The Complexity of Propositional Linear Temporal Logics

A. P. SISTLA

University of Massachusetts, Amherst, Massachusetts

AND

E. M. CLARKE

Carnegie-Mellon University, Pittsburgh, Pennsylvania

Abstract. The complexity of satisfiability and determination of truth in a particular finite structure are considered for different propositional linear temporal logics. It is shown that these problems are NP-complete for the logic with **F** and are PSPACE-complete for the logics with **F**, **X**, with **U**, with **U**, **S**, **X** operators and for the extended logic with regular operators given by Wolper.

Categories and Subject Descriptors: F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*complexity of proof procedures*; F.3.1 [**Logics and Meanings of Programs**]: Specifying and Verifying and Reasoning about Programs—*logics of programs*; *mechanical verification*; I.2.2 [**Artificial Intelligence**]: Automatic Programming—*program verification*

General Terms: Theory

Additional Key Words and Phrases: Branching time temporal logic, decision procedures, linear time temporal logic, NP-complete, PSPACE-complete, satisfiability

1. *Introduction*

Linear temporal logic was introduced in [6] as an appropriate formal system for reasoning about parallel programs. This logic permits the description of a program's execution history without the explicit introduction of program states or time. Moreover, important correctness properties, such as mutual exclusion, deadlock freedom, and absence of starvation, can be elegantly expressed in this system. Proving that a parallel program satisfies some correctness property consists of deducing the formula for that property from *program axioms* that characterize the possible interleaving of atomic statements of the individual processes. An important special case occurs when the program is finite state. In this case, the program axioms and correctness specification can be expressed in the *propositional* version of the logic, and provability becomes *decidable*. A number of researchers (see [5]) have attempted to use such a decision procedure for constructing correct finite-state programs.

In this paper, we examine the inherent complexity of decision procedures for *validity*, *satisfiability*, and *truth in structures generated by binary relations* for

Authors' addresses: A. P. Sistla, Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA 01003; E. M. Clarke, Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA 15213.

propositional linear temporal logics with the operators **F** (eventually), **G** (invariantly), **X** (next-time), **U** (until), and **S** (since).

Structures generated by binary relations (R-structures) model finite-state concurrent programs. The problem of determining truth in an R-structure consists of verifying whether a given formula holds on a path starting from a node of the structure. An algorithm for solving this problem can be used to determine whether a concurrent program fails to meet its specification on some execution.

We first consider the logic L(**F**) in which **F** is the only temporal operator. We prove that the problem of determining truth in an R-structure in NP-complete. Indeed, we prove that this problem is difficult even for simple formulas without nesting of temporal operators. This result is surprising since the corresponding problem for branching-time logics has been shown to be in P [1].

These proof techniques are extended to prove a linear size model theorem for L(**F**). Thus, we have independently established, by a different technique, the result obtained in [4] that satisfiability for this logic is in NP.

We next consider the complexity of full linear temporal logic. Although it is possible to translate propositional linear temporal logic into the language of the structures (N, <, P_1, P_2, . . .) where N is the set of natural numbers, < is the natural ω-ordering, and P_1, P_2, . . . are monadic predicates, any decision procedure for satisfiability of formulas in the latter logic must be nonelementary [7]. A tableau-based decision procedure for propositional linear temporal logic was given in [10]; however, this procedure requires exponential space. We give a polynomial-space-bounded decision procedure for satisfiability of formulas in L(**U**, **S**, **X**). We show that satisfiability for the logics L(**F**, **X**), L(**U**), L(**U**, **X**), L(**U**, **S**, **X**) and for the extended temporal logic given by Wolper in [10] are PSPACE-complete. These results are surprising because all of these logics have different expressive powers (some are more powerful than others). Finally, we show that the problem of determining truth in an R-structure is PSPACE-complete for the previously mentioned logics.

This paper is organized as follows: Section 2 defines the syntax and semantics of the linear temporal logic that we use in the remainder of the paper. In Section 3, we prove the results for L(**F**). Section 4 contains the PSPACE-completeness results for L(**F**, **X**), L(**U**), and L(**U**, **S**, **X**). In Section 5 we show how our results can be extended to the logic given in [10].

2. *Notation and Basic Definitions*

We use the following convention for symbols:

$P, Q, R, \ldots$	denote atomic formulas and are drawn from the finite set $\mathscr{P}$.
$f, g, h, \ldots$	denote formulas.
$s, t, u, \ldots$	denote finite or infinite sequences. We always assume $s = (s_0, s_1, \ldots)$.
$S, T, W, \ldots$	denote structures.

If $O_1, \ldots, O_k \in \{\mathbf{X}, \mathbf{F}, \mathbf{G}, \mathbf{U}, \mathbf{S}\}$ are distinct operators, then L($O_1, \ldots, O_k$) denotes the propositional temporal logic restricted to these operators, for example, L(**F**, **G**), L(**X**, **F**, **G**), and so on.

A *well-formed formula* in propositional linear temporal logic is either an atomic proposition or of the form $\neg f_1$, $f_1 \wedge f_2$, $\mathbf{X} f_1$, $f_1 \mathbf{U} f_2$, $f_1 \mathbf{S} f_2$ where f_1 f_2 are well-formed formulas. In addition, the following abbreviations are used:

$$f_1 \vee f_2 \equiv \neg(\neg f_1 \wedge \neg f_2), \qquad f_1 \supset f_2 \equiv \neg f_1 \vee f_2, \qquad \mathbf{F}f \equiv \text{True } \mathbf{U} f, \ \mathbf{G}f \equiv \neg \mathbf{F} \neg f.$$

Let $\tilde{L}(\mathbf{F}, \mathbf{X})$ be the logic that uses the Boolean connectives $\wedge$, $\vee$ and the temporal operators $\mathbf{F}$, $\mathbf{X}$, with negations allowed only on the atomic propositions.

A structure $S = (s, \xi)$, where $s = (s_0, s_1, \ldots)$ is an ω-sequence of states in which all the states are distinct and ξ: $\{s_0, s_1, \ldots\} \to 2^{\mathscr{P}}$. Intuitively, ξ specifies which atomic propositions are true in each state. An interpretation is a pair (S, δ) where S is a structure as previously defined, and δ is a state in the sequence s. Since we have assumed all states in s to be distinct, every state in s has a unique position. We define the truth of a formula in an interpretation inductively as follows:

$S, s_i \models P$	where	P is atomic iff $P \in \xi(s_i)$;
$S, s_i \models f_1 \wedge f_2$	iff	$S, s_i \models f_1$ and $S, s_i \models f_2$;
$S, s_i \models \neg f_1$	iff	not $(S, s_i \models f_1)$;
$S, s_i \models \mathbf{X} f_1$	iff	$S, s_{i+1} \models f_1$;
$S, s_i \models f_1 \mathbf{U} f_2$	iff	$\exists k \geq i$ such that $S, s_k \models f_2$ and $\forall j,\ i \leq j < k,\ S, s_j \models f_1$;
$S, s_i \models f_1 \mathbf{S} f_2$	iff	$\exists k \leq i$ such that $S, s_k \models f_2$ and $\forall j,\ k < j \leq i,\ S, s_j \models f_1$.

Length(f) denotes the length of the formula f, which is the number of symbols in f, and SF(f) is the set of subformulas of f or their negations after eliminating double negations. It can easily be shown by induction on f that card(SF(f)) $\leq$ 2(length(f)).

An R-structure T is a triple (N, R, η), where N is a finite set of states (also called *nodes*). $R \subseteq N \times N$ is a total binary relation (i.e., $\forall t \in N\ \exists t' \in N$ such that $(t, t') \in R$), and η: $N \to 2^{\mathscr{P}}$. A *path* p in T is an infinite sequence $(p_0, p_1, \ldots)$ where $\forall i \geq 0$, $p_i \in N$, and $(p_i, p_{i+1}) \in R$. For a path p in an R-structure $T = (N, R, \eta)$, we let S_p denote the structure (s, ξ) where $\forall i \geq 0$, $\xi(s_i) = \eta(p_i)$.

The global behavior of a finite-state parallel program can be modeled as an R-structure. In the R-structure, each path starting from the initial state represents a possible interleaving of executions of the individual processes in the program. In many cases the correctness requirements of the concurrent system can be expressed by a formula of propositional linear temporal logic. The system will be correct iff every possible execution sequence satisfies this formula; that is, every path beginning at the initial state in the corresponding R-structure satisfies the formula. For these reasons, the following problem (which we call the *determination of truth in an R-structure*) is important in verifying finite-state parallel programs:

> Given an R-structure T, a state $\delta \in N$, and a formula $f \in L$, is there a path p in T starting from δ such that S_p, $s_0 \models f$?

