Annotation-Based Deduction in Temporal Logic [‡]

Hugh McGuire*, Zohar Manna*, and Richard Waldinger**

* Computer Science Department, Stanford University, Stanford, California 94305, U.S.A.; mcguire@cs.stanford.edu , manna@cs.stanford.edu .
** Artificial Intelligence Center, SRI International, Menlo Park, California 94025, U.S.A.; waldinger@ai.sri.com .

Abstract. This paper presents a deductive system for predicate temporal logic with induction.
Representing temporal operators by first-order expressions enables temporal deduction to use the already developed techniques of first-order deduction. But when translating from temporal logic to first-order logic is done indiscriminately, the ensuing quantifications and comparisons of time expressions encumber formulas, hindering deduction. So in the deductive system presented here, translation occurs more carefully, via *reification* rules. These rules paraphrase selected temporal formulas as nontemporal first-order formulas with *time annotations*. This time reification process suppresses quantifications (the process is analogous to quantifier skolemization) and uses addition instead of complicated combinations of comparisons. Some ordering conditions on arithmetic expressions can arise, but such are handled automatically by a special-purpose unification algorithm plus a decision procedure for Presburger arithmetic.
This deductive system is relatively complete.

Contents

‡ This research was supported in part by the National Science Foundation under grant CCR-92-23226, by the Advanced Research Projects Agency under contract NAG2-703 and grant NAG2-892, and by the United States Air Force Office of Scientific Research under contract F49620-93-1-0139.

1 Introduction

Temporal logic has been found valuable for the specification of concurrent and reactive software and hardware systems (see, e.g., [MP91]). To use such specifications, however, it is necessary to have techniques for reasoning about temporal-logic formulas.

For example, model-checking or -exploring ('semantic tableau') systems such as [CES86, P86, KM93] are effective for temporal formulas which are essentially propositional. But so far, they have been inapplicable to general predicate temporal formulas. The deductive system of [AM90] extends nonclausal resolution for predicate logic [MW80] to handle temporal operators directly, but that system requires a cut rule which gratuitously introduces new formulas into proofs, and it imposes possibly distracting restrictions on its rules.

An alternative to reasoning directly with temporal operators is to translate temporal formulas into nontemporal first-order predicate logic, using quantifications and comparisons of explicit time parameters to express the temporal operators. Then, one can apply the accumulated technology for reasoning in first-order logic. Such an approach is used in [W89] and [O88] for modal logics other than temporal logic. But performing this translation indiscriminately may confuse otherwise simple modal-logic proofs, in which the modal operators are intuitively meaningful; proofs requiring induction are particularly susceptible to this disservice. (See also [O93] re efficiency.)

This paper presents a deductive system which employs such translation, but with some finesse. Temporal operators are translated gradually and selectively, and the resulting expressions representing time are simple: time is identified with the natural numbers, so temporal operators can be represented by addition (and subtraction). Then, reasoning about terms which represent time is done automatically by unification (as with "t" in the following simple example) plus a decision procedure for Presburger arithmetic derived from [S79].

An example of such a proof, of the formula $(\Box p \to p)$, is:

	Assertions	Goals	Explanations
1		$0 : (\Box p \to p)$	Given
2	$0 : \boxed{\Box p}^{-}$		1, split "$\to$"
3		$0 : \boxed{p}^{+}$	1, split "$\to$"
4	$t : \boxed{p}^{-}$		2, reify "$\Box^{-}$"
5		*true*	3&4, resolution, unifier $\{t \leftarrow 0\}$

This structure formalizes the goal-directed style of proofs that humans prefer to construct and find easiest to understand.

The deductive system here also provides induction for temporal operators, and rules such as rewriting, for convenience. This system subsumes the prior one of [AM90]; i.e., any valid formula which can be proven in that system can also be proven here.

2 The Language: Annotated Temporal Logic (ATL)

The basic language here is the Linear Temporal Logic of [MP91]; the complete language of ATL adds time annotations to formulas. For example, in the ATL formula $[3 : \Box p]$, the numeral "3" is the time annotation. This formula is satisfied if its basic subformula $\Box p$ is satisfied at time 3.

2.1 Syntax

The *basic* language here has function symbols, variable symbols, predicate symbols, the equality symbol "$=$", the six boolean operators "*true*", "*false*", "$\neg$", "$\wedge$", "$\vee$", and "$\rightarrow$", the quantification operators "$\forall$" and "$\exists$", and temporal operators. The temporal operators (which apply to formulas) comprise "$\Box$", "$\Diamond$", "$\bigcirc$", and "$\mathcal{U}$" — named "henceforth", "eventually", "next", and "until", respectively — as well as their 'mirror' operators which refer to the past: "$\boxminus$", "$\diamondminus$", "$\ominus$", and "$\mathcal{S}$" — named "heretofore", "once", "previously", and "since", respectively. The operators "$\mathcal{U}$" and "$\mathcal{S}$" are binary; the other temporal operators are unary. An additional temporal operator is "$\odot$" — named "next-value" — which applies to terms (and is unary). Lastly, re basic symbols, some zero-arity function symbols and all zero-arity predicate symbols (i.e., *propositions*) are *dynamic*, i.e., 'flexible'. (The significance of dynamic symbols is semantic; see below.)

In addition to the preceding basic symbols, the complete language of ATL requires *time symbols*: the natural numerals ("0", "1", etc.), the addition function symbol "$+$", the predecessor function symbol "*pred*", other uninterpreted function symbols (e.g., "h_1", "h_2"), variable symbols (e.g., "t_1", "t_2", "δ_1", "δ_2"), and comparison symbols "$<$" and "$\leq$" (in addition to "$=$"). Time symbols are not dynamic.

