Translating PLTL into WS1S: Application Description

B. Hirsch* and U. Hustadt
Department of Computer Science, University of Liverpool
Liverpool L69 7ZF, United Kingdom,
{B.Hirsch,U.Hustadt}@csc.liv.ac.uk

Abstract

This paper introduces a technique for translating propositional linear time logic PLTL into the weak second-order logic of one successor WS1S. Using the *Mona* tool, a decision procedure for WS1S, we obtain a decision procedure for PLTL. We demonstrate the viability of this approach by an empirical comparison with *STeP*, *SMV*, and the Logics Workbench on a class of semi-randomly generated PLTL-formulae.

1 Introduction

Temporal logics have long been recognised as being appropriate languages for specifying a wide variety of important computational properties in computer science and artificial intelligence [3]. Hence, key techniques for utilising temporal logic specifications have been investigated, including verification via proof [13, 14], verification via model-checking [2], direct execution, and combinations of some of these techniques [9]. In particular, verification via model-checking has seen sustained effort in converting theoretical approaches into reasonably efficient methods and providing tool support for practical applications.

Although there is also an extensive body of work on verification via proof, most of it is only of theoretical nature and only a small number of maintained tools exists, in particular, if we look for fully automated decision procedures for logics like PLTL or computation tree logic CTL*. In this paper we describe a translation from PLTL into the weak second-order logic of one successor WS1S, which will allow us to utilise the *Mona* tool as a decision procedure for the satisfiability problem in PLTL.

The *Mona* tool [6] provides decision procedures for WS1S and WS2S by translating WS1S and WS2S formulae into minimum deterministic finite automata and guided tree automata, respectively. Although the satisfiability problem of WS1S and WS2S is non-elementary, *Mona* has been successfully applied in a number of applications [1]. Furthermore, since the translation we propose is computable in polynomial time and also the translation back into PLTL can be computed in polynomial time, we are considering a fragment of WS1S whose satisfiability problem is PSPACE-complete. Thus, considerations of the computational complexity of the satisfiability problem alone provide no a priori insight into the practicality of the approach.

Consequently, it is important to evaluate this approach by further theoretical and empirical analysis. Concerning the latter we can build upon of the preliminary investigations by Hustadt and Schmidt [7] who have performed empirical comparisons of various decision procedures for PLTL on a class of semi-randomly generated PLTL-formulae. We will use the same class to compare our approach with *STeP*, *SMV*, and the Logics Workbench.

*This work is based on ideas investigated by the first author while working on his MSc project under the supervision of U. Hustadt and M. de Rijke.

2 PLTL

Let P be a set of propositional variables. The set of formulae of *propositional linear time logic* PLTL (over P) is inductively defined as follows: (i) every propositional variable in P is a formula of PLTL, (ii) if φ and ψ are formulae of PLTL, then $\neg\varphi$, $(\varphi \vee \psi)$, $\bigcirc\varphi$, $\Diamond\varphi$, $\Box\varphi$, $\varphi \mathcal{U} \psi$, and $\varphi \mathcal{W} \psi$ are formulae of PLTL. Other boolean connectives including $\top$, $\bot$, $\wedge$, $\rightarrow$, and $\leftrightarrow$ are defined using $\neg$, and $\vee$. A formula of the form $\Diamond\varphi$ is a $\Diamond$-*formula*. Analogously, for the remaining temporal operators.

We also use $\top$ and $\bot$ to denote the empty conjunction and disjunction, respectively. A *literal* is either p or $\neg p$ where p is a propositional variable. By $|\varphi|$ we denote the *length of a formula* φ which is the number of symbols in φ, and $\mathrm{SF}(\varphi)$ is the set of subformulae of φ or their negations (after eliminating double negations).

PLTL-formulae are interpreted over ordered pairs $\mathcal{I} = \langle \mathcal{S}, \iota \rangle$ where (i) $\mathcal{S}$ is an infinite sequence of *states* $(s_i)_{i \in \mathbf{N}}$ and (ii) ι is a function assigning to each state a subset of P.

We define a semantic relation $\models$ between pairs consisting of a PLTL-interpretation $\mathcal{I} = \langle \mathcal{S}, \iota \rangle$ and a state $s_i \in S$, and PLTL-formulae (omitting $\mathcal{U}$- and $\mathcal{W}$-formulae) as follows.

$\mathcal{I}, s_i \models p$ iff $p \in \iota(s_i)$ $\qquad$ $\mathcal{I}, s_i \models \neg\varphi$ iff $\mathcal{I}, s_i \not\models \varphi$

$\mathcal{I}, s_i \models \varphi \vee \psi$ iff $\mathcal{I}, s_i \models \varphi$ or $\mathcal{I}, s_i \models \psi$ $\qquad$ $\mathcal{I}, s_i \models \bigcirc\varphi$ iff $\mathcal{I}, s_{i+1} \models \varphi$

$\mathcal{I}, s_i \models \Box\varphi$ iff for all $j \in \mathbb{N}$, $j \geq i$ implies $\mathcal{I}, s_j \models \varphi$

$\mathcal{I}, s_i \models \Diamond\varphi$ iff there exists $j \in \mathbb{N}$ such that $j \geq i$ and $\mathcal{I}, s_j \models \varphi$