3. *The Complexity of* $L(\mathbf{F})$

Let $S = (s, \xi)$ be a structure and let $s'' = (s_j, s_{j+1}, \ldots)$ be the maximal suffix of s such that for each s_k in s'' the following condition holds: $\forall l\ \exists i$ such that $i > l$ and $\xi(s_i) = \xi(s_k)$; that is, there exist infinitely many states in s'' that have the same assignment of atomic propositions as s_k. It is easily seen that such an s'' exists (because $\mathscr{P}$ is finite) and is unique. Let $s = s' \cdot s''$. Define *init*(S) $= s'$, *final*(S) $= s''$, *range*(S) $= \{\xi(s_k) \mid s_k$ is in $s''\}$, and *size*(S) = length(init(S)) + card(range(S)). Thus, range(S) is the set of all assignments of atomic propositions that occur infinitely often in s. Note that init(S) is a finite sequence (and can be the null sequence!), final(S) is an infinite sequence, and range(S) is a subset of $2^{\mathscr{P}}$.

The following lemmas are used in proving the main results of this section. Lemma 3.1 shows that all states in final(S) with the same assignment of atomic prepositions satisfy the same formulas in L(**F**).

LEMMA 3.1. *Let $S = (s, \xi)$ be a structure and let s_j, s_k be states in final(S) such that $\xi(s_j) = \xi(s_k)$; then for all $f \in L(\mathbf{F})$, $S, s_j \models f$ iff $S, s_k \models f$.*

PROOF. The proof is by structural induction on f. If f is an atomic proposition, then the lemma holds trivially. Assume that the lemma holds for f_1, f_2. Then it is easily seen that the lemma holds for $f_1 \wedge f_2$, $\neg f_1$. We must prove that the lemma holds for $f = \mathbf{F}f_1$. Suppose $S, s_j \models \mathbf{F}f_1$. Then there is a state s_l such that $l \geq j$ and $S, s_l \models f_1$. Since s_l is in final(S), there are infinitely many m such that $\xi(s_m) = \xi(s_l)$ and (by induction) $S, s_m \models f_1$. Hence, there is an $m \geq k$ such that $S, s_m \models f_1$; that is $S, s_k \models f$. □

LEMMA 3.2. *Let $S = (s, \xi)$, $T = (t, \pi)$ be structures such that length(init(S)) = length(init(T)) for all $j <$ length(init(S)); $\xi(s_j) = \pi(t_j)$; $\xi(s_0) = \pi(t_0)$ (this is necessary for the case in which length(init(S)) = 0) and range(S) = range(T); then for all $f \in L(\mathbf{F})$, $S, s_0 \models f$ iff $T, t_0 \models f$.*

Lemma 3.2 can also be proved by induction on f, and it states that formulas in L(**F**) cannot distinguish the order of occurrence of states in final(S).

Let $s = (s_0, s_1, \ldots)$, $t = (t_0, t_1, \ldots)$ be finite or infinite sequences with all states in s, t being distinct. t *is a subsequence of* s (written $t \leq s$) iff there exists integers $i_0, i_1, \ldots$ such that $i_0 < i_1 < i_2 < \cdots$ and, for all $j \geq 0$, $s_{i_j} = t_j$. Let $S = (s, \xi)$ be a structure and $t \leq s$. t is an *acceptable subsequence of* s (written $t \preceq s$) if any s_j in final(S) is contained in t; then every state s_k in final(S) such that $\xi(s_k) = \xi(s_j)$ is also contained in t. We assume that the structure with respect to which $\preceq$ is defined is understood from the context. For notational convenience, we define *init*, *final*, *range*, and *size* for acceptable subsequences also. Let $t \preceq s$. We define init(t) to be the longest prefix of t that is a subsequence of init(S) and final(t) to be the suffix of t starting from the state appearing immediately after init(t). Range(t), size(t) are defined exactly the same as the corresponding definitions for a structure. Note that init(t), final(t), range(t), and size(t) are defined with respect to S, which we assume is understood from the context. Let u_1, u_2 be acceptable subsequences of s and let u be the subsequence containing exactly those states appearing in u_1, u_2 (there is only one such u because all states in s are distinct); then u is an acceptable subsequence of s and size(u) < size(u_1) + size(u_2).

Let $\mu(f)$ denote the number of occurrences of the symbol **F** in the formula f. Using deMorgan's laws and the identities $\neg\mathbf{F}g = \mathbf{G}\neg g$, $\neg\mathbf{G}g = \mathbf{F}\neg g$, any formula $f' \in L(\mathbf{F})$ can be converted to an equivalent formula f in which all negations apply to atomic propositions only and by at most doubling the length of the formula.

LEMMA 3.3. *Let $S = (s, \xi)$ be a structure and let $t \preceq s$ by such that, for all $j \geq 0$, $S, t_j \models f$ where $f \in L(\mathbf{F}, \mathbf{G})$ and in which all negations are applied to atomic propositions only. Then (a) there exists a u such that $u \preceq s$, size(u) $< \mu(f)$, and (b) for all structures $W = (w, \varphi)$, where $u \leq w \preceq s$, φ is the restriction of ξ to the states in w, the following condition holds:*

For any i, if w_i is present in t, then $W, w_i \models f$.

PROOF. The proof is by induction on the structure of the formula f.

Basis: $f = P$ or $\neg P$. u = null sequence satisfies the lemma.

Induction:

(i) $f = f_1 \vee f_2$. Let t' be the subsequence of t containing all t_j such that $S, t_j \models f_1$ and t'' be the subsequence of t containing all t_j such that $S, t_j \models f_2$. By induction hypothesis, there exist u_1, u_2 such that size(u_1) $\leq \mu(f_1)$, size(u_2) $\leq \mu(f_2)$, and (b) holds for t', u_1, f_1 and t'', u_2, f_2. Let $u \preceq s$ be the sequence containing the states of u_1 and u_2. Then Size(u) $\leq$ size(u_1) + size(u_2) $\leq \mu(f)$, and it is easily seen that (b) holds for u, f.

(ii) $f = f_1 \wedge f_2$. The argument is similar to (i).

(iii) $f = \mathbf{F}f_1$.

Case 1: *t is finite.* Let t_n be the last state of t. $S, t_n \models \mathbf{F}f_1$. Hence, there is an s_j appearing after t_n in s such that $S, s_j \models f_1$. If s_j is in init(S), then let $t' = (s_j)$; *otherwise*, let t' be the subsequence of all states s_k in final(S), such that $\xi(s_k) = \xi(s_j)$. For all s_k in t', $S, s_k \models f_1$. By induction hypothesis, there is a $u' \preceq s$ such that size(u') $\leq \mu(f_1)$ and (b) is satisfied for u', f_1, with $t = t'$. Now let $u \preceq s$ be the sequence containing all states of u' and t'. Then size(u) $\leq$ size(t') + size(u') $\leq 1 + \mu(f_1) = \mu(f)$ and (b) holds.

Case 2: *t is infinite.* There exist infinitely many k such that $S, s_k \models \mathbf{F}f_1$. Let t' be the infinite sequence of states in final(S) such that for all $j \geq 0$ $S, t'_j \models f_1$ and $\xi(t'_j) = \xi(t'_{j+1})$. Clearly, size($t'$) = 1. The remainder of the argument is same as in Case 1.

