A Non-Reified Temporal Logic

Fahiem Bacchus[*]
Computer Science
University of Waterloo
Waterloo, Ontario
Canada, N2L–3G1
fbacchus@dragon.waterloo.edu

Josh Tenenberg[†]
Computer Science
University of Rochester
Rochester, New York
U.S.A., 14627
josh@cs.rochester.edu

Johannes A. Koomen[‡]
Computer Science
University of Rochester
Rochester, New York
U.S.A., 14627
koomen@cs.rochester.edu

Abstract

A temporal logic is presented for reasoning about propositions whose truth values might change as a function of time. The temporal propositions consist of formulae in a sorted first-order logic, with each atomic predicate taking some set of temporal arguments as well as a set of non-temporal arguments. The temporal arguments serve to specify the predicate's dependence on time. By partitioning the terms of the language into two sorts, temporal and non-temporal, time is given a special syntactic and semantic status without having to resort to reification. The benefits of this logic are that it has a clear semantics and a well studied proof-theory. Unlike the first-order logic presented by Shoham, propositions can be expressed and interpreted with respect to any number of temporal arguments, not just with respect to a pair of time points (an interval). We demonstrate the advantages of this flexibility. In addition, nothing is lost by this added flexibility and more standard and useable syntax. To prove this assertion we show that the logic completely subsumes Shoham's temporal logic [1].

1 Introduction

Many problems in artificial intelligence require reasoning about events or states of the world that have temporal extent. Standard first-order logics have proven useful for reasoning about static propositions and their consequences, but have not been readily adaptable to the greater demands of temporal reasoning. For instance, "block A is on block B" can be represented as $\textsc{on}(\textsc{a}, \textsc{b})$, but "block A is on block B from 7pm to 12pm" is less obviously represented. One approach is to add to the predicates additional arguments denoting the temporal elements associated with the assertion: $\textsc{on}(7, 12, \textsc{a}, \textsc{b})$. This approach has received little attention, being typically abandoned in favor of *reified* logics [2, 3] containing truth predicates relating atemporal proposition terms

[*]This work was supported by a grant from the Faculty of Mathematics, University of Waterloo, and by NSERC grant OPG0041848.

[†]This work was supported in part by the Air Force Systems Command, Rome Air Development Center, Griffiss Air Force Base, New York 13441–5700, and the Air Force Office of Scientific Research, Bolling AFB, DC 20332 under Contract Number F30602–85–C–0008 which supports the Northeast Artificial Intelligence Consortium (NAIC).

[‡]This work was supported by NSF research grant DCR–8351665.

(e.g., $\textsc{on}(\textsc{a}, \textsc{b})$) to temporal points or intervals (e.g., $\text{HOLDS}[7, 12, \textsc{on}(\textsc{a}, \textsc{b})]$). In contrast to the recent trends, we demonstrate a logic obtained by including the additional temporal arguments, showing that this preserves the first-order structure of the propositions, has a clear semantics, and a standard proof-theory.[1] In addition, we make no ontological commitment toward interpreting the temporal objects as either points or intervals, leaving this choice instead to the axiom writer. These advantages are obtained by keeping within a classical first-order framework. We present first the syntax and semantics of our logic and discussing its salient features. We then compare our system to another recent non-reified temporal logic, developed by Shoham [1]. Shoham's logic deals with preserving the first-order structure of temporally scoped propositions, but uses a complex, non-standard semantics for which no proof theory has been provided. We show that this logic is subsumed by our approach, demonstrating that these non-standard features are not required. Finally, we make some comparisons between the non-reified approach that we use here and the reified logics that have been used previously.

2 A Non-Reified Temporal Logic

In the logic that we present, propositions are associated with time objects by including temporal arguments to the functions and predicates. For example, one can represent the assertion "the President of the USA in 1962 died in 1963" as $\textsc{died}(1963, \textsc{president}(1962, \textsc{usa}))$. Temporal objects are distinguished from non-temporal objects by partitioning both the universe of discourse and the symbols of the language used to denote the universe. One can thus specify, for each function and predicate symbol, some number, n, of temporal arguments and some number, m, of non-temporal arguments, and for each function symbol, whether it evaluates to a temporal or non-temporal object.

Representing temporal assertions by the "method of temporal arguments" (a phrase due to Haugh [5]), has long been used in database applications (e.g., Ahn [6]), but has typically been ignored in AI. One notable exception is a logic presented by Haugh [5]. Although many of the ideas first presented by Haugh are echoed here, there are several points of departure. Syntactically, we allow considerably more flexibility, by not limiting the number of temporal arguments of functions or predicates as Haugh does. Further, we permit the presence of functions that take a combination of temporal and non-temporal objects. However, the primary difference is Haugh's position that a non-standard semantics is required in order to sufficiently constrain the structure of the temporal domain. As we demonstrate in Section 3, the model theory need not provide this structure. In this way, the axiom writer is free to choose the particular axiomatic theory that represents the set of intended models, unconstrained by any *a priori* choice of structure inherent in the logic itself. Further, as with Shoham, Haugh does not provide a proof theory for his semantics.

[1]Our logic can be viewed as being in the same spirit as Green's original work on logic based planning [4]. Green used additional state arguments in his predicates, adding the states as extra individuals to the object language. The reified logics on the other hand take the approach of separating the language of states from the language which describes the domain. To express the dependence of the domain statements on the current state, the formulae of the domain language are reified, i.e., added as extra individuals in the state language. Viewing the time arguments as being state arguments gives the parallel between Green's approach and ours.

2.1 Syntax

Our logic, which we will refer to as "BTK," is a standard many-sorted logic having two disjoint sorts, for temporal and non-temporal objects. It is therefore an element of Wang's 2-sorted logical system T_2 [7]. We briefly review the syntax of a two-sorted logic.

The variables, $\mathbf{V}$, are of two different sorts, $\mathbf{V_t}$, and $\mathbf{V_u}$, and for every pair of natural numbers n and m there is a set (possibly empty) of (n,m)-ary function symbols, $\mathbf{F}^{(n,m)}$, and a set (possibly empty) of (n,m)-ary predicate symbols, $\mathbf{P}^{(n,m)}$. For both function and predicate symbols the first n arguments are temporal while the last m are non-temporal. We restrict the functions to range over the temporal sub-domain, calling these temporal functions, or over the non-temporal sub-domain, calling these non-temporal functions. Therefore the sort of a function will be uniquely determined by its range. We take the constants, $\mathbf{C}$, to be 0-*ary* function symbols. Hence, the constants are sorted as well.

