30

The Power of Temporal Proofs

by Martín Abadi

August 15, 1988

Systems Research Center
130 Lytton Avenue
Palo Alto, California 94301

Systems Research Center

DEC's business and technology objectives require a strong research program. The Systems Research Center (SRC) and three other research laboratories are committed to filling that need.

SRC began recruiting its first research scientists in 1984 — their charter, to advance the state of knowledge in all aspects of computer systems research. Our current work includes exploring high-performance personal computing, distributed computing, programming environments, system modelling techniques, specification technology, and tightly-coupled multiprocessors.

Our approach to both hardware and software research is to create and use real systems so that we can investigate their properties fully. Complex systems cannot be evaluated solely in the abstract. Based on this belief, our strategy is to demonstrate the technical and practical feasibility of our ideas by building prototypes and using them as daily tools. The experience we gain is useful in the short term in enabling us to refine our designs, and invaluable in the long term in helping us to advance the state of knowledge about those systems. Most of the major advances in information systems have come through this strategy, including time-sharing, the ArpaNet, and distributed personal computing.

SRC also performs work of a more mathematical flavor which complements our systems research. Some of this work is in established fields of theoretical computer science, such as the analysis of algorithms, computational geometry, and logics of programming. The rest of this work explores new ground motivated by problems that arise in our systems research.

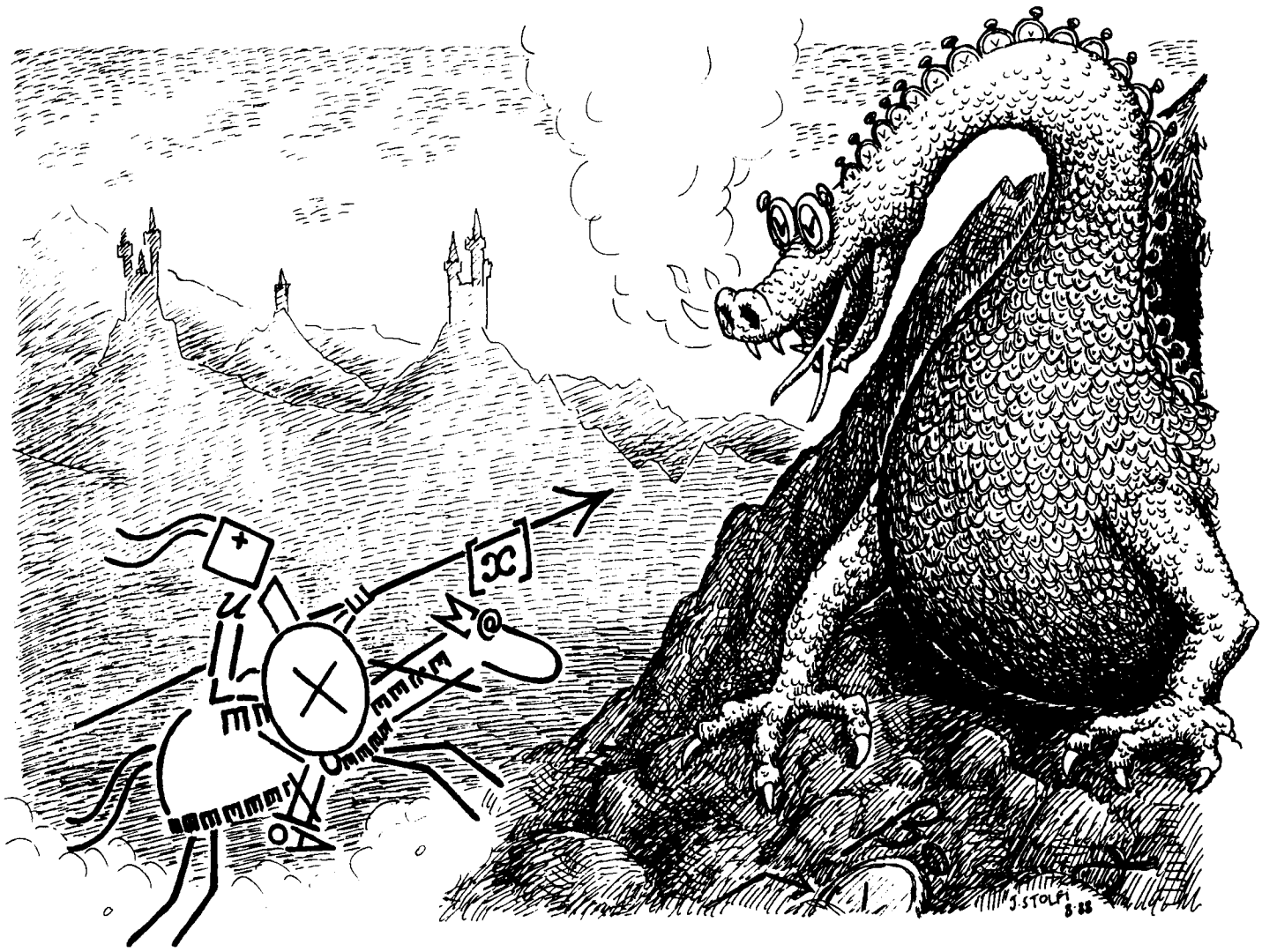
DEC has a strong commitment to communicating the results and experience gained through pursuing these activities. The Company values the improved understanding that comes with exposing and testing our ideas within the research community. SRC will therefore report results in conferences, in professional journals, and in our research report series. We will seek users for our prototype systems among those with whom we have common research interests, and we will encourage collaboration with university researchers.

Robert W. Taylor, Director

The Power of Temporal Proofs

Martín Abadi

August 15, 1988



Publication history

A shorter version of this paper was published in the proceedings of the IEEE *Symposium on Logic in Computer Science*, Ithaca, New York. June, 1987.

Author's abstract

Some methods for reasoning about concurrent programs and hardware devices have been based on proof systems for temporal logic. Unfortunately, all effective proof systems for temporal logic are incomplete for the standard semantics, in the sense that some formulas hold in every intended model but cannot be proved. We evaluate and compare the power of several proof systems for temporal logic. Specifically, we relate temporal systems to classical systems with explicit time parameters.

A typical temporal system turns out to be incomplete in a strong sense; we exhibit a short, valid formula it fails to prove. We suggest the addition of new rules to define auxiliary predicates. With these rules, we obtain nonstandard soundness and completeness results. In particular, one of the simple temporal systems we describe is as powerful as Peano Arithmetic.

Martín Abadi

Capsule review

In many systems of temporal logic, the relationship between time instants resembles the ordering of the natural numbers. This correspondence is explored more fully in this paper. The main result is that sufficiently–but not unreasonably–strong systems of temporal logic are equivalent to Peano Arithmetic. This masterful paper establishes similar correspondences for several weaker temporal logics.

Mark Manasse

Contents

Errata for "The Power of Temporal Proofs"

September 8, 1989

The proof of Theorem 6.2 in "The Power of Temporal Proofs" is seriously flawed. The claim was that all valid temporal formulas are arithmetical in the proof system T_2. The claim does not appear to be true. Indeed, Andréka, Németi, and Sain, who uncovered the problem, recently sent me a paper with a counterexample, "On the Strength of Temporal Proofs."

The main role of the claim was as a stepping stone towards Theorem 7.2. This claim, in turn, says that T_1 is complete for arithmetical formulas while T_2 is complete for all formulas. With the failure of Theorem 6.2, the result proved in 7.2 is weaker. It holds simply that both T_1 and T_2 are complete for arithmetical formulas. More precisely, Theorem 7.2 should read:

> For every formula u, $\vdash_O P(u) \Rightarrow \vdash_{T_1} u$ if u is arithmetical in $\vdash_{T_1}$.
> For every formula u, $\vdash_P P(u) \Rightarrow \vdash_{T_2} u$ if u is arithmetical in $\vdash_{T_2}$.

This final result may well be enough in practice—and, in particular, for the purposes of program verification. On the other hand, it would be pleasant to have a proof system complete for all formulas, rather than only for arithmetical ones.

Incidentally, the paper by Andréka, Németi, and Sain addresses some of the open questions in "The Power of Temporal Proofs," and I recommend it to the interested readers. It was presented at the 1989 International Symposium on Mathematical Foundations of Computer Science. I would also like to thank its authors for discovering this problem.

1. Introduction

Temporal logic has been used extensively to reason about concurrent systems. Some methods for verifying concurrent programs and hardware devices have been based on proof systems for first-order temporal logic, FTL (e.g., [Pn], [OL], [MP3]). Thus, the quality and especially the power of these verification methods depend directly on the power of the underlying FTL proof systems (e.g., [MP1]). Unfortunately, all effective FTL proof systems are incomplete for the standard semantics, in the sense that some formulas hold in every intended model but cannot be proved. Evaluating temporal proof systems and the corresponding verification methods is therefore a nontrivial problem.

In this paper, we first prove that all effective FTL proof systems are incomplete for the standard semantics and then propose alternative notions of completeness. Specifically, we consider a translation of temporal formulas into classical formulas with explicit time parameters and ask questions such as "is the temporal formula u provable in (a given system for) temporal logic if and only if its translation is provable in (a given system for) classical logic?" We study three FTL proof systems. The first one, T_0, is an extension of the usual Hilbert system of Manna and Pnueli ([MP2]) and equivalent to the resolution system of Abadi and Manna ([AM]). This basic system is incomplete in a strong sense. We exhibit a short valid formula that T_0 fails to prove. The other ones, T_1 and T_2, include T_0 with new natural rules for defining auxiliary predicates. We give simple characterizations of T_1 and T_2. For instance, our main positive result is that T_2 is as powerful as Peano Arithmetic. This characterization can be read as a nonstandard soundness and completeness theorem, since it means that the formulas provable in T_2 are exactly those that hold in every model of Peano Arithmetic.

We concentrate on Hilbert systems because they are more usual and easier to understand than systems of other kinds. However, our methods are general, and, for instance, they immediately apply in the study of resolution systems ([A1]).

Recently there has been much related work on nonstandard logics of programs (e.g., [N], [BS], [Sa1], [Sa2]), which proposes notions of completeness similar to ours. However, the existing results for temporal logic, which we discuss in more detail below, consider only provability of special classes of sentences, with restricted temporal formalisms, and in rather weak systems. In particular, they are of limited relevance to the verification of concurrent systems.

Section 2 reviews the syntax and semantics of FTL; the semantics of FTL is formulated in two equivalent ways: in terms of possible worlds and through a translation to classical logic. In section 3 we show that the standard notion of validity is intractable and suggest some approximations based on provability in formal systems of arithmetic. In section 4 we describe the basic system T_0 and prove a nonstandard completeness theorem and an incompleteness theorem. In section 5 we extend T_0 with rules to define auxiliary predicates

and obtain the systems T_1 and T_2. We give nonstandard completeness theorems for T_1 and T_2; these completeness theorems and the one in section 4 are not very informative by themselves, but we invoke them in section 6. We also discuss the connections between these systems and resolution systems with skolemization rules. In section 6 we introduce the useful notions of "clock" and "arithmetical formula." Section 7 contains the main soundness and completeness theorems for T_1 and T_2. We compare our results with previous ones in section 8 and pose some open problems in section 9. Some of the more tedious and trivial proofs are relegated to an appendix.

The material of this paper has appeared in a preliminary form in [A2]. The full work is discussed in [A1], where soundness and completeness issues for resolution systems are discussed in more detail.

2. Temporal logic

Several logics of time have been proposed (e.g., [Ka], [Bu2], [VB1]). We consider one specific temporal logic described by Manna and Pnueli ([MP1]), which is both general and relatively simple. In the intended models, time is discrete, linear, and extends infinitely toward the future. We refer to the propositional version of our logic as propositional temporal logic (PTL) and to the first-order version as first-order temporal logic (FTL). In this section we define PTL and FTL.

1. Syntax

Propositional temporal logic

A *language of* PTL is a countable collection of propositional symbols

$$p, q, r, s, \ldots .$$

Given a language, PTL formulas are built up using

- propositional symbols in the language;
- connectives: for simplicity, we assume that the only connectives are $\neg$ ("not"), $\wedge$ ("and"), and $\vee$ ("or"); we regard other connectives, such as *true*, *false*, $\supset$ ("implies") and $\equiv$ ("is equivalent to"), as abbreviations;
- modal operators: the modal operators we consider are the usual ones for discrete linear time, $\bigcirc$ ("next"), $\Box$ ("always"), $\Diamond$ ("eventually"), and the more general $\mathcal{U}$ ("until") and $\mathcal{P}$ ("precedes").

Thus, the formation rules for formulas are:

- all propositional symbols in the language under consideration are formulas;
- if u and v are formulas then so are $\neg u$, $u \wedge v$, $u \vee v$, $\bigcirc u$, $\Box u$, $\Diamond u$, $u \,\mathcal{U}\, v$, and $u \,\mathcal{P}\, v$.

First-order temporal logic

A *language of* FTL consists of a countable collection of predicate and function symbols

$$p, q, r, s, \ldots, a, b, c, f, g, h, \ldots.$$

We associate a nonnegative integer with each symbol in a language, as its arity. Propositional symbols and constant symbols are simply predicate symbols and function symbols with arity 0, respectively. Given a language, FTL formulas are built up using

- predicate and function symbols in the language;
- the equality symbol $=$, which we treat as an additional predicate symbol;
- variable symbols, such as $x, y, z, x_0, y_0, z_0, x_1, y_1, z_1, \ldots$;
- connectives and modal operators, as in PTL;
- the quantifiers $\forall$ and $\exists$.

The formation rules for terms are:

- all variable symbols are terms;
- if f is a function symbol of arity k and $t_1, \ldots, t_k$ are terms then $f(t_1, \ldots, t_k)$ is a term.

The formation rules for atomic formulas are:

- if p is a predicate symbol of arity k and $t_1, \ldots, t_k$ are terms then $p(t_1, \ldots, t_k)$ is an atomic formula;
- if t_1 and t_2 are terms then $t_1 = t_2$ is an atomic formula.

Other formulas are obtained as follows:

- formulas are constructed from other formulas by application of connectives and modal operators, as in PTL;
- if x is a variable and u is a formula then $\forall x.u$ and $\exists x.u$ are formulas.

Thus, all PTL formulas are also FTL formulas.

Flexible and rigid symbols

In PTL, all propositional symbols are *flexible*, that is, we intend to give them time-dependent meanings.

In FTL, it is particularly convenient and natural to give time-independent meanings to some symbols, which we call *rigid* symbols, and time-dependent meanings to other symbols, which we call *flexible* symbols. Thus, in a given language of FTL, each symbol is either flexible or rigid. The equality symbol $=$ is always rigid: two individuals are either always identical or always different (of course, if a and b are flexible symbols then a may equal b some times and may be different from b at other times; we attribute this to changes in the denotation of a and b rather than to changes in the denotation of $=$).

A term or a formula is said to be rigid if it contains no occurrences of flexible symbols; otherwise, it is said to be flexible.

For example, if *busy* is a flexible (time-dependent) unary predicate symbol and *printer* is a rigid (time-independent) constant symbol, then

$$busy(printer) \ \wedge \ \bigcirc\Box\neg busy(printer)$$

is a flexible formula. We intend to give the same value to *printer* in all states, and to make the property of being *busy* time-dependent.

2. A possible-worlds semantics

Informally, FTL formulas are evaluated over sequences of states. If u and v are formulas then

- $\bigcirc u$ means "u is true in the next state";
- $\Box u$ means "u is always true (from now on)";
- $\Diamond u$ means "u is eventually true";
- $u \,\mathcal{U}\, v$ means "u is true until v is true"; in particular, u is true forever if v is never true (therefore, $\mathcal{U}$ is often called "weak until" or "unless");
- $u \,\mathcal{P}\, v$ means "u precedes v"; that is, u must hold sometime before the first time when v holds and must hold eventually if v never holds.

A formal semantics is described in terms of *possible worlds* ([HC]). Given a language, a *model* $\mathcal{M}$ is a tuple $\langle D, W, w_0, R_1, R_2, \alpha, I\rangle$.

- The *domain* D is a non-empty set.

- W, the *frame* is a set with a distinguished element w_0. Intuitively, W is the set of worlds and w_0 the present world. (We require that there be just one domain D common to all worlds, rather than a domain D_w for each element w of W, as in some other modal logics.)
- R_1 and R_2 are two binary *accessibility relations* on W. Intuitively, R_1 corresponds to "next" and R_2 to "eventually."
- α is an *assignment* of values to the variables, that is, a mapping from the set of variables to D.
- The *interpretation* I gives a meaning to predicate and function symbols: for each world w, I maps each predicate symbol p to a relation $I(w,p)$ over D and each function symbol f to a function $I(w,f)$ over D. The meaning of rigid symbols must be the same at all worlds.

The evaluation function and the satisfaction relation

If $d_1,\ldots,d_n$ are elements of D then

$$\mathcal{M} \circ \langle x_1 \leftarrow d_1,\ldots,x_n \leftarrow d_n\rangle$$

denotes the model obtained from $\mathcal{M}$ by modifying its assignment function α to map the variables $x_1,\ldots,x_n$ to $d_1,\ldots,d_n$, respectively. If w is a world in W then

$$\mathcal{M}@w$$

denotes the model obtained from $\mathcal{M}$ by modifying its present world to be w.

We define inductively the binary *evaluation* function τ, which evaluates terms in models, and the binary *satisfaction* relation between models and formulas, $\models$:

- For terms:

$$\begin{aligned}\tau(\mathcal{M},x) &= \alpha(x),\\ \tau(\mathcal{M},f(t_1,\ldots,t_k)) &= I(w_0,f)\big(\tau(\mathcal{M},t_1),\ldots,\tau(\mathcal{M},t_k)\big).\end{aligned}$$

- For atomic formulas:

$$\begin{aligned}\mathcal{M} \models p(t_1,\ldots,t_k) &\Leftrightarrow I(w_0,p)\big(\tau(\mathcal{M},t_1),\ldots,\tau(\mathcal{M},t_k)\big),\\ \mathcal{M} \models t_1 = t_2 &\Leftrightarrow \tau(\mathcal{M},t_1) = \tau(\mathcal{M},t_2).\end{aligned}$$

- For connectives:

$$\mathcal{M} \models \neg u \quad \Leftrightarrow \quad \mathcal{M} \not\models u,$$

$$\mathcal{M} \models u \wedge v \Leftrightarrow [\mathcal{M} \models u \ \wedge \ \mathcal{M} \models v],$$
$$\mathcal{M} \models u \vee v \Leftrightarrow [\mathcal{M} \models u \ \vee \ \mathcal{M} \models v].$$

- For quantifiers:

$$\mathcal{M} \models \forall x.u \Leftrightarrow \forall d \in D.[\mathcal{M} \circ \langle x \leftarrow d \rangle \models u],$$
$$\mathcal{M} \models \exists x.u \Leftrightarrow \exists d \in D.[\mathcal{M} \circ \langle x \leftarrow d \rangle \models u].$$

- For modal operators:

$$\mathcal{M} \models \bigcirc u \ \Leftrightarrow \ \exists w_1.[w_0 R_1 w_1 \ \wedge \ \mathcal{M}@w_1 \models u],$$
$$\mathcal{M} \models \Box u \ \Leftrightarrow \ \forall w_1.[w_0 R_2 w_1 \ \supset \ \mathcal{M}@w_1 \models u],$$
$$\mathcal{M} \models \Diamond u \ \Leftrightarrow \ \exists w_1.[w_0 R_2 w_1 \ \wedge \ \mathcal{M}@w_1 \models u],$$
$$\mathcal{M} \models u \,\mathcal{U}\, v \ \Leftrightarrow \ \forall w_1. \left[w_0 R_2 w_1 \supset \left[\begin{array}{l} \mathcal{M}@w_1 \models u \\ \quad \vee \\ \exists w_2.(w_0 R_2 w_2 \ \wedge \ w_2 R_2 w_1 \ \wedge \ \mathcal{M}@w_2 \models v) \end{array} \right] \right],$$
$$\mathcal{M} \models u \,\mathcal{P}\, v \ \Leftrightarrow \ \exists w_1. \left[w_0 R_2 w_1 \wedge \left[\begin{array}{l} \mathcal{M}@w_1 \models u \\ \quad \wedge \\ \forall w_2.(w_0 R_2 w_2 \ \wedge \ w_2 R_2 w_1 \ \supset \ \mathcal{M}@w_2 \models \neg v) \end{array} \right] \right].$$

Thus, $\Box u$ is equivalent to $u \,\mathcal{U}\, \mathit{false}$. Furthermore, $\Diamond u$ and $u \,\mathcal{P}\, v$ are equivalent to $\neg \Box \neg u$ and $\neg((\neg u) \,\mathcal{U}\, v)$, respectively.

Standard models, satisfiability, and validity

The model $\mathcal{M}$ is *standard* if $\langle W, w_0, R_1, R_2 \rangle$ is isomorphic to $\langle N, 0, s, \leq \rangle$, that is, the natural numbers with the constant zero, the successor function, and the less-than relation. In particular, R_1 is a function and R_2 is the reflexive transitive closure of R_1 in all standard models.

We are interested in standard models because they are the intended models of temporal logic. On the other hand, we need to consider other models as well, in particular when we study soundness and completeness issues.

The formula u is *satisfiable* if some standard model $\mathcal{M}$ satisfies u, that is, $\mathcal{M} \models u$. The formula u is *valid* if all standard models satisfy u; we denote this by $\models u$.

Free variables are implicitly universally quantified as far as validity is concerned: u is valid exactly when $\forall x.u$ is valid.