(iv) $f = \mathbf{G}f_1$. If t_0 is in init(t), then let t' = suffix of s starting from t_0; otherwise, let t' = final(S). Clearly, for all $j \geq 0$, $S, t'_j \models f_1$. By induction hypothesis, there is a $u' \preceq s$ such that size(u') $\leq \mu(f_1)$ and (b) holds for u', f_1 and with $t = t'$. Since t' is a suffix of s, it is easily observed that (b) holds for f and t with $u = u'$. □

THEOREM 3.4. *If $f \in L(\mathbf{F})$ is satisfiable, then there exists a structure $S = (s, \xi)$ such that size(S) $\leq$ length(f) and $S, s_0 \models f$.*

PROOF SKETCH. Assume $f \in L(\mathbf{F})$ and is satisfiable. If neither of the symbols $\neg$, $\wedge$ appear in f, then f is either an atomic proposition or is a sequence of $\mathbf{F}$ symbols followed by an atomic proposition. In this case there is a trivial structure of size 1 in which f is true. Now consider the case in which there is at least one occurrence of either $\neg$ or $\wedge$. It is easily seen that length(f) $\geq \mu(f) + 2$. Let $V = (v, \varphi)$ be a structure such that $V, v_0 \models f$. We repeatedly apply deMorgan's laws and the identities $\neg\mathbf{F}h = \mathbf{G}\neg h$, $\neg\mathbf{G}h = \mathbf{F}\neg h$ to f until we get a formula g in which all negations are applied to the atomic propositions only. It is easily seen that $\mu(g) \leq \mu(f)$. Let t be the sequence given as follows. If length(init(v)) > 0, then $t = (v_0)$; otherwise, t is the sequence containing all states v_i such that $\varphi(v_i) = \varphi(v_0)$. Clearly, $t \preceq v$, size(t) = 1, and, owing to Lemma 3.1, for all $i \geq 0$, $V, t_i \models f$. Now, applying Lemma 3.3 for g with $S = V$, we get a sequence $u \preceq v$, such that size(u) $\leq \mu(f)$ and u satisfies the condition given in Lemma 3.3. Let $s \preceq v$ be the sequence containing all the states of t and u. Then $s \preceq v$ and size(s) $\leq$ size(t) + size(u) $\leq 1 + \mu(f)$. If s is a finite sequence, then extend it by adding to it a suffix x such that all states in x are in final(V) and all these states satisfy the same set of atomic propositions. Let $S = (s, \xi)$, where ξ is the restriction of φ to the states appearing in s. Then, from Lemma 3.3, $S, s_0 \models f$. Clearly, size(S) $\leq \mu(f) + 2 \leq$ length(f). □

THEOREM 3.5. *The following problems are NP-complete for the logic $L(\mathbf{F})$:*

(*i*) *determination of truth in an R-structure,*
(*ii*) *satisfiability.*

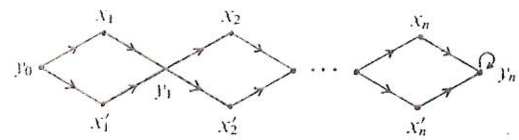

FIGURE 1

PROOF

(i) We prove that determining truth in an R-structure is NP-hard by reducing 3-SAT to this problem. Let $g = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ be a Boolean formula in 3-CNF where $C_i = l_{i1} \vee l_{i2} \vee l_{i3}$ (for $1 \le i \le m$), $l_{ik} = x_j$ or $\neg x_j$ ($1 \le k \le 3$) for some j such that $1 \le j \le n$. $x_1, x_2, \ldots, x_n$ are the variables appearing in g. Let $T = (N, R, \eta)$ be the R-structure defined as follows:

$$\mathscr{P} = \{C_i \mid 1 \le i \le m\},$$
$$N = \{x_i \mid 1 \le i \le n\} \cup \{x_i' \mid 1 \le i \le n\} \cup \{y_i \mid 0 \le i \le n\},$$
$$R = \{(y_{i-1}, x_i), (y_{i-1}, x_i'), (x_i, y_i), (x_i', y_i) \mid 1 \le i \le n\} \cup \{(y_n, y_n)\},$$
$$\eta(x_i) = \{C_j \mid x_i \text{ appears as a literal in } C_j, \text{ i.e., for some } k,\ 1 \le k \le 3,\ l_{jk} = x_i\},$$
$$\eta(x_i') = \{C_j \mid \neg x_i \text{ appears as a literal in } C_j\},$$
$$\eta(y_j) = \varnothing.$$

T can be described by the graph shown in Figure 1.

It can easily be proved that g is satisfiable iff there exists a path p in T starting from y_0 such that $(S_p, s_0) \models \mathbf{F}C_1 \wedge \mathbf{F}C_2 \wedge \cdots \wedge \mathbf{F}C_m$. This reduction is a polynomial reduction. Hence, determination of truth in an R-structure is NP-hard for the language L(**F**). □

Let $T = (N, R, \eta)$ be an R-structure. Any path q in T can be uniquely decomposed into q', q'' such that $q = q' \cdot q''$, any state that appears in q'' appears in it infinitely often, and q'' is the maximal such suffix. All the states in q'' belong to a strongly connected component in the graph of T. Let $S_q, s_0 \models f$. From Lemma 3.3, it follows that there is a subsequence u of q', such that length(u) $\le$ length(f) so that any path $p = p' \cdot q''$, where p' contains u as a subsequence and is obtained from q' by deleting some cycles, has the property that $S_p, s_0 \models f$.

Now, it easily follows that there is a path p in T starting from q_0 such that $S_p, s_0 \models f$ and $p = p' \cdot q''$ where length(p') $\le$ length(f)$\cdot$card(N). From Lemma 3.1, it follows that, if s_i, s_j are any two states in S_p corresponding to the same node in q'', then s_i, s_j in S_p satisfy the same formulas. From Lemma 3.2 it is easily seen that, if r is any path that has p' as a prefix followed by a suffix that contains the same nodes as in q'' repeating infinitely often, then $S_r, s_0 \models f$. Using these facts, we present the following nondeterministic algorithm to verify that there is a path p in T starting from q_0 such that $S_p, s_0 \models f$. A nondeterministic Turing machine (TM) M guesses p' and the set C of states appearing in q''. Next, it verifies that p' is a finite path starting from q_0 in T, that the subgraph containing nodes of C is strongly connected, and that there is an edge from the last state of p' to a state in C. Then M uses the following algorithm to verify if $(S_p, s_0) \models f$. For each node x in p' or C, M maintains a set label(x) that, at the end, contains the set of subformulas of f true at x and that is initially set to the empty set. For each $g \in$ SF(f) in the increasing order of length(g) and for each node x, label(x) is updated using the following rules:

(1) Add $g = P$ to label(x) iff $P \in \eta(x)$.
(2) Add $g = \neg g_1$ to label(x) iff $g_1 \notin$ label(x).
(3) Add $g = g_1 \wedge g_2$ to label(x) iff $g_1, g_2 \in$ label(x).
(4) Add $g = \mathbf{F}g_1$ to label(x) iff $g_1 \in$ label(y) for some $y \in C$ or x is in p' and there is a node y in p' after x such that $g_1 \in$ label(y).

M accepts iff $f \in \text{label}(q_0)$ at the end of this procedure.

It can easily be shown that the previous algorithm works correctly and that it is polynomial-time bounded in card(N) + length(f). Thus, determination of truth in an R-structure is NP-complete.

(ii) Satisfiability is NP-hard because Boolean satisfiability is NP-hard.

Left $f \in L(\mathbf{F})$. From Theorem 3.4, if f is satisfiable, then it is satisfiable in structure $S = (s, \xi)$, where size(S) $\leq$ length(f). A nondeterministic TM M that checks for satisfiability of f operates as follows: M guesses init(S) and range(S) such that length(init(S)) $\leq$ length(f), card(range(S)) $<$ length(f). Next it uses a labeling algorithm similar to the previous one to accept or reject f. Clearly, M is polynomial time bounded in length(f). □

It is to be noted that Lemmas 3.1 and 3.2 are used in the proof of Theorem 3.5. The satisfiability problem for L(**F**) is also shown to be in NP in [4] by proving a linear-size model theorem. However, our techniques are different from those used in [4], and these techniques are also used for the problem of truth in an R-structure.

LEMMA 3.6. *Let $f \in \tilde{L}(\mathbf{F}, \mathbf{X})$ and $S = (s, \xi)$ be such that $S, s_i \models f$; then there exists a finite sequence u containing s_i such that $u \leq s$, length(u) $\leq$ length(f), and for all structures $V = (v, \varphi)$ where $u \leq v \leq s$ and φ is a restriction of ξ, $V, s_i \models f$.*

PROOF. The lemma is proved by induction on the structure of f. The lemma is easily seen to hold for the base cases when f is an atomic proposition or negation of an atomic proposition. We prove the induction step for the case when $f = \mathbf{X}f_1$. Let $f = \mathbf{X}f_1$ and $S, s_i \models f$. Then $S, s_{i+1} \models f_1$. Let u' be the sequence corresponding to f_1 given by the induction hypothesis and u be the sequence starting with s_i and followed by u'. (Note that any two successive states in s also appear successively in any subsequence of s containing them.) It is easily seen that u satisfies the lemma for f. The proof for the case when $f = \mathbf{F}f_1$ is exactly similar, and for the other cases in which $f = f_1 \wedge f_2$ or $f = f_1 \vee f_2$, the proof is easily seen. □

THEOREM 3.7. *The following problems are NP-complete for $\tilde{L}(\mathbf{F}, \mathbf{X})$ also:*

(*i*) *determination of truth in an R-structure,*
(*ii*) *satisfiability.*

PROOF. Determination of truth in an R-structure is shown to be NP-hard by the same proof given for L(**F**). This is because the formulas used in this proof are also formulas in $\tilde{L}(\mathbf{F}, \mathbf{X})$. Satisfiability is NP-hard since 3-SAT is NP-hard.