If $\mathcal{I}, s_i \models \varphi$ then we say φ is *true* or *holds* at s_i in $\mathcal{I}$. If s_i is a state in $\mathcal{I}$, then let $[s_i]_{\mathcal{I},\varphi} = \{\psi \in \mathrm{SF}(\varphi) \mid \mathcal{I}, s_i \models \psi\}$. An interpretation $\mathcal{I}$ *satisfies* a formula φ iff φ holds at s_0 in $\mathcal{I}$ and it *satisfies* a set N of formulae iff for every formula $\psi \in N$, $\mathcal{I}$ satisfies ψ. In this case, $\mathcal{I}$ is a *model* for φ and N, respectively, and we say φ and N are (PLTL-)satisfiable. The satisfiability problem of PLTL is PSPACE-complete [12].

Arbitrary PLTL-formulae can be transformed into a *separated normal form* (SNF) in a satisfiability equivalence preserving way using a renaming technique replacing non-atomic subformulae by new propositions and removing all occurrences of the $\mathcal{U}$ and $\mathcal{W}$ operator [5]. The result is a set of *SNF clauses* of the following form (which differs slightly from [5]).

$$\bigvee_{i=1}^{n} L_i \qquad \text{(initial clause)}$$

$$\Box((\bigvee_{j=1}^{m} K_j) \vee (\bigvee_{i=1}^{n} \bigcirc L_i)) \qquad \text{(global clause)}$$

$$\Box((\bigvee_{j=1}^{m} K_j) \vee \Diamond L) \qquad \text{(sometime clause)}$$

Here, K_j, L_i, and L (with $1 \leq j \leq m$, $0 \leq m$, and $1 \leq i \leq n$, $0 \leq n$) denote literals.

Thus, in the following we consider without loss of generality only PLTL-formulae without occurrences of the temporal connectives $\mathcal{U}$ and $\mathcal{W}$.

3 WS1S

WS1S is the weak second-order logic of 1 successor function which is interpreted over the natural numbers. Formally, let $\mathcal{V}_1 = \{x, x_1, x_2, \dots\}$ and $\mathcal{V}_2 = \{X, X_1, X_2, \dots\}$ be sets of first-order and second-order variables, respectively, and s be a successor function. Then the ordered tuple $\Sigma = \langle \mathcal{V}_1, \mathcal{V}_2, \mathsf{s} \rangle$ is a *WS1S-signature*.

The *set of terms* $\mathcal{T}(\Sigma)$ over a WS1S-signature Σ is inductively defined as follows: (i) $0 \in \mathcal{T}(\Sigma)$, (ii) every variable $x \in \mathcal{V}_1$ is in $\mathcal{T}(\Sigma)$, and (iii) if $t \in \mathcal{T}(\Sigma)$ then $\mathsf{s}(t) \in \mathcal{T}(\Sigma)$. The *set of well-formed formulae* $\mathcal{L}(\Sigma)$ over a WS1S-signature Σ is inductively defined as follows: (i) if

$X \in \mathcal{V}_2$ and $t \in \mathcal{T}(\Sigma)$ then $X(t) \in \mathcal{L}(\Sigma)$, (ii) if t_1, $t_2 \in \mathcal{T}(\Sigma)$ then $t_1 < t_2$, $t_1 = t_2 \in \mathcal{L}(\Sigma)$, (iii) if $x \in \mathcal{V}_1$, $X \in \mathcal{V}_2$, φ_1, $\varphi_2 \in \mathcal{L}(\Sigma)$ then $\neg\varphi_1$, $\varphi_1 \vee \varphi_2 \in \mathcal{L}(\Sigma)$, $\exists x(\varphi) \in \mathcal{L}(\Sigma)$, and $\exists X(\varphi) \in \mathcal{L}(\Sigma)$. Again, additional boolean connectives can be defined using the ones included in $\mathcal{L}(\Sigma)$. Other connectives which can be defined in terms of $\mathcal{L}(\Sigma)$ include $\forall x(\varphi) := \neg\exists x(\neg\varphi)$, $\forall X(\varphi) := \neg\exists X(\neg\varphi)$, $X_1 \subseteq X_2 := \forall x(\neg X_1(x) \vee X_2(x))$, $x \leq y := (x = y) \vee (x < y)$, and $x + 1 := \mathsf{s}(x)$.

A *WS1S-interpretation* $\mathfrak{M}_{\sigma,\tau}$ over a signature Σ is an ordered tuple $\langle \mathbf{T}, s, <, \sigma, \tau\rangle$, where (i) $\mathbf{T}$ is the set of natural numbers, elements of $\mathbf{T}$ are called *states*, (ii) s is a successor function such that $s(n) = n + 1$ for every $n \in \mathbf{T}$, (iii) $<$ denotes the 'less than' relation on the natural numbers, (iv) $\sigma : \mathcal{V}_1 \mapsto \mathbf{T}$ is a valuation function for first-order variables, and (v) $\tau : \mathcal{V}_2 \mapsto \mathit{Fin}(2^{\mathbf{T}})$ is a valuation function for second-order variables, where $\mathit{Fin}(2^{\mathbf{T}})$ denotes the set of all finite subsets of $\mathbf{T}$. We omit the indices σ and τ whenever possible. Let $\sigma[x := n]$ and $\tau[X := N]$ denote assignment functions identical to σ and τ, respectively, except $\sigma[x := n]$ assigns n to x and $\tau[X := N]$ assigns N to X, respectively.