3. An arithmetical semantics

So far we have described models and the satisfaction relation in terms of possible worlds. An alternative way to interpret FTL is to translate temporal formulas into classical formulas that contain some arithmetic symbols. This arithmetical semantics is equivalent to the possible-world semantics. However, it leads to some insights about the complexity of FTL and about alternatives to the standard notion of completeness, typical of Correspondence Theory, which studies connections between modal and classical systems ([VB2]).

For each temporal language L, we define a countable, two-sorted, classical language L_O ("O" stands for "order").

The sort S_N, which we interpret as the sort of natural numbers, is equipped with the constant symbol 0, the function symbol s, the predicate symbol $\leq$, and the equality symbol $=$. It is technically practical to consider that in actuality 0 and s are just convenient, informal abbreviations for the unary predicate symbol 0_p and the binary predicate symbol s_p, respectively. Countably many variables of the number sort are denoted by letters like i; terms of this sort are denoted by letters like m.

The equality symbol $=$ and all predicate and function symbols in L are also in L_O, to apply to terms of sort S_D, which we interpret as the sort of data. The arity of rigid symbols in L_O is the same as in L. Terms of sort S_N cannot occur as arguments of these rigid symbols. The arity of flexible symbols in L_O is their arity in L incremented by one. Terms of sort S_N occur as last arguments of these flexible symbols, and in no other argument position.

For example, a typical formula of L_O is $\exists x \forall i.[s(j) \leq i \supset p(f(x, a, s(i)), i)]$. Here, i and j are intended to range over numbers; s and $\leq$ have their usual intended meanings. On the other hand, x ranges over some arbitrary data domain; a is uninterpreted in this same domain; f is uninterpreted, with this domain as range, and two pieces of data and one number as arguments; p is uninterpreted, with two arguments, one piece of data and one number.

A two-sorted classical model for L_O consists of the following components: two sets D_N and D_D, the universes for the sorts S_N and S_D, respectively; relations on D_N to interpret 0_p, s_p, and $\leq$; relations of the appropriate types to interpret the other predicate and function symbols; and two functions to assign values to variables, one for S_D and one for S_N.

As usual, the satisfaction relation, $\models_O$, is defined inductively over formulas. By $\mathcal{M} \models_O w$ we mean that the two-sorted classical model $\mathcal{M}$ satisfies the L_O formula w, with no assumptions on the properties of $\mathcal{M}$ (in particular, D_N need not even be countable). We can give a standard semantics to L_O in the natural way, requiring that D_N be the natural numbers, $\leq$ the usual less-than relation between numbers, etc.. The interpretation

of the data symbols is left open. By $\models_O w$ we mean that all standard models satisfy w. Thus, $\models_O$ represents standard validity.

Any temporal formula in a language L can be translated into a classical formula in the corresponding language L_O; informally, we may think of this classical formula as the meaning of the temporal formula. The function P maps FTL formulas to their translations in arithmetic.

For all u, let $P(u) = P^*(u, 0)$, where P^* is an auxiliary translation function defined by:

$$P^*(p(t_1, \ldots, t_k), m) = \begin{cases} p(P^*(t_1, m), \ldots, P^*(t_k, m)) \\ \qquad \text{if } p \text{ is a rigid symbol} \\ p(P^*(t_1, m), \ldots, P^*(t_k, m), m) \\ \qquad \text{if } p \text{ is a flexible symbol} \end{cases}$$

$$P^*(\bigcirc u, m) = P^*(u, s(m))$$

$$P^*(\Box u, m) = \forall i \geq m.P^*(u, i) \qquad (i \text{ and } j \text{ are new variables})$$

$$P^*(\Diamond u, m) = \exists i \geq m.P^*(u, i)$$

$$P^*(u \,\mathcal{U}\, v, m) = \forall i \geq m.(P^*(u, i) \vee \exists j.(m \leq j \leq i \wedge P^*(v, j)))$$

$$P^*(u \,\mathcal{P}\, v, m) = \exists i \geq m.(P^*(u, i) \wedge \forall j.(m \leq j \leq i \supset \neg P^*(v, j)))$$

and P^* preserves connectives, quantifiers, and variables.

Furthermore, there is a natural function $\mathcal{C}$ to convert a possible-world model $\mathcal{M}$ into a two-sorted classical model. The data domain in $\mathcal{C}(\mathcal{M})$ is D, the domain of $\mathcal{M}$; the number domain in $\mathcal{C}(\mathcal{M})$ is W, the set of possible worlds in $\mathcal{M}$; 0, s, and $\leq$ correspond to w_0, R_1, and R_2, respectively. The assignment and the interpretation are not affected.

The following simple propositions express that the possible-world semantics and the arithmetical semantics are fundamentally the same.

Proposition 2.1.

$$\mathcal{M} \models u \Leftrightarrow \mathcal{C}(\mathcal{M}) \models_O P(u).$$

Proof:
A simple inductive argument on the structure of u yields the result. ∎

The function $\mathcal{C}$ is a bijection. Let $\mathcal{D}$ be its inverse.

Proposition 2.2.

$$\mathcal{D}(\mathcal{M}) \models u \Leftrightarrow \mathcal{M} \models_O P(u).$$

Proof:
Given a classical model $\mathcal{M}$, proposition 2.1 guarantees that

$$\mathcal{D}(\mathcal{M}) \models u \Leftrightarrow \mathcal{C}(\mathcal{D}(\mathcal{M})) \models_O P(u).$$

The result follows immediately, because $\mathcal{C}(\mathcal{D}(\mathcal{M})) = \mathcal{M}$. ∎

Note that $\mathcal{C}$ maps standard possible-world models into standard classical models. Also, $\mathcal{D}$ maps standard classical models into standard possible-world models. Thus, the propositions yield:

Corollary 2.3.

$$\models u \Leftrightarrow \models_O P(u).$$

Proof:
Suppose that $\neg u$ holds in the standard possible-world model $\mathcal{M}$. Then $P(\neg u)$ holds in the standard classical model $\mathcal{C}(\mathcal{M})$. Since $\neg P(u) = P(\neg u)$, $\neg P(u)$ holds in the standard classical model $\mathcal{C}(\mathcal{M})$. To check the other direction of the equivalence, suppose that $\neg P(u)$ holds in the standard classical model $\mathcal{M}$, that is, $P(\neg u)$ holds in $\mathcal{M}$. Therefore, $\neg u$ holds in the standard possible-world model $\mathcal{D}(\mathcal{M})$. ∎

Thus, in a precise sense, if we redefine the meaning of a FTL formula u to be the meaning of $P(u)$, the semantics remains the same.

3. On standard incompleteness

We prove that standard validity is Π_1^1-complete (see, for example, [R]). In particular, no effective system for FTL can be complete in the standard sense. Therefore, we propose more realistic and practical notions of completeness.

1. The complexity of validity

A formula u is Π_1^1 if $u = (\forall R_1 \ldots \forall R_k \forall F_1 \ldots \forall F_{k'}.v)$ for some classical first-order formula v, and 0, s, $\leq$, $+$, $\times$, $R_1,\ldots,R_k$, $F_1,\ldots,F_{k'}$ are all the predicate and function symbols in v. The complexity class Π_1^1 includes all problems no harder than the truth problem for Π_1^1 formulas.

The following proposition and its corollary state that the validity problem is in the class Π_1^1. Intuitively, the question of the validity of a temporal formula can be reduced to the question of the truth of a Π_1^1 formula—we replace times with numbers.

Proposition 3.1.

$\models_O$ *is in* Π^1_1.

Proof:
If a formula has a standard model then we can construct a standard term model. As usual, the domain of this model is a set of terms modulo equality in the original model. Since the languages under consideration are countable, this term model is also countable.

In turn, if a formula has a countable model then we can construct a model with N as data domain. To do this, we number the elements of the original model and say that a relation holds for a tuple of numbers in the new model if it holds for the corresponding data in the original model.

Thus, $\models_O u$ if u holds in all standard models with N as domain. Therefore, all uninterpreted symbols in u can be taken to range over relations and functions on numbers. We conclude that $\models_O u$ if and only if $\forall R_1 \ldots \forall R_k \forall F_1 \ldots \forall F_{k'}.u$ holds for the natural numbers, where 0, s, $\leq$, $R_1,\ldots,R_k$, $F_1,\ldots,F_{k'}$ are all the predicate and function symbols in u. This reduces the question of the validity of u to the question of the truth of a Π^1_1 formula. ∎

Since the translation function P is primitive recursive, we have:

Corollary 3.2.

$\models$ *is in* Π^1_1.

The symbol $\models$ refers to validity over the standard models, which have frames isomorphic to the natural numbers. We exploit this to show that the upper bound of the previous corollary is actually tight, that is, $\models$ is Π^1_1-complete:

Theorem 3.3.

$\models$ *is* Π^1_1*-hard.*

Proof:
It suffices to show a recursive translation function that embeds Π^1_1 formulas in FTL. More precisely, we want a function E to map Π^1_1 formulas to temporal formulas such that u is true if and only if $\models E(u)$.

Given u of the form $\forall R_1 \ldots \forall R_k \forall F_1 \ldots \forall F_{k'}.v$, with v a first-order formula, we may drop the second-order quantifiers, since the notion of validity contains an implicit universal quantification on free predicate and function symbols. More precisely, let $E(u) = E'(v)$, where E' is an auxiliary function. To define E', we first "simulate" numbers in FTL. Let 0, s, $\leq$, $+$, and $\times$ be rigid uninterpreted symbols of the appropriate arities and a be a flexible constant symbol. Let A be the sentence

$$(\forall x.\, \Diamond\, a = x) \wedge (\forall x.\, \Box(a = x \supset \bigcirc \Box\, a \neq x))$$

$$\wedge (a = 0)$$
$$\wedge (\forall x \forall y.(s(x) = y \equiv \Diamond(a = x \wedge \bigcirc a = y)))$$
$$\wedge (\forall x \forall y.(x \leq y \equiv \Diamond(a = x \wedge \Diamond a = y)))$$
$$\wedge (\forall x.x + 0 = x)$$
$$\wedge (\forall x \forall y.x + s(y) = s(x + y))$$
$$\wedge (\forall x.x \times 0 = 0)$$
$$\wedge (\forall x \forall y.x \times s(y) = x \times y + x).$$

The sentence A defines the numbers and the corresponding operations, by representing the number i as the value of a at time i. Furthermore, A guarantees that all elements of the domain represent numbers. More precisely, suppose that $\mathcal{M}$ is a model for a language that includes the symbols 0, s, $\leq$, $+$, $\times$, $R_1, \ldots, R_k$, and $F_1, \ldots, F_{k'}$. Let D be the domain of $\mathcal{M}$ and let $0_{\mathcal{M}}$, $s_{\mathcal{M}}$, $\leq_{\mathcal{M}}$, $+_{\mathcal{M}}$, $\times_{\mathcal{M}}$ be the interpretations for 0, s, $\leq$, $+$, $\times$, respectively. If A holds in $\mathcal{M}$ then the structure $\langle D, 0_{\mathcal{M}}, s_{\mathcal{M}}, \leq_{\mathcal{M}}, +_{\mathcal{M}}, \times_{\mathcal{M}} \rangle$ is isomorphic to the natural numbers with the usual arithmetic operations. Now let $E'(v) = (A \supset v)$. The embedding E has the desired properties. ∎

Remark: The proof does not use any flexible symbols other than one flexible constant symbol. Therefore, the theorem holds even for restricted FTL languages where all predicate symbols and (non-constant) function symbols are rigid.

Also, the proof does not require quantification in modal contexts: A does not contain quantifiers in modal contexts, and v does not contain modal operators. As usual, all quantifiers can be extracted (using skolemization); thus, the proof indicates that the validity problem for FTL sentences of the form $\exists \vec{x}.u$, where u is quantifier-free, remains Π^1_1-complete. On the other hand, the validity problem for quantifier-free formulas is clearly decidable, since it is essentially a propositional temporal problem. ∎

Remark: The theorem was first proved by Parikh ([Pa2]). We have proved it again independently. A third proof could exploit the theory of dominoes, as Harel's intractability proof for first-order dynamic logic ([Ha2]). ∎

2. Weaker notions of completeness

Thus, the standard concept of validity is overly demanding from a theorem-proving perspective: no practical system could possibly be complete with respect to $\models$. Like arithmetic ([Go]), temporal logic has no recursively enumerable axiomatization, that is, it does not have any useful axiomatization at all. Weaker, recursively enumerable alternatives to the standard concept of validity may be more appropriate and useful in the study of FTL proof systems.

For instance, consider equipping a two-sorted classical language L_O with a proof system. Take $\vdash_O$ to mean provability in first-order logic. Of course, we include the usual equality axioms (reflexivity, symmetry, transitivity, substitutivity), and functionality axioms to the effect that there is a unique 0 and that successors are unique; we also give natural axioms for 0, s, and $\leq$ ([MW]):

$$\vdash_O \forall i.(s(i) \neq 0),$$

$$\vdash_O \forall i \forall j.(s(i) = s(j) \supset i = j),$$

$$\vdash_O \forall i.((i \leq 0) \equiv (i = 0)),$$

$$\vdash_O \forall i \forall j.(i \leq s(j) \equiv (i = s(j) \vee i \leq j)),$$

$$\vdash_O u[0] \wedge (\forall i.u[i] \supset u[s(i)]) \supset (\forall i.u[i]).$$

Given a proof concept $\vdash$ for FTL and a formula u, we may ask whether

$$\vdash u \text{ if and only if } \vdash_O P(u).$$

The proof concept $\vdash_O$ takes into account a large class of nonstandard models, that is, provability with $\vdash_O$ is equivalent to validity over a class of models much larger than the class of standard models. The equivalence "$\vdash u$ if and only if $\vdash_O P(u)$" is the least restrictive requirement we find acceptable: any FTL system incomplete with respect to $\vdash_O$ should probably be replaced with a system that translates temporal formulas into classical formulas and then uses $\vdash_O$.

Now suppose that the function symbols $+$ and $\times$ are added to L_O—again, it is practical to regard them as abbreviations for the ternary predicate symbols $+_p$ and $\times_p$, respectively. The new language is L_P ("P" stands for "Peano"). The usual Peano axioms for $+$ and $\times$ are added to $\vdash_O$:

$$\vdash_P \forall i.(i + 0 = i),$$

$$\vdash_P \forall i \forall j.(i + s(j) = s(i + j)),$$

$$\vdash_P \forall i.(i \times 0 = 0),$$

$$\vdash_P \forall i \forall j.(i \times s(j) = i \times j + i).$$

Also, the induction schema is extended to the language with $+$ and $\times$. We obtain $\vdash_P$. Note that $\vdash_P$ is strictly more powerful than $\vdash_O$ ([BS]). Given a proof concept $\vdash$ for FTL and a formula u, we may ask whether

$$\vdash u \text{ if and only if } \vdash_P P(u).$$

Since Peano Arithmetic proves almost all valid sentences of practical interest, completeness with respect to $\vdash_P$ is an attractive requirement.

4. A basic proof system

We start the study of FTL proof systems with a description of a basic Hilbert system T_0. This system is an extension to our languages of the Hilbert system of Manna and Pnueli ([MP2]), equivalent to the resolution system of Abadi and Manna ([AM]). We prove a very nonstandard completeness theorem and a nonstandard incompleteness theorem for T_0.

1. The proof system

Many Hilbert systems have been given for linear-time propositional temporal logics. Gabbay, Pnueli, Shelah, and Stavi proposed the following one ([GPSS]):

- If $\vdash_H u$ and $\vdash_H (u \supset v)$ then $\vdash_H v$.
- If $\vdash_H u$ then $\vdash_H \Box u$.
- If u is a tautology then $\vdash_H u$.
- $\vdash_H \Box(p \supset q) \supset (\Box p \supset \Box q)$.
- $\vdash_H \bigcirc(\neg p) \equiv \neg \bigcirc p$.
- $\vdash_H \bigcirc(p \supset q) \supset (\bigcirc p \supset \bigcirc q)$.
- $\vdash_H \Box p \supset p \wedge \bigcirc \Box p$.
- $\vdash_H \Box(p \supset \bigcirc p) \supset (p \supset \Box p)$.
- $\vdash_H \Box p \supset (p \,\mathcal{U}\, q)$.
- $\vdash_H (p \,\mathcal{U}\, q) \equiv q \vee (p \wedge \bigcirc(p \,\mathcal{U}\, q))$.

This system is complete for PTL with the operators $\bigcirc$, $\Box$, and $\mathcal{U}$. When the two axioms involving $\mathcal{U}$ are deleted, the system is complete for PTL with the operators $\bigcirc$ and $\Box$. Finally, the axioms $\vdash_H \Diamond p \equiv \neg \Box \neg p$ and $\vdash_H (p \,\mathcal{P}\, q) \equiv \neg((\neg p) \,\mathcal{U}\, q)$ handle $\Diamond$ and $\mathcal{P}$.

Manna and Pnueli have presented a similar Hilbert system for PTL and have extended it to a variant of FTL where only constant symbols may be flexible. This extension is based on the addition of traditional quantifier rules and a variant of the Barcan axiom, $(\forall x. \Box u) \supset (\Box \forall x.u)$.

We describe a basic Hilbert system T_0 for FTL that also relies on traditional quantifier rules and variants of the Barcan axiom:

- If $\vdash_{T_0} u$ and $\vdash_{T_0} (u \supset v)$ then $\vdash_{T_0} v$.

- If $\vdash_{T_0} u$ then $\vdash_{T_0} \Box u$.
- If $\vdash_{T_0} (u \supset v)$ and x is not free in u then $\vdash_{T_0} (u \supset \forall x.v)$.
- If u is an instance of a schema valid in PTL then $\vdash_{T_0} u$.
- If u is a rigid formula then $\vdash_{T_0} u \equiv \bigcirc u$.
- If u is an equality axiom then $\vdash_{T_0} u$.
- $\vdash_{T_0} \exists x.\neg u \equiv \neg \forall x.u$.
- $\vdash_{T_0} (\forall x.w) \supset w\theta$ where θ is the substitution $\{x \leftarrow t\}$ and does not create any new bound occurrences of variables or any new occurrences of flexible terms in the scope of modal operators.
- $\vdash_{T_0} (\forall x. \bigcirc u) \equiv (\bigcirc \forall x.u)$.
- If x is not free in v then $\vdash_{T_0} [\forall x.(u \,\mathcal{U}\, v)] \equiv [(\forall x.u) \,\mathcal{U}\, v]$.

The first axiom schema, "if u is an instance of a schema valid in PTL then $\vdash_{T_0} u$," conveniently abstracts away all details of how PTL proofs are constructed. Of course, the schema could be replaced with any complete system for PTL, such as the one described above. In particular, we would have an induction schema, $\Box(u \supset \bigcirc u) \supset (u \supset \Box u)$.

The last two axiom schemas resemble the Barcan axiom. They attempt to capture the fact that the domain of discourse is time-independent.

2. A very nonstandard completeness theorem

In this section we present a nonstandard strong-completeness theorem for T_0. We say that the theorem is very nonstandard because it states the equivalence between provability in T_0 and validity over a large class of models. The theorem answers an immediate, natural question about T_0 and is useful in later sections.

A set of formulas Σ is *T_0-consistent* if $\not\vdash_{T_0} \neg(u_1 \wedge \ldots \wedge u_n)$ for all $u_1, \ldots, u_n \in \Sigma$. A *model of T_0* is a model where all the theorems of T_0 hold at every world. The system T_0 is trivially sound with respect to models of T_0; the following theorem says that T_0 is also complete with respect to models of T_0. The proof of the theorem is based on a new variation on well-known techniques for constructing a model from a consistent set of formulas.

Theorem 4.1. *Very nonstandard strong completeness*

If the set of formulas Σ is T_0-consistent then Σ holds in some model of T_0.

Proof:
The proof of strong completeness requires only usual techniques (e.g., [Gar], pp. 273–276) when the logic does not include $\mathcal{U}$ and $\mathcal{P}$. When the logic does include $\mathcal{U}$ and $\mathcal{P}$, these techniques fail. Moreover, typical known results for PTL (e.g., [GPSS]) do not extend to FTL because the models constructed in their proofs satisfy formulas with $\mathcal{U}$ and $\mathcal{P}$ only after some steps impossible in first-order logics. Burgess ([Bu1]) has proven other completeness theorems for logics with the operators "since" and "until." However, his results do not immediately apply to a logic without past operators, because in his system reasoning about the future may include intermediate deductions about the past, and it is not obvious that these can be avoided.

We give a new construction of models of modal logics, to obtain a model M for a consistent set of formulas Σ. We emphasize the difficulties found in the propositional case. Our technique has the feature of extending to the first-order logic without any new insights.

We prove the theorem for languages with no flexible function symbols. This limited version is all we use later on (in section 6) and entails no loss of generality (as is checked in the appendix).

Throughout the proof, we claim that certain formulas are theorems of T_0 when they are instances of simple valid PTL schemas. Similarly, we use some derived inference rules, such as "if $\vdash_{T_0} u$ then $\vdash_{T_0} \bigcirc u$," that follow from the rule "if $\vdash_{T_0} u$ then $\vdash_{T_0} \Box u$" by propositional temporal reasoning.

In the propositional case, given a T_0-consistent set of formulas Σ we define a model M_0 as follows:

$W = \{(\Sigma_0, u_0) | \Sigma_0$ *is a maximally consistent set of formulas and* u_0 *is a formula*$\}$,

$w_0 = (\Sigma^*, \mathit{false})$ where Σ^* is any maximally consistent extension of $\Sigma \cup \{u | \vdash_{T_0} u\}$,

$R_1 = \{((\Sigma_0, u_0), (\Sigma_1, u_1)) | \mathit{for\ all}\ u, \bigcirc u \in \Sigma_0 \ \Rightarrow\ u \in \Sigma_1\}$,

$R_2 = \{((\Sigma_0, u_0), (\Sigma_1, u_1)) | u_1 \notin \Sigma_0, u_1 \notin \Sigma_1, \mathit{and\ for\ all}\ u, u\,\mathcal{U}\,u_1 \in \Sigma_0 \ \Rightarrow\ u \in \Sigma_1\}$,

p holds at (Σ_0, u_0) if and only if $p \in \Sigma_0$.