From Lemma 3.6, it easily follows that a formula $f \in \tilde{L}(\mathbf{F}, \mathbf{X})$ is satisfiable iff it is satisfiable in a structure S with size(S) $\leq$ length(f). Using the same techniques given in the proof of Theorem 3.5 and using Lemma 3.6, it is straightforward to see that the two problems in Theorem 3.7 are in NP for $\tilde{L}(\mathbf{F}, \mathbf{X})$. □

4. *The Complexity of $L(\mathbf{F}, \mathbf{X})$, $L(\mathbf{U})$, and $L(\mathbf{U}, \mathbf{S}, \mathbf{X})$*

The main results of this section are summarized in the following theorem.

THEOREM 4.1. *The following problems are PSPACE-complete for the logics $L(\mathbf{F}, \mathbf{X})$, $L(\mathbf{U})$, and $L(\mathbf{U}, \mathbf{S}, \mathbf{X})$:*

(*i*) *satisfiability,*
(*ii*) *determination of truth in an R-structure.*

The proof of the Theorem 4.1 is based on the following lemmas.

Let $S = (s, \xi)$, $T = (t, \pi)$ be structures such that, for some $m \geq 0$, the following conditions are satisfied:

$$\forall i, 0 \leq i \leq m, t_i = s_i; \qquad \forall i, i > m + 1, t_i = s_{i-1},$$

and π is an extension of ξ such that

$$\pi(t_{m+1}) = \pi(t_m),$$

that is, T is obtained by duplicating the mth state in S successively once. The following lemma is easily proved by induction on the formula f.

LEMMA 4.2. *For any $f \in L(\mathbf{U})$, $T, t_m \models f$ iff $T, t_{m+1} \models f$ and for any δ in s, $S, \delta \models f$ iff $T, \delta \models f$.*

Lemma 4.2 states that, by duplicating a state successively, we do not change the truth value of a formula in L(**U**). Note that the lemma is not true for L(**U**, **X**).

LEMMA 4.3. *Determining truth in an R-structure is polynomial-time reducible to the satisfiability problem for L(**F**, **X**), L(**U**), and L(**U**, **S**, **X**).*

PROOF. Let $T = (N, R, \eta)$ be an R-structure and let $f \in$ L(**U**, **S**, **X**). Let $\mathscr{P}_2$ be the set of atomic propositions appearing in f. Let $\mathscr{P}_1 = \{P_x \mid x \in N\}$ and $\mathscr{P}_1 \cap \mathscr{P}_2 = \emptyset$. $\mathscr{P}_1$ contains one new atomic proposition for each state in N.

Let $x \in N$ and g_1 be the conjunction of all Q such that $Q \in \eta(x)$, let $Q \in \mathscr{P}_2$, g_2 be the disjunction of all Q such that $Q \in (\mathscr{P}_2 - \eta(x))$, and let g_3 be the disjunction of all P_y such that $(x, y) \in R$ and

$$f_x = P_x \supset (g_1 \wedge \neg g_2 \wedge \mathbf{X} g_3).$$

Let h_1 be the disjunction of all P_x such that $x \in N$, let h_2 be the conjunction of all f_x such that $x \in N$, and let h_3 assert that exactly one proposition in $\mathscr{P}_1$ is true at any point and

$$f' = \mathbf{G}(h_1 \wedge h_2 \wedge h_3).$$

Any structure $T = (t, \pi)$ such that $T, t_0 \models f'$ has the following property. At each state in t, exactly one proposition in $\mathscr{P}_1$ is true. If P_x is true at a state, then all propositions in $\eta(x)$ are true in that state, all propositions in $(\mathscr{P}_2 - \eta(x))$ are false in that state, and in the next state P_y is true for exactly one y such that $(x, y) \in R$. Let $q \in N$ and $f'' = f' \wedge f \wedge P_q$. It can easily be seen that there is a path p in S starting from q such that $S_p, s_0 \models f$ iff f'' is satisfiable. If $f \in$ L(**F**, **X**), then $f'' \in$ L(**F**, **X**).

In f', we can avoid the **X** operator using the **U** operator in the following way. We replace the formula $\mathbf{X} g_3$ by g' defined as follows:

$$g' = \begin{cases} (P_x \mathbf{U} g_3), & \text{if } (x, x) \notin R, \\ (\mathbf{G}(P_x) \vee [P_x \mathbf{U} (g_3 \wedge \neg P_x)]), & \text{otherwise.} \end{cases}$$

If $(x, x) \notin R$, then g' causes P_x to repeat successively a finite number of times before P_y is true for some y that is a neighbor of x. However, Lemma 4.2 states that this does not change the truth value of the formula f; that is, f is true in a structure in which P_x does not repeat iff it is true in a structure in which P_x repeats successively a finite number of times. It is easily seen that there is a path p in T starting from q such that $S_p, s_0 \models f$ iff f'' is satisfiable. These reductions are clearly polynomial reductions. □

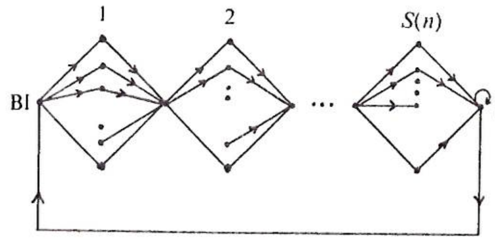

FIGURE 2

LEMMA 4.4. *Determination of truth in an R-structure is PSPACE-hard for* $L(\mathbf{F}, \mathbf{X})$ *and* $L(\mathbf{U})$.

PROOF. Let $M = (Q, \Sigma, \zeta, V_A, V_R, V_I)$ be a one-tape deterministic TM where Q is the set of states, Σ is the alphabet, $\zeta: Q \times \Sigma \to Q \times \Sigma \times \{L, R\}$, V_A, V_R, V_I are the accepting, rejecting and initial states, respectively. Let M be $S(n)$-space bounded such that $S(n)$ is bounded by a polynomial in n. We assume that after M enters the state V_A or V_R it remains in that state without updating the contents of the tape. An id of M is appropriately defined. Let $a = a_1 a_2 \cdots a_n$ be an input to M.

Let $T = (N, R, \eta)$ be an R-structure shown in Figure 2.

Let $\mathscr{P} = \{P_\sigma \mid \sigma \in (Q \times \Sigma) \cup \Sigma\} \cup \{\text{BI}, \text{EI}\}$ be the set of atomic propositions. The structure in Figure 2 has $S(n)$ diamonds connected in a chain, and, in each diamond, there are card$(Q \times \Sigma \cup \Sigma)$ number of vertical vertices. In each diamond, on each vertical vertex, exactly one atomic proposition is true, and every atomic proposition of the form P_σ, where $\sigma \in (Q \times \Sigma \cup \Sigma)$, is true on some vertical vertex of the diamond. Each subpath between BI and EI represents an id of M, and a path from BI represents a sequence of ids of M.

Using $2S(n)$ $\mathbf{X}$ operators, the relation between the contents of a tape cell in successive ids can be asserted. Because of this, polynomial-length bounded formulas in $L(\mathbf{F}, \mathbf{X})$ can be obtained asserting the following conditions: All the ids on a path p starting from BI are valid, the first id is the initial id containing the input string $a_1 a_2 \cdots a_n$, each successive id follows from the previous one-by-one move of M, and the final id appears on the path.