Terms and wffs are defined in the standard fashion, with the only restriction being that arguments of the correct sort must be given for each function and predicate. We will use "t" to denote temporal terms, and "c" to denote non-temporal terms, sometimes with subscripts. The sort of a term is determined by the sort of its outermost symbol. In addition, we will call predicates that take only temporal arguments *temporal predicates*, and predicates that take only non-temporal arguments *non-temporal predicates.*

A set of inference rules is provided by Wang in [7]. For our present purposes we need not include them here. A BTK language along with the inference rules and proper axioms is a BTK *system.*

2.2 Semantics

A model is defined to be the tuple $\mathcal{M} = \langle\langle T, U\rangle, \sigma\rangle$. T and U are non-empty universes, and σ is an interpretation function that maps each (n,m)-ary temporal function to an (n,m)-ary function from $T^n \times U^m$ to T, each (n,m)-ary non-temporal function to an (n,m)-ary function from $T^n \times U^m$ to U, and each (n,m)-ary predicate to an (n,m)-ary predicate on $T^n \times U^m$. Meaning is assigned to the formulae by standard first-order rules for interpreting the atomic formulas, truth-functional connectives, and quantifiers, except that each quantified variable ranges only over the appropriate universe. We denote the interpretation of ψ under σ by ψ^σ.

3 Relativization and Proof Theory

Rather than using a 2-sorted logic for BTK, we could instead have used a standard (one-sorted) logic. Thus, for every BTK system we could have a corresponding BTK$'$ system, where there is only a single universe, and thus only a single sort for the variables and functions. In addition, the one-place predicates *Temporal* and *Non-Temporal* are part of every BTK$'$ language. The BTK$'$ system is then defined analogously to that of BTK, with the addition of the following theorems:

1. $\exists x, y.\ \mathit{Temporal}(x) \wedge \mathit{Non\text{-}Temporal}(y)$

2. $\forall x.\ \mathit{Temporal}(x) \oplus \mathit{Non\text{-}Temporal}(x)$,

where $\oplus$ is exclusive-or. A statement ϕ in BTK can be "relativized" to a statement ϕ' in BTK$'$, by substituting simultaneously in ϕ, for each expression of the form $\forall x.\alpha$, where x is a temporal variable, an expression of the form

$$\forall x.\ \mathit{Temporal}(x) \rightarrow \alpha,$$

and for each expression of the form $\forall z.\alpha$, where z is a non-temporal variable, an expression of the form

$$\forall z.\mathit{Non\text{-}Temporal}(z) \rightarrow \alpha^{2}.$$

We then get the following result trivially from Wang, (attributed to Herbrand [8]):

> A statement of any system BTK is provable in BTK if and only if its relativization in the corresponding system BTK$'$ is provable in BTK$'$.

BTK$'$ is a standard first-order system and as such its proof theory, and automated use of this proof theory, has been well studied. The theorem implies that any first-order proof theory can be trivially used as a proof theory for BTK: one only has to relativize every statement of a given BTK system and do deduction in the first order BTK$'$. In addition, by relativizing a BTK system in this fashion, one can automate deduction by using standard automated theorem proving techniques.

It should be noted, however, that one need not relativize the logic in order to obtain either a proof theory or an automated theorem prover for a sorted logic. This is because sorted proof theories and their automation have been well studied. For example, Walther [9] has developed an automated theorem prover for a sorted clause form logic, based upon resolution and paramodulation. In fact, he gives some strong arguments to indicate that reasoning directly with the sorted logic would be far more efficient. It is a trivial exercise to cast BTK as a variant of Walther's clause form sorted logic and to use his automated reasoner.

The major difficulty involved in reasoning in a BTK system lies in reasoning with the temporal terms. Halpern and Shoham [10] have demonstrated that for modal temporal logics the complexity of reasoning is highly dependent on the nature of the temporal domain. A similar situation holds for BTK.

Many different complete proof theories exist for first-order logic (e.g., the ones given by Barwise in [11]). These proof theories give mechanical procedures for generating all valid first-order formulae. As long as we can completely axiomatize the special properties of the defined relations, we can use one of these complete proof theories to generate all formulas valid for these relations. First-order domains, however, have no special structure, they consist simply of a collection of relations defined over an unstructured domain of discourse. The temporal sub-domain, on the other hand, does possess special structure, and it may not be possible to provide a complete axiomatization of this structure. For example, if one requires that the temporal domain T be the set of integers, then it is well known that there is no complete axiomatization of the integers in languages which include multiplication and addition.[3] In other words, if one places no restrictions on the set of legal BTK models, in particular, if one places no requirements on the structure of the temporal domain, then complete proof theories can be provided for BTK, by the above relativization result

[2] We are taking existentially quantified variables as defined from universally quantified variables

[3] This follows directly from Gödel's incompleteness result, see, e.g., Barwise [11].

and the existence of complete proof theories for first-order logic, or by the use of complete proof theories for sorted first-order logic, like Walther's. On the other hand if one restricts the set of legal BTK models to be models where the temporal domain T has some special structure one cannot necessarily guarantee a proof theory complete for these models: even the relativized first-order BTK$'$ will not have a complete proof theory.

Although temporal structures like the integers cannot be characterized by a set of first-order axioms, there are many other temporal structures that can be. These include temporal domains that are linearly ordered, models of Peano arithmetic, and totally ordered fields. This last is particularly useful. The reals are an instance of a totally ordered field. Hence, if we choose such a temporal structure we will be able to axiomatize its behavior and be assured that all deductions carried out with this axiomatization will be sound with respect to the reals. Furthermore, it is well known that every totally ordered field has a subfield which is isomorphic to the rationals. This means that we can include in our language temporal constants representing any rational time point. When one considers the fact that our computers can only represent rationals (and only a finite set of rationals at that), it should be clear that one can capture a great deal of useful reasoning about real time points by restricting oneself to the rationals.

Another interesting type of temporal domain which has a complete axiomatization occurs when the primitive temporal objects are intervals. For example, Ladkin [12] demonstrates that the axiomatic theory of the *Interval Calculus* provided by Allen and Hayes [13] precisely characterizes the unbounded linear orders.

If the temporal domain of BTK, T, is defined to be any one of these temporal structures, or any other structure which we can characterize by a set of axioms, a complete proof theory can be easily generated. One just adds the axiomatization of the temporal domain to the axiomatization of first-order logic. The first-order rules of inference will provide a complete proof theory when they operate on the union of the temporal and first-order axioms. This can be done in either the sorted context or, via relativization of the temporal axioms, in the unsorted context.

To make this more precise we make the following definitions.[4]

Definition 1 A class of temporal structure $\mathcal{T}$ is said to be *characterized* by an axiomatization AX (i.e., a recursive set of axioms) if we have that $T \models AX$ iff $T \in \mathcal{T}$. That is, the models of AX are exactly the class of temporal structures.