The function σ mapping first-order variables to $\mathbf{T}$ is lifted to a function $\hat{\sigma}$ mapping terms in $\mathcal{T}(\Sigma)$ to elements of $\mathbf{T}$ as follows: $\hat{\sigma}(x) = \sigma(x)$, $\hat{\sigma}(\mathsf{0}) = 0$, and $\hat{\sigma}(\mathsf{s}(t)) = s(\hat{\sigma}(t)) = \hat{\sigma}(t) + 1$.

Let Σ be a WS1S-signature. We inductively define a relation $\models$ between WS1S-interpretations $\mathfrak{M} = \langle \mathbf{T}, s, <, \sigma, \tau\rangle$ over Σ and WS1S-formulae as follows.

$$
\begin{array}{lll}
\mathfrak{M}_{\sigma,\tau} \models X(t) & \text{iff } \hat{\sigma}(t) \in \tau(X) & \\
\mathfrak{M}_{\sigma,\tau} \models t_1 = t_2 & \text{iff } \hat{\sigma}(t_1) = \hat{\sigma}(t_2) & \mathfrak{M}_{\sigma,\tau} \models t_1 < t_2 \text{ iff } \hat{\sigma}(t_1) < \hat{\sigma}(t_2) \\
\mathfrak{M}_{\sigma,\tau} \models \neg\varphi & \text{iff } \mathfrak{M}_{\sigma,\tau} \not\models \varphi & \mathfrak{M}_{\sigma,\tau} \models \varphi \vee \psi \quad \text{iff } \mathfrak{M}_{\sigma,\tau} \models \varphi \text{ or } \mathfrak{M}_{\sigma,\tau} \models \psi \\
\mathfrak{M}_{\sigma,\tau} \models \exists x(\varphi) & \text{iff there exists a } n \in \mathbf{T} \text{ such that } \mathfrak{M}_{\sigma[x:=n],\tau} \models \varphi & \\
\mathfrak{M}_{\sigma,\tau} \models \exists X(\varphi) & \text{iff there exists an } N \in \mathit{Fin}(2^{\mathbf{T}}) \text{ such that } \mathfrak{M}_{\sigma,\tau[X:=N]} \models \varphi &
\end{array}
$$

For readability we will drop the super- and subscripts whenever possible.

If $\mathfrak{M}_{\sigma,\tau} \models \varphi$ then we say φ is *true* in $\mathfrak{M}_{\sigma,\tau}$ and that $\mathfrak{M}_{\sigma,\tau}$ *satisfies* φ. Similarly, if $X \in \mathcal{V}_2$, $\mathfrak{M}_{\sigma,\tau}$ is a WSnS-interpretation with $n \in \mathbf{T}$, then X is *true at* n iff $n \in \tau(X)$. If φ is a WS1S-formula over Σ such that there is an interpretation $\mathfrak{M}_{\sigma,\tau}$ over Σ such that φ is true in $\mathfrak{M}_{\sigma,\tau}$, then φ is *satisfiable* and $\mathfrak{M}_{\sigma,\tau}$ is a WS1S-*model* for φ. The complexity of the satisfiability problem of WS1S is non-elementary [10].

4 Coping with Infinity

Given that the domain underlying both the interpretation of PLTL and WS1S is isomorphic to the natural numbers and that the language of WS1S allows us to quantify over elements of this domain and at the same time provides us with a successor function s and a binary relation $<$ interpreted as the successor function and 'less than' relation on the natural number, respectively, the translation of PLTL to WS1S seems to be straightforward. Given a set P of propositional variables, we uniquely associate with each element $p \in \mathsf{P}$ a second-order variable X_p. Then part of a translation of PLTL to WS1S could be given by

$$\varpi(p, x) = X_p(x) \qquad \varpi(\bigcirc\varphi, x) = \varpi(\varphi, \mathsf{s}(x)) \qquad \varpi(\Box\varphi, x) = \forall y(x \leq y \rightarrow \varpi(\varphi, y))$$

Extending ϖ to cover the remaining connectives of PLTL is straightforward. The translation $\overline{\Pi}(\varphi)$ of a PLTL-formula φ over P can then be given by $\varpi(\varphi, \mathsf{0})$.

However, we only need to look at a very simple example to see that this translation to WS1S is not satisfiability equivalence preserving. Consider $\Box p$ and its translation $\overline{\Pi}(\Box p) = \forall y(\mathsf{0} \leq y \rightarrow X_p(y))$. If we try to construct a model for $\forall y(\mathsf{0} \leq y \rightarrow X_p(y))$, the valuation

function τ has to map X_p to $\mathbf{T}$, since for all elements n of $\mathbf{T}$, $0 \leq n$ is true. However, $\mathbf{T}$ is an infinite set while only finite subsets of $\mathbf{T}$ can be in the codomain of τ. Thus, $\overline{\Pi}(\Box p)$ is unsatisfiable. Of course, $\Box p$ is satisfiable, any interpretation $\langle \mathcal{S}, \iota \rangle$ with $p \in \iota(s_i)$ for every $s_i \in \mathcal{S}$ is a model for $\Box p$.