Let f_a be the conjunction of formulas asserting these conditions. It is easily seen that there is a path p from BI in T such that $S_p, s_0 \models f_a$ iff M accepts a. For any input a, f_a can be obtained in polynomial time. We briefly sketch how we can avoid the use of $\mathbf{X}$ operator in f_a by using $\mathbf{U}$ operator. We introduce additional propositions $Q_0, Q_1, \ldots, Q_{s(n)}$ to mark the left and right end points of successive diamonds. Let e_1 be the disjunction of all Q_i for $0 \le i \le s(n)$. The position of the ith cell in an id is indicated by the truth of the formula $g_i = (\neg e_1 \mathbf{U} Q_i)$; that is, g_i is true at a state iff that state corresponds to the ith cell in an id. Let e_2 be the disjunction of all P_σ such that $\sigma \in Q \times \Sigma$. The following formula asserts that each id has exactly one composite symbol:

$$\mathbf{G}[\text{BI} \supset \text{BI}\ \mathbf{U}\ ((\neg e_2 \wedge \neg \text{BI})\ \mathbf{U}\ (e_2 \wedge d_1))],$$

where

$$d_1 = \neg e_1\ \mathbf{U}\ (e_1 \wedge (\neg e_2\ \mathbf{U}\ \text{BI})).$$

The formula $[\neg g_i\ \mathbf{U}\ (g_i \wedge P_\sigma)]$ asserts that the ith cell in the first id contains the symbol σ. From this it is straightforward to see how we can assert that the first id is the initial id. Let h_1, h_2 be propositional formulas. Then the following formula asserts that, if the state corresponding to the ith cell in an id satisfies h_1, then the state corresponding to the jth cell in the next id satisfies h_2:

$$G[(g_i \wedge h_1) \supset (\neg BI\ \mathbf{U}\ d_2)],$$

where

$$d_2 = BI \wedge [\neg g_j\ \mathbf{U}\ (g_j \wedge h_2)].$$

From this it is easy to see how we can assert that the contents of successive ids are properly related. Now it is straightforward to see how the formula f_a can be obtained. The resulting formula $f_a \in L(\mathbf{U})$. □

Let $S = (s, \xi)$ be a structure and $f \in L(\mathbf{U}, \mathbf{S}, \mathbf{X})$. For any state s_i in s, let $[s_i]_{S,f} = \{g \in SF(f) \mid S, s_i \models g\}$. Note that the number of such subsets $\leq 2^{\text{length}(f)}$.

LEMMA 4.5. *In $S = (s, \xi)$, if s_i, s_j are two states such that $[s_i]_{S,f} = [s_j]_{S,f}$, then for the structure $S' = (s', \xi')$, where $s' = (s_0, s_1, \ldots, s_{i-1}, s_j, s_{j+1}, \ldots)$ and ξ' is restriction of ξ to states in s, the following property holds:*

$$[s_k]_{S,f} = [s_k]_{S',f} \qquad \forall s_k \text{ such that } s_k \text{ is present in } s \text{ and in } s'.$$

Lemma 4.5 can be proved by induction on the length of the formula f.

A formula g is said to be a U-*formula* if it is of the form $g_1\ \mathbf{U}\ g_2$. Let $g = g_1\ \mathbf{U}\ g_2$ be a U-formula such that $S, s_i \models g$. We say that g is *fulfilled* before s_j iff $j \geq i$ and there is an l such that $i \leq l \leq j$ and $S, s_l \models g_2$. A structure $S = (s, \xi)$ is said to be *ultimately periodic* with starting index i and period m if $\forall k \geq i\ \xi(s_k) = \xi(s_{k+m})$.

LEMMA 4.6. *For the structure $S = (s, \xi)$ let i, p be intergers such that $p > 0$, $[s_i]_{S,f} = [s_{i+p}]_{S,f}$, and every U-formula in $[s_i]_{S,f}$ is fulfilled before s_{i+p}. Let $S' = (s', \xi')$ be an ultimately periodic structure with starting index i and period p such that $\forall k < i + p\ \xi(s_k) = \xi'(s'_k)$. Then for any $g \in SF(f)$, the following conditions hold:*

(*a*) $\forall k < i + p\ S, s_k \models g$ *iff* $S', s'_k \models g$,
(*b*) $\forall k \geq i\ S', s'_k \models g$ *iff* $S', s'_{k+p} \models g$.

PROOF. We prove (a), (b) by structural induction on g.

Basis: If g is atomic, then (a), (b) follow trivially.

Induction: Assume (a), (b) hold for $g_1, g_2 \in SF(f)$. By a simple argument, it can easily be shown that (a), (b) hold for $g = \neg g_1$, $g_1 \wedge g_2$. In Case 1 and Case 2, we prove that (a), (b) hold for $g = g_1\ \mathbf{U}\ g_2$, $g_1\ \mathbf{S}\ g_2$: a similar argument can be given for $g = \mathbf{X} g_1$.

Case 1: $g = g_1\ \mathbf{U}\ g_2$. We prove (a). (b) can be proved similarly. Assume for some $k < i + p$ $S, s_k \models g$. Assume $k < i$. From the hypothesis of the lemma it follows that, for some l such that $k \leq l < i + p$, $S, s_l \models g_2$ and $\forall j\ k \leq j < l$, $S, s_j \models g_1$. By the induction hypothesis, the above property holds for S' also. Hence, $S', s'_k \models g$. Now assume $i \leq k < i + p$. The interesting case occurs when $\forall j\ k \leq j < i + p$, $S, s_j \models \neg g_2$, $S, s_j \models g_1$. In this case, $S, s_{i+p} \models g$ and, hence, $S, s_i \models g$. From the hypothesis of the lemma and the induction hypothesis for (b), it can easily be seen that $S', s'_k \models g$. The implication in the other direction can also be proved similarly.

Case 2: $g = g_1 \mathbf{S}\, g_2$. Then for $k < i + p$ $S\, s_k \models g$

iff $(\exists l \le k\ S,\ s_l \models g_2$ and $\forall j\ l < j \le k\ S,\ s_j \models g_1)$

iff $(\exists l \le k\ S',\ s'_l \models g_2$ and $\forall j\ l < j \le k\ S',\ s'_j \models g_1)$

(due to induction hypothesis)

iff $S',\ s'_k \models g$.

We would like to prove that (b) also holds for g. Assume for $k \ge i$, $S',\ s'_k \models g$. Then there exists $l \le k$ such that $S',\ s'_l \models g_2$ and for all j, such that $l < j \le k$, $S',\ s'_j \models g_1$. For $k \ge i + p$ or ($k < i + p$ and $l \ge i$), the result can easily be seen. So we consider the case in which $l < i \le k < i + p$. In this case, owing to the induction hypothesis for (a), it can be seen that $S,\ s_l \models g_2$ and for all j such that $l < j \le i$ $S,\ s_j \models g_1$. Hence, $S,\ s_i \models g$. Owing to the hypothesis of the lemma, we see that $S,\ s_{i+p} \models g$. Thus, one of the following two cases holds:

(i) $\exists m(k < m < i + p$ *and* $S,\ s_m \models g_2$ *and* $\forall j$ *such that* $m < j \le i + p$ $S,\ s_j \models g_1)$.

By the induction hypothesis for (a), Condition (i) is also satisfied by S'. Owing to the induction hypothesis for (b), it follows that for all j such that $i + p \le j \le k + p$ $S',\ s'_j \models g_1$. Hence, $S',\ s'_{k+p} \models g$.

(ii) $\forall j\ i \le j \le i + p$ $S,\ s_j \models g_1$.

Owing to the induction hypothesis for (a), Condition (ii) holds for S' also. Owing to the induction hypothesis for (b),

$$\forall j\ k \le j \le k + p\ S',\ s'_j \models g_1.$$

Hence, $S',\ s'_{k+p} \models g$. The induction step for the reverse implication in (b) can be similarly proved. □

THEOREM 4.7 (ULTIMATELY PERIODIC MODEL THEOREM). *A formula* $f \in L(\mathbf{U}, \mathbf{S}, \mathbf{X})$ *is satisfiable iff it is satisfiable in an ultimately periodic structure* $S = (s, \xi)$ *with starting index* $l \le 2^{1+length(f)}$ *period* $p \le 4^{1+length(f)}$ *and* $\forall k \ge l[s_k]_{S,f} = [s_{k+p}]_{S,f}$, *and every U-formula in* $[s_k]_{S,f}$ *is fulfilled before* s_{k+p}.