Clearly, the class of linearly ordered temporal structures is characterized by the first-order axioms of a linear order, i.e., the axioms (1) $\forall x. x \leq x$, (2) $\forall xy. x \leq y \wedge y \leq x \rightarrow y = x$, (3) $\forall xyz. x \leq y \wedge y \leq z \rightarrow x \leq z$,

[4]We need to be precise as it is not necessarily the case that we can combine two complete axiomatizations and retain completeness. Completeness is closely tied to the expressiveness of the language. For example, although the reals cannot be characterized in first-order logic (i.e., we cannot write a set of first-order axioms that has only the reals as a model), Tarski [14] has shown that the first-order theory of real closed fields (RCF) is complete for the reals. That is, a formula written in the language of RCF is valid if and only if it is true of the reals. This result rests on the limited expressiveness of the language of RCF. RCF is capable of expressing only a limited set of assertions about the reals, and a complete proof theory exists for this limited set, but not for larger sets. When we combine our temporal and atemporal languages into a BTK system we are increasing the expressiveness of our temporal language, through the mixed functions and predicates. Hence, we may have an axiomatization that, for our original temporal language, is complete with respect to a particular temporal domain, but when we combine that temporal language with an atemporal language, to form a BTK system, we may lose completeness. We have now increased the expressiveness of our temporal language and may have exceeded the capabilities of the original axiomatization.

and (4) $\forall xy. x \le y \lor y \le x$. Similarly, the class of temporal structures that are totally ordered fields are characterized by the first-order axioms of a totally ordered field (see, e.g., Shoenfield [15]). However, there does not exist any axiomatization that characterizes the integer or the real temporal structures.

Let BTK$_{\mathcal{T}}$ be a BTK system in which any member of $\mathcal{T}$ is admissible as the temporal domain T, and let $AX_{\mathcal{T}}$ be an axiomatization which characterizes $\mathcal{T}$. Let AX_{FO} be any complete axiomatization of first-order logic. Then we have:

Theorem 2 *The axiom system consisting of $AX_{\mathcal{T}} \cup AX_{FO}$ is a complete axiomatization of BTK$_{\mathcal{T}}$.*
Proof: The axiom system is simply a collection of first-order axioms; hence, for any satisfiable formula α we can construct a Henkin model. This model will satisfy α and all of the axioms. In particular, it will satisfy $AX_{\mathcal{T}}$. Since $AX_{\mathcal{T}}$ characterizes $\mathcal{T}$, the temporal domain will be a member of $\mathcal{T}$, and therefore a legitimate model for BTK$_{\mathcal{T}}$. Thus, a model exists for every satisfiable formula, and as a standard consequence the axiom system is complete.

An argument made by Shoham [1] is that a logic based on the method of temporal arguments, such as BTK, is insufficient for the demands of temporal reasoning:

> This option is not acceptable from our standpoint, although there is nothing technically wrong with it. The problem is that if time is represented as an argument (or several arguments) to predicates, there is nothing general you can say about the temporal aspect of assertions. For example, you cannot say that "effects cannot precede their causes"; at most you can say that about specific causes and effects. Indeed, this first option accords no special status to time—neither conceptual nor notational—which goes against the very spirit of our enterprise.

Haugh [5] has given some counter arguments to this claim, but with BTK we can give a more precise refutation. We will show that BTK subsumes the logic developed by Shoham (to be referred to as "STL"). Given this result, it is the case that STL can represent the sentence "effects cannot precede their causes" only if BTK can: STL is no more expressive than a logic obtained by adding additional time arguments to the predicates.[5] In addition, time *is* given a special status in BTK by using a sorted logic that distinguishes temporal objects from all other objects, both semantically (conceptual), and syntactically (notational).

4 Shoham's Logic

In this section we briefly describe STL and discuss the main differences between it and our temporal logic. Shoham's logic is presented in [1].

STL is sorted in much the same way as BTK. There are a set of temporal constants and variables as well as non-temporal constants and variables. However the treatment of function and relation symbols is different. STL has temporal functions, but these functions can only take temporal arguments—they are a special case of BTK temporal functions, i.e., temporal functions with $m = 0$. Furthermore, STL allows no user defined temporal relations, just the predefined ones

[5]Later we will return to the question of whether or not a non-reified logic like BTK, or STL, can in fact express this sentence.

'$\leq$' and '$=$'. Non-temporal functions and relations are also treated differently. Syntactically they do not take any temporal arguments, although semantically they are always evaluated with respect to a pair of time *points* (an interval).

The atomic formulae of STL are of two types—formulae formed from the two temporal relations $=$ and $\leq$, e.g., $t_1 = t_2$ or $t_1 \leq t_2$, where t_1, t_2 are both temporal terms,[6] and formulae formed via the TRUE construct. Using Shoham's definition,

> If t_a and t_b are temporal terms, $c_1, \ldots, c_m$ are non-temporal terms, and R is a m-ary relation symbol, then
> $$\mathrm{TRUE}(t_a, t_b, R(c_1, \ldots, c_m))$$
> is an atomic formula.

For example, the sentence "block A is on block B between 7 and 12" would be expressed in STL as

$$\mathrm{TRUE}(7, 12, \textsc{on}(\textsc{a}, \textsc{b})).$$

TRUE is not a relation in STL, nor is it a modal operator; rather, it is a *reifying* context. It asserts that the proposition $R(c_1, \ldots, c_m)$ is true over the interval specified by t_a and t_b. The time points t_a and t_b do not appear as direct arguments to the relation symbol R, nor to any functions which may appear in the c_i's, but they affect the semantic interpretation of these symbols.

The rest of the formulae of STL are built up in the standard manner, by closing off under negation, conjunction and universal quantification. As in BTK, quantification can occur over the time points or over the ordinary individuals, dependent on the sort of variable used.

Semantically STL has, like BTK, a universe of temporal objects and a universe of individuals. Unlike BTK, STL requires that the temporal objects be time *points*, and requires that all of the atomic formulae include exactly two temporal arguments (denoting the starting and ending points of the temporal interval over which the proposition holds). The interpretation function maps the temporal function symbols to functions over the universe of time points. The mapping of the non-temporal function and relation symbols is, however, determined not only by the symbol itself but also by the two time points which occur in the TRUE construct.[7] In particular, there is a mapping from non-temporal function symbols and a pair of time points to functions over the universe of individuals. Similarly, there is a mapping from non-temporal relation symbols and a pair of time points to relations over the individuals. Each non-temporal function symbol denotes many different functions over the non-temporal individuals. The particular function that it denotes is determined by the time points in its TRUE context, and likewise for non-temporal relation symbols.[8] Once the particular non-temporal function or relation is identified by the time points the rest of the interpretation proceeds in a standard manner. A fuller description of Shoham's logic is provided in the Appendix.

Given the use of what resembles a truth predicate, STL bears a syntactic similarity to reified logics, and in fact, Shoham argues that it is "a new reified temporal logic" [1, p. 103]. However,

[6] Terms in STL are formed in the standard manner, i.e., constants and variables, or functions applied to the proper number of terms. Note, however, that in STL there are no mixed functions, i.e., functions of temporal and non-temporal terms.