Our solution to this problem is based on a result by Sistla and Clarke [12]. They have shown that if a PLTL-formula is satisfiable, then it is satisfiable in a so-called ultimately periodic interpretation. Formally, a PLTL-interpretation $\mathcal{I} = \langle \mathcal{S}, \iota \rangle$ is *ultimately periodic with starting index b and period p* iff for every $j \geq b$, $\iota(s_j) = \iota(s_{j+p})$. Let ψ be a formula of the form $\Diamond \psi_1$ such that $\mathcal{I}, s_i \models \psi$. We say that ψ is *fulfilled before* s_j iff $i \leq j$ and there is an l, $i \leq l \leq j$, such that $\mathcal{I}, s_l \models \psi_1$. Then a PLTL-formula φ is satisfiable iff it is satisfiable in an ultimately periodic interpretation $\mathcal{I} = \langle \mathcal{S}, \iota \rangle$ with starting index $b \leq 2^{|\varphi|}$, period $p \leq |\varphi| 2^{|\varphi|}$, and for every $j \geq b$, $[s_j]_{\mathcal{I},\varphi} = [s_{j+p}]_{\mathcal{I},\varphi}$ and every $\Diamond$-formula in $[s_j]_{\mathcal{I},\varphi}$ is fulfilled before s_{j+p}.

An ultimately periodic interpretation $\mathcal{I} = \langle \mathcal{S}, \iota \rangle$ with starting index b and period p can be finitely represented by the tuple $\langle \mathcal{S}_{<b+p}, \iota|_{\mathcal{S}_{<b+p}}, s_b \rangle$ where $\mathcal{S}_{<b+p} = (s_i)_{i<b+p}$. For our purposes a slightly different representation which includes the state s_{b+p} as well as s_b and is given by $\mathcal{I}_{fin} = \langle \mathcal{S}_{\leq e}, \iota|_{\mathcal{S}_{\leq e}}, s_b, s_e \rangle$, where $e = b + p$ and $\mathcal{S}_{\leq e} = (s_i)_{i \leq e}$, turns out to be more convenient. It is then possible to define the semantics of PLTL-formulae with respect to these finite representations of ultimately periodic PLTL-interpretations instead of the interpretations themselves. To this end, we can first define a function N on $\mathcal{S}_{\leq e}$ which determines for each state $s_i \in \mathcal{S}_{\leq e}$ the 'next state' in the ultimately periodic interpretation represented by $\mathcal{I}_{fin}$:

$$N(s_i) = \begin{cases} s_{i+1}, \text{ if } i < e \\ s_{b+1}, \text{ if } i = e \end{cases}$$

By N^* we denote the reflexive, transitive closure of N. Finally, we define a semantic relation $\models$ between a PLTL-formula φ and a pair $\mathcal{I}_{fin}$ and a state s_i:

$$\begin{array}{ll} \mathcal{I}_{fin}, s_i \models p \quad \text{iff } p \in \iota|_{\mathcal{S}_{\leq e}}(s_i) & \mathcal{I}_{fin}, s_i \models \bigcirc \varphi \text{ iff } \mathcal{I}_{fin}, N(s_i) \models \varphi \\ \mathcal{I}_{fin}, s_i \models \Box \varphi \text{ iff for all } s_j \in \mathcal{S}_{\leq e}, (s_i, s_j) \in N^* \text{ implies } \mathcal{I}_{fin}, s_j \models \varphi & \\ \mathcal{I}_{fin}, s_i \models \Diamond \varphi \text{ iff there exists } s_j \in \mathcal{S}_{\leq e}, (s_i, s_j) \in N^* \text{ such that } \mathcal{I}_{fin}, s_j \models \varphi & \end{array}$$

The definition of the semantics of the boolean connectives remains unchanged. This reformulation of the semantics of PLTL forms a suitable basis for a translation to WS1S as we will see in the following section.

5 The Translation

The translation we propose is a semantics-based translation. However, it is not based on the standard semantics presented in Section 2, but on the semantics presented in Section 4 where the semantic structures are finite representations of ultimately periodic PLTL-interpretations. We uniquely associate with every propositional variable p a second-order variable X_p. In addition we use first-order variables x_b and x_e. These will be mapped by the valuation function σ of a WS1S-interpretation to states of the interpretation which correspond to the states s_b and s_e of a ultimately periodic PLTL-interpretation as described in the previous section.