PROOF. Let f be a satisfiable formula. Since f may refer to the past, it may not be satisfiable at the beginning of a structure. For this reason we consider $g = \mathbf{F}f$. Clearly, there exists a structure $T = (t, \eta)$ such that $T,\ t_0 \models g$. Let l, m be integers such that $[t_l]_{T,g} = [t_{l+m}]_{T,g}$ and the condition (*) holds:

(*) Every U-formula in $[t_l]_{T,g}$ is fulfilled before t_{l+m}.

It is easily seen that l, m exist. Now we repeatedly apply the reductions of Lemma 4.5 to states between t_0 and t_l, or to states between t_l and t_{l+m} (excluding t_0, t_l, t_{l+m}) without violating (*), until no more such reductions are possible. In the resulting sequence,

(a) there are at most $2^{\text{length}(g)}$ states before t_l,
(b) there are at most $\text{length}(g) \cdot 2^{\text{length}(g)}$ states between t_l and t_{l+m}.

(a) follows trivially if we observe that, in the resulting sequence, there are no two states before t_l that satisfy exactly the same set of subfomulas of g. If (b) does not hold, then there exist at least length(g) + 1 states between t_l and t_{l+m} that satisfy the same set of subformulas of g; that is, there exist at least length(g) intervals between these states. It is easily seen that there exist at least one interval among these, such that every U-formula in SF(g) that is satisfied at s_l is fulfilled at a state outside this interval and between t_l and t_{l+m}, since there are fewer than length(g) U-formulas in SF(g). Hence, we could have carried a reduction of Lemma 4.5 for the two end states of this interval without violating (*). This contradicts our assumption.

Let t' be the resulting sequence after the reductions, and let $T' = (t', \eta')$ be the structure, where η' is the restriction of η to the states in t'. There exist integers $i \leq 2^{\text{length}(g)}$, $p \leq 4^{\text{length}(g)}$ such that t'_i, t'_{i+p} satisfy the same subformulas of g; any U-formula in SF(g) that holds at t'_i is fulfilled before t'_{i+p}. Using Lemma 4.6, we obtain a periodic structure S with starting index $i \leq 2^{\text{length}(g)}$, period $p \leq 4^{\text{length}(g)}$ satisfying the condition of the lemma, such that $S, s_0 \models g$. □

In [2], a theorem similar to the above is proved for a restricted version of PDL.

Proof of Theorem 4.1. Let f be a formula in L(**U**, **S**, **X**). As above, we consider $g = \mathbf{F}f(=\text{True } \mathbf{U} f)$. Note that f is satisfiable iff g is satisfiable at the beginning of an ultimately periodic structure. We describe below a nondeterministic TM M that checks for satisfiability of g. M guesses two numbers $n_1 \leq 2^{\text{length}(g)}$, $n_2 \leq 4^{\text{length}(g)}$, which are supposed to be the starting index and period of an ultimately periodic structure. Next, M guesses the subformulas that are true at the beginning and verifies that g is in this set. At this point, it checks for Boolean consistency, and it checks that any subformula $f_1 \mathbf{S} f_2$ is in this set iff f_2 is in this set.

Subsequently, M guesses the subformulas that are true in the next state and verifies their consistency with the subformulas that are guessed to be true in the present state. If $\text{Sub}_{\text{present}}$, Sub_{next} are the formulas guessed to be true at the present state and the next state, respectively, it verifies that the following conditions hold for each $g \in \text{SF}(f)$:

$$
\begin{array}{lll}
g = \mathbf{X} f_1 \in \text{Sub}_{\text{present}} & \text{iff} & f_1 \in \text{Sub}_{\text{next}}, \\
g = f_1 \mathbf{U} f_2 \in \text{Sub}_{\text{present}} & \text{iff} & f_2 \in \text{Sub}_{\text{present}} \\
 & & \text{or} \quad (f_1 \in \text{Sub}_{\text{present}} \text{ and } f_1 \mathbf{U} f_2 \in \text{Sub}_{\text{next}}), \\
g = f_1 \mathbf{S} f_2 \in \text{Sub}_{\text{next}} & \text{iff} & f_2 \in \text{Sub}_{\text{next}} \\
 & & \text{or} \quad (f_1 \in \text{Sub}_{\text{next}} \text{ and } f_1 \mathbf{S} f_2 \in \text{Sub}_{\text{present}}).
\end{array}
$$

It also checks the Boolean consistency whenever it guesses a set of subformulas to be true at any state; that is, for each $g \in \text{SF}(f)$,

$$
\begin{array}{lll}
g = f_1 \wedge f_2 \in \text{Sub}_{\text{present}} & \text{iff} & f_1, f_2 \in \text{Sub}_{\text{present}}; \\
g = \neg f_1 \in \text{Sub}_{\text{present}} & \text{iff} & f_1 \notin \text{Sub}_{\text{present}}.
\end{array}
$$

M continues this process, each time incrementing the counter. When the counter is n_1, it notes that it is in the periodic part of the structure. It saves the set of subformulas $\text{Sub}_{\text{period}}$, guessed to be true at the beginning of the period, and it reinitializes the counter. It continues guessing the subformulas in the next state and incrementing the counter. At each instance, it has to keep three sets of subformulas: those that are true in present state, those true in the next state, and those true at the beginning of the period. When the counter has value n_2, it stops guessing and takes $\text{Sub}_{\text{period}}$ to be the set of subformulas true in the next state. At each step in this procedure, it checks the consistency of the subformulas guessed. It also verifies the following condition. Each formula of the form $(f_1 \mathbf{U} f_2) \in \text{Sub}_{\text{period}}$ is eventually fulfilled within the period, that is f_2 is present in the set of subformulas guessed to be true somewhere within the period. It can easily be proved by induction that M accepts an input formula iff it is satisfiable. Clearly, M uses space linear in length(f). Using Savitch's theorem [8], it follows that there is a polynomial space-bounded deterministic TM that decides satisfiability. □

5. *Complexity of Extensions of the Logic*

In [10], propositional linear temporal logic is extended with the addition of operators corresponding to *regular right linear grammars*. A regular right linear grammar is a regular grammar in which all the production rules are of the form $N \to aM$ or $N \to a$, where N, M are the nonterminals in the grammar and a is a string of terminal symbols. Let R be a regular right linear grammar with terminal symbols $a_1, a_2, \ldots, a_n$ and nonterminal symbols $N_1, N_2, \ldots, N_m$. If $f_1, f_2, \ldots, f_n$ are formulas in the logic, then so is $N_j(f_1, f_2, \ldots, f_n)$ for $1 \le j \le m$. For a structure $S = (s, \xi)$, $S, s_k \models N_j(f_1, f_2, \ldots, f_n)$ iff there exists a finite or infinite string $a_{i_0}, a_{i_1}, a_{i_2}, \ldots$ generated by R from N_j such that, for all $l \ge 0$, $S, s_{l+k} \models f_{i_l}$. If $\alpha = a_{i_0}, a_{i_1}, \ldots$ has the previous property, then we say that α makes $N_j(f_1, \ldots, f_n)$ true at s_k.

For notational convenience, we assume that the nonterminal symbols in different grammars are denoted by different letters with subscripts.

Example. Consider the grammar $N_0 \to a_1\, a_2\, N_0$. It generates the infinite string $a_1a_2a_1a_2 \cdots$. $S, s_0 \models N_0$ (True, P) iff P holds at all even states in s.

For convenience, we assume that each production rule in the grammar has, at most, one terminal symbol. Note that, for any grammar, we can obtain an equivalent grammar with the previous property by increasing the size of the grammar by at most a constant factor. For any formula f in this logic, we define SF(f) inductively as follows:

If $f = P$, then $\mathrm{SF}(f) = \{P\}$,
$f = f_1 \wedge f_2$ or $f_1\, \mathbf{U}\, f_2$ or $f_1\, \mathbf{S}\, f_2$ then $\mathrm{SF}(f) = \mathrm{SF}(f_1) \cup \mathrm{SF}(f_2) \cup \{f\}$,
$f = \neg f_1$ or $\mathbf{X}F_1$ then $\mathrm{SF}(f) = \mathrm{SF}(f_1) \cup \{f\}$,
$f = \mathrm{N}_j(f_1, f_2, \ldots, f_n)$ where N_j is a nonterminal in the previous regular grammar, then $\mathrm{SF}(f) = \mathrm{SF}(f_1) \cup \mathrm{SF}(f_2) \cup \cdots \cup \mathrm{SF}(f_n) \cup \{N_j(f_1, f_2, \ldots, f_n) \mid 1 \le j \le m\}$.