Terms and *incomplete* (unannotated) formulas are constructed from basic symbols and time symbols as is customary, applying function symbols, predicate symbols, and operators to appropriate arguments. A *time term* is either a time numeral, a time variable, or the application of a time function symbol to arguments which do not contain "$\odot$" or any dynamic symbols.

An elementary complete annotated formula of ATL has the form "$[\tau : \varphi]$", where τ is a time term — in this context called a *time annotation* — and φ is an incomplete formula. An elementary complete *time formula* is an unannotated comparison of two time terms. Further complete formulas are constructed from these elementary ones using nontemporal boolean operators and quantifiers as is customary (except that quantifiers are not allowed to bind variables in time annotations).

2.2 Semantics

A *model* is a structure with a *domain* — i.e., a nonempty set of *objects* — and a sequence of *states* indexed by the natural numbers $\mathbb{N}$. For an index $n \in \mathbb{N}$, state n of model σ is denoted by the expression "$\langle\sigma, n\rangle$". A model assigns concrete objects, functions, and relations (over the domain) to the language's nondynamic variable symbols, function symbols, and predicate symbols, respectively; a state does the same for the dynamic symbols. A model interprets the time numerals ("0" etc.) as the natural numbers $\mathbb{N}$, "+" as addition, "*pred*" as predecessor,[1] and so on (for the time symbols). For example, a time function symbol (e.g., "h") is interpreted as a function whose range is $\mathbb{N}$.

Interpretation of terms and unannotated formulas is as customary, with functions, operators, quantifiers, etc. An expression which contains no temporal operators or dynamic symbols is interpreted via the global assignments of a model; otherwise, it is interpreted by a state. Some examples for temporal operators are:

$$\langle\sigma, n\rangle \models \bigcirc\varphi \quad \text{iff} \quad \langle\sigma,\, n+1\rangle \models \varphi\,.$$

$$\langle\sigma, n\rangle \models \ominus\varphi \quad \text{iff} \quad 0 < n \ \text{ and } \ \langle\sigma,\, n-1\rangle \models \varphi\,.$$

$$\langle\sigma, n\rangle \models (\varphi\ \mathcal{U}\ \psi) \quad \text{iff} \quad \begin{array}{l}\text{there exists a } k \geq n \text{ such that } \langle\sigma, k\rangle \models \psi, \\ \text{and, for every } i \text{ such that } n \leq i < k,\ \langle\sigma, i\rangle \models \varphi\,.\end{array}$$

$$\langle\sigma, n\rangle \models (\varphi\ \mathcal{S}\ \psi) \quad \text{iff} \quad \begin{array}{l}\text{there exists a } k \leq n \text{ such that } \langle\sigma, k\rangle \models \psi, \\ \text{and, for every } i \text{ such that } k < i \leq n,\ \langle\sigma, i\rangle \models \varphi\,.\end{array}$$

The value of the term $\odot\alpha$ in the nth state of model σ is the value of the term α in the $(n+1)$st state of σ.

A model σ interprets an annotated formula $[\tau : \varphi]$ as follows: Let $n \in \mathbb{N}$ be σ's interpretation of the time annotation τ. Then, $\sigma \models [\tau : \varphi]$ iff $\langle\sigma, n\rangle \models \varphi$.
Interpretation of further complete formulas is then as customary.

A formula is *valid* if it is satisfied by all models.

A *theory* is specified by a set A of formulas which are called the *axioms* of the theory. A formula φ is *valid within the theory* specified by the set A if for every model σ that satisfies all the axioms A, $\sigma \models \varphi$.

3 Deduction I: Fundamentals

Our deductive framework, using deductive tableaux, is that of [MW93].

3.1 Notation: Deductive Tableaux

A *deductive tableau* is a table with four columns and any positive number of rows. The leftmost column simply contains row numbers; the middle two columns, titled "Assertions" and "Goals", contain complete ATL formulas (only one per row); and the rightmost column, titled "Explanations", contains texts

[1] A model's interpretation of "*pred*(0)" is arbitrary.

which explain how rows were derived. Each assertion or goal formula may have one (or more) of its subformulas highlighted with a box (boxes). For an example of a deductive tableau, see the Introduction.

The intuitive interpretation of a deductive tableau[2] is that the column of goals contains a formula being proved valid, followed by reductions of it to more tractable goals, and the column of assertions contains formulas which can be assumed true, to be used in the proof. A box highlights a subformula in a row when the subformula is particularly significant to a deduction rule which is applied to that row in the proof; this subformula is called a *target* of the rule.

Associated with every subformula in a deductive tableau is a characteristic called *polarity*, which is determined by the parity of the number of negations — whether explicit, or implicit via "$\rightarrow$" — within whose scope the subformula lies; one additional negation is implicit for assertions, reflecting their status of 'implying' the goals. Even parity of these negations occasions *positive* polarity, and odd parity occasions *negative* polarity. A superscript of either "$^+$" or "$^-$" on a formula indicates the formula's polarity.

3.2 Outline of the Deductive Scheme

An ATL proof that a basic formula φ is valid begins with the following deductive tableau:

	Assertions	Goals	Explanations
1		$0 : \varphi$	Given

Then, rows are added via rules. The proof succeeds when *true* is derived as a reduced goal formula.

Quasi-deductive primitive operations, acting within rows instead of adding new ones, include renaming variables and automatically simplifying expressions, for example changing $\neg\neg\psi$ to ψ and $0+\tau$ to τ.