Formally, the translation morphism tr is defined as follows:

$$\begin{array}{lll} tr_x(p) = X_p(x) & tr_x(\neg\varphi) = \neg tr_x(\varphi) & tr_x(\varphi \vee \psi) = tr_x(\varphi) \vee tr_x(\psi) \\ tr_x(\bigcirc\varphi) = \exists y(tr_{xy} \wedge tr_y(\varphi)) & tr_x(\Box\varphi) = \forall y(tr^*_{xy} \rightarrow tr_y(\varphi)) & tr_x(\Diamond\varphi) = \exists y(tr^*_{xy} \wedge tr_y(\varphi)) \end{array}$$

where

$$tr_{xy} = ((x < x_e) \wedge (y = \mathsf{s}(x))) \vee ((x = x_e) \wedge (y = \mathsf{s}(x_b)))$$
$$tr^*_{xy} = \exists X(X(x) \wedge X(y) \wedge \forall y_1, z_1((X(y_1) \wedge tr_{y_1 z_1}) \rightarrow X(z_1)) \wedge \forall Y((Y(x) \wedge \forall y_2, z_2((Y(y_2) \wedge tr_{y_2 z_2}) \rightarrow Y(z_2))) \rightarrow X \subseteq Y))$$

and we define the translation Π of a PLTL-formula φ to WS1S by $\Pi(\varphi) = tr_0(\varphi)$.

Theorem 5.1. *Let φ be PLTL-formula. Then φ is satisfiable if and only if $\Pi(\varphi)$ is satisfiable.*

Proof. First, let $\mathfrak{M} = \langle \mathbf{T}, s, <, \sigma, \tau \rangle$ be a model of $\Pi(\varphi)$. We show how to construct a PLTL-interpretation $\mathcal{I} = \langle \mathcal{S}, \iota \rangle$ that satisfies φ. $\Pi(\varphi)$ contains free first-order variables x_b and x_e, and for each propositional variable p occurring in φ it contains a corresponding second-order variable X_p. Let b and e be $\sigma(x_b)$ and $\sigma(x_e)$, respectively, and let $p = e - b$. We obtain $\mathcal{I}$ by unravelling the subset $\mathbf{T}_{\leq e} = \{i \in \mathbb{N} \mid i \leq e\}$. We define each state s_i in $\mathcal{S}$ by $s_i = i$ and we define a function $r : \mathcal{S} \mapsto \mathbf{T}$ as follows: $r(s_i) = i$, for $i \leq e$, and $r(s_i) = ((i - (b+1)) \mod p) + (b+1)$, for $i > e$. Finally, the valuation function ι is defined by $\iota(s_i) = \{p \in \mathsf{P} \mid r(s_i) \in \tau(X_p)\}$.

We show by induction on the structure of φ that for any subformula ϑ of φ and any state s_i, $\mathcal{I}, s_i \models \vartheta$ iff $\mathfrak{M}_{\sigma',\tau} \models tr_x(\vartheta)$ where $\sigma' = \sigma[x := r(s_i)]$.

Assume that ϑ is a propositional variable p occurring in φ. By definition, $tr_x = X_p(x)$ and $\mathfrak{M}_{\sigma',\tau} \models X_p(x)$ iff $\sigma'(x) = r(s_i) \in \tau(X_p)$. By definition of ι, $p \in \iota(s_i)$ iff $r(s_i) \in \tau(X_p)$. Finally, $\mathcal{I}, s_i \models p$ iff $p \in \iota(s_i)$. Thus, $\mathcal{I}, s_i \models p$ iff $\mathfrak{M}_{\sigma[x:=r(s_i)],\tau} \models tr_x(p)$.

The cases for the boolean connectives are straightforward.

For $\vartheta = \bigcirc\psi$ we have to show that for any s_i, $\mathcal{I}, s_i \models \bigcirc\psi$ iff $\mathfrak{M}_{\sigma',\tau} \models tr_x(\bigcirc\psi)$. By definition, $\mathcal{I}, s_i \models \bigcirc\psi$ iff $\mathcal{I}, s_{i+1} \models \psi$ and by induction hypothesis $\mathcal{I}, s_{i+1} \models \psi$ iff $\mathfrak{M}_{\sigma'[y:=r(s_{i+1})],\tau} \models tr_y(\psi)$. Let $\sigma'' = \sigma'[y := r(s_{i+1})]$. Note that $\sigma''(x_b) = \sigma'(x_b) = \sigma(x_b)$ and $\sigma''(x_e) = \sigma'(x_e) = \sigma(x_e)$. By definition of r, $r(s_{i+1}) = r(s_i) + 1$ and $r(s_i) < e$, or $r(s_{i+1}) = b+1$ and $r(s_i) = e$. So, $\sigma''(y) = \sigma''(x)+1$ and $\sigma''(x) < \sigma''(x_e)$, or $\sigma''(y) = \sigma''(x_b)+1$ and $\sigma''(x) = \sigma''(x_e)$. In both cases we have $\mathfrak{M}_{\sigma'',\tau} \models tr_{xy}$. So, $\mathcal{I}, s_{i+1} \models \bigcirc\psi$ iff $\mathfrak{M}_{\sigma'',\tau} \models tr_{xy}$ and $\mathfrak{M}_{\sigma'',\tau} \models tr_y(\psi)$ iff $\mathfrak{M}_{\sigma',\tau} \models \exists y(tr_{xy} \wedge tr_y(\psi))$ iff $\mathfrak{M}_{\sigma',\tau} \models tr_x(\bigcirc\psi)$.