Then we remove all worlds not reachable from w_0, to obtain the model M.

Intuitively, the definitions identify the world $w = (\Sigma_0, u_0)$ with the set of formulas Σ_0 that hold at w. The second component of a world, a formula u_0, is a new technical device whose intuitive meaning is explained below. Lindenbaum's Lemma guarantees that if Σ is T_0-consistent then $\Sigma \cup \{u | \vdash_{T_0} u\}$ has a maximally consistent extension. Therefore, we may choose the initial world w_0 to satisfy Σ and all the theorems of T_0. As usual, we define accessibility in such a way that one world is accessible from another world if this relation is acceptable in view of the formulas that the worlds contain. Thus, wR_1w' whenever if

$\bigcirc u$ is in w then u is in w'. In usual completeness proofs, R_2 is defined similarly: wR_2w' whenever if $\Box u$ is in w then u is in w'. The second argument of w', the formula u', makes possible an important refinement: only the formulas that need to hold until u' must be in w'. In other words, we intend to read wR_2w' as "w' comes after w, but before u' holds."

All elements of Σ appear in the initial world. Also, all theorems of T_0 appear in all worlds, since if $\vdash_{T_0} u$ then $\vdash_{T_0} \bigcirc u$ and $\vdash_{T_0} u \,\mathcal{U}\, u_1$.

Note that if we had a rigid proposition symbol p then p would receive the same value in all worlds, since $\vdash_{T_0} p \equiv \bigcirc p$, and hence (by propositional temporal reasoning) $\vdash_{T_0} [p \supset \bigcirc p] \wedge [\neg p \supset \bigcirc \neg p]$ and $\vdash_{T_0} [p \supset (p \,\mathcal{U}\, u_1)] \wedge [\neg p \supset ((\neg p) \,\mathcal{U}\, u_1)]$.

Now we prove that all elements of Σ hold in the initial world and all theorems of T_0 hold in all worlds, via a more general lemma:

Lemma 4.2. *Truth lemma*

For every world (Σ_0, u_0), membership in Σ_0 and truth in (Σ_0, u_0) are equivalent, that is,

$$u \in \Sigma_0 \;\Leftrightarrow\; \mathcal{M}@(\Sigma_0, u_0) \models u.$$

Proof:
The proof proceeds by induction on the depth of modal operators in u. The base case, where u is classical, is straightforward. For the inductive step, the proof proceeds by induction on the structure of u. We establish the result for the base case, that is, for formulas where the main connective is a modal operator. The cases for $\mathcal{U}$ and $\mathcal{P}$ subsume those for $\Box$ and $\Diamond$, since $\vdash_{T_0} (\Box u) \equiv (u \,\mathcal{U}\, \mathit{false})$ and $\vdash_{T_0} (\Diamond u) \equiv (u \,\mathcal{P}\, \mathit{false})$. We omit the routine arguments for the classical connectives, which constitute the inductive step.

- Suppose that $\bigcirc v \in \Sigma_0$. We want to show that $\bigcirc v$ holds at Σ_0, that is, $v \in \Sigma_1$ for some (Σ_1, u_1) such that $(\Sigma_0, u_0) R_1 (\Sigma_1, u_1)$. Note that if $\vdash_{T_0} \neg u$ then $\vdash_{T_0} \neg \bigcirc u$; moreover, $\vdash_{T_0} (\bigcirc u \wedge \bigcirc u') \supset \bigcirc(u \wedge u')$ and $\vdash_{T_0} \neg \bigcirc u \equiv \bigcirc \neg u$. Hence, since Σ_0 is consistent, $\Sigma_0^+ = \{u | \bigcirc u \in \Sigma_0\}$ is consistent as well: suppose that

  $$\vdash_{T_0} \neg(u^1 \wedge \ldots \wedge u^k) \;\text{ for some } u^1, \ldots, u^k \in \Sigma_0^+,$$

  then

  $$\vdash_{T_0} \neg \bigcirc(u^1 \wedge \ldots \wedge u^k),$$

  and hence

  $$\vdash_{T_0} \neg((\bigcirc u^1) \wedge \ldots \wedge (\bigcirc u^k)),$$

with $(\bigcirc u^1), \ldots, (\bigcirc u^k) \in \Sigma_0$. Let Σ_1 be a maximally consistent extension of Σ_0^+ (there is one, by Lindenbaum's Lemma) and u_1 an arbitrary formula. Clearly, $v \in \Sigma_1$ and $(\Sigma_0, u_0) R_1 (\Sigma_1, u_1)$.

- Suppose that $v \mathcal{P} v' \in \Sigma_0$. We want to show that $v \mathcal{P} v'$ holds at (Σ_0, u_0), that is, for some (Σ_1, u_1) such that $(\Sigma_0, u_0) R_2 (\Sigma_1, u_1)$, $v \in \Sigma_1$ and, for all (Σ_2, u_2) between (Σ_0, u_0) and (Σ_1, u_1) in the R_2 relation, $v' \notin \Sigma_2$. Note that if $\vdash_{T_0} \neg u$ then $\vdash_{T_0} \Box \neg u$ and hence $\vdash_{T_0} \neg (u \mathcal{P} v')$; also $\vdash_{T_0} (u \mathcal{P} v') \wedge (u' \mathcal{U} v') \supset ((u \wedge u') \mathcal{P} v')$. Hence, since Σ_0 is consistent, $\Sigma_0^+ = \{v\} \cup \{u | u \mathcal{U} v' \in \Sigma_0\}$ is consistent as well: suppose that

$$\vdash_{T_0} \neg (v \wedge u^1 \wedge \ldots \wedge u^k) \text{ for some } u^1, \ldots, u^k \in \Sigma_0^+,$$

then

$$\vdash_{T_0} \neg \big((v \wedge u^1 \wedge \ldots \wedge u^k) \mathcal{P} v'\big),$$

and hence

$$\vdash_{T_0} \neg \big((v \mathcal{P} v') \wedge (u^1 \mathcal{U} v') \wedge \ldots \wedge (u^k \mathcal{U} v')\big),$$

with $(v \mathcal{P} v'), (u^1 \mathcal{U} v'), \ldots, (u^k \mathcal{U} v') \in \Sigma_0$. Let Σ_1 be a maximally consistent extension of Σ_0^+ (there is one, by Lindenbaum's Lemma) and $u_1 = v'$. Clearly, $v \in \Sigma_1$. Also, $(\Sigma_0, u_0) R_2 (\Sigma_1, v')$:
1) $v' \notin \Sigma_0$, since $v \mathcal{P} v' \in \Sigma_0$ and $\vdash_{T_0} v \mathcal{P} v' \supset \neg v'$.
2) $v' \notin \Sigma_1$, since $\vdash_{T_0} (\neg v') \mathcal{U} v'$, and hence $(\neg v') \mathcal{U} v' \in \Sigma_0$ and $\neg v' \in \Sigma_1$.
3) If $u \mathcal{U} v' \in \Sigma_0$ then $u \in \Sigma_1$, by the construction of Σ_1.
Now suppose that for some (Σ_2, u_2) we have $(\Sigma_0, u_0) R_2 (\Sigma_2, u_2) R_2 (\Sigma_1, v')$. By the definition of R_2, $v' \notin \Sigma_2$, as desired.

- Suppose that $v \mathcal{U} v' \in \Sigma_0$. We want to show that $v \mathcal{U} v'$ holds at (Σ_0, u_0), that is, for all (Σ_1, u_1) such that $(\Sigma_0, u_0) R_2 (\Sigma_1, u_1)$, $v \in \Sigma_1$ or, for some (Σ_2, u_2) between (Σ_0, u_0) and (Σ_1, u_1) in the R_2 relation, $v' \in \Sigma_2$. Consider an arbitrary (Σ_1, u_1) such that $(\Sigma_0, u_0) R_2 (\Sigma_1, u_1)$. Either $v \mathcal{U} u_1 \in \Sigma_0$ or $v' \mathcal{P} u_1 \in \Sigma_0$ because $\vdash_{T_0} v \mathcal{U} v' \supset [(v \mathcal{U} u_1) \vee (v' \mathcal{P} u_1)]$. In the former case, $v \in \Sigma_1$ by the definition of R_2. In the latter case, we assume that $v \notin \Sigma_1$ and construct a world between (Σ_0, u_0) and (Σ_1, u_1) where v' appears. If $v' \in \Sigma_1$ then we take $(\Sigma_2, u_2) = (\Sigma_1, u_1)$. Clearly, $v' \in \Sigma_1$. The accessibility conditions are fulfilled: since $\vdash_{T_0} u \mathcal{U} u_1 \supset (u \vee u_1)$ and $u_1 \notin \Sigma_1$, if $u \mathcal{U} u_1 \in \Sigma_2$ then $u \in \Sigma_1$. Now suppose that $v' \notin \Sigma_1$. Since $v \notin \Sigma_1$ we get $(\neg v) \mathcal{P} u_1 \in \Sigma_0$. Then $v' \mathcal{P} (\neg v \wedge \neg v') \in \Sigma_0$ follows, since $\vdash_{T_0} ((\neg v) \mathcal{P} u_1) \wedge (v \mathcal{U} v') \supset (v' \mathcal{P} (\neg v \wedge \neg v'))$. Note that if $\vdash_{T_0} \neg u$ then $\vdash_{T_0} \neg (u \mathcal{P} v')$; also, $\vdash_{T_0} (u \mathcal{P} v') \wedge (u' \mathcal{U} v') \supset ((u \wedge u') \mathcal{P} v')$. Hence, since Σ_0 is consistent, $\Sigma_0^+ = \{v'\} \cup \{u' | u' \mathcal{U} (\neg v \wedge \neg v') \in \Sigma_0\}$ is consistent as well (the argument is similar to the one in the previous case). Let Σ_2 be a

maximally consistent extension of Σ_0^+ (there is one, by Lindenbaum's Lemma) and $u_2 = (\neg v \wedge \neg v')$. Clearly, $v' \in \Sigma_2$. To show that $(\Sigma_2, \neg v \wedge \neg v')$ is between (Σ_0, u_0) and (Σ_1, u_1), we check:
1) $(\neg v \wedge \neg v') \notin \Sigma_0$ follows from $v \,\mathcal{U}\, v' \in \Sigma_0$ and $\vdash_{T_0} v \,\mathcal{U}\, v' \supset (v \vee v')$.
2) $(\neg v \wedge \neg v') \notin \Sigma_2$ follows from $v' \in \Sigma_2$.
3) If $u \,\mathcal{U}\, (\neg v \wedge \neg v') \in \Sigma_0$ then $u \in \Sigma_2$, by the construction of Σ_2.
4) $u_1 \notin \Sigma_2$: Since $(\neg v \wedge \neg v') \in \Sigma_1$, $(\neg v \wedge \neg v') \,\mathcal{P}\, u_1 \in \Sigma_0$. Then note that $\vdash_{T_0} (\neg v \wedge \neg v') \,\mathcal{P}\, u_1 \supset (\neg u_1) \,\mathcal{U}\, (\neg v \wedge \neg v')$. Thus, $(\neg u_1) \,\mathcal{U}\, (\neg v \wedge \neg v') \in \Sigma_0$ and hence $\neg u_1 \in \Sigma_2$ (since $(\Sigma_0, u_0) R_2 (\Sigma_1, u_1)$, as (1), (2), and (3) guarantee).
5) $u_1 \notin \Sigma_1$ follows from the hypothesis that $(\Sigma_0, u_0) R_2 (\Sigma_1, u_1)$.
6) If $u \,\mathcal{U}\, u_1 \in \Sigma_2$ then $u \in \Sigma_1$: Equivalently, we show that if $u \in \Sigma_1$ then $u \,\mathcal{P}\, u_1 \in \Sigma_2$: Suppose that $u \in \Sigma_1$. Then $\neg v \wedge \neg v' \wedge u \in \Sigma_1$. Therefore, $(\neg v \wedge \neg v' \wedge u) \,\mathcal{P}\, u_1 \in \Sigma_0$. Note that

$$\vdash_{T_0} \big((\neg v \wedge \neg v' \wedge u) \,\mathcal{P}\, u_1\big) \supset \big[\big((\neg v \wedge \neg v' \wedge u) \,\mathcal{P}\, u_1\big) \,\mathcal{U}\, (\neg v \wedge \neg v')\big],$$
$$\vdash_{T_0} \big[\big((\neg v \wedge \neg v' \wedge u) \,\mathcal{P}\, u_1\big) \,\mathcal{U}\, (\neg v \wedge \neg v')\big] \supset \big((u \,\mathcal{P}\, u_1) \,\mathcal{U}\, (\neg v \wedge \neg v')\big).$$

Thus, $(u \,\mathcal{P}\, u_1) \,\mathcal{U}\, (\neg v \wedge \neg v') \in \Sigma_0$ and $u \,\mathcal{P}\, u_1 \in \Sigma_2$ (since $(\Sigma_0, u_0) R_2 (\Sigma_1, u_1)$, as (1), (2), and (3) guarantee).

In all three cases, duality considerations ease the proof of the other direction of the equivalence.

- Suppose that $\bigcirc v$ holds at Σ_0. We want to show that $\bigcirc v \in \Sigma_0$. If $\bigcirc v$ holds at Σ_0 then some (Σ_1, u_1) such that $(\Sigma_0, u_0) R_1 (\Sigma_1, u_1)$ contains v. Since $\neg v \notin \Sigma_1$, the definition of R_1 yields $\bigcirc \neg v \notin \Sigma_0$. Since $\vdash_{T_0} \neg \bigcirc v \equiv \bigcirc \neg v$, we obtain $\bigcirc v \in \Sigma_0$.

- Suppose that $v \,\mathcal{P}\, v'$ holds at Σ_0. We want to show that $v \,\mathcal{P}\, v' \in \Sigma_0$. We suppose that $v \,\mathcal{P}\, v' \notin \Sigma_0$ to show that $v \,\mathcal{P}\, v'$ does not hold at Σ_0, and thus obtain a contradiction. Since $v \,\mathcal{P}\, v' \notin \Sigma_0$ and $\vdash_{T_0} \neg (v \,\mathcal{P}\, v') \equiv ((\neg v) \,\mathcal{U}\, v')$, we have that $(\neg v) \,\mathcal{U}\, v' \in \Sigma_0$. Since the depth of modal operators in $\neg (v \,\mathcal{P}\, v')$ and $(\neg v) \,\mathcal{U}\, v'$ is the same, this implies that $(\neg v) \,\mathcal{U}\, v'$ holds at Σ_0. Hence, $v \,\mathcal{P}\, v'$ does not hold at Σ_0.

- Suppose that $v \,\mathcal{U}\, v'$ holds at Σ_0. We want to show that $v \,\mathcal{U}\, v' \in \Sigma_0$. We suppose that $v \,\mathcal{U}\, v' \notin \Sigma_0$ to show that $v \,\mathcal{U}\, v'$ does not hold at Σ_0, and thus obtain a contradiction. Since $v \,\mathcal{U}\, v' \notin \Sigma_0$ and $\vdash_{T_0} \neg (v \,\mathcal{U}\, v') \equiv ((\neg v) \,\mathcal{P}\, v')$, we have that $(\neg v) \,\mathcal{P}\, v' \in \Sigma_0$. Since the depth of modal operators in $\neg (v \,\mathcal{U}\, v')$ and $(\neg v) \,\mathcal{P}\, v'$ is the same, this implies that $(\neg v) \,\mathcal{P}\, v'$ holds at Σ_0. Hence, $v \,\mathcal{U}\, v'$ does not hold at Σ_0. ∎

This concludes the propositional model construction. The first-order model construction is based on the propositional one. As is typical for first-order completeness proofs, the

sets of formulas that appear in the construction of worlds are not only maximally consistent but also saturated. (Recall that $\Sigma_0 \vdash_{T_0} u$ if for some $u_1, \ldots, u_n \in \Sigma_0$, $\vdash_{T_0} (u_1 \wedge \ldots \wedge u_n) \supset u$; Σ_0 is *omega-complete* if $\Sigma_0 \vdash_{T_0} u[t]$ for every term t implies $\Sigma_0 \vdash_{T_0} \forall x.u[x]$, where x is a new variable; Σ_0 is *saturated* if it is both omega-complete and maximally consistent.)

Fortunately, the extension to FTL requires only two new propositions—all other details are very similar to those given in [Gar], pp. 273–276; in particular, the extension is considerably simplified by the presence of the Barcan formula. We show that if Σ_0 is a saturated set then $\{u | \bigcirc u \in \Sigma_0\}$ and $\{u | u \,\mathcal{U}\, v' \in \Sigma_0\}$ are omega-complete (this is analogous to the key step in lemma 3 in p. 275 of [Gar]).

Proposition 4.3.

If Σ_0 is a saturated set then $\{u | \bigcirc u \in \Sigma_0\}$ is omega-complete.

Proof:
Suppose that

$$\{v | \bigcirc v \in \Sigma_0\} \vdash_{T_0} u[t]$$

for every term t. If $\vdash_{T_0} v \supset u[t]$ then $\vdash_{T_0} (\bigcirc v) \supset (\bigcirc u[t])$; also, $\vdash_{T_0} (\bigcirc u_0) \wedge (\bigcirc u_1) \supset \bigcirc(u_0 \wedge u_1)$. Hence, $\Sigma_0 \vdash \bigcirc u[t]$ for every t, and $\Sigma_0 \vdash_{T_0} \forall x.(\bigcirc u[x])$ since Σ_0 is omega-complete. Furthermore, $\vdash_{T_0} \forall x.(\bigcirc u[x]) \supset \bigcirc(\forall x.u[x])$. Since Σ_0 is maximally consistent, $\bigcirc(\forall x.u[x]) \in \Sigma_0$. Thus, $(\forall x.u[x]) \in \{v | \bigcirc v \in \Sigma_0\}$ and, immediately,

$$\{v | \bigcirc v \in \Sigma_0\} \vdash_{T_0} (\forall x.u[x]),$$

as desired. ∎

Proposition 4.4.

If Σ_0 is a saturated set then $\{u | u \,\mathcal{U}\, v' \in \Sigma_0\}$ is omega-complete for every v'.

Proof:
Suppose that

$$\{v | v \,\mathcal{U}\, v' \in \Sigma_0\} \vdash_{T_0} u[t]$$

for every term t. If $\vdash_{T_0} v \supset u[t]$ then $\vdash_{T_0} (v \,\mathcal{U}\, v') \supset (u[t] \,\mathcal{U}\, v')$; also, $\vdash_{T_0} (u_0 \,\mathcal{U}\, v') \wedge (u_1 \,\mathcal{U}\, v') \supset ((u_0 \wedge u_1) \,\mathcal{U}\, v')$. Hence, $\Sigma_0 \vdash (u[t]) \,\mathcal{U}\, v'$ for every t, and $\Sigma_0 \vdash_{T_0} \forall x.[(u[x]) \,\mathcal{U}\, v']$ since Σ_0 is omega-complete. Furthermore, $\vdash_{T_0} \forall x.[(u[x]) \,\mathcal{U}\, v'] \supset [(\forall x.u[x]) \,\mathcal{U}\, v']$, because the new variable x does not occur in v'. Since Σ_0 is maximally consistent, $(\forall x.u[x]) \,\mathcal{U}\, v' \in \Sigma_0$. Thus, $(\forall x.u[x]) \in \{v | v \,\mathcal{U}\, v' \in \Sigma_0\}$ and, immediately,

$$\{v | v \,\mathcal{U}\, v' \in \Sigma_0\} \vdash_{T_0} (\forall x.u[x]),$$

as desired. ∎ ∎

Remark: The proof of theorem 4.1 does not require all schemas valid in PTL. For instance, we do not exploit the connections between $\bigcirc$ and $\mathcal{U}$. Thus, the same completeness proof applies to some other modal logics with different axioms about time. ▮

3. A nonstandard incompleteness theorem

The completeness theorem of the previous section may appear as a first promising step in proving that T_0 is reasonably powerful. Furthermore, systems equivalent to T_0 are often empirically satisfactory. As we prove here, however, $\vdash_{T_0}$ is surprisingly weak—not even complete with respect to $\vdash_O$:

Theorem 4.5. *Nonstandard incompleteness*

There is a formula u_0 such that $\vdash_O P(u_0)$ but $\not\vdash_{T_0} u_0$.

Proof:
Fix the language to contain only the flexible constant symbol a and the flexible predicate symbol p. Let u_0 be

$$[(\forall x.\, \Diamond\, a = x) \wedge p(a) \wedge (\forall x \forall y.(p(x) \wedge \Diamond(a = x \wedge \bigcirc a = y)) \supset p(y))] \supset (\forall x.p(x)).$$

Intuitively, u_0 says that if a enumerates the domain in a sequence of instants, p holds for the first element in the enumeration, and if it holds for an element then it holds for its successor in the enumeration, then it must hold for all elements in the domain. In a sense, u_0 establishes a connection between induction on time and induction in a domain.

Significantly, $\not\vdash_{T_0} u_0$. To show this, it suffices to construct a model $\mathcal{M}$ of T_0 where u_0 fails. Let the domain and the set of worlds in the model, D and W, both equal $N + Z$, that is, a copy of $\{0, 1, \ldots\}$ and a copy of $\{\ldots, -1', 0', 1', \ldots\}$. The initial world w_0 is 0, R_1 is the union of the successor functions on N and Z, R_2 is $\leq$ with $m \leq n'$ for all m and n (in other words, we put N before Z). In the initial world, $p(x)$ holds if and only if x is in N; in all other worlds, $p(x)$ is always false. In world i, a has value i.