If the validity being proved is relative to a theory, then any axiom of the theory can be added to the deductive tableau as an assertion. Previously proved lemmas can be added similarly.

Such axioms and lemmas for the natural numbers can be used for time.

3.3 The Rewriting Rule

This rule generates a new row from a prior one by replacing a target subformula with an equivalent (sub)formula. Examples of rewriting schemas are:

$$\neg(\varphi_1 \wedge \varphi_2) \;\Leftrightarrow\; (\neg\varphi_1 \vee \neg\varphi_2) \qquad \text{and} \qquad (\forall\nu_1)(\forall\nu_2)\varphi \;\Leftrightarrow\; (\forall\nu_2)(\forall\nu_1)\varphi \;.$$

Nontemporal rewritings are listed in [MW93]. Temporal ones are as follows.

First:

$$\begin{aligned} \Box\varphi \;&\Leftrightarrow\; \varphi \wedge \bigcirc\Box\varphi \\ \Diamond\varphi \;&\Leftrightarrow\; \varphi \vee \bigcirc\Diamond\varphi \\ (\varphi_1 \,\mathcal{U}\, \varphi_2) \;&\Leftrightarrow\; \varphi_2 \vee \big[\varphi_1 \wedge \bigcirc(\varphi_1 \,\mathcal{U}\, \varphi_2)\big] \end{aligned}$$

[2] [MW93] gives formal semantics for deductive tableaux.

When used from left to right, these rewritings are called *expansions*.

Next, there are rewritings which specify that the temporal operator "$\bigcirc$" distributes over or commutes with nontemporal boolean operators ("$\neg$", "$\wedge$", "$\vee$", and "$\rightarrow$") and quantifiers. For example: $\bigcirc(\varphi_1 \wedge \varphi_2) \;\Leftrightarrow\; (\bigcirc\varphi_1 \,\wedge\, \bigcirc\varphi_2)$.

There are some similar rewritings involving the other temporal operators:

$$\begin{array}{rcl@{\qquad}rcl}
\Box(\varphi_1 \wedge \varphi_2) & \Leftrightarrow & \Box\varphi_1 \wedge \Box\varphi_2 & (\varphi_1 \wedge \varphi_2)\ \mathcal{U}\ \varphi_3 & \Leftrightarrow & (\varphi_1\ \mathcal{U}\ \varphi_3) \,\wedge\, (\varphi_2\ \mathcal{U}\ \varphi_3) \\
\Diamond(\varphi_1 \vee \varphi_2) & \Leftrightarrow & \Diamond\varphi_1 \vee \Diamond\varphi_2 & \varphi_1\ \mathcal{U}\ (\varphi_2 \vee \varphi_3) & \Leftrightarrow & (\varphi_1\ \mathcal{U}\ \varphi_2) \,\vee\, (\varphi_1\ \mathcal{U}\ \varphi_3)
\end{array}$$

$$\begin{array}{rcl@{\qquad}rcl}
\Box(\forall\nu)\varphi & \Leftrightarrow & (\forall\nu)\Box\varphi & (\forall\nu)\varphi\langle\nu\rangle\ \mathcal{U}\ \psi & \Leftrightarrow & (\forall\nu)\big(\varphi\langle\nu\rangle\ \mathcal{U}\ \psi\big) \\
\Diamond(\exists\nu)\varphi & \Leftrightarrow & (\exists\nu)\Diamond\varphi & \psi\ \mathcal{U}\ (\exists\nu)\varphi\langle\nu\rangle & \Leftrightarrow & (\exists\nu)\big(\psi\ \mathcal{U}\ \varphi\langle\nu\rangle\big)
\end{array}$$

(In each of the quantification cases for $\mathcal{U}$, the formula ψ must not contain any free occurrence of the variable ν.)

There are similar rewritings for the operators "$\boxminus$", "$\diamondminus$", "$\ominus$", and "$\mathcal{S}$".

3.4 The Time Distribution Rule

Time annotations distribute over or commute with nontemporal boolean operators ("$\neg$", "$\wedge$", "$\vee$", and "$\rightarrow$") and quantifiers. For example, the formula $[3 : (\Box p \rightarrow p)]$ is equivalent to the formula $\big([3 : \Box p] \rightarrow [3 : p]\big)$. The following rewritings provide this *time distribution*:

$$\begin{array}{rcl@{\qquad}rcl}
\big[\tau : (\neg\varphi)\big] & \Rightarrow & \neg[\tau : \varphi] & \big[\tau : (\varphi_1 \rightarrow \varphi_2)\big] & \Rightarrow & [\tau : \varphi_1] \rightarrow [\tau : \varphi_2] \\
\big[\tau : (\varphi_1 \wedge \varphi_2)\big] & \Rightarrow & [\tau : \varphi_1] \wedge [\tau : \varphi_2] & \big[\tau : (\forall\nu)\varphi\big] & \Rightarrow & (\forall\nu)[\tau : \varphi] \\
\big[\tau : (\varphi_1 \vee \varphi_2)\big] & \Rightarrow & [\tau : \varphi_1] \vee [\tau : \varphi_2] & \big[\tau : (\exists\nu)\varphi\big] & \Rightarrow & (\exists\nu)[\tau : \varphi]
\end{array}$$

(In each of the quantification cases, the time annotation τ must not contain ν.)

Time distribution can be (automatically) combined with the normal operations of other deduction rules, as desired. (For clear examples of such combination, see applications of the splitting rule — e.g., in the sample deductive tableau in the Introduction.)

3.5 The Duality Rule

Applied to a negated or simply *false* formula which is in one column (Assertions or Goals), this rule adds the formula's inverse to the other column.