For $\vartheta = \Box\psi$, we have to show that for any s_i, $\mathcal{I}, s_i \models \Box\psi$ iff $\mathfrak{M}_{\sigma',\tau} \models tr_x(\Box\psi)$. If $\mathcal{I}, s_i \models \Box\psi$, then for all $j \in \mathbb{N}$, if $j \geq i$ then $\mathcal{I}, s_j \models \psi$. We have to show that $\mathfrak{M}_{\sigma',\tau} \models \forall y(tr^*_{xy} \rightarrow tr_y(\psi))$. Let n be an element of $\mathbf{T}$ such that $\mathfrak{M}_{\sigma'[y:=n],\tau} \models tr^*_{xy}$. We can show by induction that $\mathfrak{M}_{\sigma'[y:=n],\tau} \models tr^*_{xy}$ implies that there is a sequence $n_0, \ldots, n_k$ of elements of $\mathbf{T}$ such that $n_0 = r(s_i)$, $n_k = n$, and for every l, $0 \leq l < k$, $\mathfrak{M}_{\sigma[x:=n_l, y:=n_{l+1}],\tau} \models tr_{xy}$. By definition of tr_{xy}, for every l, $0 \leq l < k$, either $n_{l+1} = n_l + 1$ and $n_l < e$, or $n_{l+1} = b + 1$ and $n_l = e$. Again, by induction we can show that for every l, $0 \leq l \leq k$, $r(s_{i+l}) = n_l$. So, in particular, $r(s_{i+k}) = n_k = n$, that is, there is a $j \geq i$ such that $r(s_j) = n$. By induction hypothesis, for any s_l, $\mathcal{I}, s_l \models \psi$ iff $\mathfrak{M}_{\sigma'[y:=r(s_l)]} \models tr_y(\psi)$. Since $\mathcal{I}, s_j \models \psi$ we infer that $\mathfrak{M}_{\sigma'[y:=r(s_j)]} \models tr_y(\psi)$. Thus, for an arbitrary element n of $\mathbf{T}$, $\mathfrak{M}_{\sigma'[y:=n]} \models tr^*_{xy} \rightarrow tr_y(\psi)$, and consequently $\mathfrak{M}_{\sigma',\tau} \models \forall y(tr^*_{xy} \rightarrow tr_y(\psi))$.

Conversely, assume $\mathfrak{M}_{\sigma',\tau} \models tr_x(\Box\psi)$. So, $\mathfrak{M}_{\sigma',\tau} \models \forall y(tr^*_{xy} \rightarrow tr_y(\psi))$. We show by induction that that for every $j \in \mathbb{N}$, if $j \geq i$ then $\mathfrak{M}_{\sigma'[y:=r(s_j)],\tau} \models tr^*_{xy}$ and $\mathcal{I}, s_j \models \psi$. For $j < i$ this is trivially true. For $j = i$ we have to show that $\mathfrak{M}_{\sigma'[y:=r(s_i)],\tau} \models tr^*_{xy}$ and $\mathcal{I}, s_i \models \psi$. Both are straightforward to check. In the induction step, we consider $j + 1 > i$ and we can assume that $\mathfrak{M}_{\sigma'[y:=r(s_l)],\tau} \models tr^*_{xy}$ for every l, $i \leq l \leq j$. This, together with the observation that $\mathfrak{M}_{\sigma'[x:=r(s_j), y:=r(s_{j+1})],\tau} \models tr_{xy}$ holds, implies $\mathfrak{M}_{\sigma'[y:=r(s_{j+1})],\tau} \models tr^*_{xy}$. Then we again use

$\mathfrak{M}_{\sigma',\tau} \models tr_x(\Box\psi)$ to show that $\mathfrak{M}_{\sigma'[y:=r(s_{j+1})],\tau} \models tr_y(\psi)$, from which we infer $\mathcal{I}, s_{j+1} \models \psi$. Thus, $\mathcal{I}, s_i \models \Box\psi$.

Since $\Box$ and $\Diamond$ are duals, we are done.

Second, let $\mathcal{I} = \langle \mathcal{S}, \iota \rangle$ be an ultimately periodic PLTL-model for a satisfiable PLTL-formula φ with starting index b and period p. We show how to construct a a WS1S-interpretation $\mathfrak{M} = \langle \mathbf{T}, s, <, \sigma, \tau \rangle$ that satisfies $\Pi(\varphi)$. Let $\mathbf{T}$ be the set of natural numbers, s the successor function $+1$, and $<$ the 'less than' relation. Let σ be an arbitrary mapping from $\mathcal{V}_1$ to $\mathbf{T}$ such that $\sigma(x_b) = b$ and $\sigma(x_e) = b + p$. Let τ be an arbitrary mapping from $\mathcal{V}_2$ to $Fin(2^{\mathbf{T}})$ such that for every propositional letter p in φ, $\tau(X_p) = \{i \in \mathbb{N} \,|\, \mathcal{I}, s_i \models p \text{ and } i \leq b + p\}$. We define a mapping $r : \mathcal{S} \mapsto \mathbf{T}$ as follows: $r(s_i) = i$ for every $i \leq e$, and $r(s_i) = ((i - (b+1)) \bmod p) + (b+1)$ for every $i > e$.