The formula

$$(\forall x.\, \Diamond\, a = x) \wedge p(a) \wedge (\forall x \forall y.(p(x) \wedge \Diamond(a = x \wedge \bigcirc a = y)) \supset p(y))$$

holds in $\mathcal{M}$. However, $(\forall x.p(x))$ does not hold. Therefore, u_0 is falsified in this model. We still need to check that $\mathcal{M}$ is a model of T_0. We show that all instances of schemas valid in PTL hold at every world; all the other axioms and rules of T_0 are sound for constant-domain possible-world models. In fact, we only consider the schemata $(\bigcirc \neg u) \equiv \neg(\bigcirc u)$, $\Box u \supset (u \wedge \bigcirc \Box u)$, $u \mathcal{U} v \equiv [v \vee u \wedge \bigcirc(u \mathcal{U} v)]$, and $\Box(u \supset \bigcirc u) \supset (u \supset \Box u)$—since these

schemata, added to others which hold for all possible-world models, yield the complete PTL axiom system of Gabbay, Pnueli, Shelah, and Stavi ([GPSS]).

- The schema $(\bigcirc \neg u) \equiv \neg(\bigcirc u)$ holds at all worlds in $\mathcal{M}$ since R_1 is a function.
- The schema $\Box u \supset (u \wedge \bigcirc \Box u)$ follows from $u \,\mathcal{U}\, v \supset [v \vee u \wedge \bigcirc(u \,\mathcal{U}\, v)]$ (proved below), since $\Box u \equiv (u \,\mathcal{U}\, \mathit{false})$ holds in all possible-world models.
- The schema $u \,\mathcal{U}\, v \supset [v \vee u \wedge \bigcirc(u \,\mathcal{U}\, v)]$ holds at all worlds in $\mathcal{M}$: Assume that $u \,\mathcal{U}\, v$ holds at some arbitrary world w_1. Since R_2 is reflexive, either v or u must hold at w_1 (by the semantics of $\mathcal{U}$ for possible-world models). If v holds then $v \vee u \wedge \bigcirc(u \,\mathcal{U}\, v)$ holds. Otherwise, consider the world w_2 such that $w_1 R_1 w_2$. It suffices to show that $u \,\mathcal{U}\, v$ holds at w_2, that is, that u holds until v holds in the "future" of w_2. Since $R_1 \subset R_2$ and R_2 is transitive, any w_3 such that $w_2 R_2 w_3$ also satisfies $w_1 R_2 w_3$. By our hypothesis, either u holds at w_3 or v holds at some world w_4 such that $w_1 R_2 w_4$ and $w_4 R_2 w_3$. Since v does not hold at w_1 and $R_2 - (R_1 \circ R_2) = \{(w, w) | w \in W\}$, w_4 must satisfy $w_2 R_2 w_4$ and $w_4 R_2 w_3$. In short, for any w_3 such that $w_2 R_2 w_3$, either u holds or v holds at some w_4 such that $w_2 R_2 w_4$ and $w_4 R_2 w_3$, that is, $u \,\mathcal{U}\, v$ holds at w_2.
- The schema $[v \vee u \wedge \bigcirc(u \,\mathcal{U}\, v)] \supset u \,\mathcal{U}\, v$ holds at all worlds in $\mathcal{M}$: Assume that $v \vee u \wedge \bigcirc(u \,\mathcal{U}\, v)$ holds at some arbitrary world w_1. If v holds at w_1, $u \,\mathcal{U}\, v$ holds as well, by the reflexivity of R_2 and the semantics of $\mathcal{U}$. Otherwise, assume u holds at w_1 and $u \,\mathcal{U}\, v$ holds at its successor world, w_2. Therefore, for all w_3 such that $w_2 R_2 w_3$ either u holds or for some w_4 such that $w_2 R_2 w_4$ and $w_4 R_2 w_3$, v holds. Since u holds at w_1 and $R_2 - (R_1 \circ R_2) = \{(w, w) | w \in W\}$, we can derive that for all w_3 such that $w_1 R_2 w_3$ either u holds or, for some w_4 such that $w_2 R_2 w_4$ and $w_4 R_2 w_3$, v holds. Since $R_1 \subset R_2$ and R_2 is transitive, any such w_4 must also satisfy $w_1 R_2 w_4$, so for all w_3 such that $w_1 R_2 w_3$ either u holds or for some w_4 such that $w_1 R_2 w_4$ and $w_4 R_2 w_3$, v holds, that is, $u \,\mathcal{U}\, v$ holds at w_1.
- The induction schema, $\Box(u \supset \bigcirc u) \supset (u \supset \Box u)$, is satisfied at all worlds in $\mathcal{M}$: Suppose that for a formula u we have u and $\Box(u \supset \bigcirc u)$ at some world w_1. We want to show that $\Box u$ holds at w_1. Certainly, u must hold at the world w_2 such that $w_1 R_1 w_2$. For all formulas v, for all i, j such that $1 R_2 i$ and $1 R_2 j$, that is, for all worlds accessible from world 1, v holds at world i if and only if v holds at world j. (The proof is a trivial inductive argument on the syntactic structure of v, where the base case concerns atomic formulas and the inductive step concerns formulas built up with the various connectives.) In particular, all worlds w_3 such that $w_2 R_2 w_3$ are indistinguishable from world w_2. Therefore, u must hold at all w_3 such that $w_2 R_2 w_3$. Since $R_2 - (R_1 \circ R_2) = \{(w, w) | w \in W\}$ and u holds at w_1, u holds at all w_3 such that $w_1 R_2 w_3$, that is, $\Box u$ holds at w_1.

Intuitively, however, u_0 is true. In fact, it is true in the standard semantics, and even in the weak nonstandard semantics determined by $\vdash_O$, since $\vdash_O P(u_0)$. The basic

idea behind the proof of $P(u_0)$ is that $p(a(i),0)$ implies $p(a(s(i)),0)$ for all i, so induction yields $p(a(i),0)$ for all i. Note that here p is interpreted at time 0 while its argument a refers to other times. In the modal system where time is implicit this double reference is impossible; therefore, we cannot formulate a temporal version of the simple inductive proof just described within a classical language. ▮

5. Auxiliary definitions

As we showed in the previous section, T_0 is surprisingly limited. An analysis of its incompleteness and of how informal temporal theorem proving is carried out gives rise to new rules. Similar rules can be added to other proposed FTL systems.

In the first subsection we describe two extensions of T_0. In the second subsection we give some preliminary completeness theorems that we use to obtain stronger ones in section 7. In the third subsection, we briefly discuss the connection between rules to discharge definitions and skolemization rules.

1. Two systems with auxiliary definitions

In practice, proofs often involve auxiliary predicate and function symbols. For instance, if we define the auxiliary rigid predicate q such that $q(x)$ holds if and only if $p(x)$ holds at the initial world, we obtain a proof for the sentence u_0 exhibited in the previous incompleteness theorem 4.5. We show that $\Box\, q(a)$ holds inductively, and then use $(\forall x.\, \Diamond\, a = x)$ to derive $\forall x.\, \Diamond\, q(x)$. Since q is rigid, this simplifies to $\forall x.q(x)$. By the definition of q, we reach the desired conclusion, $\forall x.p(x)$.

Useful auxiliary objects are not always rigid. In some cases, flexible objects have been introduced in informal proofs. Thus, Hailpern and Owicki have given inductive definitions for "history variables" and used them in proofs ([HO]).

We propose to allow definitions for rigid and flexible auxiliary predicates in formal proofs. Some restrictive kinds of definitions are actually sufficient for completeness purposes, but general forms seem more elegant and practical. Definitions for auxiliary functions could be formulated similarly. However, for simplicity, we derive them from definitions for predicates (since any provably functional predicate can be manipulated as the corresponding function).

Rigid predicates are defined explicitly by formulas of the form

$$\forall x_1 \ldots \forall x_k.p(x_1,\ldots,x_k) \equiv u,$$

where p is the new rigid predicate symbol being defined and p does not occur in u.

For flexible predicates, we are content with primitive-recursive definitions of the form

$$\forall x_1 \ldots \forall x_n . [p(x_1, \ldots, x_n) \equiv u \ \wedge \ \Box((\bigcirc p(x_1, \ldots, x_n)) \equiv v)],$$

where p is the new flexible predicate symbol, p does not occur in u, and p does not occur in the scope of any modal operator in v. We also require that p does not occur in the scope of $\forall$ or $\neg$ in v in order to keep definitions simple; this requirement is essential for our soundness results and seems generally easy to satisfy in practice. These definitions are analogous to primitive-recursive definitions in classical logic. Sometimes we refer to them simply as recursive definitions.

Definitions may be iterated, in the sense that defined symbols may be used in new definitions.

Given a temporal language, we add an infinite supply of new rigid and flexible predicate symbols for definitions to use. We require that definitions define only these predicate symbols. Note that there is a largest (typically infinite) set D_e of possible explicit definitions when only explicit definitions are considered, up to renaming of the defined predicate symbols. Similarly, there is a largest set D_{er} of possible explicit and recursive definitions when both explicit and recursive definitions are considered.

We extend the Hilbert system T_0 with sound rules to exploit definitions. We allow the discharge of explicit definitions at the end of proofs to obtain the system T_1 and the concept $\vdash_{T_1}$ from T_0 and $\vdash_{T_0}$:

- If $\vdash_{T_0} w$ then $\vdash_{T_1} w$.
- If $\vdash_{T_1} (d \supset w)$ and d defines a rigid predicate not occurring in w then $\vdash_{T_1} w$.

We allow the discharge of both explicit and primitive-recursive definitions at the end of proofs to obtain the system T_2 and the concept $\vdash_{T_2}$ from T_0 and $\vdash_{T_0}$:

- If $\vdash_{T_0} w$ then $\vdash_{T_2} w$.
- If $\vdash_{T_2} (d \supset w)$ and d defines a predicate not occurring in w then $\vdash_{T_2} w$.

Remark: We could allow the discharge of definitions at any point in proofs (rather than only at the end). However, this would unnecessarily complicate our proof systems. In particular, the soundness theorem of section 7 would still apply as an upper bound on the power of T_1 and T_2 after some minor modifications. ▮

2. Two very nonstandard completeness theorems

Consider two models $\mathcal{M}$ and $\mathcal{M}'$, with interpretation functions I and I', respectively.

We say that $\mathcal{M}'$ *expands* $\mathcal{M}$ (or is an *expansion* of $\mathcal{M}$) if $\mathcal{M}$ and $\mathcal{M}'$ are identical except that I' extends I to give meanings to some new predicate and function symbols. Expansions preserve the meaning of formulas that do not contain the new symbols:

Proposition 5.1.

If $\mathcal{M}'$ expands $\mathcal{M}$ and the formula u does not contain any of the symbols that $\mathcal{M}'$ interprets but $\mathcal{M}$ does not interpret, then $\mathcal{M} \models u \Leftrightarrow \mathcal{M}' \models u$.

Proof:
We prove that for all assignments α and all worlds w

$$(\mathcal{M} \cdot \alpha)@w \models u \Leftrightarrow (\mathcal{M}' \cdot \alpha)@w \models u$$

by induction on the structure of u. Both the base case and the inductive step are trivial. ∎

We say that $\mathcal{M}'$ *e-expands* $\mathcal{M}$ if $\mathcal{M}'$ expands $\mathcal{M}$ and satisfies each definition in D_e. The model $\mathcal{M}$ is a *model of* T_1 if some model of T_0 e-expands $\mathcal{M}$. Intuitively, models of T_1 are models of T_0 that can be expanded to satisfy all explicit definitions while remaining models of T_0.

T_1 is trivially sound with respect to models of T_1. A set of formulas Σ is T_1-*consistent* if $\not\vdash_{T_1} \neg(u_1 \wedge \ldots \wedge u_n)$ for all $u_1, \ldots, u_n \in \Sigma$. Theorem 4.1 immediately yields that T_1 is strongly complete with respect to models of T_1:

Proposition 5.2. *Very nonstandard strong completeness*

If the set of formulas Σ is T_1-consistent then Σ holds in some model of T_1.

Proof:
If Σ is T_1-consistent, then $\not\vdash_{T_1} \neg(u_1 \wedge \ldots \wedge u_n)$ for all $u_1, \ldots, u_n \in \Sigma$. The rule for definitions guarantees that $\not\vdash_{T_0} \neg(d_1 \wedge \ldots \wedge d_m \wedge u_1 \wedge \ldots \wedge u_n)$ for all $u_1, \ldots, u_n \in \Sigma$ and $d_1, \ldots, d_m \in D_e$. Therefore, $\Sigma \cup D_e$ is T_0-consistent, and has a model $\mathcal{M}$ of T_0 by theorem 4.1. In particular, $\mathcal{M}$ satisfies Σ. Furthermore, since $\mathcal{M}$ e-expands itself, $\mathcal{M}$ is also a model of T_1. ∎

Similarly, we say that $\mathcal{M}'$ *er-expands* $\mathcal{M}$ if $\mathcal{M}'$ expands $\mathcal{M}$ and satisfies each definition in D_{er}. The model $\mathcal{M}$ is a *model of* T_2 if some model of T_0 er-expands $\mathcal{M}$. Intuitively, models of T_2 are models of T_0 that can be expanded to satisfy all explicit and all recursive definitions while remaining models of T_0.

T_2 is trivially sound with respect to models of T_2. A set of formulas Σ is T_2-*consistent* if $\not\vdash_{T_2} \neg(u_1 \wedge \ldots \wedge u_n)$ for all $u_1, \ldots, u_n \in \Sigma$. Theorem 4.1 immediately yields that T_2 is strongly complete with respect to models of T_2:

Proposition 5.3. *Very nonstandard strong completeness*

If the set of formulas Σ is T_2-consistent then Σ holds in some model of T_2.

Proof:
The argument is identical to the previous one. ∎

Thus, models of T_1 and T_2 can be expanded to satisfy all the appropriate definitions at once. As we argue now, they can also be expanded to satisfy any chosen definitions. Let $\mathcal{M}$ be a model of T_1 (or T_2), and $\mathcal{M}'$ one such "full" expansion. Given a subset of D_e (or D_{er}) with definitions for the auxiliary predicates $p_1, p_2, \ldots$, we may restrict the interpretation function in $\mathcal{M}'$ to give meanings to the symbols in the original language, to $p_1, p_2, \ldots$, and to no other auxiliary symbol. The model we obtain, $\mathcal{M}''$, is an expansion of $\mathcal{M}$ and a model of T_1 (or T_2), since $\mathcal{M}'$ is an e-expansion (or an er-expansion) of $\mathcal{M}''$. Therefore, models of T_1 and T_2 can be expanded to satisfy arbitrary explicit or recursive definitions while remaining models of T_1 and T_2.

3. On resolution systems

In some resolution systems, skolemization has a role similar to that of the rules to discharge definitions because, intuitively, skolemization introduces an auxiliary function (instead of an auxiliary predicate).

For instance, suppose that there is a resolution proof of $(d \supset u)$, that is, refutation of $\neg(d \supset u)$, where d is a definition for the new rigid predicate $p(\vec{x})$ with the formula $w[\vec{x}]$. We can construct a resolution proof of u, that is, a refutation of $\neg u$. Intuitively, we introduce a skolem function instead of a defined predicate. The skolem function maps $\vec{x}$ to some distinguished element a if and only if $w[\vec{x}]$ holds in the present. A slight complication arises in that we need to consider the trivial case where the domain contains a single element.

In particular, the resolution system **R** ([A1]) includes a skolemization rule that introduces rigid skolem function symbols. Thus, our results on T_1 carry over to **R**. On the other hand, the resolution system of Abadi and Manna ([AM]) does not include a skolemization rule and is analogous to T_0.

6. Clocks and arithmetical formulas

In this section we define the class of arithmetical formulas. We have a completeness result for arithmetical formulas for T_1. Arithmetical formulas are also useful in the study of T_2, although our completeness theorem for T_2 is not restricted to arithmetical formulas.

1. Definitions and examples

Sometimes a clock is a useful device in proofs. For instance, a program counter may help us prove properties of a program even though the program never refers to the program counter. In some logics of programs, clocks are actually not only useful but sometimes necessary ([Pa1]).

Informally, a clock is a formula c that distinguishes a set of tuples of elements of the domain at each point, without repetitions. The distinguished tuples in a world can be thought of as the "time" of that world. (For instance, the formula $c[x]$ may be *program-counter* $= x$.) More precisely, c satisfies the *clock condition*

$$C(c): \ (\Box \exists \vec{x}.c[\vec{x}]) \wedge (\Box \forall \vec{x}.(c[\vec{x}] \supset \bigcirc \Box \neg c[\vec{x}])).$$

The formula c is *a clock for* u if we can use $C(c)$ to show u, that is, if we can prove $C(c) \supset u$ then we can prove u (or, as we often say, "u *reduces to* $C(c) \supset u$"). More precisely, consider a proof concept $\vdash$ (for instance, one of $\vdash_{T_0}$, $\vdash_{T_1}$, $\vdash_{T_2}$). The formula c is a clock for u in $\vdash$ if

$$\vdash C(c) \supset u \text{ implies } \vdash u.$$

Thus, the provability of u with a clock suffices to guarantee the provability of u. Note that we do not require that c be in the original temporal language under consideration: c may include auxiliary predicate symbols introduced in definitions. The formula u is *arithmetical* in $\vdash$ if there exists a clock for u in $\vdash$ (the name was chosen because arithmetical formulas have a most natural interpretation in arithmetical universes ([Ha1])).

Examples:

- The formula $\exists x \exists y.(x \neq y)$ is trivially not arithmetical, in any sound system: it can be proved using a clock (if a clock exists, then the domain contains two distinct values) but not without a clock. We do not know of more subtle examples of formulas which are not arithmetical.

- Consider $\vdash_{T_0}$. Suppose we are interested in

  $$u: \ \left[A \wedge \Box \forall z.(a = z \supset \bigcirc a > z)\right] \supset \forall y. \Diamond a > y,$$

  where A is some basic collection of axioms about arithmetic, a is a flexible constant symbol, and all other symbols are rigid. The formulas in the antecedent, A and $\Box \forall z.(a = z \supset \bigcirc a > z)$, imply that a must take a different value at each instant. Therefore, we can prove

  $$\left[A \wedge \Box \forall z.(a = z \supset \bigcirc a > z)\right] \supset C(a = x),$$

  and derive $\left[C(a = x) \supset u\right] \supset u$ by propositional reasoning. Therefore, $a = x$ is a clock for u.

- Consider $\vdash_{T_2}$. Suppose we are interested in

$$u: \ \left[\forall x.(s(x) \neq 0) \wedge \forall x \forall y.(s(x) = s(y) \supset x = y)\right] \supset v$$

for some formula v. For instance, we may imagine that v expresses some temporal property of programs on numbers, and that is why some basic facts about 0 and s appear in u. In T_2, we can give a definition d_c for a flexible predicate c that is a clock for u:

$$\forall x.\left[(c(x) \equiv (x = 0)) \wedge \Box((\bigcirc c(x)) \equiv \exists y.(c(y) \wedge x = s(y)))\right].$$

Intuitively, c is the clock that gives the times 0, 1, 2, ... Note that

$$\left[\forall x.(s(x) \neq 0) \wedge \forall x \forall y.(s(x) = s(y) \supset x = y) \wedge d_c\right] \supset C(c) \qquad (*)$$

is provable in T_2.

To check that c is a clock for u, we simply need to reduce u to $C(c) \supset u$. Suppose that $C(c) \supset u$ has a proof in T_2. Let D be the conjunction of the definitions involved in this proof and in the proof of $(*)$. Then $(D \wedge C(c)) \supset u$ is provable in T_0. By propositional reasoning,

$$\left[\forall x.(s(x) \neq 0) \wedge \forall x \forall y.(s(x) = s(y) \supset x = y) \wedge d_c \wedge D\right] \supset u$$

and hence $(d_c \wedge D) \supset u$ are also provable in T_0. We discharge the definitions, to conclude that u is provable in T_2. ∎

As the examples suggest, many of the formulas that arise in reasoning about computations are arithmetical. For the systems T_0 and T_1 it suffices that the formulas in question mention some (provably) non-repeating term—which may represent a program counter or just some program variable. Furthermore, all instances of valid PTL schemas are trivially arithmetical, since they are provable. It is therefore reasonable to suggest that many important FTL formulas are arithmetical in $\vdash_{T_0}$ and $\vdash_{T_1}$.

For the system of most interest to us, T_2, arithmetical formulas are even easier to find. Typically, infinite-state systems operate on domains such as the integers, the lists, or the strings. By proposition 6.1 (below), formulas that involve some basic theory of such a domain are arithmetical in $\vdash_{T_2}$. This basic theory needs to refer only to elementary facts about a "successor" operation, e.g., successor for integers, concatenation for lists and for strings; intuitively, this "successor" operation suffices to construct a clock, that ticks by applying the operation to its current value.