3.6 The Splitting Rule

This rule disassembles a formula, easing further processing of its components.

Assertions	Goals
	$\varphi_1 \rightarrow \varphi_2$
φ_1	
	φ_2

Assertions	Goals
	$\varphi_1 \vee \varphi_2$
	φ_1
	φ_2

Assertions	Goals
$\varphi_1 \wedge \varphi_2$	
φ_1	
φ_2	

4 Deduction II: Time Reification

Quantifier elimination is presented here in some detail because it is crucial to time reification.

4.1 The Quantifier-Elimination Rule (Skolemization)[3]

A quantifier $(\mathcal{Q}\nu)$ has *universal force* if either its quantification operator $\mathcal{Q}$ is "$\forall$" and its polarity is positive, or if its quantification operator is "$\exists$" and its polarity is negative; otherwise, $(\mathcal{Q}\nu)$ has *existential force.*

The schema of the quantifier-elimination rule is: $[\tau : (\mathcal{Q}\nu)\varphi] \;\Rightarrow\; [\tau : \tilde{\varphi}]$,[4] where formula $\tilde{\varphi}$ is constructed from formula φ as follows:

- If the quantifier $(\mathcal{Q}\nu)$ being eliminated has universal force, then a function symbol "f_{new}" — new to the deductive tableau — is introduced. Additionally, let $\overline{\nu}$ be a list of the following variables of the entire row formula to which this rule is being applied: all the free variables, plus all the variables bound by quantifiers of existential force within whose scope the target subformula $[\tau : (\mathcal{Q}\nu)\varphi]$ lies. Then, the new subformula $\tilde{\varphi}$ is constructed from the original subformula φ by substituting the term "$f_{\text{new}}(\overline{\nu})$" for each occurrence of variable ν which was bound by the target quantifier $(\mathcal{Q}\nu)$.
- If the quantifier $(\mathcal{Q}\nu)$ being eliminated has existential force (and does not lie within the scope of any quantifier of universal force, in which case elimination of $(\mathcal{Q}\nu)$ would be disallowed), then the 'new' subformula $\tilde{\varphi}$ is simply identical to the original subformula φ; thus, $(\mathcal{Q}\nu)$ is simply removed.

(When eliminating $(\mathcal{Q}\nu)$ might change the quantifier binding of any variable via 'quantifier capturing', such a variable must be renamed.)

4.2 Introduction of the Time Reification Rule

Time reification removes a temporal operator from a formula via a process which, actually, is equivalent to the three-step operation of (1) translating the formula into nontemporal first-order predicate logic, (2) eliminating quantifiers introduced by step (1), and (3) translating back into temporal logic.

[AM90] gives a typical scheme for translating basic temporal formulas to semantically equivalent nontemporal formulas. An illustrative example is:

$$\bigcirc\bigcirc\bigcirc\Box p \quad\longmapsto\quad (\forall t)\big[(t \geq 3) \rightarrow p(t)\big] \;\; .$$

Reification, applied to the equivalent ATL formula, translates analogously for the "$\bigcirc$"-operators (yielding "3") but differently for the operator "$\Box$":

$$[0 : \bigcirc\bigcirc\bigcirc\Box p] \quad \overset{(1)}{\longmapsto} \quad (\forall\delta)\, p(3+\delta) \;\; .$$

[3] Skolemization in other systems has somewhat reversed operations since the goal there is refutation while the goal here is validation.

[4] Or, for unannotated time formulas, simply: $(\mathcal{Q}\nu)\varphi \;\Rightarrow\; \tilde{\varphi}$.

(The choice of "δ" versus "t" indicates a time increment instead of an absolute time.) Assuming negative polarity, reification would continue as follows:

$$(\forall\delta)\, p(3+\delta) \quad \xmapsto{(2)} \quad p(3+\delta) \quad \xmapsto{(3)} \quad [3+\delta \,:\, p] \ .$$

Another example of ATL translation is:

$$\big[0 \,:\, \bigcirc\bigcirc\bigcirc(p \;\mathcal{U}\; q)\big] \quad \xmapsto{(1)} \quad (\exists\delta_1)\Big(q(3+\delta_1) \;\wedge\; (\forall\delta_2)\big[(\delta_2 < \delta_1) \rightarrow p(3+\delta_2)\big]\Big) \, .$$

Reification of $\bigcirc$

As an illustrative example, translating the formula $[t \,:\, \bigcirc p]$ of ATL into nontemporal predicate logic yields $p(t{+}1)$. Translating back yields $[t+1 \,:\, p]$. Generalizing (facilely), the schema for reification of the operator "$\bigcirc$" is:

$$[\tau \,:\, \bigcirc\varphi] \;\; \Rightarrow \;\; [\tau+1 \,:\, \varphi] \ .$$

Reification of $\Box$ and $\Diamond$

An example for the operator "$\Box$" is given above; the operator "$\Diamond$" is treated analogously, with "$\exists$" instead of "$\forall$". The reification schemas are:

$$\begin{array}{llll} \Big[\tau \,:\, (\Box\varphi)^+\Big] \;\Rightarrow\; \big[\tau + h_{\text{new}}(\overline{\nu}) \,:\, \varphi\big] & & \Big[\tau \,:\, (\Box\varphi)^-\Big] \;\Rightarrow\; \big[\tau + \delta_{\text{new}} \,:\, \varphi\big] \\ \Big[\tau \,:\, (\Diamond\varphi)^-\Big] \;\Rightarrow\; \big[\tau + h_{\text{new}}(\overline{\nu}) \,:\, \varphi\big] & & \Big[\tau \,:\, (\Diamond\varphi)^+\Big] \;\Rightarrow\; \big[\tau + \delta_{\text{new}} \,:\, \varphi\big] \end{array}$$