[7] This is the only place that a non-temporal relation or function can appear.

[8] In this sense, the non-temporal functions can be viewed as *fluents* [16].

since formulae are not treated as object denoting terms, and TRUE, despite its resemblance, is not formally a predicate, we view STL as being closer to the spirit of an *intensional* logic. The semantics that Shoham provides bears a striking resemblance to Dowty's *temporal models* for his tensed logic [17, p. 113], the primary difference being that in STL, the time objects are represented explicitly in the formulae, while Dowty provides them implicitly in the model.

4.1 Comparison of Shoham's Temporal Logic to BTK

There are several implications of Shoham's approach. One is that every non-temporal function and relation is *always* dependent on *exactly* two time points. Thus, for example, it is clumsy to specifying that a function is dependent on only one time point, $\textsc{location}(\textsc{space-shuttle}, t_1)$, or that a relation is "eternal," i.e., not dependent on time, $\textsc{block}(A)$. Since the time dependency is specified semantically the syntax is completely rigid on this matter. In BTK, there is neither a syntactic commitment to the number of temporal objects that any function or predicate may depend on, nor is there any commitment to interpreting the temporal objects as either intervals or points. It is our position that these choices should not be constrained by the logic, but should be left to the axiom writer to decide.

An major problem with STL is that there is no simple way of referring to one temporally referenced object within the context of another temporal interval, such as the example "the President of 1962 died in 1963." This is because Shoham requires all non-temporal terms to be evaluated with respect to the *same* temporal terms, i.e., those specified in the TRUE context. To express such a statement in Shoham's logic one has to resort to the more cumbersome use of equality and quantification:

$$\forall x[\mathrm{TRUE}(1962, 1963, \textsc{president}(\textsc{usa}) = x) \rightarrow \mathrm{TRUE}(1963, 1964, \textsc{died}(x))]$$

This can be compared with the expression of this statement in BTK given in section 2. We will have more to say about reasoning with Shoham's logic below.

A further problem is that Shoham does not allow for temporal predicates, except for the predefined ones $\leq$ and $=$.[9] Thus one would have to extend his formalism to, for instance, embed the MEETS predicate and axioms of [18] within STL.

A major difficulty with Shoham's approach is that, since he has chosen to move away from standard (or sorted) first order syntax, first order proof theory, which is purely syntactic, no longer applies. Hence, Shoham's logic requires a new proof theory. This means that *one cannot justify the use of Shoham's logic for reasoning about temporal propositions*. There is no reasoning procedure specified that provides any formal guarantees of soundness or completeness.

It may not be very difficult to provide a proof theory for Shoham's logic, but this in itself would not suffice to provide a useful tool for reasoning. One would also have to develop some understanding of the properties of such a proof theory, especially if one wishes to construct automated theorem provers based on it. This may not be an easy task since, as indicated above, there are some examples which force the use of equality which is known add complexity, especially for automation. The fact that our temporal logic has a standard syntax means that we can take advantage of 20 years

[9] What we mean here is that the semantic model Shoham defines does not allow for "user defined" temporal relations. He does allow an arbitrary set of temporal functions.

of research in automated reasoning, and many more years of theoretical work on understanding first-order proof theory.

Our temporal logic is a simple sorted first order logic. It is simple because the sorts do not intersect. Proof theories for sorted first order logics already exist, and are applicable *as is* to our logic. In addition, considerable work has been done on automating such proof theories, [9]. Furthermore, if one chooses to interpret our logic as non-sorted, then standard FOL proof theory applies, as do automated theorem provers for first order logic.

One would hope that there are compensations in using STL in exchange for abandoning standard proof theory. This is, however, not the case. The next section will show that nothing is lost in moving from Shoham's temporal logic to the logic proposed in this paper. It shows that STL is subsumed by our logic in the precise sense that any STL model can be transformed to a BTK model in such a way that there is a one to one correspondence between the sentences satisfied by the STL model and the sentences satisfied by the BTK model. These results also show that there is one way of doing reasoning in STL: *translate it into BTK.*

5 Subsumption of Shoham's Logic

We show that Shoham's logic (STL) is subsumed by the logic proposed in this paper (BTK) by defining two transformations, a syntactic transformation, π_{syn}, and a semantic transformation, π_{sem}.[10] π_{syn} maps sentences of STL to sentences of BTK, while π_{sem} maps models of STL to models of BTK. Using these two transformations we will show that any STL model can be transformed into a BTK model in such a way that the set of sentences satisfied by the BTK model[11] includes the transformed set of STL sentences satisfied by the STL model. In other words, any set of STL sentences can be rewritten as a set of BTK sentences without eliminating any models which satisfy those sentences.

The syntactic transformation is based on a simple idea. In Shoham's logic all predicate symbols and non-temporal function symbols are interpreted with respect to the two time terms which appear as the first two arguments of the "TRUE" construct. In transforming STL to BTK we take these two time terms and add them as explicit temporal arguments to the predicate symbol, and similarly we add them as extra arguments to the non-temporal functions. Temporal functions are unaffected by the transformation, and none of the symbols are altered—they are just rearranged.

The only technical point is that non-temporal terms can be built up from nested application of non-temporal functions. In this case it is necessary to propagate the two temporal arguments recursively to all embedded function terms. For example, the non-temporal term $f(g(h(c)))$ in STL, where f, g, and h are non-temporal functions and c is a non-temporal constant, must be converted to a term of the form $f(t_1, t_2, g(t_1, t_2, h(t_1, t_2, c)))$, where t_1 and t_2 are the propagated temporal terms. Here each of the functions f, g, and h has been converted to functions with two extra temporal arguments.

[10] Ladkin [19] uses a similar approach to map Allen's interval calculus [20] to the language of rational numbers. In doing this he is able to give decision procedures for the interval calculus.

[11] A model, $\mathcal{M}$, satisfies a sentence, α, written $\mathcal{M} \models \alpha$, if $\alpha^\sigma = \top$, i.e., if α is true under the interpretation of the model, σ.

The following examples should give a good idea of the nature of the syntactic transformation. The STL expressions

1. $\mathrm{TRUE}(t_1, t_2, \textsc{colour}(\textsc{house}17, \textsc{red}))$.
2. $\mathrm{TRUE}(t_3, t_4, \textsc{gender}(\textsc{president}(\textsc{usa}), \textsc{male}))$.
3. $\mathrm{TRUE}[t_1, f_t(t_1), P(h(g(B)))]$.

will be transformed to the BTK expressions:

1. $\textsc{colour}(t_1, t_2, \textsc{house}17, \textsc{red})$.
2. $\textsc{gender}(t_3, t_4, \textsc{president}(t_3, t_4, \textsc{usa}), \textsc{male})$.
3. $P[t_1, f_t(t_1), h(t_1, f_t(t_1), g(t_1, f_1(t_1), B))]$.

