma) \) if and only if \( M \models H(\phi(\sigma), \sigma) \). The proof is by induction on the complexity of \( \phi \).

Basis: If \( \phi(\sigma) \) is an STLA formula, then \( H(\phi(\sigma), \sigma) \) is \( \phi(\sigma) \) itself and the basis follows.

Induction step: The case of the logical connectives \( \land \) and \( \lor \) and the rigid variable quantifier \( \exists \) immediately follows from the definition of the translation and the induction hypothesis.
Let \( \varphi(x, \bar{x}) \) be of the form \( \exists x \psi(x, \bar{x}) \). Then \( \mathcal{H}(\varphi(x, \bar{x}), \sigma) \) is

\[
\exists x (\Box[\sigma \neq \sigma']_x \land \mathcal{H}(\psi(x, \bar{x}), \sigma)).
\]

Assume \( M \models \exists x \psi(\bar{d}, x, \bar{d}) \). That is, for some \( a \in D^N_M \), \( M \models \psi(\bar{d}, a, \bar{d}) \). By the induction hypothesis, \( M \models \mathcal{H}(\psi(\bar{d}, a, \bar{d}), \sigma) \). Also, by the definition of \( \sigma \), \( M \models \Box[\sigma \neq \sigma']_a \). Thus, \( M \models \Box[\sigma \neq \sigma']_a \land \mathcal{H}(\psi(\bar{d}, a, \bar{d}), \sigma) \), implying, by Remark 3.7, \( M \models \exists x (\Box[\sigma \neq \sigma']_x \land \mathcal{H}(\psi(x, \bar{x}), \sigma)) \), even without adding or removing stuttering steps.

Conversely, assume that \( M \models \exists x (\Box[\sigma \neq \sigma']_x \land \mathcal{H}(\psi(x, \bar{x}), \sigma)) \). That is, for some \( \sigma_1 \) and \( \bar{d}_1 \) such that

\[
\exists x (\Box[\sigma \neq \sigma']_x \land \mathcal{H}(\psi(x, \bar{x}), \sigma)).
\]

and some \( a \in D^N_M \), \( M \models \Box[\sigma_1 \neq \sigma'_1]_a \land \mathcal{H}(\psi(\bar{d}, a, \bar{d}), \sigma_1) \). Since \( \Box[s \neq s']_a \land \mathcal{H}(\psi(\bar{d}, a, \bar{d}), s) \) is a TLA formula and satisfiability of TLA formulas is invariant under stuttering, \( M \models \Box[\sigma_1 \neq \sigma'_1]_a \land \mathcal{H}(\psi(\bar{d}, a, \bar{d}), \sigma_1) \). In particular, \( M \models \Box[\sigma_1 \neq \sigma'_1]_a \). That is, \( \sigma_1 \) changes, at least, as fast as \( \sigma \) does. Since \( \sigma \) changes in each state, \( \exists x (\Box[\sigma_1 \neq \sigma'_1]_x \land \mathcal{H}(\psi(\bar{d}, x, \bar{d}), \sigma)) \), which together with (6.2) imply \( \exists x (\Box[\sigma_1 \neq \sigma'_1]_x \land \mathcal{H}(\psi(\bar{d}, x, \bar{d}), \sigma)) \). Therefore, \( M \models \mathcal{H}(\psi(\bar{d}, a, \bar{d}), \sigma) \). By the induction hypothesis, \( M \models \psi(\bar{d}, a, \bar{d}) \), implying \( M \models \exists x \psi(\bar{d}, x, \bar{d}) \).

Now we pass to the proof of the second part of the theorem. Let \( M \models S'_{\varphi}(\sigma) \land \mathcal{H}(\psi(\bar{d}, \bar{d}), \sigma) \). Since \( S'(s) \land \mathcal{H}(\psi(\bar{d}, \bar{d}), s) \) is a TLA formula and satisfiability of TLA formulas is invariant under stuttering, \( M \models S'_{\varphi}(\sigma) \land \mathcal{H}(\psi(\bar{d}, \bar{d}), \sigma) \). Let \( \sigma(i) = \sigma_i, i = 0, 1, \ldots \). Since, in particular, \( M \models S'_{\varphi}(\sigma) \), i.e. \( \sigma \) changes, at least, as fast as the assignment to any flexible variable of \( \varphi \), \( M \models \sigma_i \neq \sigma_{i+1}, i = 0, 1, \ldots \). Thus, by the ‘only if’ part of the proof of the first part of the theorem, \( M \models \varphi(\bar{d}, \bar{d}) \), i.e. \( \mathcal{E} = \bar{d} \).

7 Simulating finite domains by infinite domains

In this section, using the ‘relativization technique’, we show how finite domains can be embedded into infinite ones. This step is required because we need infinitely many domain elements to simulate time when we embed SOTL or TLA into ERTLA.

We shall use the following notation. Let \( R \) be a new unary predicate symbol and let \( \varphi \) be a formula over the original language. We denote formulas \( \exists x (R(x) \land \varphi(x)) \) and \( \exists x (\Box R(x) \land \varphi(x)) \) by \( \exists_R x \varphi(x) \) and \( \exists_R x \varphi(x) \), respectively.