Given a formula $s_p[\vec{x}, \vec{y}]$, where $\vec{x}$ and $\vec{y}$ have the same length, let $Inj(s_p)$ denote

$$\forall\vec{x}\forall\vec{y}\forall\vec{z}.((s_p[\vec{x},\vec{y}] \wedge s_p[\vec{x},\vec{z}]) \supset \vec{y} = \vec{z})$$
$$\wedge$$
$$\forall\vec{x}\forall\vec{y}\forall\vec{z}.((s_p[\vec{x},\vec{z}] \wedge s_p[\vec{y},\vec{z}]) \supset \vec{x} = \vec{y})$$
$$\wedge$$
$$\forall\vec{x}\forall\vec{y}.\,\Box(s_p[\vec{x},\vec{y}] \equiv \bigcirc s_p[\vec{x},\vec{y}])$$

and $Range(s_p) \subset Domain(s_p)$ denote

$$\exists\vec{x}.(\exists\vec{y}.s_p[\vec{x},\vec{y}] \wedge \forall\vec{z}.\neg s_p[\vec{z},\vec{x}])$$
$$\wedge$$
$$\forall\vec{x}.(\exists\vec{y}.s_p[\vec{y},\vec{x}] \supset \exists\vec{z}.s_p[\vec{x},\vec{z}]).$$

The formula $Inj(s_p)$ asserts that s_p denotes a rigid injective partial function; the formula $Range(s_p) \subset Domain(s_p)$ that there is at least one "0-like" element to start a sequence of function applications.

Proposition 6.1.

If $\vdash_{T_2} [(Inj(s_p) \wedge (Range(s_p) \subset Domain(s_p))) \supset u] \supset u$ *for some formula* $s_p[\vec{x},\vec{y}]$ *then* u *is arithmetical in* $\vdash_{T_2}$.

Proof:
The flexible predicate c is defined by

$$\forall\vec{x}.[(c(\vec{x}) \equiv \exists\vec{y}.s_p[\vec{x},\vec{y}] \wedge \forall\vec{z}.\neg s_p[\vec{z},\vec{x}]) \wedge \Box((\bigcirc c(\vec{x})) \equiv \exists\vec{y}.(c(\vec{y}) \wedge s_p[\vec{y},\vec{x}]))].$$

The clock condition $C(c)$ can be proved from $Inj(s_p) \wedge (Range(s_p) \subset Domain(s_p))$ and the definition of c. As in the previous example, the reduction of u to $C(c) \supset u$ follows by propositional reasoning and definition discharges. ∎

2. A basic theorem

Intuitively, we would like to know that many formulas are arithmetical, because this makes easier completeness arguments: since we may use clocks to prove arithmetical formulas, these formulas may be easier to prove than arbitrary ones. As pointed out above, both propositional formulas and formulas that refer to a "successor" operation are arithmetical. In practice, most formulas fall into one of these two groups; this informal observation has a formal counterpart, which is the central idea in the following proof that all valid formulas are arithmetical in T_2.

Theorem 6.2.

If $\models u$ *then* u *is arithmetical in* $\vdash_{T_2}$.

Proof:
We prove the theorem for formulas with no function symbols. A proposition in the appendix shows that this restriction does not entail any loss of generality.

We suppose that $\models u$ but u is not arithmetical in $\vdash_{T_2}$ and derive a contradiction. Since u is not arithmetical in $\vdash_{T_2}$, for all formulas c we have that $\vdash_{T_2} C(c) \supset u$ but $\not\vdash_{T_2} u$. The completeness of T_2 for models of T_2 yields that $\neg u$ holds in some model $\mathcal{M}$ of T_2. Now we argue that $\mathcal{M}$ must be rather simple. Later on we derive that $\neg u$ holds in some standard model $\mathcal{M}_s$, thus contradicting the assumption $\models u$.

Given a n-ary predicate symbol p we expand $\mathcal{M}$ with a relation for the rigid $2n$-ary symbol $<$, defined by

$$\vec{x} < \vec{y} \equiv [(\neg p(\vec{x}) \wedge p(\vec{y})) \ \mathcal{P} \ (p(\vec{x}) \wedge \neg p(\vec{y}))].$$

Intuitively, $\vec{x}$ is less than $\vec{y}$ if, the first time that p distinguishes one from the other, p is false for $\vec{x}$ and true for $\vec{y}$. In other words, $\vec{x}$ is less than $\vec{y}$ if the sequence of values of p at $\vec{x}$ is lexicographically less than the sequence of values of p at $\vec{y}$.

We also expand $\mathcal{M}$ with relations for the associated equivalence relation $\simeq$ and the associated order $\leq$:

$$\vec{x} \simeq \vec{y} \equiv \neg(\vec{x} < \vec{y} \vee \vec{y} < \vec{x}),$$

$$\vec{x} \leq \vec{y} \equiv (\vec{x} \simeq \vec{y} \vee \vec{x} < \vec{y}).$$

Suppose that the relation $<$ gives rise to an infinite chain of tuples of elements in the domain. Without loss of generality, we may assume that the chain is infinite "forward" (otherwise, we consider $>$ instead). Now we define a successor function after the (arbitrary) starting point $\vec{z}$:

$$s(\vec{x}, \vec{y}) \equiv [\vec{z} \leq \vec{x} \wedge \vec{x} < \vec{y} \wedge \forall \vec{x}_1.(\vec{x}_1 \leq \vec{x} \vee \vec{y} \leq \vec{x}_1)].$$

A clock can be constructed:

$$\big(c(\vec{x}) \equiv (\vec{x} \simeq \vec{z})\big) \wedge \Box((\bigcirc c(\vec{x})) \equiv \exists \vec{y}.(c(\vec{y}) \wedge s(\vec{y}, \vec{x}))\big).$$

The expansion of $\mathcal{M}$ we obtain satisfies $C(c)$, still satisfies $\neg u$, and still is a model of T_2. On the other hand, $C(c) \supset u$ holds in all models of T_2, since $\vdash_{T_2} C(c) \supset u$. Thus, we derive a contradiction. Hence, $<$ cannot give rise to infinite chains.

This implies that for each predicate symbol p in u there are only finitely many $\simeq$-equivalence classes. In other words, for each p there are only finitely many possible patterns for $p(\vec{x})$ as $\vec{x}$ varies. This allows us to "split" p into a rigid component and a flexible propositional component.

The model $\mathcal{M}$ can be expanded with time-independent relations for the rigid symbols $r_1, \ldots, r_k$; we define $r_i(\vec{x})$ to hold if $\vec{x}$ is in the equivalence class i. Also, we can introduce

relations for the flexible proposition symbols $q_1, \ldots, q_k$, to represent the possible patterns of $p(\vec{x})$. More precisely, q_i is true in a certain world if p holds for the $\vec{x}$'s in the equivalence class i in that world. After all these expansions to $\mathcal{M}$, we have a model $\mathcal{M}'$ of $\neg u$.

In $\mathcal{M}'$,

$$\Box \forall \vec{x}.[p(\vec{x}) \equiv \bigvee_{i \leq k}(r_i(\vec{x}) \wedge q_i)]. \qquad (*)$$

For each predicate symbol p there is a different number k and a different set of defined symbols $r_1, \ldots, r_k, q_1, \ldots, q_k$ for which a similar equivalence holds.

The original formula $\neg u$ can be rewritten using the equivalences $(*)$: each atomic formula is replaced with the corresponding disjunction. The formula obtained, v, is equivalent to $\neg u$ in all models where the equivalences hold.

An inductive argument on the structure of v shows that v is equivalent to a Boolean combination of propositional temporal formulas and formulas where all symbols are rigid. Typical transformations are to rewrite $\Box(u_1 \vee u_2)$ to $u_1 \vee \Box\, u_2$, if u_1 is rigid, and to rewrite $\forall x.(u_1 \vee u_2)$ to $(\forall x.u_1) \vee u_2$, if u_2 is propositional. Therefore, v is equivalent to a formula of the form

$$(u_1^1 \wedge u_2^1) \vee \ \ldots \ \vee (u_1^n \wedge u_2^n),$$

with u_1^i rigid and u_2^i propositional for all i.

Since v holds in $\mathcal{M}'$, for some i the formulas u_1^i and u_2^i must hold in $\mathcal{M}'$. By the standard completeness theorem for PTL ([GPSS]) u_2^i must also have a standard model. We may take this model with the same domain and the same interpretation of rigid symbols as $\mathcal{M}'$ —since these choices do not affect the truth-value of the formulas u_2^i and the model remains standard. Finally, the model can be expanded with relations for the flexible predicate symbols defined with the equivalences $(*)$. The model we obtain is $\mathcal{M}_s$. Since u_1^i and u_2^i hold in $\mathcal{M}_s$, v holds as well. Since the equivalences hold, $\mathcal{M}_s$ satisfies $\neg u$. Thus, we have constructed a standard model for $\neg u$ and contradicted the hypothesis that $\models u$. ∎

Remark: The proof of theorem 6.2 requires that the logic include the operator $\mathcal{P}$, even if the formula u under consideration includes only $\bigcirc$, $\Box$, and $\Diamond$. Fortunately, we can refine the proof to guarantee that the theorem holds even for a logic without $\mathcal{U}$ and $\mathcal{P}$. The only step we reformulate is the definition of the rigid predicate symbol $<$.

First we define the flexible predicate symbol $p_1(\vec{x}, \vec{y})$ so that $p_1(\vec{x}, \vec{y})$ holds if and only if $p(\vec{x}) \wedge \neg p(\vec{y})$ has already held, that is,

$$p_1(\vec{x}, \vec{y}) \equiv (p(\vec{x}) \wedge \neg p(\vec{y})) \ \wedge \ \Box[(\bigcirc p_1(\vec{x}, \vec{y})) \equiv (p_1(\vec{x}, \vec{y}) \ \vee \ \bigcirc(p(\vec{x}) \wedge \neg p(\vec{y})))].$$

Similarly, we define $p_2(\vec{x}, \vec{y})$ so that $p_1(\vec{x}, \vec{y})$ holds if and only if $\neg p(\vec{x}) \wedge p(\vec{y})$ has already held. Then $<$ is defined by

$$\vec{x} < \vec{y} \equiv \Diamond[p_2(\vec{x}, \vec{y}) \wedge \neg p_1(\vec{x}, \vec{y})]. \quad ∎$$

7. Soundness and completeness

The main results of this section are the soundness and completeness of $\vdash_{T_1}$ and $\vdash_{T_2}$ with respect to $\vdash_O$ and $\vdash_P$, respectively. The completeness theorem for $\vdash_{T_1}$ refers only to arithmetical formulas. The more important completeness theorem for $\vdash_{T_2}$ applies to all formulas.

The soundness theorem states that modal temporal reasoning can be transformed into classical reasoning with explicit time parameters; in fact, the transformation is based on a step by step simulation.

Theorem 7.1. *Soundness*

For every formula u, $\vdash_{T_1} u \Rightarrow \vdash_O P(u)$ *and* $\vdash_{T_2} u \Rightarrow \vdash_P P(u)$.

Proof:
The argument is similar for both systems. A modal proof of u consists of a proof within T_0 and some definition discharges within T_1 or T_2. We show how to simulate the first part (classically) and how to eliminate the second part.

First, we show how to construct a classical proof of $P(\Box v)$ from a modal proof of v in T_0, for an arbitrary v. The construction proceeds by induction on the structure of the proof of v. We consider $\Box v$ rather than v in order to handle the case where the last rule in the proof is the one that introduces $\Box$. The extra $\Box$ is easy to delete at the end of this construction, that is, if $P(\Box v)$ is provable then so is $P(v)$.

- For all axioms v of T_0, $P(\Box v)$ is provable:
  - Let v be an instance of a schema valid in PTL. A completeness result for PTL ([GPSS]) enables us to consider only the cases where v is one of a few simple axioms for PTL. All of the proofs are routine.
  - Let v be an instance of $u \equiv \bigcirc u$ for some rigid formula u. Then $P(\Box v) = \forall i \geq 0.[P^*(u,i) \equiv P^*(u,s(i))]$. Since u is rigid, the systems of arithmetic prove $P^*(u,m) \equiv P^*(u,m')$ for any m and m' (this can easily be checked with an induction on the structure of u). It immediately follows that they prove $P(\Box v)$ as well.
  - Let v be an equality axiom. Then $P(\Box v) = \forall i \geq 0.P^*(v,i)$ and $P^*(v,i)$ is one of

$$(x = x),$$
$$(x = y \supset y = x),$$
$$(x = y \wedge y = z \supset x = z),$$
$$(x = y \supset P^*(t,i)\theta = P^*(t,i)),$$
$$(x = y \supset P^*(w,i)\theta \equiv P^*(w,i)),$$

for some term t and some formula w, where $\theta = \{x \leftarrow y\}$ and y does not occur bound in w. In all cases, $P^*(v,i)$ is a classical equality axiom; therefore, the systems of arithmetic prove $P^*(v,i)$, and hence $P(\Box v)$.

- Let v be $\exists x.\neg w \equiv \neg\forall x.w$. Then

$$P(\Box v) = \forall i \geq 0.\big(\exists x.\neg P^*(w,i) \equiv \neg\forall x.P^*(w,i)\big).$$

Since the systems of arithmetic properly include the predicate calculus, they certainly prove $\exists x.\neg P^*(w,i) \equiv \neg\forall x.P^*(w,i)$ for any $P^*(w,i)$, and hence $P(\Box v)$.

- Let v be $(\forall x.w) \supset w\theta$ for some formula w and some substitution $\theta = \{x \leftarrow t\}$ that does not create any new bound occurrences of variables or any occurrences of flexible terms in the scope of modal operators in w. Either x does not occur in the scope of modal operators in w or t does not contain any flexible symbols. In either case,

$$P^*(w\theta,i) = P^*(w,i)\{x \leftarrow P^*(t,i)\}.$$

Then

$$P(\Box v) = \forall i \geq 0.\big[(\forall x.P^*(w,i)) \supset \big(P^*(w,i)\{x \leftarrow P^*(t,i)\}\big)\big].$$

The quantifiers in $P^*(w,i)$ are those in w and some additional quantifiers over numbers. The former could not bind any of the data variables in $P^*(t,i)$, since these variables occur in t and the rule application does not create any new bound occurrences of variables in the original modal proof. The latter could not bind any number variables in $P^*(t,i)$, since the substitution does not create any new occurrences of flexible terms in modal contexts. Therefore, $\{x \leftarrow P^*(t,i)\}$ does not create any new bound occurrences of variables in $P^*(w,i)$. Since the systems of arithmetic properly include the predicate calculus, they certainly prove

$$(\forall x.P^*(w,i)) \supset \big(P^*(w,i)\{x \leftarrow P^*(t,i)\}\big),$$

that is, $(\forall x.P^*(w,i)) \supset P^*(w\theta,i)$, and hence $P(\Box v)$.

- Let v be $(\forall x.\bigcirc u) \equiv (\bigcirc \forall x.u)$. Note that

$$P^*((\forall x.\bigcirc u),i) = \forall x\forall j.(s_p(i,j) \supset P^*(u,j)),$$
$$P^*((\bigcirc \forall x.u),i) = \forall j.(s_p(i,j) \supset \forall x.P^*(u,j)),$$

and that these two formulas are provably equivalent. The provability of $P(\Box v)$ follows.

- Let v be $[\forall x.(u\,\mathcal{U}\,u')] \equiv [(\forall x.u)\,\mathcal{U}\,u']$, with x is not free in u'. As in the previous case, $P^*(\forall x.(u\,\mathcal{U}\,u'), i)$ and $P^*((\forall x.u)\,\mathcal{U}\,u', i)$ are provably equivalent. The provability of $P(\Box\, v)$ follows.

- All rules in T_0 can be simulated (e.g., if we can infer w from v then we can infer $P(\Box\, w)$ from $P(\Box\, v)$):

  - Assume that $P(\Box\, w_0)$ and $P(\Box(w_0 \supset w_1))$ are provable to show that $P(\Box\, w_1)$ is provable as well. We have

    $$P(\Box\, w_0) = \forall i \geq 0.P^*(w_0, i),$$
    $$P(\Box(w_0 \supset w_1)) = \forall i \geq 0.\big(P^*(w_0, i) \supset P^*(w_1, i)\big).$$

    By classical reasoning it follows that

    $$\forall i \geq 0.\big[P^*(w_0, i) \wedge \big(P^*(w_0, i) \supset P^*(w_1, i)\big)\big],$$

    and then $\forall i \geq 0.P^*(w_1, i)$, that is, $P(\Box\, w_1)$.

  - Assume that $P(\Box\, w)$ is provable to show that $P(\Box\,\Box\, w)$ is provable as well. The formula $P(\Box\, w) \supset P(\Box\,\Box\, w)$, that is,

    $$\forall i \geq 0.P^*(w, i) \supset (\forall i \geq 0)(\forall j \geq i).P^*(w, j),$$

    is provable, since $\vdash_O (i \geq 0 \wedge j \geq i) \supset j \geq 0$. Hence, we can prove $P(\Box\,\Box\, w)$.

  - Assume that $P(\Box(w_0 \supset w_1))$ is provable and x is not free in w_0 to show that $P(\Box(w_0 \supset \forall x.w_1))$ is provable as well. We have

    $$P(\Box(w_0 \supset w_1)) = \forall i \geq 0.(P^*(w_0, i) \supset P^*(w_1, i)).$$

    Since $\vdash_O \forall i.(i \geq 0)$, it follows that $P^*(w_0, i) \supset P^*(w_1, i)$ is provable. The variable x is not free in $P^*(w_0, i)$, since it is not free in w_0. By the classical rule of introduction for $\forall$, we derive $P^*(w_0, i) \supset \forall x.P^*(w_1, i)$, that is, $P^*(w_0, i) \supset P^*(\forall x.w_1, i)$. Therefore, we can prove

    $$\forall i \geq 0.\big[P^*(w_0, i) \supset P^*(\forall x.w_1, i)\big],$$

    that is, $P(\Box(w_0 \supset \forall x.w_1))$.

We have transformed the part within T_0 of a proof of u into an analogous classical proof. In the modal proof of u we discharge definitions to obtain the final result u from some theorem of T_0. We show that definitions are superfluous in the classical proof of $P(u)$, that is, if d is a definition and $P(d \supset v)$ is provable then so is $P(v)$.

- If d is an explicit definition and $\vdash_O P(d \supset v)$ then $\vdash_O P(v)$:
  Assume that $\vdash_O P(d \supset v)$. We have

  $$P(d \supset v) = [P(d) \supset P(v)],$$

  and $P(d)$ is of the form

  $$\forall x_1 \ldots \forall x_k . p(x_1, \ldots, x_n) \equiv w[x_1, \ldots, x_n].$$

  We may replace every occurrence of $p(t_1, \ldots, t_n)$ with the corresponding instance $w[t_1, \ldots, t_n]$ in the proof of $\vdash_O P(d) \supset P(v)$ —possibly after some renaming of bound variables to avoid unwanted captures. Every step of the proof is still legal since $\vdash_O$ does not distinguish $p(t_1, \ldots, t_n)$ from $w[t_1, \ldots, t_n]$. We obtain a proof of

  $$\vdash_O (\forall x_1 \ldots \forall x_n . w \equiv w) \supset P(v),$$

  and, therefore, a proof of $\vdash_O P(v)$.

- If d is a recursive definition and $\vdash_P P(d \supset v)$ then $\vdash_P P(v)$:
  Assume that $\vdash_P P(d \supset v)$. We have

  $$P(d \supset v) = [P(d) \supset P(v)],$$

  and $P(d)$ is the primitive-recursive definition for a predicate symbol p. By classical coding techniques from Peano Arithmetic ([Kl], sections 48 and 49), p is also definable explicitly, say by d'. The definition d' is of the form

  $$\forall x_1 \ldots \forall x_k \forall i . p(x_1, \ldots, x_n, i) \equiv w[x_1, \ldots, x_n, i].$$

  Furthermore, $\vdash_P d' \supset P(d)$, and hence also $\vdash_P d' \supset P(v)$. As above, we may replace every occurrence of $p(t_1, \ldots, t_n, m)$ with the corresponding instance $w[t_1, \ldots, t_n, m]$ in the proof of $\vdash_P d' \supset P(v)$. Again, every step of the proof is still legal. We obtain a proof of

  $$\vdash_P (\forall x_1 \ldots \forall x_n \forall i . w \equiv w) \supset P(v),$$

  and, therefore, a proof of $\vdash_P P(v)$. ∎

The completeness theorem, converse to the soundness theorem, states that classical reasoning with explicit time parameters can be transformed into modal temporal reasoning. The transformation is more than a trivial simulation, though; in particular, we exploit the existence of clocks.

Theorem 7.2. *Completeness*

For every formula u, $\vdash_O P(u) \Rightarrow \vdash_{T_1} u$ *if* u *is arithmetical in* $\vdash_{T_1}$.
For every formula u, $\vdash_P P(u) \Rightarrow \vdash_{T_2} u$.

Proof:
Since the translation function P preserves standard validity and $\vdash_P$ is sound, $\vdash_P P(u)$ implies that $\models u$, and hence that u is arithmetical in $\vdash_{T_2}$. Therefore, it suffices to prove that for every arithmetical formula u (with clock c) $\vdash_O P(u) \Rightarrow \vdash_{T_1} u$ and $\vdash_P P(u) \Rightarrow \vdash_{T_2} u$. The main steps of the proof are:

1) we use the clock to define some predicates and functions on the domain (0, s, $\leq$, and $\simeq$ for T_1, and also $+$ and $\times$ for T_2);

2) these predicates and functions satisfy the usual properties of 0, s, $\leq$, $=$, $+$, $\times$, and this can be shown within T_1 and T_2;

3) the clock also helps translate u into a FTL formula $Q(u)$ syntactically similar to $P(u)$;

4) furthermore, $Q(u)$ can be shown equivalent to u within T_1, so it suffices to construct a proof of $Q(u)$;

5) the proof of $Q(u)$ is identical in structure to that of $P(u)$, except that the usual axioms about 0, s, etc., are treated as theorems.

We present the completeness proof step by step. We may assume that there are no function symbols—though we use some as abbreviations. A proposition in the appendix checks that this entails no loss of generality. Also, at some points we claim that certain formulas are provable within T_1 and leave the corresponding arguments for the appendix.