The semantic transformation is similar. In STL each non-temporal function or relation symbol actually denotes a set of different functions or relations over the non-temporal individuals. The time points in the "TRUE" context determine which element of the set is picked out for this particular instance. In converting from an STL model to a BTK model we gather up all of the different functions associated with each function symbol and construct a single function which has two extra temporal arguments. The new BTK function has the property that when it is evaluated at a fixed pair of time points it is the same function as the function denoted by the STL symbol when that symbol is interpreted with respect to those time points. The non-temporal relations are transformed in a similar manner.

These transforms are defined formally in the appendix, where we prove the following theorem.

Theorem 3 *Given an STL sentence α and an STL model $\mathcal{M}$ then*

$$\mathcal{M} \models \alpha \quad \textit{iff} \quad \pi_{sem}(\mathcal{M}) \models \pi_{syn}(\alpha).$$

Proof The proof is straight forward, but requires the development of a fair amount of notation. See the appendix for details. ■

This theorem is a formal specification of the manner in which STL is subsumed by BTK, and it has an interesting corollary regarding proof theories.

Corollary 4 *A sound proof theory in BTK can be used to produce sound inferences in STL.*

Proof Let α and β be sentences of STL. We claim that if $\pi_{syn}(\alpha) \vdash \pi_{syn}(\beta)$ is a sound deduction in BTK, then $\alpha \models \beta$ in STL. That is, if the syntactic transformation of α can be used to deduce (soundly) the syntactic transformation of β then α entails β in STL.[12]

If $\pi_{syn}(\alpha) \vdash \pi_{syn}(\beta)$ then, by the assumption of soundness, for all BTK models, $\mathcal{M}'$, we have that $\mathcal{M}' \models \pi_{syn}(\alpha)$ implies that $\mathcal{M}' \models \pi_{syn}(\beta)$. Thus, this also holds for all models which have the special form $\pi_{sem}(\mathcal{M}_S)$ for all STL models $\mathcal{M}_S$. Hence, by theorem 3 we have that for all STL models $\mathcal{M}_S$, $\mathcal{M}_S \models \alpha$ implies $\mathcal{M}_S \models \beta$. In other words $\alpha \models \beta$ in STL. ■

[12] $\alpha \models \beta$ if any model which satisfies α (i.e., assigns truth to α) also satisfies β. A proof theory is said to be sound if $\alpha \vdash \beta$ implies $\alpha \models \beta$. It is said to be complete if $\alpha \models \beta$ implies $\alpha \vdash \beta$.

It is natural to ask a similar question about completeness. That is, can a complete proof theory in BTK produce a complete set of inferences in STL. Here, however, the answer is more difficult to determine. If we have that $\alpha \models \beta$ in STL, then we know from theorem 3 that $\pi_{syn}(\alpha)$ entails $\pi_{syn}(\beta)$ in every BTK model which is of the form $\pi_{sem}(\mathcal{M}_S)$, for some STL model, $\mathcal{M}_S$. However these are not the only BTK models, and it is quite possible that in some BTK model which is not a transformed STL model $\pi_{syn}(\alpha)$ is true while $\pi_{syn}(\beta)$ is false. Hence in BTK $\pi_{syn}(\alpha) \vdash \pi_{syn}(\beta)$ would not be sound, even though $\alpha \models \beta$ in STL. Although it is clear that these extra BTK models exist, it might still be possible to "factor" out their effect, extending the result to yield completeness. This remains an open question.

6 Translating Shoham's Ontology to BTK

One of the benefits of Shoham's logic is that it does not require the axiom writer to use a fixed ontology of temporally scoped propositions, as, for example, Allen does [2] with his introduction of *properties, events*, and *processes.* Rather, Shoham's logic allows the axiom writer to build her own ontology *axiomatically.*

We argue that Shoham's ontology extends naturally to our logic by virtue of the demonstrated translation, and that, in fact, our ontology is richer, since our logic allows intervals to be the primitive temporal objects rather than being defined by the two endpoints, as in STL. An example showing the translation of the ontology axioms should suffice to demonstrate our claim.

Shoham defines a proposition type x (where proposition types are simply relation symbols with the requisite arguments) to be *downward hereditary* "if whenever it holds over an interval it holds over all of its subintervals." Shoham's axiom schema for this is

$$\forall t_1, t_2, t_3, t_4 . [t_1 \leq t_3 \leq t_4 \leq t_2 \land t_1 \neq t_4 \land t_3 \neq t_2 \land \mathrm{TRUE}(t_1, t_2, x)] \rightarrow \mathrm{TRUE}(t_3, t_4, x).$$

for all x's of the appropriate type. This translates in BTK to the following schema, for each $(2, m)$-ary predicate of the appropriate type:

$$\begin{aligned}&\forall t_1, t_2, t_3, t_4, y_1, \ldots, y_m . \\ &\qquad [t_1 \leq t_3 \leq t_4 \leq t_2 \land t_1 \neq t_4 \land t_3 \neq t_2 \land p(t_1, t_2, y_1, \ldots, y_m)] \rightarrow p(t_3, t_4, y_1, \ldots, y_m).\end{aligned}$$

In addition, the predicate "$\leq$" must be defined axiomatically in BTK, since it is not implicitly defined as it is in STL.[13] In BTK, however, one is not forced to use time *points* so one might alternatively define downward hereditary for a system in which intervals are taken as the interpretation of time objects, as in:

$$\forall i_1, i_2, y_1, \ldots, y_m . \mathit{During}(i_2, i_1) \land p(i_1, y_1, \ldots, y_m) \rightarrow p(i_2, y_1, \ldots, y_m),$$

where it is assumed that *During* has been defined axiomatically.

This same style of translation, then, can be used for any of the other elements of Shoham's ontology: *upward-hereditary, point-downward-hereditary, liquid, gestalt,* etc.

[13] This is because an ordering relation on the entire domain does not make sense for certain temporal structures, e.g., intervals.

7 Comparing Reified and Non-Reified Logics

In BTK, propositions are related to times by adding time-denoting argument terms to each of the predicates. Semantically, a relation's time dependence is modeled by adding a set of temporal objects to each tuple in the relation. For example, if $\textsc{on}$ is a binary predicate dependent on a single temporal argument (say an interval), then in BTK it will denote a set of triples in the semantics, each triple consisting of the pair of objects that are in the $\textsc{on}$ relation and the interval during which they are in this relation. So, for example, $\textsc{on}(I, A, B)$ will be interpreted as asserting that the triple denoted by I, A, and B is a member of the set of triples denoted by $\textsc{on}$.