The \( R \)-relativization of \( \varphi \), denoted \( \varphi_R \) is the formula that is obtained from \( \varphi \) by replacing the quantifier \( \exists x \) with \( \exists R \). Formally, \( \varphi_R \) is defined, by induction, as follows:

- if \( \varphi \) does not contain quantifiers, then \( \varphi_R = \varphi \);
- \( (\neg \varphi)_R = \neg \varphi_R \);
- \( (\varphi \cup \psi)_R = \varphi_R \cup \psi_R \);
- \( (\Box \varphi)_R = \Box \varphi_R \);
- \( (\exists x \varphi)_R = \exists_R x \varphi_R \); and
- \( (\exists x \varphi)_R = \exists_R x \varphi_R \).

Next, for a formula \( \varphi \) we denote by \( F^\varphi \) the conjunction of all formulas of the form \( \forall x_1 \ldots \forall x_n \exists f(x_1, \ldots, x_n) = x \), all formulas of the form \( \exists y (y = x) \), and all formulas of the form \( \exists x \Box (y = x) \) where \( f \) is a function symbol that appears in \( \varphi \) and \( x/\bar{x} \)

\[6] Note that \( \mathcal{E} \) may contain stuttering steps, because it also depends on \( \sigma \).
is a free rigid/flexible variable of \( \varphi \). Finally, the translation of an ERTLA formula \( \varphi \) is \( \exists x R(x) \land F^R_R \land R_R \). We need conjunct \( \exists x R(x) \) to ensure that \( R \) is not empty and conjunct \( F^R_R \) is required because for each \( (n, \ldots, m) \) function symbol \( f \), each rigid variable \( x \), and each flexible variable \( \alpha \) which appear in \( \varphi \) we have ‘implicit’ axioms \( \forall x_1, \ldots, x_n \exists x f(x_1, \ldots, x_n) = x \), \( \exists y(y = x) \), and \( \exists y \square(y = x) \), respectively, which also must be relativized.

Now, temporal interpretations of the original language can be simulated by temporal interpretations over infinite domains of the original language augmented with \( R \) in the following manner.

Let \( M \) be a temporal interpretation. Consider a temporal interpretation \( M^+ \) of the language augmented with \( R \), such that \( D_{M^+} \) is an infinite domain that contains \( D_M \) as a subset, and assignments to predicate and function symbols are defined as follows. For a predicate symbol \( P \) (a function symbol \( f \)) of the original language, \( P^{M^+} (f^{M^+}) \) can be any extension of \( P^M (f^M) \) to \( D_{M^+} \) and \( R^M = D_M \). Obviously, \( M^+ \models \exists x R(x) \land F^R_R \land R_R \) and for an ERTLA formula \( \varphi(x, \alpha) \) and ‘\( M \) assignments’ \( \bar{\alpha} \) and \( \bar{\alpha} \) to \( x \) and \( \alpha \), respectively, \( M \models \varphi(\bar{\alpha}, \bar{\alpha}) \) if and only if \( M^+ \models \varphi_R(\bar{\alpha}, \bar{\alpha}) \).

Conversely, let \( M \) be a temporal interpretation of the language augmented with \( R \) such that \( M \models \exists x R(x) \land F^R_R \land R_R \). Consider a temporal interpretation \( M^- \) that is defined as follows. The domain \( D_{M^-} \) of \( M^- \) is \( R^M \). Since \( M \models \exists x R(x) \), \( D_{M^-} \) is not empty. For a predicate symbol \( P \) (a function symbol \( f \)), \( P^{M^-} (f^{M^-}) \) is the restriction of \( P^M (f^M) \) to \( D_{M^-} \). Since \( M \models F^R_R \), assignment \( P^{M^-} (f^{M^-}) \) is well defined. Then, extending \( P^{M^-} \) and \( f^{M^-} \) to \( P^M \) and \( f^M \), respectively, we obtain that \( (M^-)^+ = M \). Thus, by the previous paragraph, for an ERTLA formula \( \varphi(x, \alpha) \) and ‘\( M^- \) assignments’ \( \bar{\alpha} \) and \( \bar{\alpha} \) to \( x \) and \( \alpha \), respectively, \( M \models \varphi_R(\bar{\alpha}, \bar{\alpha}) \) if and only if \( M^- \models \varphi_R(\bar{\alpha}, \bar{\alpha}) \). Moreover, since, by definition, \((\exists y(y = x))^R \) and \((\exists y \square(y = x))^R \) are equivalent to \( R(x) \) and \( \square R(x) \), respectively, and \( M \models F^R_R \), if \( M \models \varphi_R(\bar{\alpha}, \bar{\alpha}) \), then \( \bar{\alpha} \) and \( \bar{\alpha} \) are ‘\( M^- \) assignments’ as well.

8 Summary

Combining Theorems 4.2, 4.4, 5.3, and 6.1 (and Remark 4.6, of course) with the relativization from the previous section we obtain the desired translations of TLA into SOTL and vice versa.

Finally, it immediately follows from the definitions of \( s_{OTE} \), \( F \), \( G \), and \( H \) that the translation from SOTL into TLA is linear, whereas the converse translation is quadratic.

Acknowledgements

A part of this paper is based on the M.Sc. thesis of Arkady Estrin written at the Technion – Israel Institute of Technology.

The work of Michael Kaminski was partially supported by the Fund for the Promotion of Research in the Technion. A part of this paper was written when Michael Kaminski was on his sabbatical leave at the Department of Computer Science of the National University of Singapore.

References


The Expressive Power of Temporal Logic of Actions


Received 4 October 2000