Since reification of these operators implicitly involves quantifiers, the details of the quantifier-elimination rule apply: In the two cases on the left, for "$\Box^+$" and "$\Diamond^-$", the meta-symbol "$\overline{\nu}$" denotes a list of variables as in skolemization; and in the other two cases, for "$\Box^-$" and "$\Diamond^+$", this rule is disallowed for any formula $[\tau \,:\, (\Box\varphi)^-]$ or $[\tau \,:\, (\Diamond\varphi)^+]$ that occurs within the scope of any quantifier of universal force. An example for "$\Box^+$" is:

Asstns	Goals	Explanations
	$\big[t \,:\, \neg\big(d = f(x)\big)\big] \;\wedge\; (\exists y)\Big[c \,:\, \boxed{\Box(d=y)}^{\,+}\Big]$	
	$\big[t \,:\, \neg\big(d = f(x)\big)\big] \;\wedge\; (\exists y)\big[c + h(t,x,y) \,:\, d = y\big]$	reify $\Box^+$; "h" is new

Reification of $\odot$

An example of a formula containing the operator "$\odot$" is $[t \,:\, \odot d = a]$. The symbol d here should be dynamic; otherwise, the term $\odot d$ would be equivalent to d, so simplification would delete the operator "$\odot$" in the formula. Assume that the symbol a is not dynamic. Then translating this formula into nontemporal logic yields $d(t+1) = a$. Translating back yields $[t+1 \,:\, d = a]$.

But if the symbol a were dynamic, that result would not be equivalent to the original formula. To see this, note that with a dynamic, translation of the formula $[t : \odot d = a]$ into nontemporal logic would yield $d(t+1) = a(t)$, while translation of $[t+1 : d = a]$ would yield $d(t+1) = a(t+1)$.

Considering these cases, the reification schema for the operator "$\odot$" is:

$$\big[\tau : \varphi\langle \odot\alpha_1, \ldots, \odot\alpha_n\rangle\big] \quad \Rightarrow \quad \big[\tau + 1 : \varphi\langle \alpha_1, \ldots, \alpha_n\rangle\big] \ ,$$

where (1) the target subformula $\varphi\langle\ldots\rangle$ must not contain any dynamic symbol outside of the specified terms $\odot\alpha_i$, and (2) none of the specified terms $\odot\alpha_i$ may occur within the scope of any other temporal operator.

Reification of $\ominus$, $\boxminus$, and $\diamondminus$

$$[\tau : \ominus\varphi] \ \Rightarrow \ (0 < \tau) \ \wedge \ [\mathit{pred}(\tau) : \varphi] \ .$$

Both $\Big[\tau : (\boxminus\varphi)^+\Big]$ and $\Big[\tau : (\diamondminus\varphi)^-\Big] \ \Rightarrow \ \big(h_{\text{new}}(\overline{\nu}) \leq \tau\big) \ \wedge \ \big[h_{\text{new}}(\overline{\nu}) : \varphi\big]$.

Both $\Big[\tau : (\boxminus\varphi)^-\Big]$ and $\Big[\tau : (\diamondminus\varphi)^+\Big] \ \Rightarrow \ (t_{\text{new}} \leq \tau) \ \wedge \ [t_{\text{new}} : \varphi]$.

The details for $\boxminus$ and $\diamondminus$ are the same as for the operators $\Box$ and $\Diamond$.

Reification of $\boldsymbol{\mathcal{U}}$ and $\boldsymbol{\mathcal{S}}$ (Here, the subscript "n" specifies newness.)

$$\Big[\tau : (\varphi \ \mathcal{U} \ \psi)^-\Big] \ \Rightarrow \ \big[\tau + h_{\text{n}}(\overline{\nu}) : \psi\big] \ \wedge \ \Big(\big(\delta_{\text{n}} < h_{\text{n}}(\overline{\nu})\big) \ \rightarrow \ \big[\tau + \delta_{\text{n}} : \varphi\big]\Big)$$

$$\Big[\tau : (\varphi \ \mathcal{U} \ \psi)^+\Big] \ \Rightarrow$$
$$\big[\tau + \delta_{\text{n}} : \psi\big] \ \wedge \ \Big(\big(h_{\text{n}}(\overline{\nu}, \delta_{\text{n}}) < \delta_{\text{n}}\big) \ \rightarrow \ \big[\tau + h_{\text{n}}(\overline{\nu}, \delta_{\text{n}}) : \varphi\big]\Big)$$

$$\Big[\tau : (\varphi \ \mathcal{S} \ \psi)^-\Big] \ \Rightarrow$$
$$\big(h_{\text{n}}(\overline{\nu}) \leq \tau\big) \ \wedge \ \big[h_{\text{n}}(\overline{\nu}) : \psi\big] \ \wedge \ \Big(\Big[\big(h_{\text{n}}(\overline{\nu}) < t_{\text{n}}\big) \ \wedge \ \big(t_{\text{n}} \leq \tau\big)\Big] \rightarrow \big[t_{\text{n}} : \varphi\big]\Big)$$

$$\Big[\tau : (\varphi \ \mathcal{S} \ \psi)^+\Big] \ \Rightarrow$$
$$(t_{\text{n}} \leq \tau) \ \wedge \ (t_{\text{n}} : \psi) \ \wedge \ \Big(\Big[\big(t_{\text{n}} < h_{\text{n}}(\overline{\nu}, t_{\text{n}})\big) \ \wedge \ \big(h_{\text{n}}(\overline{\nu}, t_{\text{n}}) \leq \tau\big)\Big] \ \rightarrow \ \big[h_{\text{n}}(\overline{\nu}, t_{\text{n}}) : \varphi\big]\Big)$$