Step 1:

- First we define the rigid numberhood predicate n by $\forall\vec{x}.[n(\vec{x}) \equiv \Diamond\, c[\vec{x}]]$.
- We define the rigid zero predicate 0_p by $\forall\vec{x}.[0_p(\vec{x}) \equiv c[\vec{x}]]$.
- We define the rigid successor predicate s_p by

$$\forall\vec{x}\forall\vec{y}.[s_p(\vec{x},\vec{y}) \equiv \Diamond(c[\vec{x}] \wedge \bigcirc\, c[\vec{y}])].$$

- Similarly, we define $\vec{x} \leq \vec{y}$ as $\Diamond(c[\vec{x}] \wedge \Diamond\, c[\vec{y}])$.
- Also, we define $\vec{x} \simeq \vec{y}$ as $\Diamond(c[\vec{x}] \wedge c[\vec{y}])$. The relation $\simeq$ is intended as an approximation to $=$.
- In T_2, primitive-recursive definitions can be exploited to define the rigid predicate $+_p$. First we define the flexible predicate p_1 by

$$\forall\vec{x}\forall\vec{y}.[p_1(\vec{x},\vec{y}) \equiv (\vec{x} \simeq \vec{y}) \wedge \Box((\bigcirc\, p_1(\vec{x},\vec{y})) \equiv \exists\vec{z}.(p_1(\vec{x},\vec{z}) \wedge s_p(\vec{z},\vec{y})))].$$

Intuitively, $p_1(\vec{x},\vec{y})$ holds at time k if $\vec{x}+k=\vec{y}$. Therefore, we define $+_p$ by

$\forall\vec{x}\forall\vec{y}\forall\vec{z}.[+_p(\vec{x},\vec{y},\vec{z}) \equiv \Diamond(c[\vec{y}] \wedge p_1(\vec{x},\vec{z}))]$.

- Similarly, $\times_p$ can be defined in T_2. First we define the flexible predicate p_2 by

  $\forall\vec{x}\forall\vec{y}.[p_2(\vec{x},\vec{y}) \equiv 0_p(\vec{y}) \wedge \Box((\bigcirc p_2(\vec{x},\vec{y})) \equiv \exists\vec{z}.(p_2(\vec{x},\vec{z}) \wedge +_p(\vec{z},\vec{x},\vec{y})))]$.

  Intuitively, $p_2(\vec{x},\vec{y})$ holds at time k if $\vec{x}\times k=\vec{y}$. Therefore, we define $\times_p$ by

  $\forall\vec{x}\forall\vec{y}\forall\vec{z}.[\times_p(\vec{x},\vec{y},\vec{z}) \equiv \Diamond(c[\vec{y}] \wedge p_2(\vec{x},\vec{z}))]$.

Step 2:

The defined symbols can be shown to satisfy most of their usual axioms within T_1. One important qualification is that the full substitutivity-of-equals property does not hold for $\simeq$. However, the substitutivity properties we obtain are sufficient for our purposes. Similarly, enough (but not all) instances of the induction schema can be proved; rigid predicate definitions are essential in these proofs.

More precisely, let A represent the definitions for auxiliary predicates (A includes the definitions for n, 0_p, s_p, $\leq$, and $\simeq$; when addition and multiplication are involved, A also includes the definitions for p_1, $+_p$, p_2, and $\times_p$). Then

$\vdash_{T_1} (C(c) \wedge A) \supset \exists\vec{x}.n(\vec{x})$.

This guarantees that the predicate n can be treated as a sort. Any proof of u using n as a sort can be transformed into a proof where n is a predicate (this is justified in a proposition presented in the appendix). From now on, we denote variables of this sort by letters like i, and terms of this sort by letters like m.

We write $\exists_!i.u[i]$ as an abbreviation for

$$\exists i.u[i] \wedge \forall i\forall j.[u[i] \wedge u[j] \supset i \simeq j].$$

The predicates 0_p, s_p, $+_p$, and $\times_p$ can be thought of as functions on the sort defined by n:

$\vdash_{T_1} (C(c) \wedge A) \supset \exists_!i.0_p(i)$,

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i\exists_!j.s_p(i,j)$,

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i\forall j\exists_!k.+_p(i,j,k)$,

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i\forall j\exists_!k.\times_p(i,j,k)$.

Therefore, it is convenient to use the functional abbreviations 0, s, $+$, $\times$ for 0_p, s_p, $+_p$, $\times_p$, just as in L_O and L_P.

Also, $\simeq$ is an equivalence relation and enjoys some useful substitutivity properties; $\simeq$ acts like $=$ on the sort defined by n:

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i.(i \simeq i)$,

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j.(i \simeq j \supset j \simeq i)$,

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j \forall k.((i \simeq j \wedge j \simeq k) \supset i \simeq k)$,

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j.(i \simeq j \supset \Diamond(c[i] \wedge a) \equiv \Diamond(c[j] \wedge a))$ if a is atomic,

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j.(i \simeq j \supset a[i] \equiv a[j])$,
if a is an atomic formula with predicate symbol n, 0_p, s_p, $\leq$, $\simeq$, $+_p$, or $\times_p$.

We say that a formula is *c-formed* if it is built up from atomic formulas with relation symbol n, 0_p, s_p, $\leq$, $\simeq$, $+_p$, and $\times_p$, and from formulas of the form $\Diamond(c[m] \wedge a)$, where a is atomic. The substitutivity property we need follows from the last two facts by induction:

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j.(i \simeq j \supset u[i] \equiv u[j])$ if u is c-formed.

Other theorems guarantee that the defined symbols satisfy the appropriate axioms:

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i.(s(i) \not\simeq 0)$,

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j.(s(i) \simeq s(j) \supset i \simeq j)$,

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i.((i \leq 0) \equiv (i \simeq 0))$,

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j.\big(i \leq s(j) \equiv (i \simeq s(j) \vee i \leq j)\big)$,

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i.(i + 0 \simeq i)$,

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j.(i + s(j) \simeq s(i + j))$,

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i.(i \times 0 \simeq 0)$,

$\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j.(i \times s(j) \simeq i \times j + i)$,

$\vdash_{T_1} (C(c) \wedge A) \supset [u[0] \wedge (\forall i.u[i] \supset u[s(i)]) \supset (\forall i.u[i])]$ if u is c-formed.

Step 3:

We define the translation function Q by $Q(u) = Q^*(u, 0)$, where Q^* is an auxiliary translation function such that:

$$Q^*(p(t_1, \ldots, t_k), m) = \begin{cases} p(t_1, \ldots, t_k) \\ \qquad \text{if } p \text{ is a rigid predicate symbol} \\ \Diamond(c[m] \wedge p(t_1, \ldots, t_k)) \\ \qquad \text{if } p \text{ is a flexible predicate symbol} \end{cases}$$

$$Q^*(\bigcirc u, m) = Q^*(u, s(m))$$

$$Q^*(\Box u, m) = \forall i \geq m. Q^*(u, i) \qquad (i \text{ and } j \text{ are new variables})$$

$$Q^*(\Diamond u, m) = \exists i \geq m. Q^*(u, i)$$

$$Q^*(u \,\mathcal{U}\, v, m) = \forall i \geq m. [Q^*(u, i) \vee \exists j. (m \leq j \leq i \wedge Q^*(v, j))]$$

$$Q^*(u \,\mathcal{P}\, v, m) = \exists i \geq m. [Q^*(u, i) \wedge \forall j. (m \leq j \leq i \supset \neg Q^*(v, j))]$$

and Q^* renames bound variables and preserves connectives and quantifiers.

Step 4:

The function Q provably preserves meaning. First we prove:

Lemma 7.3.

$\vdash_{T_1} (C(c) \wedge A) \supset (Q^*(u, m) \equiv \Diamond(c[m] \wedge u))$ *for all* u *and* m.

Proof:

The lemma is proved by induction on the structure of u. It suffices to show that

$\vdash_{T_1} (C(c) \wedge A) \supset Q^*(p(t_1, \ldots, t_k), m) \equiv \Diamond(c[m] \wedge p(t_1, \ldots, t_k))$,

$\vdash_{T_1} (C(c) \wedge A) \supset Q^*(\neg u, m) \equiv \Diamond(c[m] \wedge \neg u)$,

$\vdash_{T_1} (C(c) \wedge A) \supset Q^*(u_1 \wedge u_2, m) \equiv \Diamond(c[m] \wedge (u_1 \wedge u_2))$,

$\vdash_{T_1} (C(c) \wedge A) \supset Q^*(u_1 \vee u_2, m) \equiv \Diamond(c[m] \wedge (u_1 \vee u_2))$,

$\vdash_{T_1} (C(c) \wedge A) \supset Q^*(\forall x.u, m) \equiv \Diamond(c[m] \wedge \forall x.u)$,

$\vdash_{T_1} (C(c) \wedge A) \supset Q^*(\exists x.u, m) \equiv \Diamond(c[m] \wedge \exists x.u)$,

$\vdash_{T_1} (C(c) \wedge A) \supset Q^*(\bigcirc u, m) \equiv \Diamond(c[m] \wedge \bigcirc u)$,

$\vdash_{T_1} (C(c) \wedge A) \supset Q^*(\Box u, m) \equiv \Diamond(c[m] \wedge \Box u)$,

$\vdash_{T_1} (C(c) \wedge A) \supset Q^*(\Diamond u, m) \equiv \Diamond(c[m] \wedge \Diamond u)$,

$\vdash_{T_1} (C(c) \wedge A) \supset Q^*(u \,\mathcal{U}\, v, m) \equiv \Diamond(c[m] \wedge u \,\mathcal{U}\, v)$,

$\vdash_{T_1} (C(c) \wedge A) \supset Q^*(u \,\mathcal{P}\, v, m) \equiv \Diamond(c[m] \wedge u \,\mathcal{P}\, v)$,

with the induction hypothesis that the lemma holds for all proper subformulas of the formulas under consideration.

By the induction hypothesis, it suffices to show that

$\vdash_{T_1}(C(c) \wedge A) \supset p(t_1,\ldots,t_k) \equiv \Diamond(c[m] \wedge p(t_1,\ldots,t_k))$ if p is rigid,

$\vdash_{T_1}(C(c) \wedge A) \supset \Diamond(c[m] \wedge p(t_1,\ldots,t_k)) \equiv \Diamond(c[m] \wedge p(t_1,\ldots,t_k))$ if p is flexible,

$\vdash_{T_1}(C(c) \wedge A) \supset \neg\Diamond(c[m] \wedge u) \equiv \Diamond(c[m] \wedge \neg u)$,

$\vdash_{T_1}(C(c) \wedge A) \supset (\Diamond(c[m] \wedge u_1) \wedge \Diamond(c[m] \wedge u_2)) \equiv \Diamond(c[m] \wedge (u_1 \wedge u_2))$,

$\vdash_{T_1}(C(c) \wedge A) \supset (\Diamond(c[m] \wedge u_1) \vee \Diamond(c[m] \wedge u_2)) \equiv \Diamond(c[m] \wedge (u_1 \vee u_2))$,

$\vdash_{T_1}(C(c) \wedge A) \supset \forall x'.\, \Diamond(c[m] \wedge u[x']) \equiv \Diamond(c[m] \wedge \forall x.u[x])$ if x' is a new variable (in particular, x' does not occur in $c[m]$),

$\vdash_{T_1}(C(c) \wedge A) \supset \exists x'.\, \Diamond(c[m] \wedge u[x']) \equiv \Diamond(c[m] \wedge \exists x.u[x])$ if x' is a new variable (in particular, x' does not occur in $c[m]$),

$\vdash_{T_1}(C(c) \wedge A) \supset \Diamond(c[s(m)] \wedge u) \equiv \Diamond(c[m] \wedge \bigcirc u)$,

$\vdash_{T_1}(C(c) \wedge A) \supset \forall i \geq m.\, \Diamond(c[i] \wedge u) \equiv \Diamond(c[m] \wedge \Box u)$,

$\vdash_{T_1}(C(c) \wedge A) \supset \exists i \geq m.\, \Diamond(c[i] \wedge u) \equiv \Diamond(c[m] \wedge \Diamond u)$,

$$\vdash_{T_1}(C(c) \wedge A) \supset \left[\begin{array}{l} \forall i \geq m.\,[\Diamond(c[i] \wedge u) \vee \exists j.(m \leq j \leq i \wedge \Diamond(c[j] \wedge v))] \\ \quad \equiv \\ \Diamond(c[m] \wedge u\,\mathcal{U}\,v) \end{array}\right],$$

$$\vdash_{T_1}(C(c) \wedge A) \supset \left[\begin{array}{l} \exists i \geq m.\,[\Diamond(c[i] \wedge u) \wedge \forall j.(m \leq j \leq i \supset \neg\Diamond(c[j] \wedge v))] \\ \quad \equiv \\ \Diamond(c[m] \wedge u\,\mathcal{P}\,v) \end{array}\right].$$

The second formula is trivially provable. Duality considerations enable us to omit the cases for $\vee$, $\exists$, $\Diamond$, and $\mathcal{P}$. Also, the case for $\Box$ is subsumed by the case for $\mathcal{U}$ and can be omitted. The remaining cases are treated in the appendix. ▮

Theorem 7.4.

$\vdash_{T_1}(C(c) \wedge A) \supset (u \equiv Q(u))$ *for all* u.

Proof:
In the appendix we check that

$$\vdash_{T_1}(C(c) \wedge A) \supset (u \equiv \Diamond(c[0] \wedge u)) \text{ for all } u.$$

The lemma yields

$$\vdash_{T_1} (C(c) \wedge A) \supset (u \equiv Q^*(u,0)) \text{ for all } u.$$

Now it suffices to point out that $Q^*(u,0) = Q(u)$. ▮

Step 5:

The formula u is provable if $(C(c) \wedge A) \supset u$ is provable (in the appropriate system). By step 4, it suffices to show that $(C(c) \wedge A) \supset Q(u)$ is provable.

The formulas $Q(u)$ and $P(u)$ have identical syntactic structures, except that formulas of the form $\Diamond(c[m] \wedge p(t_1, \ldots, t_k))$ occur in $Q(u)$ where atoms of the form $p(t_1, \ldots, t_k, m)$ occur in $P(u)$. This difference is insignificant enough that the proof for $P(u)$ can be applied to $(C(c) \wedge A) \supset Q(u)$, with four minor modifications:

- The assumption $C(c) \wedge A$ is carried along.
- The equality symbol for numbers, $=$, is replaced with $\simeq$.
- If p is an uninterpreted predicate symbol then the atom $p(t_1, \ldots, t_k, m)$ is replaced with $\Diamond(c[m] \wedge p(t_1, \ldots, t_k))$.
- The axioms about numbers are no longer treated as axioms, but they are proved from $C(c) \wedge A$ (as in step 2). Note that since all formulas in the proof of $Q(u)$ are c-formed, the substitutivity and induction properties we obtain suffice. ▮

This concludes the proof of the completeness theorem. As a corollary, we can show that T_0 is strictly less powerful than T_1 and T_1 is strictly less powerful than T_2. Of course, all systems are incomplete in the standard sense.

Corollary 7.5.

$$\vdash_{T_0} \subset \vdash_{T_1} \subset \vdash_{T_2} \subset \models.$$

Proof:
The inclusion of all the proof concepts in $\models$ is a consequence of their soundness. Since they are all effective, they are incomplete; hence their inclusion in $\models$ is proper.

It is trivial that $\vdash_{T_0} \subseteq \vdash_{T_1} \subseteq \vdash_{T_2}$, since T_0 is a subsystem of T_1 and T_1 is a subsystem of T_2. Also, there are formulas u_1 and u_2 to separate $\vdash_{T_0}$ from $\vdash_{T_1}$ and $\vdash_{T_1}$ from $\vdash_{T_2}$, respectively:

For u_1, take

$$\begin{aligned}&[(\forall x.\, \Diamond\, a = x) \wedge (\forall x.\, \Box(a = x \supset \bigcirc \Box\, a \neq x))\\ &\wedge p(a) \wedge (\forall x \forall y.(p(x) \wedge \Diamond(a = x \wedge \bigcirc\, a = y)) \supset p(y))]\\ &\supset (\forall x.p(x)).\end{aligned}$$

Note that u_1 is very similar to u_0 exhibited for the nonstandard incompleteness theorem 4.5 (we introduce a slight difference to make u_1 obviously arithmetical). In fact, the arguments in the proof of theorem 4.5. show that $\not\vdash_{T_0} u_1$ and $\vdash_O P(u_1)$, with no modification. Also, the formula u_1 is arithmetical in T_1, with clock $(a = x)$. By the completeness theorem, it follows that $\not\vdash_{T_0} u_1$ and $\vdash_{T_1} u_1$.

As for u_2, Biró and Sain ([BS]) have produced a formula v of L_O such that $\not\vdash_O v$ and $\vdash_P v$. It is an immediate observation that v is the translation of a FTL formula, say u_2. By our soundness and completeness theorems, it follows that $\not\vdash_{T_1} u_2$ and $\vdash_{T_2} u_2$. ▮

Remark: Consider the restrictions of T_1 and T_2 that do not operate on formulas with $\mathcal{U}$ and $\mathcal{P}$. The completeness theorem and its corollary hold for these restricted systems. In fact, the proofs are special cases of the general proofs. Note, in particular, that we have pointed out that all valid formulas are arithmetical in $\vdash_{T_2}$ even when the logic does not include $\mathcal{U}$ and $\mathcal{P}$. ▮

8. Related work

The intractability of FTL has been widely accepted for some time. However, no intractability proof has been published to the best of our knowledge.

In dynamic logic, the intractability theorems initially led to an interest in completeness results for arithmetical universes ([Ha1]). We suspect these results do not carry over directly to temporal logic. On the other hand, literature on nonstandard logics of programs (e.g., [N], [BS], [Sa1], [Sa2]) discusses notions of completeness similar to those we study.

Previous works on nonstandard temporal logics differ from ours in three major respects. First, the logics under consideration often include the modal operator *First*, but not $\mathcal{U}$ and $\mathcal{P}$, and the only flexible symbols are constant symbols. Second, the works focus on weak FTL proof systems, similar to T_0 and T_1. Third, the main soundness and completeness theorems given are for special classes of sentences, such as partial and total correctness assertions for deterministic sequential programs. For instance, these theorems do not directly apply to reasoning about concurrent systems.

There have been results analogous to ours for other modal logics. In particular, Solovay ([So]) has provided an interpretation of the propositional logic of provability in Peano Arithmetic. Although our main positive result seems close in style and form to Solovay's, the constructions used in the proofs have little in common. Another fundamental difference is that the logic Solovay considers is axiomatizable, while temporal logic is highly intractable.

9. Open questions

The system we are most interested in is T_2, because it corresponds well both to Peano Arithmetic and to informal proof methods, and because it is the most powerful one of those we have considered. Still, there are some intriguing open questions on T_0 and T_1:

- We have shown that $\vdash_{T_0}$ is incomplete with respect to $\vdash_O$. Is there any simple characterization of T_0?
- We would like to know whether T_1 is actually complete for all formulas (and not just for arithmetical formulas). Consider augmenting T_1 with a rule to use a clock, that is, to derive u from $C(c) \supset u$ provided that the flexible predicate symbol c does not occur in u. As long as the domain of discourse is infinite, the rule is harmless. With this rule, all formulas become arithmetical, and hence $\vdash_{T_1}$ becomes complete with respect to $\vdash_O$. Thus, we would like to know whether a clock adds power to T_1.

10. Appendix

In the following subsections 10.1 and 10.2 we prove two general propositions to show how to replace function symbols with predicate symbols and predicates with sorts. These are simple extensions of propositions well known in classical logic ([Kl], [Gal]).

In subsection 10.3 we argue that certain formulas can be proved within T_1.

1. Eliminating function symbols

Given a formula u and a set F of uninterpreted function symbols in a given language, we define the "unnesting" of u, u^*, to be the formula obtained by repeatedly replacing occurrences of

$$p(\ldots, t[f(t_1, \ldots, t_n)], \ldots)$$

with

$$\exists x.[f(t_1, \ldots, t_n) = x \ \wedge \ p(\ldots, t[x], \ldots)]$$

where $f \in F$ and x is a new variable, until all f's in F occur only in atomic formulas of the form $f(t_1, \ldots, t_n) = x$. For the sake of uniqueness, we may choose a standard order for this rewriting.

Now u^* can be transformed into a formula u^+ with no function symbols in F: for each $f \in F$, we replace $f(t_1, \ldots, t_n) = x$ with $q_f(t_1, \ldots, t_n, x)$. Let F_u be the set of elements of F that occur in u. The formula

$$\bigwedge_{f \in F_u} [\Box \forall x_1 \ldots \forall x_n \exists! y. q_f(x_1, \ldots, x_n, y)]$$

expresses that all the predicate symbols introduced represent functions. (For rigid f's, the $\Box$ may be omitted.) Thus,

$$u' : \bigwedge_{f \in F_u} [\Box \forall x_1 \ldots \forall x_n \exists! y. q_f(x_1, \ldots, x_n, y)] \supset u^+$$

is equivalent to the original u.

When the language is sorted, u' is defined analogously and the new symbols introduced are taken in the appropriate sorts.

Proposition 10.1.

Let $\vdash$ be one of $\vdash_O$, $\vdash_P$, $\vdash_{T_0}$, $\vdash_{T_1}$, and $\vdash_{T_2}$. For every formula u, $\vdash u \Leftrightarrow \vdash u'$.

Proof:
To prove that $\vdash u \Rightarrow \vdash u'$, we assume that u has a proof and construct a proof for u'. The construction proceeds by induction on the structure of a proof for u. More precisely, we check that if v is an axiom then v' is provable and that if a rule derives v from $v_1, \ldots, v_k$ then v' can be obtained from $v_1', \ldots, v_k'$. The usual classical arguments are omitted. We present only the arguments for the temporal axioms and rules.