By contrast, in first-order reified logics such as those of [2, 3], propositions are treated as object denoting terms and related to times through a "truth" predicate, as in $\mathrm{HOLDS}(I, \textsc{on}(A, B))$. In this case $\textsc{on}(A, B)$ is a term, instead of a formula. That is, it denotes an object in the domain, a special "formula" object. The temporal terms, like I, continue to denote temporal objects. The relation defined on the domain is a binary relation that relates formula objects to temporal objects. The symbol $\textsc{on}$, that in BTK was treated as a predicate symbol, is now treated as a function symbol. The object denoting terms remain unchanged (in this example, A and B). Hence, with reified logics there is a change in the denotation of the atemporal predicate symbols, and we have an expanded domain of discourse which includes formula objects.

One does need to make these changes carefully, however. Since $\textsc{on}(A, B)$ is now a term and $\textsc{on}$ is a function, we could, if we used the standard unsorted term formation rules of first-order logic, generate new terms like $\textsc{on}(\textsc{on}(A, B), B)$. Clearly, such terms do not correspond to legal first-order formulas. To avoid such difficulties we must add a precise sort structure to the language, to distinguish those terms which denote the "real" objects of the domain, from those which denote the formula objects. Under such a sortal structure we are prohibited from applying a "real" object function, like $\textsc{on}$, to a formula object, like $\textsc{on}(A, B)$. Lifschitz [3] provides an example of such a carefully constructed sorted, reified logic.

Things can get very complicated if we allow the reification of non-atomic formulae, e.g., terms like $\mathrm{AND}(\textsc{on}(A, B), \textsc{on}(C, D))$ as does Allen [2]. It is fairly straightforward to extend the sortal structure to insure that, e.g, the logical function '$\wedge$' only takes formula objects as arguments. But we also need axioms which specify the equality of certain obviously equal formulas, e.g., axioms like $\forall x, y.\mathrm{AND}(x, y) \equiv \mathrm{AND}(y, x)$, and this is what becomes complex, and probably needlessly so. Most applications do not require the reification of more than the atomic propositions: the logical connectives can be applied outside of the HOLDS predicate. For example, instead of writing $\mathrm{HOLDS}(I, \mathrm{AND}(\textsc{on}(A, B), \textsc{on}(C, D))$ we can write $\mathrm{HOLDS}(I, \textsc{on}(A, B)) \wedge \mathrm{HOLDS}(I, \textsc{on}(C, D))$.

One advantage that is possessed by reified logics is that they allow quantification over propositions. For example, one can express the assertion "effects cannot precede their causes" in a reified logic. We could have a predicate $\textsc{causes}(x, y)$ that takes two formula terms as arguments and asserts that x causes y. With this predicate our assertion could be expressed with the following formula:

$$\forall y, t_2.(\exists x.\textsc{causes}(x, y)) \wedge \mathrm{HOLDS}(t_2, y) \rightarrow (\exists z, t_1.\textsc{causes}(z, y) \wedge \mathrm{HOLDS}(t_1, z) \wedge t_1 < t_2),$$

where x, y and z are formula variables; t_1 and t_2 are temporal variables; and quantification occurs only over the correct subset of the domain. That is, if y has some cause, x, (i.e., it is not spontaneous) and it holds during t_2 there must be some previous time point t_1 where one of its causes, z, (not necessarily the same as x) occurred.

Such a statement cannot be expressed in a non-reified logic like BTK (nor, by our results, in Shoham's logic). In BTK we can only use axiom schema like those we used to define different temporal ontologies. For example, we could write the above as an axiom schema, where the propositional variables x, y and z are no longer quantified. Instead they would have to be treated as meta-variables. Each instantiation of the schema would give the proper assertion for a particular triple of propositions. But this does not quite duplicate the above assertion. In particular, the above assertion holds for all propositions, even if we don't currently have them in our language.

In summary, reified logics have the disadvantage of being more complex, and on a practical side we have less experience with automated reasoning in such logics. But they have the corresponding advantage of being more expressive [21]. The relative merits of reified vs. non-reified logics will depend on the particular application. BTK is a very standard and easily understood formalism that is capable of a wide range of temporal reasoning, and it is likely to be sufficient in any practical temporal reasoning system. If one needs quantification over propositions that cannot be reduced to a collection of instances, however, one must resort to reified logics, or, perhaps, to some combination of the two (e.g., Allen et al. [22]).

8 Conclusion

A temporal logic has been presented for reasoning about propositions whose truth values might change as a function of time. The temporal propositions consist of formulae in a sorted first-order logic with each atomic predicate taking some set of temporal arguments which denote time objects, as well as a set of non-temporal arguments. The temporal arguments serve to specify the proposition's dependence on time. By partitioning the terms of the language into two sorts, temporal and non-temporal, time is given a special syntactic and semantic status in the logic without having to resort to reification or non-standard syntax and semantics. The benefits of this logic are that it has a clear semantics and a well understood proof-theory for which considerable experience in constructing automated reasoners already exists. Unlike the first-order logic presented by Shoham, propositions can be expressed and interpreted with respect to any number of temporal arguments, not just with respect to a pair of time objects (an interval). In addition, the axiom writer is free to consider the time objects as either points or intervals. By proving that the logic completely subsumes Shoham's, we have demonstrated that nothing is lost by this added flexibility and more standard and useable syntax.

Acknowledgements

Thanks to Andre Trudel and Jay Weber for helpful comments.

A Transformation of STL to BTK

Definition 5 *The syntactic transform, π_{syn}, which maps STL sentences to BTK sentences, is defined recursively as follows. It depends on a syntactic transformation of the non-temporal terms* *which is defined next.*

1. $\pi_{syn}(t_a \le t_b) \mapsto t_a \le t_b$, and $\pi_{syn}(t_a = t_b) \mapsto t_a = t_b$ (i.e., temporal terms and formulae are left intact).
2. $\pi_{syn}\Big(\mathrm{TRUE}(t_a, t_b, p(c_1, \ldots, c_n))\Big) \mapsto p(t_a, t_b, \pi_{syn}^{[t_a,t_b]}(c_1), \ldots, \pi_{syn}^{[t_a,t_b]}(c_n))$
3. $\pi_{syn}(\neg\alpha) \mapsto \neg\pi_{syn}(\alpha)$
4. $\pi_{syn}(\alpha \wedge \beta) \mapsto \pi_{syn}(\alpha) \wedge \pi_{syn}(\beta)$
5. $\pi_{syn}(\forall x(\alpha)) \mapsto \forall x(\pi_{syn}(\alpha))$, where x can be a variable of either sort.

Definition 6 *The syntactic transform, $\pi_{syn}^{[t_i,t_j]}$, which maps non-temporal terms of STL to terms of BTK is defined as follows.*

1. If c is a non-temporal constant or variable of STL then $\pi_{syn}^{[t_i,t_j]}(c) \mapsto c$.
2. $\pi_{syn}^{[t_i,t_j]}(f(c_1, \ldots, c_n)) \mapsto f(t_i, t_j, \pi_{syn}^{[t_i,t_j]}(c_1), \ldots, \pi_{syn}^{[t_i,t_j]}(c_n))$

The symbols of the corresponding BTK and STL languages are identical, but as is seen from the definition of π_{syn}, non-temporal functions and predicates have two extra temporal arguments.