5 Deduction III: Nonclausal Resolution

The schemas for nonclausal resolution are:[5]

Assertions	Goals
$\varphi\Big\langle[\tau : \rho^+]\Big\rangle$	
$\psi\Big\langle[\tilde{\tau} : \tilde{\rho}^-]\Big\rangle$	
$\big(\varphi\langle \mathit{true}\rangle \ \vee \ \psi\langle \mathit{false}\rangle\big) \circ \theta$	

Asstns	Goals
	$\varphi\Big\langle[\tau : \rho^+]\Big\rangle$
	$\psi\Big\langle[\tilde{\tau} : \tilde{\rho}^-]\Big\rangle$
	$\big(\varphi\langle \mathit{true}\rangle \ \wedge \ \psi\langle \mathit{false}\rangle\big) \circ \theta$

[5] As with the quantifier-elimination rule, target subformulas for the resolution rule can be unannotated time formulas.

Assertions	Goals
$\varphi\Big\langle[\tau \,:\, \rho^{+}]\Big\rangle$	
	$\psi\Big\langle[\tilde{\tau} \,:\, \tilde{\rho}^{-}]\Big\rangle$
	$\begin{pmatrix}\neg\varphi\langle \mathit{true}\rangle \\ \wedge \\ \psi\langle \mathit{false}\rangle\end{pmatrix} \circ \theta$

Assertions	Goals
	$\varphi\Big\langle[\tau \,:\, \rho^{+}]\Big\rangle$
$\psi\Big\langle[\tilde{\tau} \,:\, \tilde{\rho}^{-}]\Big\rangle$	
	$\begin{pmatrix}\varphi\langle \mathit{true}\rangle \\ \wedge \\ \neg\psi\langle \mathit{false}\rangle\end{pmatrix} \circ \theta$

In each case: (1) the formulas φ and ψ must not share any free variables; (2) neither target subformula $[\tau : \rho]$ or $[\tilde{\tau} : \tilde{\rho}]$ may contain any occurrence of any variable that is bound by a quantifier outside of the target; and (3) these target subformulas must unify via a most general unifier θ.

Additional restrictions apply if the unifier θ involves a replacement $\nu \leftarrow \alpha$ with term α containing a dynamic symbol. For then:

1. No free occurrence of variable ν in the formulas φ and ψ may lie within the scope of any temporal operator or inside any time annotation or time formula.
2. All the time annotations within whose scope(s) replaceable occurrences of ν lie must be unified (by θ).

These two restrictions ensure consistency of the time at which a dynamic symbol may be interpreted. Otherwise, such a symbol — contained in term α, placed in various contexts by the replacement $\nu \leftarrow \alpha$ — could refer to different objects, violating soundness.

When resolution fails only because the target time annotations τ and $\tilde{\tau}$ fail to unify, an option is that the equation $\tau = \tilde{\tau}$ can be conjoined with the desired result of resolution as something additional that needs to be proven. A Presburger-like decision procedure can handle time formulas such as this equation. For an example of this situation, see Section 7.2.

The Equality Rule

This rule involves resolution-based reasoning. Hence, like the resolution rule itself, this rule comprises four schemas because of the alternatives for columns. Only one such schema need be shown:

Assertions	Goals
	$\varphi\Big\langle\big[\tau \,:\, \rho\langle\alpha\rangle\big]\Big\rangle$
$\psi\Big\langle\Big[\tilde{\tau} \,:\, (\beta = \tilde{\alpha})^{-}\Big]\Big\rangle$	
	$\Big(\varphi\Big\langle\big[\tau \,:\, \rho\langle\beta\rangle\big]\Big\rangle \;\wedge\; \neg\psi\langle \mathit{false}\rangle\Big) \circ \theta$

Here, θ must unify term α with term $\tilde{\alpha}$ as well as time annotation τ with time annotation $\tilde{\tau}$. Further details are as for resolution.

6 Deduction IV: Induction

The logic may refer to a domain, such as the natural numbers or lists, for which principles of induction hold. Then, rules which implement these principles are available. An example, for the natural numbers, is:

Assertions	Goals
	$\tau : (\forall \nu)\varphi\langle\nu\rangle$
	$\tau : \Big(\varphi\langle 0\rangle \wedge [\varphi\langle m\rangle \rightarrow \varphi\langle m+1\rangle]\Big)$

where the initial row must not contain free variables and m must be new. Thus, a goal can be reduced to a base case plus an inductive step.

Induction also applies to temporal operators. For example:

Assertions	Goals
	$\tau : \Box\varphi$
	$[\tau : \varphi] \wedge \Big([\tau + c_{\mathrm{new}} : \varphi] \rightarrow [\tau + c_{\mathrm{new}} + 1 : \varphi]\Big)$

where the initial row must not contain free variables and c_{new} must be new. There are additional temporal induction schemas, e.g., for assertions of the form $[\tau : \Diamond\varphi]$ (see the extended version of this paper).

7 Examples

7.1 Validity of the Formula $\Big([p \wedge \Box(p \rightarrow \bigcirc\Diamond p)] \rightarrow \Box\Diamond p\Big)$

This example, which requires temporal induction, is used in [AM90].