- If v is an instance of a valid PTL schema, of $u \equiv \bigcirc u$ (with u rigid), of $(\forall x. \bigcirc u) \equiv (\bigcirc \forall x.u)$, or of $[\forall x.(u \,\mathcal{U}\, v)] \equiv [(\forall x.u) \,\mathcal{U}\, v]$, then v^+ is an instance of the same schema, and, therefore, is an axiom. The axiom v^+ immediately yields v'.

- Suppose v is $\Box w$ and is deduced from w. Clearly, $\Box w'$ can be deduced from w'. By propositional temporal reasoning, $\vdash (\Box w') \equiv (\Box w)'$, so $(\Box w)'$ can be deduced from w'.

- Suppose v is deduced from $d \supset v$, for a definition d. Since d^+ is also a definition, v' can be deduced from $d^+ \supset v'$. By propositional reasoning, $\vdash (d \supset v)' \equiv (d^+ \supset v')$. Therefore, v' can be deduced from $(d \supset v)'$.

To prove the other direction, $\vdash u' \Rightarrow \vdash u$, we assume that $\vdash u'$ and show that $\vdash u$. Let v be u' with $f(t_1, \ldots, t_n) = t_{n+1}$ substituted for $q_f(t_1, \ldots, t_{n+1})$, for all $f \in F$. Clearly, the same proof succeeds for v and u', since they have the same structure; thus, $\vdash v$. In all systems under consideration functions are provably functional. In particular,

$$\vdash \bigwedge_{f \in F_u} [\Box \forall x_1 \ldots \forall x_n \exists! y. f(x_1, \ldots, x_n) = y].$$

Since $v = \bigwedge_{f \in F_u} [\Box \forall x_1 \ldots \forall x_n \exists! y. f(x_1, \ldots, x_n) = y] \supset u^*$, it follows that $\vdash u^*$.

Now, it suffices to show that $\vdash u \equiv u^*$. It suffices to point out that each step of the "unnesting" is provably correct, that is,

$$\vdash p(\ldots, t[f(t_1, \ldots, t_n)], \ldots) \equiv \exists x.[f(t_1, \ldots, t_n) = x \wedge p(\ldots, t[x], \ldots)]. \quad \blacksquare$$

This proposition guarantees that some proofs need to consider only formulas of restricted forms, where some or all function symbols are forbidden:

Corollary 10.2.

Assume that T_0 is complete for models of T_0 for all sets of formulas with no flexible function symbols, that is, if no function symbol occurs in the formulas in Σ and if Σ is T_0-consistent then Σ holds in some model of T_0. Then T_0 is complete for models of T_0.

Proof:
Consider a T_0-consistent set of formulas Σ where flexible function symbols may occur. We eliminate all flexible function symbols as in the proposition. More precisely, we define:

$$F = \{f | f \textit{ is a flexible function symbol occurring in } \Sigma\},$$
$$\Sigma' = \{u^+ | u \in \Sigma\} \cup \{\Box \forall x_1 \ldots \forall x_n \exists! y. q_f(x_1, \ldots, x_n, y) | f \in F\}.$$

By the proposition, Σ' is T_0-consistent. The assumption yields that Σ' has models. In one of these models, the predicate symbols that replace the flexible function symbols are interpreted as functions. Hence, a model for Σ can be read off immediately. $\blacksquare$

Corollary 10.3.

Assume that every valid formula with no function symbols is arithmetical in $\vdash_{T_2}$, that is, if no function symbol occurs in u and $\models u$ then u is arithmetical in $\vdash_{T_2}$. Then every valid formula is arithmetical in $\vdash_{T_2}$.

Proof:
Given a valid formula u, u' is also valid. Therefore, u' is arithmetical in $\vdash_{T_2}$. Let c_0 be a clock for u'. If $\vdash_{T_2} C(c_0) \supset u'$ then $\vdash_{T_2} u'$. We obtain a formula c_1 from c_0 by replacing all occurrences of $q_f(t_1, \ldots, t_{n+1})$ with $f(t_1, \ldots, t_n) = t_{n+1}$. The formula c_1 is our candidate clock for u. Assume that $\vdash_{T_2} C(c_1) \supset u$ to show that $\vdash_{T_2} u$. By the proposition, $\vdash_{T_2} [C(c_1) \supset u]'$. Also, $\vdash_{T_2} [C(c_1) \supset u]' \equiv [C(c_0) \supset u']$ by propositional reasoning. Therefore, $\vdash_{T_2} C(c_0) \supset u'$, and hence, since c_0 is a clock for u', $\vdash_{T_2} u'$. By the proposition, $\vdash_{T_2} u$ follows. $\blacksquare$

Corollary 10.4.

Assume that $\vdash_O P(u) \Rightarrow \vdash_{T_1} u$ and $\vdash_P P(u) \Rightarrow \vdash_{T_2} u$ for every arithmetical formula u with no function symbols. Then $\vdash_O P(u) \Rightarrow \vdash_{T_1} u$ and $\vdash_P P(u) \Rightarrow \vdash_{T_2} u$ for every arithmetical formula u.

Proof:
Assume that $\vdash_O P(u) \Rightarrow \vdash_{T_1} u$ for all arithmetical u with no function symbols and that for some arithmetical v, $\vdash_O P(v)$. We show that $\vdash_{T_1} v$. Since $\vdash_O P(v)$, the proposition yields $\vdash_O (P(v))'$. Note that $\vdash_O (P(v))' \equiv P(v')$, and hence $\vdash_O P(v')$.

Moreover, if v is arithmetical then so is v' (in fact, if c is a clock for v then c^+ is a clock for v'). Since v' contains no function symbols and is arithmetical, $\vdash_{T_1} v'$ follows from our assumption. By the proposition, $\vdash_{T_1} v$.

A similar argument handles $\vdash_P$ and $\vdash_{T_2}$. ∎

2. From predicates to sorts

We show that treating a rigid predicate as a sort does not improve provability. This very minor proposition on sorts can be extended in a number of ways, but the current form suffices for our purposes.

Suppose that $\forall i.u[i]$ and $\exists i.u[i]$ are used as abbreviations for $\forall \vec{x}.(p(\vec{x}) \supset u[\vec{x}])$ and $\exists \vec{x}.(p(\vec{x}) \wedge u[\vec{x}])$, respectively, and that the usual rules for quantifiers are applied to formulas with these special sorted variables. Thus, any provability concept $\vdash$ is extended to a new provability concept $\vdash_s$ for formulas with this kind of sort abbreviations. As long as p provably corresponds to a non-empty relation, $\vdash_s$ is no more powerful than the original $\vdash$.

Proposition 10.5.

Let p be a rigid predicate symbol and $\vdash$ be one of $\vdash_{T_1}$ and $\vdash_{T_2}$. Suppose that $u[i_1, \ldots, i_n]$ has a proof within $\vdash_s$, that is, $\vdash_s u[i_1, \ldots, i_n]$. In this proof, $i_1, \ldots, i_n$ are the only variables in the sort defined by p that may occur free. Let u' be the formula

$$[(\exists \vec{x}.p(\vec{x})) \wedge p(\vec{x}_1) \wedge \ldots \wedge p(\vec{x}_n)] \supset u[\vec{x}_1, \ldots, \vec{x}_n].$$

Then $\vdash u'$.

Proof:
We show how to construct a proof of $\vdash u'$ by induction on the structure of a proof of $\vdash_s u$. More precisely, we check that if v is an axiom in $\vdash_s$ then $\vdash v'$, and that if a rule derives v from $v_1, \ldots, v_k$ within $\vdash_s$ then v' can be obtained from $v_1', \ldots, v_k'$ within $\vdash$. Again, we spell out only the arguments for the temporal axioms and rules; the remaining arguments are similar and totally classical.

- If $v[i_1, \ldots, i_n]$ is an instance of a valid PTL schema, of $u \equiv \bigcirc u$ (with u rigid), of $(\forall x. \bigcirc u) \equiv (\bigcirc \forall x.u)$, or of $[\forall x.(u \,\mathcal{U}\, v)] \equiv [(\forall x.u) \,\mathcal{U}\, v]$, then $v[\vec{x}_1, \ldots, \vec{x}_n]$ is an

instance of the same schema, and, therefore, is an axiom. The provability of v' follows immediately.

- Suppose v is $\Box w$ and is deduced from w. Clearly, $\Box w'$ can be deduced from w'. By propositional temporal reasoning, $\vdash (\Box w') \equiv (\Box w)'$, so $(\Box w)'$ can be deduced from w'.

- Suppose $v[i_1, \ldots, i_n]$ is deduced from $d[i_1, \ldots, i_n] \supset v[i_1, \ldots, i_n]$, for a definition $d[i_1, \ldots, i_n]$. Since $d[\vec{x}_1, \ldots, \vec{x}_n]$ is also a definition, v' can be deduced from $d[\vec{x}_1, \ldots, \vec{x}_n] \supset v'$. By propositional reasoning, the formulas $(d[\vec{x}_1, \ldots, \vec{x}_n] \supset v')$ and $(d[i_1, \ldots, i_n] \supset v)'$ are equivalent. Therefore, v' follows from $(d \supset v)'$. ∎

3. Some useful theorems

In steps 2 and 4 of the completeness theorem (section 7) we claim that certain formulas are provable in T_1. We justify the claim in this subsection. Most of the arguments are routine. Since T_1 is complete with respect to models of T_1, we are able to give some (slightly) semantic proofs. The corresponding syntactic proofs are long but easy to reconstruct.

The formulas $C(c)$ and A are defined as in section 7. Throughout, we consider a model $\mathcal{M}$ of T_1 where $C(c)$ and A hold.

- $\vdash_{T_1} (C(c) \wedge A) \supset \exists \vec{x}.n(\vec{x})$

The model $\mathcal{M}$ satisfies $(\exists \vec{x}.c[\vec{x}])$, since it satisfies $C(c)$. Hence, it satisfies $(\exists \vec{x}.\ \Diamond c[\vec{x}])$. Since $n(\vec{x})$ is defined as $\Diamond c[\vec{x}]$, $\mathcal{M}$ also satisfies $(\exists \vec{x}.n(\vec{x}))$.

- $\vdash_{T_1} (C(c) \wedge A) \supset \Box(c[i] \supset u) \equiv \Diamond(c[i] \wedge u) \qquad (*)$

(This "duality proposition" is a useful tool in proving the remaining theorems.)

Assume that $\Diamond(c[i] \wedge u)$ holds in $\mathcal{M}$. Suppose that $\Box(c[i] \supset u)$ does not hold, that is, that $\Diamond(c[i] \wedge \neg u)$ holds. By propositional temporal reasoning we obtain $\Diamond(c[i] \wedge \bigcirc \Diamond c[i])$, in contradiction with $C(c)$. Therefore, $\Box(c[i] \supset u)$ must hold.

Assume that $\Box(c[i] \supset u)$ holds. By the definition of n, $\Diamond c[i]$ holds; propositional temporal reasoning yields $\Diamond(c[i] \wedge u)$.

- $\vdash_{T_1} (C(c) \wedge A) \supset \Box((\bigcirc c[i]) \supset u) \equiv \Diamond((\bigcirc c[i]) \wedge u) \qquad (**)$

The proof is similar to the previous one.

- $\vdash_{T_1} (C(c) \wedge A) \supset \exists_! i.0_p(i)$

The model $\mathcal{M}$ satisfies $\exists i.c[i]$, since it satisfies $C(c)$. It follows that $\exists i.0_p(i)$ holds, by the definition of 0_p. Furthermore, if both $0_p(i)$ and $0_p(j)$ hold, then $c[i]$ and $c[j]$ hold, so $\Diamond(c[i] \wedge c[j])$ holds as well. It follows that $i \simeq j$, by the definition of $\simeq$.

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i \exists! j.s_p(i,j)$

Consider an arbitrary i. By the definition of n, $c[i]$ must hold at some world w_1 such that $w_0 R_2 w_1$. Then there is some j with property c at some world w_2 such that $w_1 R_1 w_2$: since $C(c)$ must hold at w_1, $\bigcirc \exists \vec{y}.c[\vec{y}]$ must hold at w_1. By the definition of s_p, this means that i has a successor. To see that this successor is unique (as far as $\simeq$ can distinguish), assume that both j and j' are successors to i. Then both $\bigcirc c[j]$ and $\bigcirc c[j']$ hold at w_1; hence $c[j] \wedge c[j']$ holds at w_2. It follows that $\Diamond \bigcirc(c[j] \wedge c[j'])$ and hence $\Diamond(c[j] \wedge c[j'])$ hold at w_0. The definition of $\simeq$ yields that $j \simeq j'$.

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j \exists! k. +_p (i,j,k)$

We first show that p_1 defines a flexible function from numbers to numbers at all worlds w_1 such that $w_0 R_2 w_1$. By the induction principle, it suffices to show that p_1 is a function at w_0 and that if it is a function at some w_1 such that $w_0 R_2 w_1$ then it is a function at some w_2 such that $w_1 R_1 w_2$. At w_0, $p_1(i,j)$ holds if and only if $0_p(i)$ holds; since 0_p defines a constant, p_1 defines a function. Now assume that p_1 is a function at w_1 and consider w_2. At w_2, $p_1(i,j)$ holds if and only if for some j we have $s_p(j,k)$ and $p_1(i,k)$ holds at w_1. Since p_1 is a function at w_1 and s_p is a function, p_1 is a function at w_2.

To show that $+_p$ is a function, consider arbitrary i and j. By the definition of n, there must be some w_1 that satisfies $c[j]$ such that $w_0 R_2 w_1$. For some $\simeq$-unique k, w_1 satisfies $p_1(i,k)$, and hence $+_p(i,j,k)$. To finish, we need to check that if $c[j]$ holds at both w_1 and w_1', then the same k gives us $p_1(i,k)$ at w_1 and w_1'. Suppose that $p_1(i,k)$ holds at w_1; then $\Diamond(c[j] \wedge p_1(i,k))$ holds at w_0. By $(*)$, we obtain $\Box(c[j] \supset p_1(i,k))$. Therefore, $p_1(i,k)$ holds at w_1'. If $p_1(i,k')$ holds at w_1' for some other k', we have $k \simeq k'$ because p_1 defines a function.

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j \exists! k. \times_p (i,j,k)$

The argument is exactly analogous to the one for $+_p$ (using $+_p$ instead of s_p).

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i.(i \simeq i)$

Consider an arbitrary i. By the definition of n, $\Diamond(c[i] \wedge c[i])$, that is, $i \simeq i$, holds.

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j.(i \simeq j \supset j \simeq i)$

Consider arbitrary i and j such that $i \simeq j$, that is, $\Diamond(c[i] \wedge c[j])$. By commutativity, $\Diamond(c[j] \wedge c[i])$, that is, $j \simeq i$, holds.

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j \forall k.((i \simeq j \wedge j \simeq k) \supset i \simeq k)$

Consider arbitrary i, j, and k such that $i \simeq j$ and $j \simeq k$, that is, $\Diamond(c[i] \wedge c[j])$ and $\Diamond(c[j] \wedge c[k])$. By $(*)$, $\Box(c[j] \supset c[k])$ follows. By propositional temporal reasoning, we obtain $\Diamond(c[i] \wedge c[j] \wedge c[k])$, and then derive $\Diamond(c[i] \wedge c[k])$, that is, $i \simeq k$.

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j.(i \simeq j \supset \Diamond(c[i] \wedge a) \equiv \Diamond(c[j] \wedge a))$ if a is atomic

Consider arbitrary i and j such that $i \simeq j$, that is, $\Diamond(c[i] \wedge c[j])$, and assume that $\Diamond(c[i] \wedge a)$. By $(*)$, $\Box(c[i] \supset c[j])$ follows. By propositional temporal reasoning we obtain $\Diamond(c[j] \wedge a)$. The other direction of the equivalence is similar.

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j.(i \simeq j \supset a[i] \equiv a[j])$
  if a is atomic with relation symbol n, 0_p, s_p, $\leq$, $\simeq$, $+_p$, or $\times_p$.

We prove a more general proposition instead:

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j.(i \simeq j \supset \Box(a[i] \equiv a[j]))$
  if a is atomic with relation symbol n, 0_p, s_p, $\leq$, $\simeq$, p_1, $+_p$, p_2, or $\times_p$.

We first prove the property for a atomic with relation symbol n, 0_p, s_p, $\leq$, or $\simeq$. Consider arbitrary i and j such that $i \simeq j$, that is, $\Diamond(c[i] \wedge c[j])$. By $(*)$, $\Box(c[i] \equiv c[j])$ follows. Let $d[i]$ and $d[j]$ be the formulas obtained from $a[i]$ and $a[j]$ when the symbols n, 0_p, s_p, $\leq$, and $\simeq$ are replaced with their definitions. Note that $a[i]$ and $a[j]$ are provably equivalent to $d[i]$ and $d[j]$, respectively. Since i and j occur in $d[i]$ and $d[j]$ only as arguments to c, $\Box(c[i] \equiv c[j])$ yields $\Box(d[i] \equiv d[j])$. It follows that $a[i] \equiv a[j]$, and then, since all symbols involved are rigid, $\Box(a[i] \equiv a[j])$.

For p_1, the proof is inductive. It suffices to show that substitutivity holds at w_0 and that if it holds at some w_1 such that $w_0 R_2 w_1$ then it holds at some w_2 such that $w_1 R_1 w_2$. At w_0, substitutivity holds because p_1 is defined in terms of $\simeq$, for which we have already proved substitutivity. Now assume that substitutivity holds at w_1 to prove that it holds at some successor world w_2. The meaning of p_1 at w_2 is defined in terms of s_p and p_1 at w_1; we have proved substitutivity for s_p and assumed it for p_1 at w_1. Therefore, we have substitutivity for p_1 at w_2.

Now substitutivity for $+_p$ follows because $+_p$ is defined in terms of c and p_1. A similar argument enables us to prove substitutivity for p_2 from substitutivity for 0_p and $+_p$, and substitutivity for $\times_p$ from substitutivity for p_2.

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i.(s(i) \not\simeq 0)$

Consider an arbitrary i and a successor of i, j. By the definition of n, we obtain $\Diamond \bigcirc c[j]$. We have that $\bigcirc \Box \neg c[0]$ (from $C(c)$ and the definition of 0_p). Propositional temporal reasoning yields $\Diamond(c[j] \wedge \neg c[0])$; $(*)$ yields $\Box(c[j] \supset \neg c[0])$, that is, $j \not\simeq 0$.

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j.(s(i) \simeq s(j) \supset i \simeq j)$

Consider arbitrary i and j and their respective successors k and k', and assume that $k \simeq k'$. We obtain $\Diamond(c[i] \wedge \bigcirc c[k])$ and $\Diamond(c[j] \wedge \bigcirc c[k])$ by substitutivity and the definition of s_p. Propositional temporal reasoning yields

$$\Diamond(c[i] \wedge c[j]) \vee \Diamond \bigcirc(c[k] \wedge \bigcirc \Diamond c[k]).$$

The second disjunct does not hold since $C(c)$ holds. Hence, $\Diamond(c[i] \wedge c[j])$, that is, $i \simeq j$, holds.

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i.((i \leq 0) \equiv (i \simeq 0))$

Consider an arbitrary i and assume that $i \leq 0$, that is, $\Diamond(c[i] \wedge \Diamond c[0])$. Propositional temporal reasoning yields $c[i] \vee \bigcirc \Diamond c[0]$. Since we have $\bigcirc \Box \neg c[0]$, we can eliminate the first disjunct and derive $c[i]$. The definition of 0_p implies $c[0]$, and we obtain $\Diamond(c[i] \wedge c[0])$, that is, $i \simeq 0$.

Now assume that $i \simeq 0$, that is, $\Diamond(c[i] \wedge c[0])$. Propositional temporal reasoning yields $\Diamond(c[i] \wedge \Diamond c[0])$, that is, $i \leq 0$.

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j.\big(i \leq s(j) \equiv (i \simeq s(j) \vee i \leq j)\big)$

Consider arbitrary i and j such that $i \leq s(j)$, that is, for some k such that $\Diamond(c[j] \wedge \bigcirc c[k])$, $\Diamond(c[i] \wedge \Diamond c[k])$. Propositional temporal reasoning yields

$$\Diamond(c[i] \wedge c[k]) \vee \Diamond(c[i] \wedge \Diamond \bigcirc c[k]).$$

If the first disjunct holds, we have $i \simeq k$, that is, $i \simeq s(j)$. If the second disjunct holds, we have $\Box((\bigcirc c[k]) \supset c[j])$ (by $(**)$), and hence propositional temporal reasoning yields $\Diamond(c[i] \wedge \Diamond c[j])$, that is, $i \leq j$.

Now assume that $i \simeq s(j)$, that is, $\Diamond(c[i] \wedge c[s(j)])$. Propositional temporal reasoning yields $\Diamond(c[i] \wedge \Diamond c[s(j)])$, that is, $i \leq s(j)$. Finally, assume that $i \leq j$, that is, $\Diamond(c[i] \wedge \Diamond c[j])$. Consider a successor k of j. By the definition of s_p, we have $\Diamond(c[j] \wedge \bigcirc c[k])$, and, by $(*)$, $\Box(c[j] \supset \bigcirc c[k])$. Propositional temporal reasoning yields $\Diamond(c[i] \wedge \Diamond \bigcirc c[k])$, and then $\Diamond(c[i] \wedge \Diamond c[k])$, that is, $i \leq k$.

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i.(i + 0 \simeq i)$

Consider an arbitrary i. Suppose $k \simeq i + 0$, that is, $\Diamond(c[0] \wedge p_1(i,k))$. By $(*)$, $\Box(c[0] \supset p_1(i,k))$ follows, and then $p_1(i,k)$, by the definition of 0_p. The definition of p_1 immediately yields $i \simeq k$.