Next we define the semantic transformation π_{sem}, but in order to do this we first need to provide more detail about the models of STL.

A model of STL is defined to be the tuple

$$\mathcal{M} = \Big\langle TW, \preceq, W, TFN, FN, RL, M \Big\rangle$$

Where:

1. TW is a universe of time points,
2. $\preceq$ is an ordering relation on TW,
3. W is a universe of individuals,
4. TFN is a set of temporal functions, $TW^n \mapsto TW$,
5. FN is a set of non-temporal functions, $W^n \mapsto W$,
6. RL is a set of non-temporal relations in W^n,
7. M is the tuple of interpretation functions $\langle M_1, M_2, M_3, M_4, M_5 \rangle$, where:
   (a) M_1 is a mapping from the time constants to TW,
   (b) M_2 is a mapping from the non-temporal constants to W,

(c) M_3 is a mapping from the temporal functions to TFN,

(d) M_4 is a mapping from $TW \times TW \times f \mapsto FN$, where f is the set of non-temporal function symbols,

(e) M_5 is a mapping from $TW \times TW \times r \mapsto RL$, where r is the set of non-temporal relation symbols.

In the following we denote temporal terms by t_i (for various subscripts i) and non-temporal terms by c_i (note, terms are syntactic entities). We use hatted $\hat{t}$ (usually with subscripts) to denote time points. These are semantic entities which are members of TW, the universe of time points. In addition, we use hatted $\hat{c}$ or $\hat{a}$ (again usually with subscripts) to denote individuals from the semantic domain W.

The meaning of an expression ψ, $M(\psi)$, is defined as follows:

1. If ψ is a temporal variable, then $M(\psi) = VA_t(\psi)$ where VA_t is a variable assignment function over TW.

2. If ψ is a temporal constant, then $M(\psi) = M_1(\psi)$.

3. If ψ is a temporal term of the form $f(t_1, \ldots, t_n)$, then
$$M(\psi) = M_3(f)(M(t_1), \ldots, M(t_n)).$$

4. If ψ is a non-temporal term, then meaning is assigned to ψ with respect to two time points as follows:

   (a) If ψ is a non-temporal constant, then for all time points $\hat{t_1}$, $\hat{t_2}$,
   $$M(\hat{t_1}, \hat{t_2}, \psi) = M_2(\psi).$$

   (b) If ψ is a non-temporal variable, then for all time points $\hat{t_1}$, $\hat{t_2}$,
   $$M(\hat{t_1}, \hat{t_2}, \psi) = VA_w(\psi),$$
   where VA_w is a variable assignment function over W.

   (c) If ψ is a non-temporal function term of the form $f(c_1, \ldots, c_n)$, then for all time points $\hat{t_1}$,$\hat{t_2}$,
   $$M(\hat{t_1}, \hat{t_2}, \psi) = M_4(\hat{t_1}, \hat{t_2}, f)[M(\hat{t_1}, \hat{t_2}, c_1), \ldots, M(\hat{t_1}, \hat{t_2}, c_n)].$$

And finally, a wff ϕ is satisfied under interpretation $\mathcal{M}$ and variable assignment VA, (written $\mathcal{M} \models_{VA} \phi$) as follows:

1. $\mathcal{M} \models_{VA} t_1 = t_2$ iff $M(t_1) = M(t_2)$.

2. $\mathcal{M} \models_{VA} t_1 \leq t_2$ iff $M(t_1) \preceq M(t_2)$.

3. $\mathcal{M} \models_{VA}$ TRUE$(t_1, t_2, p(c_1, \ldots, c_n))$ iff
$$\langle M[M(t_1), M(t_2), c_1], \ldots, M[M(t_1), M(t_2), c_n] \rangle \in M_5[M(t_1), M(t_2), p].$$

Truth is assigned to non-atomic formulae in the standard fashion. Note that predicates are interpreted with respect to two time points, just as were the non-temporal functions.

We now define the semantic transformation of an STL model.

Definition 7 *The semantic transformation of $\mathcal{M}$, $\pi_{sem}(\mathcal{M})$, is a BTK model constructed as follows.*

1. $T = TW$, the universe of time points is the same.

2. $U = W$, the universe of individuals is the same.

3. If f is a temporal function symbol of STL, then $f^\sigma = M_3(f)$.

4. If f is an n-ary non-temporal function symbol of STL, then the following set of $n+3$ ordered tuples is the interpretation of f under $\pi_{sem}(\mathcal{M})$:

$$f^\sigma = \{\langle \hat{t_1}, \hat{t_2}, \hat{c}_1, \ldots, \hat{c}_n, \hat{a}\rangle | \quad \hat{t_1}, \hat{t_2} \in TW \text{ and } (M_4(\hat{t_1}, \hat{t_2}, f))(\hat{c}_1, \ldots, \hat{c}_n) = \hat{a}\}.$$

   Note that this set does in fact define a function. Given any tuple $\langle \hat{t_1}, \hat{t_2}, \hat{c}_1, \ldots, \hat{c}_n\rangle$, M_4 maps f, $\hat{t_1}$, and $\hat{t_2}$ to a unique function over W^n $(= U^n)$. Hence, the $\hat{c}_i$'s will then map to a unique element $\hat{a}$ of W $(= U)$.

5. If P is a predicate symbol of STL, then

   a) if P is $\leq$, then $P^\sigma = \preceq$, i.e., the semantic ordering relation on TW;

   b) if P is $=$, then $P^\sigma = \{\langle \hat{t}, \hat{t}\rangle | \hat{t} \in T\}$;

   c) if P is an n-ary non-temporal predicate symbol of STL then the following set of $n+2$-ary tuples is the interpretation of P under $\pi_{sem}(\mathcal{M})$:

$$P^\sigma = \{\langle \hat{t_1}, \hat{t_2}, \hat{c}_1, \ldots, \hat{c}_n\rangle | \quad \hat{t_1}, \hat{t_2} \in TW \text{ and } \langle \hat{c}_1, \ldots, \hat{c}_n\rangle \in M_5(\hat{t_1}, \hat{t_2}, P).\}$$

6. To complete the definition of σ, we choose an arbitrary mapping of the temporal variables to T and an arbitrary mapping of the non-temporal variables to U. Finally, we maintain the STK denotations of all constants, i.e., $t^\sigma = M_1(t)$ for all time constant symbols t, and $c^\sigma = M_2(c)$ for all non-temporal constant symbols c.

Now we can prove the main technical result.