We extend SF(f) by adding to it $\neg g$ for every g present in the previously defined set after eliminating double negations. With the above definition of SF(f), it can easily be seen that Lemma 4.5 holds for this logic. We prove Theorem 4.1 for ETL also.

Let $S = (s, \xi)$ be a structure and $N_i(f_1, f_2, \ldots, f_q)$ be a formula. For $j \ge 0$, we define a labeled tree $\Gamma(S, j, N_i)$ in which each node is labeled with a subformula of the form $N_r(f_1, \ldots, f_q)$ as follows: The root of the tree is labeled with $N_i(f_1, f_2, \ldots, f_q)$. Let x be a node of the tree at a depth l that is labeled with $N_k(f_1, \ldots, f_q)$. For each production rule of the form $N_k \to a_pN_r$, such that $S, s_{j+l} \models f_p$, x has a son with label $N_r(f_1, \ldots, f_q)$.

LEMMA 5.1. *$S, s_j \models \neg N_i(f_1, \ldots, f_q)$ iff there is no finite string that makes $N_i(f_1, \ldots, f_q)$ true at s_j and the tree $\Gamma(S, j, N_i)$ is finite.*

PROOF. Assume $S, s_j \models \neg N_i(f_1, \ldots, f_q)$. Clearly, there is no finite string that makes $N_i(f_1, \ldots, f_q)$ true at s_j. Now contrary to the lemma, assume that $\Gamma(S, j, N_i)$ is infinite. Since this tree is finitely branching from Konig's infinitary lemma, it follows that $\Gamma(S, j, N_i)$ has an infinite path. From this it is easily seen that there is an infinite string $a_{i_0}, a_{i_1}, \ldots$ generated by the grammar starting from N_i such that, for all $l \ge 0$, $S, s_{j+l} \models f_{i_l}$. Hence, $S, s_j \models N_i(f_1, \ldots, f_q)$, which is a contradiction.

Now assume that $\Gamma(S, j, N_i)$ is finite and that there is no finite string that makes $N_i(f_1, \ldots, f_q)$ true at s_j. Now if $S, s_j \models N_i(f_1, \ldots, f_q)$, then there should be an infinite string that makes $N_i(f_1, \ldots, f_q)$ true at s_j and this would make $\Gamma(S, j, N_i)$ infinite, a contradiction. Hence, $S, s_j \models \neg N_i(f_1, \ldots, f_q)$. □

Let $S, s_j \models \neg N_i(f_1, \ldots, f_q)$. Then $\Gamma(S, j, N_i)$ is finite. Let l be the maximum depth of any node in $\Gamma(S, j, N_i)$ and let m be any integer such that $m \geq l + j$. Then we say that *$\neg N_i(f_1, \ldots, f_q)$ at s_j is fulfilled* before s_m.

LEMMA 5.2. *Let S, i, p, f be as in Lemma 4.6, with the additional constraint that for every subformula of the form $\neg N_j(f_1, \ldots, f_q)$ in $[s_i]_{S,f}$, $\neg N_j(f_1, \ldots, f_q)$ at s_i is fulfilled before s_{i+p}. Also let S' be as in Lemma 4.6. Then (a), (b) of Lemma 4.6 also hold for ETL.*

PROOF. We prove (a), (b) of Lemma 4.6 by induction on the structure of g. Clearly, it is enough if we prove the induction step for the case in which $g = N_u(f_1, \ldots, f_q)$, where N_u is a nonterminal in a grammar with terminal symbols $a_1, a_2, \ldots, a_q$, since the other cases are proved in Lemma 4.6. Let $S, s_k \models g$ for $k < i + p$. Assume $k < i$. Now we have the following two cases:

(i) g is satisifed at s_k owing to a string of length $\leq (i - k)$ generated from N_u; that is, there is a finite string $a_{m_0}, a_{m_1}, \ldots, a_{m_{r-1}}$ generated from N_u, where $r \leq (i - k)$ and, for all l, $0 \leq l < r$ S, $s_{(k+l)} \models f_{m_l}$. Using the induction hypothesis for (a), (b), it is easily seen that S' $s'_k \models g$.

(ii) g is satisfied at s_k owing to a string of length $> (i - k)$ generated from N_u. From this it is easily seen that there is a finite string $\alpha = a_{m_0}, a_{m_1}, \ldots, a_{m_{(r-1)}}$ and a nonterminal N_v, where $r = i - k$ such that N_v can be reached from N_u, with the string α using the production rules of the grammar and, for all l $0 \leq l < r$, S, $s_{k+l} \models f_{m_l}$. Now we have two subcases: (a) N_v is satisfied at s_i owing to a finite string $\beta = a_{j_0}, a_{j_1}, \ldots, a_{j_{t-1}}$ of length $\leq p$. In this case, using the induction hypothesis, it is easily seen that S', $s'_k \models g$. (b) N_v is satisfied at s_i owing to a string of length $>p$. In this case, there is a string $\beta = a_{j_0}, a_{j_1}, \ldots, a_{j_{p-1}}$ and a nonterminal N_w such that N_w can be reached from N_v, with the string β using the production rules of the grammar, $S, s_{i+p} \models N_w(f_1, \ldots, f_q)$ and, for all l $0 \leq l < p$, S, $s_{i+l} \models f_{j_l}$. Since s_i, s_{i+p} satisfy the same subformulas, it follows that $S, s_i \models N_w(f_1, \ldots, f_q)$. Now by repeatedly using the previous argument, the following is easily seen: There is a finite or infinite string $\gamma = a_{j_0}, a_{j_1}, \ldots$, generated from N_v such that, for all l, $0 \leq l <$ length of γ, S, $s_{i+l} \models f_{j_l}$ where $t = l \bmod p$. Now from the induction hypothesis for (a), (b) it is seen that for all l $0 \leq l <$ length of γ S', $s'_l \models f_{j_l}$ and hence S', $s'_i \models N_v(f_1, \ldots, f_q)$. From this it follows that S', $s'_k \models g$.

A similar argument can be given for the case when $i \leq k < i + p$.

Assume $S, s_k \models \neg g$, where $k \leq i$. Then clearly $\neg g$ at s_k is fulfilled before s_{i+p}. Hence, by the induction hypothesis for (a), (b) it follows that S', $s'_k \models \neg g$. Assume $i < k < i + p$. The interesting case occurs when $\neg g$ at s_k is not fulfilled before $s_{(i+p-1)}$. Then, let $N_{t_1}, \ldots, N_{t_r}$ be the labels of the nodes at depth $(i + p - k)$ in $\Gamma(S, k, N_u)$. Clearly $\forall v$ such that $1 \leq v \leq r$, S, $s_{i+p} \models \neg N_{t_v}(f_1, \ldots, f_q)$ and hence $S, s_i \models \neg N_{t_v}(f_1, \ldots, f_q)$. Clearly, all these formulas at s_i are fulfilled before s_{i+p}. Now from the induction hypotheses for (a), (b), it easily follows that S', $s'_k \models \neg g$.

It is straightforward to see that (b) holds for g. □

THEOREM 5.3. *A formula f in ETL is satisfiable iff it is satisfiable in an ultimately periodic structure $S = (s, \xi)$ with starting index $l \leq 2^{1+card(SF(f))}$, period $p \leq c^{card(SF(f))}$ for some constant c, and $\forall k \geq l$, $[s_k]_{S,f} = [s_{k+p}]_{S,f}$ and all the*

U-formulas and all subformulas of the form $\neg N_u(f_1, \ldots, f_q)$ *in* $[s_k]_{S,f}$ *are fulfilled before* s_{k+p}.

PROOF. As in the proof of Theorem 4.7, let $g = \mathbf{F}f$ and $T = (t, \eta)$ be such that $T, t_0 \models g$. Let l, m be intergers such that $[t_l]_{T,g} = [t_{l+m}]_{T,g}$ and

(*) Every U-formula in $[t_l]_{T,g}$ is fulfilled before t_{l+m}, and for every subformula of the form $\neg N_r(f_1, \ldots, f_q) \in [t_l]_{T,g}$, $\neg N_r(f_1, \ldots, f_q)$ at t_l is fulfilled before t_{l+m}.