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j.(i + s(j) \simeq s(i + j))$

Consider arbitrary i and j. Suppose $k \simeq i + j$, that is, $\Diamond(c[j] \wedge p_1(i,k))$. The definitions of s_p and p_1 imply

$$\Diamond((\bigcirc c[s(j)]) \wedge (\bigcirc p_1(i, s(k)))).$$

By propositional temporal reasoning we obtain

$$\Diamond(c[s(j)] \wedge p_1(i, s(k))),$$

that is, $i + s(j) \simeq s(k)$.

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i.(i \times 0 \simeq 0)$

  The argument is exactly analogous to the one for $i + 0 \simeq i$.

- $\vdash_{T_1} (C(c) \wedge A) \supset \forall i \forall j.(i \times s(j) \simeq i \times j + i)$

  Consider arbitrary i and j. Suppose $k \simeq i \times j$, that is, $\Diamond(c[j] \wedge p_2(i,k))$. The definitions of s_p and p_2 imply

$$\Diamond((\bigcirc c[s(j)]) \wedge (\bigcirc p_2(i, k + i))).$$

By propositional temporal reasoning we obtain

$$\Diamond(c[s(j)] \wedge p_2(i, k + i)),$$

that is, $i \times s(j) \simeq k + i$.

- $\vdash_{T_1} (C(c) \wedge A) \supset [u[0] \wedge (\forall i.u[i] \supset u[s(i)]) \supset (\forall i.u[i])]$ if u is c-formed

  Given a c-formed formula $u[\vec{x}]$, assume $u[0]$ and $(\forall i.u[i] \supset u[s(i)])$. Define the rigid predicate symbol p to be u at the initial world, that is, $\forall \vec{x}.(p(\vec{x}) \equiv u[\vec{x}])$, and expand $\mathcal{M}$ with a relation for the defined rigid symbol p. Now we prove that $\Box \forall i.(c[i] \supset p(i))$. By the induction principle, it suffices to show that $\forall i.(c[i] \supset p(i))$ holds at w_0 and that if it holds at some w_1 such that $w_0 R_2 w_1$ then it holds at some w_2 such that $w_1 R_1 w_2$.

  Since $u[0]$ holds at w_0, $u[i]$ holds at w_0 for any i such that $i \simeq 0$ (by substitutivity). If $c[i]$ holds at w_0, then $i \simeq 0$ (by the definition of $\simeq$) and hence $u[i]$ holds at w_0. The definition of p immediately yields $p(i)$.

  Now assume that $\forall i.(c[i] \supset p(i))$ holds at w_1 to show that it holds at some successor world w_2. Consider an arbitrary i such that $c[i]$ holds at w_1, and hence $p(i)$ holds at w_1. Since p is rigid, $p(i)$ holds at w_0 as well. The definition of p guarantees that $u[i]$ holds at w_0, and hence that $u[s(i)]$ holds at w_0. Consider an arbitrary j such that $c[j]$ holds at w_2, and hence $j \simeq s(i)$ holds at w_0. Substitutivity yields $u[j]$. From the definition of p we obtain that $p(j)$ holds at w_0, and hence at w_2 as well.

The definition of n implies that $\forall i.\ \Diamond c[i]$. We obtain $\forall i.\ \Diamond p(i)$. Since p is a rigid symbol, this entails $\forall i.p(i)$, that is, $\forall i.u[i]$.

- $\vdash_{T_1}(C(c) \wedge A) \supset p(t_1,\ldots,t_k) \equiv \Diamond(c[m] \wedge p(t_1,\ldots,t_k))$ if p is rigid

Assume that $p(t_1,\ldots,t_k)$ holds. Since p is rigid, $\Box p(t_1,\ldots,t_k)$ holds as well. For any m, $\Diamond c[m]$ holds (by the definition of n). Propositional temporal reasoning yields $\Diamond(c[m] \wedge p(t_1,\ldots,t_k))$.

Assume that $\Diamond(c[m] \wedge p(t_1,\ldots,t_k))$ holds. Then $p(t_1,\ldots,t_k)$ holds at some world w_1 such that $w_0 R_2 w_1$. Since p is rigid, $p(t_1,\ldots,t_k)$ must also hold at w_0.

- $\vdash_{T_1}(C(c) \wedge A) \supset \neg \Diamond(c[m] \wedge u) \equiv \Diamond(c[m] \wedge \neg u)$

Propositional temporal reasoning yields this syntactic variant of $(*)$.

- $\vdash_{T_1}(C(c) \wedge A) \supset (\Diamond(c[m] \wedge u_1) \wedge \Diamond(c[m] \wedge u_2)) \equiv \Diamond(c[m] \wedge (u_1 \wedge u_2))$

Assume that both $\Diamond(c[m] \wedge u_1)$ and $\Diamond(c[m] \wedge u_2)$ hold. Then $\Box(c[m] \supset u_1)$ holds (by $(*)$); $\Diamond(c[m] \wedge (u_1 \wedge u_2))$ follows by propositional temporal reasoning.

Assume that $\Diamond(c[m] \wedge (u_1 \wedge u_2))$ holds. Then both $\Diamond(c[m] \wedge u_1)$ and $\Diamond(c[m] \wedge u_2)$ follow trivially, and so does their conjunction.

- $\vdash_{T_1}(C(c) \wedge A) \supset \forall x'.\ \Diamond(c[m] \wedge u[x']) \equiv \Diamond(c[m] \wedge \forall x.u[x])$
  (x' does not occur in $c[m]$)

Assume that $\forall x'.\ \Diamond(c[m] \wedge u[x'])$ holds. By $(*)$, $\forall x'.\ \Box(c[m] \supset u[x'])$ holds. This formula is equivalent to $\Box \forall x'.(c[m] \supset u[x'])$. Since the variable x' does not occur in $c[m]$, we can derive $\Box(c[m] \supset \forall x.u[x])$. By $(*)$, $\Diamond(c[m] \wedge \forall x.u[x])$ holds.

Assume that $\Diamond(c[m] \wedge \forall x.u[x])$ holds. By $(*)$, $\Box(c[m] \supset \forall x.u[x])$ holds. For a new variable x', we obtain $\Box \forall x'.(c[m] \supset u[x'])$. This formula is equivalent to $\forall x'.\ \Box(c[m] \supset u[x'])$. By $(*)$, $\forall x'.\ \Diamond(c[m] \wedge u[x'])$ holds.

- $\vdash_{T_1}(C(c) \wedge A) \supset \Diamond(c[s(m)] \wedge u) \equiv \Diamond(c[m] \wedge \bigcirc u)$

Assume that $\Diamond(c[s(m)] \wedge u)$ holds. By $(*)$, $\Box(c[s(m)] \supset u)$ holds. Together with the definition of s_p, this yields $\Diamond(c[m] \wedge \bigcirc(c[s(m)] \wedge u))$, and hence also $\Diamond(c[m] \wedge \bigcirc u)$.

Assume that $\Diamond(c[m] \wedge \bigcirc u)$ holds. Together with the definition of s_p, this yields $\Diamond((\bigcirc c[s(m)]) \wedge (\bigcirc u))$. The conclusion $\Diamond(c[s(m)] \wedge u)$ follows by propositional temporal reasoning.

- $\vdash_{T_1}(C(c) \wedge A) \supset \left[\begin{array}{l} \forall i \geq m.\left[\Diamond(c[i] \wedge u) \vee \exists j.(m \leq j \leq i \wedge \Diamond(c[j] \wedge v))\right] \\ \quad \equiv \\ \Diamond(c[m] \wedge u\, \mathcal{U}\, v) \end{array}\right]$

Assume that

$$\forall i \geq m.[\Diamond(c[i] \wedge u) \vee \exists j.(m \leq j \leq i \wedge \Diamond(c[j] \wedge v))]$$

holds. Suppose that $\neg \Diamond(c[m] \wedge u \,\mathcal{U}\, v)$ holds, to derive a contradiction. By $(*)$, we obtain $\Diamond(c[m] \wedge \neg(u \,\mathcal{U}\, v))$, and then $\Diamond(c[m] \wedge ((\neg u) \,\mathcal{P}\, v))$. Since $C(c)$ holds, for some k we have

$$\Diamond[c[m] \wedge ((c[k] \wedge \neg u) \,\mathcal{P}\, v)]. \qquad (\dagger)$$

It follows that $\Diamond(c[k] \wedge \neg u)$ and $k \geq m$. Hence, our assumption yields

$$\Diamond(c[k] \wedge u) \vee \exists j.(m \leq j \leq k \wedge \Diamond(c[j] \wedge v)).$$

Since we have $\Diamond(c[k] \wedge \neg u)$, $(*)$ enables us to eliminate the first disjunct. Thus, for some j such that $m \leq j \leq k$, $\Diamond(c[j] \wedge v)$ holds, and then $(\dagger)$ yields

$$\Diamond[c[m] \wedge ((c[k] \wedge \neg u) \,\mathcal{P}\, v) \wedge \Diamond(c[j] \wedge v)].$$

By propositional temporal reasoning, we obtain

$$[(c[j] \wedge v \wedge \Diamond(c[k] \wedge \neg u)) \,\mathcal{P}\, v] \vee \Diamond(c[k] \wedge \bigcirc \Diamond c[j]).$$

The first disjunct leads to $v \,\mathcal{P}\, v$, an unsatisfiable formula. The second disjunct is ruled out by $j \leq k$ together with $C(c)$. In both cases we have contradictions.

Now assume that $\Diamond(c[m] \wedge u \,\mathcal{U}\, v)$ holds. Suppose that

$$\neg\forall i \geq m.[\Diamond(c[i] \wedge u) \vee \exists j.(m \leq j \leq i \wedge \Diamond(c[j] \wedge v))]$$

holds, to derive a contradiction. Then for some $i \geq m$ we have $\Box(c[i] \supset \neg u)$, and hence $\Diamond(c[i] \wedge \neg u)$ (by $(*)$), and for all j between m and i we have $\Box(c[j] \supset \neg v)$. We derive

$$\Diamond[c[m] \wedge u \,\mathcal{U}\, v \wedge \Diamond(c[i]) \wedge ((\Diamond c[i]) \,\mathcal{U}\, c[i])]$$

from $i \geq m$ and $C(c)$, and then

$$\Diamond[c[m] \wedge u \,\mathcal{U}\, v \wedge \Diamond(c[i]) \wedge ((\exists j.c[j] \wedge \bigcirc \Box \neg c[m] \wedge \Diamond c[i]) \,\mathcal{U}\, c[i])]$$

from $C(c)$. The definition of $\leq$ enables us to conclude

$$\Diamond[u \,\mathcal{U}\, v \wedge \Diamond(c[i]) \wedge ((\exists j.m \leq j \leq i \wedge c[j]) \,\mathcal{U}\, c[i])].$$

Since $\Box(c[i] \supset \neg u)$ and $m \leq j \leq i \supset \Box(c[j] \supset \neg v)$ (and $\Box(c[i] \supset \neg v)$ in particular), propositional temporal reasoning yields

$$\Diamond[u \,\mathcal{U}\, v \wedge \Diamond(\neg u) \wedge ((\exists j.\neg v) \,\mathcal{U}\, (\neg u \wedge \neg v))].$$

This is equivalent to

$$\Diamond[u \,\mathcal{U}\, v \wedge \Diamond(\neg u) \wedge (\neg v \,\mathcal{U}\, (\neg u \wedge \neg v))],$$

an unsatisfiable formula.

- $\vdash_{T_1}(C(c) \wedge A) \supset (u \equiv \Diamond(c[0] \wedge u))$ for all u

  Assume that u holds. Since $c[0]$ holds, $c[0] \wedge u$ holds, and so does $\Diamond(c[0] \wedge u)$.

  Assume that $\Diamond(c[0] \wedge u)$ holds. By $(*)$, $\Box(c[0] \supset u)$ holds as well. Since $c[0]$ holds, u follows.

Acknowledgments

I am grateful to Gianluigi Bellin, Sol Feferman, Rob Goldblatt, Leo Harrington, John Lamping, Mark Manasse, and two anonymous referees, for comments and fruitful discussions on this work, and to Cynthia Hibbard, for editorial help.

References

[A1] M. Abadi, *Temporal-Logic Theorem Proving*, PhD Thesis, Computer Science Department, Stanford University, 1987.

[A2] M. Abadi, "The power of temporal proofs," Second Annual Symposium on Logic in Computer Science, 1987, pp. 123–130.

[AM] M. Abadi and Z. Manna, "A timely resolution," First Annual Symposium on Logic in Computer Science, 1986, pp. 176–186.

[BS] B. Biró and I. Sain, "Peano Arithmetic for the time scale of nonstandard logics of programs," unpublished manuscript, Mathematical Institute of the Hungarian Academy of Sciences, 1983.

[Bu1] J. Burgess, "Axioms for tense logic," *Notre Dame Journal of Formal Logic*, Vol. 23, No. 4, Oct. 1982, pp. 367–383.

[Bu2] J. Burgess, "Basic tense logic," in *Handbook of Philosophical Logic, Vol. II* (D. Gabbay and F. Guenthner, eds.), D. Reidel Publishing Co., Dordrecht, 1984, pp. 89–133.

[Gal] J. H. Gallier, *Logic for Computer Science: Foundations of Automatic Theorem Proving*, Harper & Row, New York, 1986.

[Gar] J. Garson, "Quantification in modal logic," in *Handbook of Philosophical Logic, Vol. II* (D. Gabbay and F. Guenthner, eds.), D. Reidel Publishing Co., Dordrecht, 1984, pp. 249–307.

[GPSS] D. Gabbay, A. Pnueli, S. Shelah, and J. Stavi, "The temporal analysis of fairness," Seventh ACM Symposium on Principles of Programming Languages, 1980, pp. 163–173.

[Go] K. Gödel, "On formally undecidable propositions of Principia Mathematica and related systems I," in *From Frege to Gödel* (J. Heijenoort, ed.), Harvard University Press, Cambridge, Massachusetts, 1967, pp. 525–581.

[Ha1] D. Harel, "Dynamic logic," in *Handbook of Philosophical Logic, Vol. II* (D. Gabbay and F. Guenthner, eds.), D. Reidel Publishing Co., Dordrecht, 1984, pp. 497–604.

[Ha2] D. Harel, "Effective transformations on infinite trees, with applications to high undecidability, dominoes, and fairness," *Journal of the ACM*, Vol. 33, No. 1, Jan. 1986, pp. 224–248.

[HC] G.E. Hughes and M.J. Cresswell, *An Introduction to Modal Logic*, Methuen & Co., London, 1968.

[HO] B. Hailpern and S. Owicki, "Modular verification of computer communication protocols," *IEEE Trans. on Communications*, Vol. COM-31, No. 1, Jan. 1983, pp. 56–68.

[Ka] J.A.W. Kamp, "Tense logic and the theory of linear order," PhD Thesis, University of California, Los Angeles, May 1968.

[Kl] S.C. Kleene, *Introduction to Metamathematics*, North Holland Publishing Co., Amsterdam, 1952.

[MP1] Z. Manna and A. Pnueli, "How to cook a temporal proof system for your pet language," Tenth ACM Symposium on Principles of Programming Languages, Jan. 1983, pp. 141–154.

[MP2] Z. Manna and A. Pnueli, "Verification of concurrent programs: A temporal proof system," Report No. STAN-CS-83-967, Computer Science Department, Stanford University, June 1983.

[MP3] Z. Manna and A. Pnueli, "Adequate proof principles for invariance and liveness properties of concurrent programs," *Science of Computer Programming*, Vol. 4, No. 3, Dec. 1984, pp. 257–289.

[MW] Z. Manna and R. Waldinger, *The Logical Basis for Computer Programming, Vol. 1*, Addison-Wesley, Reading, Massachusetts, 1985.

[N] I. Németi, "Nonstandard dynamic logic," in *Logics of Programs* (D. Kozen, ed.), Springer-Verlag LNCS 131, 1981, pp. 311-348.

[OL] S. Owicki and L. Lamport, "Proving liveness properties of concurrent programs," *ACM Transactions on Programming Languages and Systems*, Vol. 4, No. 3, July 1982, pp. 455–495.

[Pa1] R. Parikh, "A decidability result for second order process logic," 19th Annual IEEE Symposium on Foundations of Computer Science, 1978, pp. 177–183.

[Pa2] R. Parikh, private communication.

[Pn] A. Pnueli, "The temporal semantics of concurrent programs," *Theoretical Computer Science* 13, 1981, pp. 45–60.

[R] H. Rogers, *Theory of Recursive Functions and Effective Computability*, McGraw-Hill Book Company, New York, 1967.

[Sa1] I. Sain, "The reasoning powers of Burstall's (modal logic) and Pnueli's (temporal logic) program verification methods," in *Logics of Programs* (R. Parikh, ed.), Springer-Verlag LNCS 193, 1985, pp. 302–320.

[Sa2] I. Sain, "Relative program verifying powers of the various temporal logics," unpublished manuscript, Mathematical Institute of the Hungarian Academy of Sciences, 1985.

[So] R. Solovay, "Provability interpretations of modal logic," *Israel Journal of Mathematics*, Vol. 25, 1976, pp. 287–304.

[VB1] J. Van Benthem, *The Logic of Time*, D. Reidel Publishing Co., Dordrecht, 1982.

[VB2] J. Van Benthem, "Correspondence Theory," in *Handbook of Philosophical Logic, Vol. II* (D. Gabbay and F. Guenthner, eds.), D. Reidel Publishing Co., Dordrecht, 1984, pp. 167–247.

SRC Reports

"A Kernel Language for Modules and Abstract Data Types."
R. Burstall and B. Lampson.
Research Report 1, September 1, 1984.

"Optimal Point Location in a Monotone Subdivision."
Herbert Edelsbrunner, Leo J. Guibas, and Jorge Stolfi.
Research Report 2, October 25, 1984.

"On Extending Modula-2 for Building Large, Integrated Systems."
Paul Rovner, Roy Levin, John Wick.
Research Report 3, January 11, 1985.

"Eliminating go to's while Preserving Program Structure."
Lyle Ramshaw.
Research Report 4, July 15, 1985.

"Larch in Five Easy Pieces."
J. V. Guttag, J. J. Horning, and J. M. Wing.
Research Report 5, July 24, 1985.

"A Caching File System for a Programmer's Workstation."
Michael D. Schroeder, David K. Gifford, and Roger M. Needham.
Research Report 6, October 19, 1985.

"A Fast Mutual Exclusion Algorithm."
Leslie Lamport.
Research Report 7, November 14, 1985.

"On Interprocess Communication."
Leslie Lamport.
Research Report 8, December 25, 1985.

"Topologically Sweeping an Arrangement."
Herbert Edelsbrunner and Leonidas J. Guibas.
Research Report 9, April 1, 1986.

"A Polymorphic λ-calculus with Type:Type."
Luca Cardelli.
Research Report 10, May 1, 1986.

"Control Predicates Are Better Than Dummy Variables For Reasoning About Program Control."
Leslie Lamport.
Research Report 11, May 5, 1986.

"Fractional Cascading."
Bernard Chazelle and Leonidas J. Guibas.
Research Report 12, June 23, 1986.

"Retiming Synchronous Circuitry."
Charles E. Leiserson and James B. Saxe.
Research Report 13, August 20, 1986.

"An $O(n^2)$ Shortest Path Algorithm for a Non-Rotating Convex Body."
John Hershberger and Leonidas J. Guibas.
Research Report 14, November 27, 1986.

"A Simple Approach to Specifying Concurrent Systems."
Leslie Lamport.
Research Report 15, December 25, 1986. Revised January 26, 1988

"A Generalization of Dijkstra's Calculus."
Greg Nelson.
Research Report 16, April 2, 1987.

"*win* and *sin*: Predicate Transformers for Concurrency."
Leslie Lamport.
Research Report 17, May 1, 1987.

"Synchronizing Time Servers."
Leslie Lamport.
Research Report 18, June 1, 1987.

"Blossoming: A Connect-the-Dots Approach to Splines."
Lyle Ramshaw.
Research Report 19, June 21, 1987.

"Synchronization Primitives for a Multiprocessor: A Formal Specification."
A. D. Birrell, J. V. Guttag, J. J. Horning, R. Levin.
Research Report 20, August 20, 1987.

"Evolving the UNIX System Interface to Support Multithreaded Programs."
Paul R. McJones and Garret F. Swart.
Research Report 21, September 28, 1987.

"Building User Interfaces by Direct Manipulation."
Luca Cardelli.
Research Report 22, October 2, 1987.

"Firefly: A Multiprocessor Workstation."
C. P. Thacker, L. C. Stewart, and E. H. Satterthwaite, Jr.
Research Report 23, December 30, 1987.

"A Simple and Efficient Implementation for Small Databases."
Andrew D. Birrell, Michael B. Jones, and Edward P. Wobber.
Research Report 24, January 30, 1988.

"Real-time Concurrent Collection on Stock Multiprocessors."
John R. Ellis, Kai Li, and Andrew W. Appel.
Research Report 25, February 14, 1988.

"Parallel Compilation on a Tightly Coupled Multiprocessor."
Mark Thierry Vandevoorde.
Research Report 26, March 1, 1988.

"Concurrent Reading and Writing of Clocks."
Leslie Lamport.
Research Report 27, April 1, 1988.

"A Theorem on Atomicity in Distributed Algorithms."
Leslie Lamport.
Research Report 28, May 1, 1988.

"The Existence of Refinement Mappings."
Leslie Lamport.
Research Report 29, August 14, 1988.

by Martín Abadi

Systems Research Center
130 Lytton Avenue
Palo Alto, California 94301