	Assertions	Goals	Explans
1		$0 : \begin{pmatrix} [p \wedge \Box(p \rightarrow \bigcirc\Diamond p)] \\ \rightarrow \\ \Box\Diamond p \end{pmatrix}$	Given
2	$0 : \boxed{p}^{-}$		1, split
3	$0 : \boxed{\Box(p \rightarrow \bigcirc\Diamond p)}^{-}$		1, split
4		$0 : \boxed{\Box\Diamond p}$	1, split
5		$\left[0 : \boxed{\Diamond p}^{+}\right] \wedge \Big([c_1 : \Diamond p] \rightarrow [c_1 + 1 : \Diamond p]\Big)$	4, temporal induction (of $\Box$)

6		$\left[t_2 : \boxed{p}^{+}\right]$ $\wedge\ (\ldots \rightarrow \ldots)$	5, reify $\diamond^{+}$
7		$\left[c_1 : \diamond p\right]$ $\rightarrow$ $\left[c_1 + 1 \ : \ \diamond p\right]$	6&2, resolution, unifier $\{t_2 \leftarrow 0\}$
8	$c_1 : \boxed{\diamond p}$		7, split
9		$c_1 + 1 \ : \ \boxed{\diamond p}^{+}$	7, split
10	$\boxed{t_3 : \ (p \rightarrow \bigcirc\diamond p)}$		3, reify $\square^{-}$
11	$[t_3 : p] \rightarrow \left[t_3 : \boxed{\bigcirc\diamond p}\right]$		10, distribution
12	$[t_3 : p] \rightarrow \left[t_3 + 1 : \boxed{\diamond p}^{-}\right]$		11, reify $\bigcirc$
13		$c_1 : \boxed{p}^{+}$	12&9, resolution, unifier $\{t_3 \leftarrow c_1\}$
14	$\boxed{c_1 : \ (p \vee \bigcirc\diamond p)}$		8, expand $\diamond$
15	$\left[c_1 : \boxed{p}^{-}\right] \vee [c_1 : \bigcirc\diamond p]$		14, distribution
16	$c_1 : \boxed{\bigcirc\diamond p}$		15&13, resolution; duality
17	$c_1 + 1 : \boxed{\diamond p}^{-}$		16, reify $\bigcirc$
18		*true*	17&9, resolution

Highlights of this proof are:

- First, the splitting rule conveniently demarcates the given formula's premises and desired conclusion (goal #4). Time distribution is used implicitly.
- One rationalization of the decision to apply temporal induction to goal #4 is that when one attempts the simpler operation, namely reification (of the operator "$\square$"), one gets stuck later in the proof.
- Goal #5, being the result of the induction rule, comprises two conjuncts: the base case and the inductive step. In this proof, the base case (the left conjunct) is proven first.
- Goal #7 is split for convenience.
- Assertion #8 is the inductive hypothesis, and goal #9 is the desired inductive conclusion. Experience with induction proofs leads one to attempt to reduce the latter to the former. Assertion #3 is used for this reduction.
- When goal #13 arises, assertion #8 can be used.
- The last two deduction-steps handle a case due to the disjunctive nature (consider assertion #14) of the inductive hypothesis.

7.2 Validity of the Formula $\Big(\Box\Diamond\big((\neg p)\ \mathcal{S}\ q\big) \ \rightarrow\ \Box\big(p \rightarrow \Diamond q\big)\Big)$

	Assertions	Goals	Explanations
1		$0: \begin{pmatrix} \Box^{-}\Diamond^{-}\big((\neg p)\ \mathcal{S}\ q\big) \\ \rightarrow \\ \Box^{+}(p \rightarrow \Diamond q) \end{pmatrix}$	Given
2	$t_1 + h_2(t_1)\ :\ \boxed{(\neg p)\ \mathcal{S}\ q}^{-}$		1, split; reify $\Box^{-}$ & $\Diamond^{-}$
3		$c_3\ :\ p \rightarrow \Diamond^{+}q$	1, split; reify $\Box^{+}$
4	$c_3 : \boxed{p}^{-}$		3, split
5		$\underline{c_3 + \delta_4}\ :\ q^{+}$	3, split; reify $\Diamond^{+}$
6	$h_5(t_1) \leq\ t_1 + h_2(t_1)$		2, reify $\mathcal{S}^{-}$; split
7	$h_5(t_1)\ :\ \boxed{q}^{-}$		2, reify $\mathcal{S}^{-}$; split
8	$\big(h_5(t_1) < t_6\big) \wedge \big(t_6 \leq t_1 + h_2(t_1)\big)$ $\rightarrow$ $\Big[t_6 : \neg\boxed{p}^{+}\Big]$		2, reify $\mathcal{S}^{-}$; split
9		$h_5(t_1) < c_3$ $\wedge$ $\boxed{c_3 \leq t_1 + h_2(t_1)}^{+}$	8&4, resolution, $\{t_6 \leftarrow c_3\}$; and duality

At this point in the proof, one could resolve goal #5, containing q^{+}, with assertion #7, containing q^{-}; the system would automatically generate the goal $\big(h_5(t_1)\ =\ c_3 + \delta_4\big)$ and pass this new formula plus the deductive tableau's other time formulas — assertion #6 and goal #9 — to a Presburger-like decision procedure, altogether in the form '$\langle$assertions$\rangle \rightarrow \langle$goals$\rangle$':

$$(\forall t_1)\Big(h_5(t_1) \leq\ t_1 + h_2(t_1)\Big) \ \rightarrow\ \begin{pmatrix} (\exists t_1)\Big(\big(h_5(t_1) < c_3\big)\ \wedge\ \big(c_3 \leq\ t_1 + h_2(t_1)\big)\Big) \\ \bigvee \\ (\exists\, t_1, \delta_4)\Big(h_5(t_1)\ =\ c_3 + \delta_4\Big) \end{pmatrix}$$

This formula is indeed valid (for the natural numbers), so these operations would satisfactorily finish the proof.