Theorem 3 *Given an STL sentence α and an STL model $\mathcal{M}$ then*

$$\mathcal{M} \models \alpha \quad \textit{iff} \quad \pi_{sem}(\mathcal{M}) \models \pi_{syn}(\alpha).$$

Proof The cases where α is of the form $t_1 = t_2$ or $t_1 \leq t_2$ are trivial. The non-trivial case is α of the form

$$\text{TRUE}(t_1, t_2, p(c_1, \ldots, c_n)).$$

We need only consider this case where all of the terms are ground, i.e., variable free, since the formulae of STL and BTK are built up in an identical manner and the universes over which the quantified variables can range are identical. (α is a sentence so all variables are quantified.) $\pi_{syn}(\alpha)$ is of the form

$$p(t_1, t_2, \pi_{syn}^{[t_1,t_2]}(c_1), \ldots, \pi_{syn}^{[t_1,t_2]}(c_n)).$$

$\pi_{sem}(\mathcal{M})$ will be a model for this sentence iff

$$\langle t_1^\sigma, t_2^\sigma, \pi_{syn}^{[t_1,t_2]}(c_1)^\sigma, \ldots, \pi_{syn}^{[t_1,t_2]}(c_n)^\sigma \rangle \in p^\sigma,$$

where σ is the interpretation function of $\pi_{sem}(\mathcal{M})$. By definition, $\mathcal{M}$ is a model of α iff

$$\langle M[M(t_1), M(t_2), c_1], \ldots, M[M(t_1), M(t_2), c_n] \rangle \quad \in M_5[M(t_1), M(t_2), p].$$

Clearly from the construction of $\pi_{sem}(\mathcal{M})$ all temporal terms are given the same denotation in BTK as in STL, i.e., $t^\sigma = M(t)$ for all temporal terms t. We also claim that all non-temporal terms, *in a given TRUE context*, are given the same denotation in BTK as in STL. If the term is a constant this follows directly from the definition of π_{sem}, i.e., $c^\sigma = M_2(c)$. If the term is of the form $f(c_1, \ldots, c_n)$ and is within the temporal context determined by temporal terms t_1 and t_2, then if we take $(c_i)^\sigma = M(M(t_1), M(t_2), c_i)$ for all i by induction, then

$$\begin{aligned}
&\pi_{syn}[f(c_1, \ldots, c_n)]^\sigma \\
&= [f(t_1, t_2, \pi_{syn}^{[t_1,t_2]}(c_1), \ldots, \pi_{syn}^{[t_1,t_2]}(c_n))]^\sigma \\
&= f^\sigma((t_1)^\sigma, (t_2)^\sigma, \pi_{syn}^{[t_1,t_2]}(c_1)^\sigma, \ \ldots, \pi_{syn}^{[t_1,t_2]}(c_n)^\sigma) \\
&= [M_4(M(t_1), M(t_2), f)][M(M(t_1), M(t_2), c_1), \ldots, \\
&\qquad M(M(t_1), M(t_2), c_n)] \\
&= M(f(c_1, \ldots, c_n)).
\end{aligned}$$

Hence all of the terms are given an identical denotation. But using the definition of p^σ we have that

$$\begin{aligned}
&\langle t_1^\sigma, t_2^\sigma, \pi_{syn}^{[t_1,t_2]}(c_1)^\sigma, \ldots, \pi_{syn}^{[t_1,t_2]}(c_n)^\sigma \rangle \in p^\sigma \quad \text{iff} \\
&\langle M(t_1), M(t_2), M[M(t_1), M(t_2), c_1], \ldots, M[M(t_1), M(t_2), c_n] \rangle \in p^\sigma \quad \text{iff} \\
&\langle M[M(t_1), M(t_2), c_1], \ldots, M[M(t_1), (t_2), c_n] \rangle \in M_5[M(t_1), M(t_2), p]
\end{aligned}$$

Q.E.D. ■

References

[1] Yoav Shoham. Temporal logics in AI: Semantical and ontological considerations. *Artificial Intelligence*, 33(1):89–104, 1987.

[2] James F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23(2):123–154, 1984.

[3] Vladimir Lifschitz. A theory of action. *Proceedings of the 10th IJCAI*, 1987.

[4] Cordell C. Green. Application of theorem proving to problem solving. In *Proceedings of the 1st IJCAI*, pages 219–239, 1969. Also in [23, pages 202–222].

[5] Brian A. Haugh. Non-standard semantics for the method of temporal arguments. In *Proceedings of the 10th IJCAI*, pages 449–455, 1987.

[6] I. Ahn. Towards an implementation of database management systems with temporal support. In *Proceedings of the International Conference on Data Engineering*, pages 374–381. IEEE Computer Society Press, 1986.

[7] Hao Wang. Logic of many-sorted theories. *Journal of Symbolic Logic*, 17, 1952.

[8] Jacques Herbrand. *Recherches sur la theorie de la demonstration.* PhD thesis, Paris, 1930.

[9] Christoph Walther. *A Many-Sorted Calculus Based on Resolution and Paramodulation.* Research Notes in Artificial Intelligence. Pitman, London, 1987.

[10] J.Ỹ. Halpern and Y. Shoham. A propositonal modal logic of time intervals. *Proceedings of the Symposium on Logic in Computer Science*, 1986.

[11] Jon Barwise. First-order logic. In Jon Barwise, editor, *Handbook of Mathematical Logic.* North-Holland, 1977.

[12] Peter B. Ladkin. The completeness of a natural system for reasoning with time intervals. In *Proceedings of the 10th IJCAI*, pages 462–467, 1987.

[13] James F. Allen and Patrick J. Hayes. A commonsense theory of time. In *Proceedings of the 9th IJCAI*, pages 528–531, 1985.

[14] A. Tarski. *A Desision Method for Elementary Algebra and Geometry, 2nd Edition.* University of California Press, 1951.

[15] Joseph R. Shoenfield. *Mathematical Logic.* Addison-Wesley, 1967.

[16] John McCarthy and Patrick Hayes. Some philosophical problems from the standpoint of artificial intelligence. In *Machine Intelligence 4*. Edinburgh University Press, 1969. Also in [23, pages 431–450].

[17] David R. Dowty, Robert E. Wall, and Stanley Peters. *Introduction to Montague Semantics.* Synthese Language Library. D. Reidel, Holland, 1981.

[18] Patrick J. Hayes and James F. Allen. Short time periods. In *Proceedings of the 10th IJCAI*, 1987.

[19] Peter B. Ladkin. Satisfying first-order constraints about time intervals. In *Proceedings of the 7th AAAI*, pages 512–517, 1988.

[20] James F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[21] A. G.M̃. Koomen, Johannes. *Reasoning about Recurrence*. PhD thesis, University of Rochester, Dept. of Computer Science, Rochester, NY, July 1989. (also available as URCS TR–307).

[22] James Allen, Henry Kautz, Richard Pelavin, and Josh Tenenberg. *Formal Models of Plan Reasoning (forthcoming)*. Morgan-Kaufman, 1990.

[23] Bonnie Lynn Webber and Nils J. Nilsson, editors. *Readings in Artificial Intelligence*. Morgan Kaufmann, 1980.