Along the lines of the first case, we can then show by induction on the structure of φ that for every subformula ϑ of φ and every state $s_i \in \mathcal{S}$, $\mathcal{I}, s_i \models \vartheta$ iff $\mathfrak{M}_{\sigma[x:=r(s_i)],\tau} \models tr_x(\vartheta)$. Since $\mathcal{I}, s_0 \models \varphi$, $\mathfrak{M}_{\sigma[x:=r(s_0)],\tau} \models tr_x(\varphi)$ holds. By definition, $r(s_0) = 0$, which implies $\mathfrak{M} \models tr_0(\varphi)$. Thus, $\mathcal{I}$ satisfies $\Pi(\varphi)$. □

6 Experiments

In [7] Hustadt and Schmidt consider a class of semi-randomly generated PLTL-formulae in separated normal form which is intended to highlight differences between tableaux-based or automata-based decision procedures for PLTL and temporal resolution [4, 5]. It turns out that this class is also suitable to exemplify differences between various tableaux-based and automata-based decision procedures.

The random generator we have used in our experiments takes as input the number of propositional variables n, the number of clauses l, the number of literals per step clause k, and a probability p. A semi-randomly generated formula φ is a conjunction of global and sometime clauses of the following form

$$\Box(\bigcirc L_1^1 \vee \ldots \vee \bigcirc L_k^1) \wedge \ldots \wedge \Box(\bigcirc L_1^l \vee \ldots \vee \bigcirc L_k^l)$$
$$\wedge \Box(\neg p_1 \vee \Diamond p_2) \wedge \Box(\neg p_2 \vee \Diamond p_3) \wedge \ldots \wedge \Box(\neg p_n \vee \Diamond p_1)$$

where for each global clause, the literals $L_1^i, \ldots, L_k^i$ are generated by choosing k distinct variables randomly from the set $\{p_1, \ldots, p_n\}$ of n propositional variables and by determining the polarity of each literal with probability p. The sometime clauses included in φ only depend on the parameter n. In the following we call the formulae generated in this way *semi-random SNF formulae*. In all our experiments the parameters k and p have been fixed to 3 and 0.5, respectively.

The absence of initial clauses and the restricted form of global clauses in semi-random SNF formulae indicates that they form a strict subclass of the class of sets (or conjunctions) of SNF clauses. Thus, not every PLTL-formula is equivalent to some semi-random SNF formula.

Concerning the satisfiability of semi-random SNF formulae we observe that similar to random kSAT formulae, a semi-random SNF formula is likely to be satisfiable if the number l of global clauses is small and it is likely to be unsatisfiable if the number l of clauses is large. Figure 1 shows the percentage of satisfiable semi-random SNF formulae for two values of the parameter n, namely, $n = 5$ and $n = 12$. For each value of n and particular values of l we have generated 100 semi-random SNF formulae which were tested for satisfiability. For ratios l/n between 2 and 4 we note a phase transition in the satisfiability of semi-random SNF formula which becomes more abrupt with greater values of n.

In our experiments we have compared four systems: (i) the Logics Workbench (LWB) 1.0 which contains an implementation of Schwendimann's one-pass tableau calculus for PLTL [11] (written in C++), (ii) a system written by Klaus Schneider at the University of Karlsruhe which combines a PLTL-to-SMV translator (written in Java) and the SMV model checker (written in C) which we will call *TLSMV* in the following, (iii) the Stanford Temporal Prover *STeP* which includes a tableaux-based decision procedure for PLTL developed by McGuire [8], and (iv) the combination of the translation described in Section 5 (written in Prolog) and the *Mona* tool (written in C). The Prolog program for the translation can be downloaded from the second author's homepage. The experiments have been performed on a PC with a 800MHz Pentium III processor, 256MB main memory, and 512MB virtual memory running RedHat Linux 6.2. For each individual satisfiability test a time-limit of 1000 CPU seconds was used.

Figures 3 to 5 show the CPU time percentile graphs of the four systems on random SNF formulae for $n = 5$ and $n = 12$. It is obvious that the combination of translation to WS1S and *Mona* shows a better performance than any of the other three systems on almost all formulae considered in the experiments described in this paper. It is certainly premature to make the same claim for other classes of formulae and there will be classes where our approach is inferior to others (in particular, temporal resolution shows the same good performance on $\mathcal{C}_{ran}$ as can be seen in [7]), but it can be said that the experiments support our view that the combination of translation and automata-based techniques can provide viable decision procedures for PLTL.

References

[1] A. Ayari, D. A. Basin, and A. Podelski. LISA: A specification language based on WS2S. In *Proc. CSL'97*, LNCS 1414, pp. 18–34. Springer, 1998.

[2] E. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. MIT Press, 2000.

[3] E. A. Emerson. Temporal and modal logic. In *Handbook of Theoretical Computer Science*, pp. 997–1072. Elsevier, 1990.

[4] M. Fisher. A resolution method for temporal logic. In *Proc. IJCAI'91*, pp. 99–104. Morgan Kaufman, 1991.

[5] M. Fisher, C. Dixon, and M. Peim. Clausal Temporal Resolution. *ACM Transactions on Computational Logic*, 2(1):12–56, 2001.

[6] J. G. Henriksen et al. Mona: Monadic second-order logic in practice. In *Proc. TACAS'95*, LNCS 1019, pp. 89–119. Springer, 1996.