If t_i is a state appearing between t_l and t_{l+m}, then let $C(t_i) = \{N_u(f_1, \ldots, f_q) \mid$ for some subformula $\neg N_r(f_1, \ldots, f_q) \in [t_l]_{T,g}$, $N_u(f_1, \ldots, f_q)$ is the label of a node in $\Gamma(T, l, N_r)$ at a depth $(i - l)\}$. We say that two states t_i, t_j between t_l, t_{l+m} are equivalent if $C(t_i) = C(t_j)$ and $[t_i]_{T,g} = [t_j]_{T,g}$. It can easily be seen that the number of such equivalence classes $\leq 4^{\mathrm{card}(\mathrm{SF}(g))}$. If t_i, t_j are such equivalent states and if we delete all states between t_i and t_j(excluding t_j but including t_i), then, in the resulting structure, $\neg N_r(f_1, \ldots, f_q)$ still holds at t_l, since it remains fulfilled before t_{l+m}. Now we carry out this reduction repeatedly for the states between t_l and t_{l+m} without violating (*) until no more such reductions can be carried out. We also repeatedly carry out the reduction of Lemma 4.5 to states before t_l until no more such reductions can be carried out. In the resulting structure, there are at most $2^{\mathrm{card}(\mathrm{SF}(g))}$ states before t_l and at most $\mathrm{card}(\mathrm{SF}(g)) \cdot 4^{\mathrm{card}(\mathrm{SF}(g))}$ states between t_l and t_{l+m}. The remainder of the proof is same as that for Theorem 4.7. □

THEOREM 5.4. *The following problems are PSPACE-complete for ETL*:

(*i*) *satisfiability*,
(*ii*) *determination of truth in an R-structure.*

PROOF. We first consider the satisfiability problem. We assume that the grammars corresponding to the regular operators are encoded as part of the input. In this case, if the length of the input is n, then $\mathrm{card}(\mathrm{SF}(f)) \leq 2n$, where f is the input formula. We modify the proof of Theorem 4.1 as follows: The two guessed integers n_1, n_2 should be $\leq 2^{2n}$, c^{2n}, respectively, where c is the constant of Theorem 5.3. In addition, the operation of M is to be modified as follows:

> At any time $N_j(f_1, f_2, \ldots, f_q)$ is in the set of subformulas guessed to be true at any state, iff either (1) there is a production rule of the form $N_j \to a_k N_l$ in the grammar, f_k is present in the set of formulas guessed to be true in the present state, and $N_l(f_1, f_2, \ldots, f_q)$ is present in the set of formulas guessed to be true in the next state, or (2) there is a production rule of the form $N_j \to a_k$ and f_k is present in the set of subformulas guessed to be true in the present state.

For each formula of the form $\neg N_j(f_1, f_2, \ldots, f_q)$ present in the set of subformulas guessed to be true at the beginning of the periodic part, M keeps a set of subformulas denoted by $\varphi(N_j(f_1, f_2, \ldots, f_q))$. Roughly, if $N_k \in \varphi(N_j(f_1, \ldots, f_q))$, then $N_k(f_1, \ldots, f_q)$ should be false at the present state. At the beginning of the periodic part, this set contains only N_j. If $\varphi_{\mathrm{present}}$, φ_{next} denote the value of φ in the present and next state, then φ is updated as follows: $\varphi_{\mathrm{next}}(N_j(f_1, f_2, \ldots, f_q)) = \{N_l \mid$ there is a production rule $N_p \to a_k N_l$ in the grammar such that $N_p \in \varphi_{\mathrm{present}}(N_j(f_1, f_2, \ldots, f_q))$ and f_k is present in the set of subformulas guessed to be true in the present state$\}$. M makes sure that $\varphi(N_j(f_1, f_2, \ldots, f_q))$ becomes empty at some point within the periodic part of the structure. This guarantees that $\Gamma(S, i, N_j)$ is finite where S is the guessed periodic structure, and i is the beginning of the period. The consistency checks of the previous paragraph guarantee that there is no finite string that makes $N_j(f_1, \ldots, f_q)$ true at the beginning of the period. Owing to

TABLE I

Logic	Satisfiability	Validity	Truth in an R-Structure
L(**F**) $\tilde{\mathrm{L}}$(**F**, **X**)	NP-complete	Co-NP-complete	NP-complete
L(**F**, **X**) L(**U**) L(**U**, **X**) L(**U**, **S**, **X**) Linear time logic with Regular operators	PSPACE-complete	PSPACE-complete	PSPACE-complete

Lemma 5.1, these conditions guarantee that $\neg N_j(f_1, \ldots, f_q)$ is true at the beginning of the period.

It can easily be proved that M accepts an input formula in the extended logic iff the formula is satisfiable. It is easily seen that M is polynomial-space bounded. The problem of determining truth in an R-structure is in PSPACE, since it is reducible to the satisfiability problem. □

6. *Conclusion*

In this paper we have examined the complexity of satisfiability and truth in a particular structure for various propositional linear temporal logics. We have determined that these problems are NP-complete for L(**F**) and PSPACE-complete for L(**F**, **X**), L(**U**), L(**U**, **S**, **X**), and Wolper's extended logic (see Table I). It should be observed that the satisfiability problem is PSPACE-complete for L(**F**, **X**) whereas it is only NP-complete for $\tilde{\mathrm{L}}$(**F**, **X**), as the later logic does not permit arbitrary alternation of ¬, **F**.

These complexity results first appeared in an early version of this paper [9] in 1982. Subsequently, satisfiability for L(**U**, **X**) was also shown to be in PSPACE in the journal version of [2] by using a different technique that does not work for L(**U**, **S**, **X**). One of the theorems used in [9] to prove that satisfiability for the extended temporal logic of Section 5 is in PSPACE contained an error. The error was independently corrected by us in the version submitted to this journal and by Wolper in the journal version of [10].

Finally, it is interesting to compare our results with the corresponding results for branching-time logics. Since branching-time formulas are interpreted over the states of a structure, rather than over executions sequences, determining truth in a particular structure is much easier and, in many cases, is in polynomial time P [1]. Satisfiability, on the other hand, can be shown to be exponential-time hard for branching-time logics with a next-time operator and is shown to be PSPACE-complete in [3] for many branching time logics with only the **F** and **G** operators. Thus, deciding satisfiability is apparently more difficult for the branching-time logics than for the corresponding linear-time logics.

REFERENCES

1. Clarke, E. M., and Emerson, A. Design and synthesis of programming skeletons using branching time temporal logic. *IBM Conference on Logics of Programs.* Lecture Notes in Computer Science, vol. 131, Springer-Verlag, New York, 1981.
2. Halpern, J. Y., and Reif, J. H. The propositional dynamic logic of deterministic, well-structured programs. In *Proceedings of the 22nd Symposium on Foundations of Computer Science* (Nashville, Tenn.) IEEE, New York, 1981, pp. 322–334. Also in *Theoret. Comput. Sci., 27* (1983), 127–165.

3. LADNER, R. The computational complexity of provability in systems of modal propositional logic. *SIAM J. Comput. 6* (1977), 467–480.
4. ONO, H. AND NAKAMURA, A. On the size of refutation Kripke models for some linear modal and tense logics. *Studia Logica 39* (1980), 325–333.
5. MANNA, Z., AND WOLPER, P. Synthesizing concurrent programs from temporal logic specifications. *IBM Conference on Logics of Programs*. Lecture Notes in Computer Science, vol. 131, Springer-Verlag, New York, 1981.
6. PNUELI, A. The temporal logic of programs. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science* (Providence, R.I.). IEEE, New York, 1977, pp. 46–57.
7. ROBERTSON, E. L. Structure of complexity in weak monadic second order theories of the natural numbers. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science* (Providence, R.I.). IEEE, New York, 1977, pp. 161–171.
8. SAVITCH, W. J. Relationships between nondeterministic and deterministic tape complexities. *J. Comput. Syst. Sci. 4*, 2 (1970), 177–192.
9. SISTLA, A. P., AND CLARKE, E. M. The complexity of propositional linear temporal logics. In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing* (San Francisco, Calif. May 5–7). ACM, New York, 1982, pp. 159–168.
10. WOLPER, P. Temporal logic can be more expressive. In *Proceedings of 22nd Symposium on Foundations of Computer Science* (Nashville, Tenn.). IEEE, New York, 1981, pp. 340–348. Also in *Inf. Control 56* (1983), 72–99.

RECEIVED APRIL 1982; REVISED OCTOBER 1984; ACCEPTED JANUARY 1985