Alternatively, one may continue the proof 'manually' as follows:

10	$\boxed{x\ \leq\ x + y}^{-}$		axiom for natural numbers
11		$\boxed{h_5(c_3) < c_3}^{+}$	10&9, resolution, $\big\{t_1 \leftarrow c_3,\ x \leftarrow c_3,\ y \leftarrow h_2(c_3)\big\}$

12	$\boxed{x < y}^{-} \vee (\exists z)(x = y + z)$		property of naturals
13	$\boxed{(\exists z)\big(h_5(c_3) = c_3 + z\big)}^{-}$		12&11, resolution, $\big\{x \leftarrow h_5(c_3),\ y \leftarrow c_3\big\}$; duality
14	$\boxed{h_5(c_3) = \underline{c_3 + c_7}}^{-}$		13, eliminate $\exists^{-}$
15		$h_5(c_3) : \boxed{q}^{+}$	14&5, equality, $\{\delta_4 \leftarrow c_7\}$
16		*true*	15&7, resolution, $\{t_1 \leftarrow c_3\}$

7.3 Verification of List-processing

An assumption of list-processing code is that for any list l and at any time, if some dynamic symbol d is set equal to the list, and then (as time passes) d gets repeatedly truncated (while doing so is possible), then eventually d will equal the empty list *NIL*. Such a statement may be formalized as follows:

$$(\forall l)\Box\left(\left[(d = l) \wedge \Box\Big(\neg(d = \mathit{NIL}) \rightarrow \big(\odot d = \mathit{tail}(d)\big)\Big)\right] \rightarrow \Diamond(d = \mathit{NIL})\right).$$

This formula can be proven in ATL, using induction for lists, in thirty-three steps.

8 Properties of the ATL Deductive System

(Proofs of these properties are given in the extended version of this paper.)

8.1 Soundness

When all ATL formulas and operations are translated into nontemporal predicate logic, the soundness established in [MW93] applies; i.e., any formula which can be proven in ATL is indeed valid.

8.2 Relative Completeness

The ATL system can perform all the operations of the deductive system of [AM90]. Hence, as [AM90] identified a class of 'arithmetical' formulas (of predicate temporal logic) for which their deductive system is complete, the ATL system is also complete for that class.

8.3 Decision Procedure for Propositional Temporal Logic

The ATL deductive system can be used for a decision procedure for formulas of propositional temporal logic by mimicking more standard 'semantic' decision-algorithms such as the graph-based one of [KM93]. (This decision procedure using ATL also resembles the construction in [AM90]'s proof of completeness of their deductive system for propositional temporal formulas.)

9 Conclusion

The deductive system described here allows established techniques for reasoning in predicate logic with induction to be extended to temporal logic. This system has several features which enable it to avoid complications of removal of temporal operators, and it invokes special unification and decision procedures to reduce the burden on the user. It enables proofs of temporal formulas to be constructed with more formality, ease, and clarity than was previously possible.

References

[AM90] Abadi, M., and Manna, Z.: "Nonclausal Deduction in First-Order Temporal Logic," in *Journal of the Association for Computing Machinery* (*JACM*), Volume 37 (1990), Number 2 (April), pp. 279–317.

[CES86] Clarke, E., Emerson, E., and Sistla, A.: "Automatic Verification of Finite-State Concurrent Systems Using Temporal Logic Specifications," in *ACM Transactions on Programming Languages and Systems*, Volume 8 (1986), Number 2 (April), pp. 244–263.

[KM93] Kesten, Y., Manna, Z., McGuire, H., and Pnueli, A.: "A Decision Algorithm for Full Propositional Temporal Logic," in Courcoubetis, C. (editor): *Computer Aided Verification (5th International Conference, CAV '93)* (LNCS #697), pp. 97–109. Springer-Verlag, Berlin, 1993.

[MP91] Manna, Z., and Pnueli, A.: *The Temporal Logic of Reactive and Concurrent Systems: Specification.* Springer-Verlag, New York, 1991.

[MW80] Manna, Z., and Waldinger, R.: "A Deductive Approach to Program Synthesis," in *ACM Transactions on Programming Languages and Systems*, Volume 2 (1980), pp. 90–121.

[MW93] Manna, Z., and Waldinger, R.: *The Deductive Foundations of Computer Programming.* Addison-Wesley, Reading, Massachusetts, 1993.

[O88] Ohlbach, H.: "A Resolution Calculus for Modal Logics," in Lusk, E., and Overbeek, R. (editors): *9th International Conference on Automated Deduction (Proceedings)* (LNCS #310), pp. 500–516. Springer-Verlag, Berlin, 1988.

[O93] Ohlbach, H.: "Translation Methods for Non-Classical Logics — An Overview," in *Automated Deduction in Nonstandard Logics* (Technical Report #FS-93-01), pp. 113–125. AAAI Press, Menlo Park, California, 1993.

[P86] Plaisted, D.: "A Decision Procedure for Combinations of Propositional Temporal Logic and Other Specialized Theories", in *Journal of Automated Reasoning*, Volume 2 (1986), pp. 171–190.

[S79] Shostak, R.: "A practical decision procedure for arithmetic with function symbols," in *JACM*, Volume 26 (1979), Number 2 (April), pp. 351–360.

[W89] Wallen, L.: *Automated Proof Search in Nonclassical Logics.* The MIT Press, Cambridge, Massachusetts, 1989.

This article was processed using the LaTeX macro package with LLNCS style