[7] U. Hustadt and R. A. Schmidt. Formulae which highlight differences between temporal logic and dynamic logic provers. In *Issues in the Design and Experimental Evaluation of Systems for Modal and Temporal Logics*, Technical Report DII 14/01, pp. 68–76. Unversitá degli Studi di Siena, 2001.

[8] Y. Kesten, Z. Manna, H. McGuire, and A. Pnueli. A decision algorithm for full propositional temporal logic. In *Proc. CAV'93*, LNCS 697, pp. 97–109. Springer, 1993.

[9] Z. Manna and the STeP group. STeP: The Stanford Temporal Prover. Technical report STAN-CS-TR-94-1518, Stanford University, 1994.

[10] A. R. Meyer. Weak monadic second order theory of successor is not elementary recursive. In *Proc. Logic Colloquium*, LNCS 453, pp. 132–154. Springer, 1975.

[11] S. Schwendimann. *Aspects of Computational Logic*. PhD thesis, Universität Bern, Switzerland, 1998.

[12] A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logic. *Journal of the ACM*, 32(3):733–749, 1985.

[13] M. Vardi and P. Wolper. Reasoning about infinite computations. *Inform. and Computat.*, 115:1–37, 1994.

[14] P. Wolper. The tableau method for temporal logic: An overview. *Logique et Analyse*, 28:119–136, 1985.

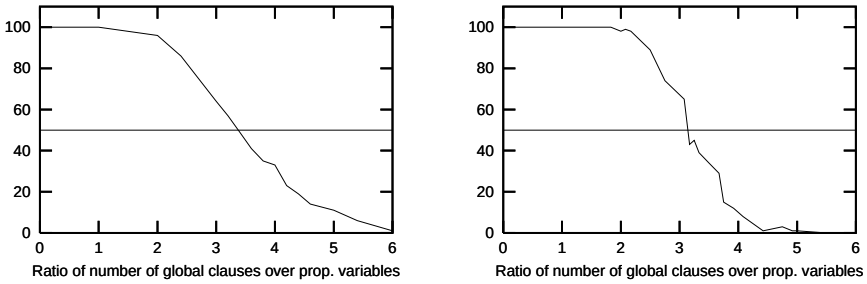

Figure 1: Satisfiability of random PLTL-formulae for $n = 5$ (left) and $n = 12$ (right).

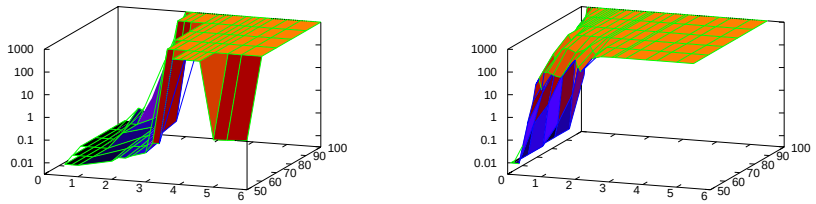

Figure 2: CPU time percentiles of the LWB 1.0 for $n = 5$ (left) and $n = 12$ (right).

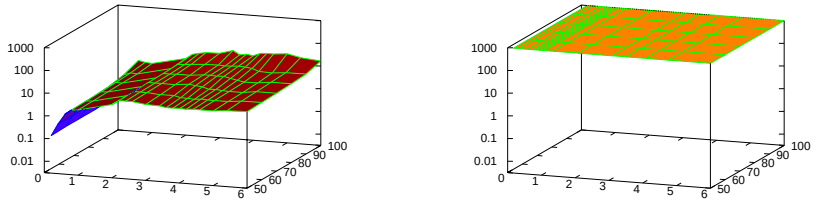

Figure 3: CPU time percentiles of *TLSMV* for $n = 5$ (left) and $n = 12$ (right).

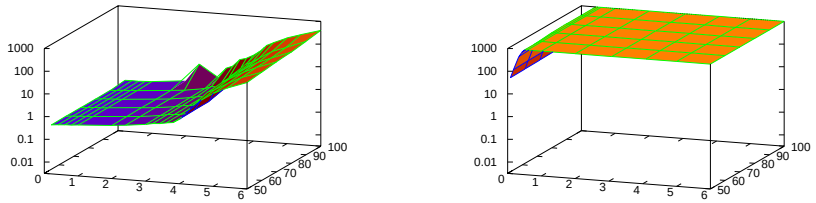

Figure 4: CPU time percentiles of *STeP* for $n = 5$ (left) and $n = 12$ (right).

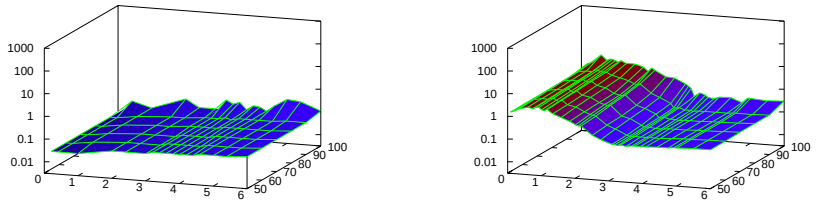

Figure 5: CPU time percentiles of *Mona* for $n = 5$ (left) and $n = 12$ (right).