Time revisited[1]

STEPHANIE A. MILLER AND LENHART K. SCHUBERT[2]
Department of Computing Science, University of Alberta, Edmonton, Alta., Canada T6G 2H1

Received January 4, 1989
Revision accepted March 19, 1990

Temporal reasoning is essential for many artificial intelligence applications. To date, most research has concentrated on temporal inference in isolation without considering the role it can play in a more general reasoning environment. This paper takes an efficient temporal reasoner and extends its inferential capabilities to handle both strict and nonstrict relations. The resulting temporal specialist is incorporated into a system intended for low-level reasoning in natural language understanding. The specialist assists the resolution-based theorem prover in function evaluation, literal evaluation, and generalized resolving and factoring. The combined system can do some proofs in just a few steps that would normally require many. An example from the fully operational hybrid system is included.

Key words: temporal reasoning, special inference methods, theory resolution, knowledge representation.

Le raisonnement temporel est essentiel dans de nombreuses applications relatives à l'intelligence artificielle. Jusqu'à présent, la plupart des travaux de recherche ont porté sur l'inférence temporelle dans l'isolation sans tenir compte du rôle qu'elle peut jouer dans un environnement de raisonnement plus général. Cet article traite d'un raisonneur temporel efficace dont les capacités inférentielles ont été augmentées afin de pouvoir traiter les relations strictes et non strictes. Le spécialiste temporel qui en résulte est intégré à un système conçu pour le raisonnement bas-niveau en langage naturel. Le spécialiste assiste le démonstrateur de théorèmes basé sur la résolution dans l'évaluation des fonctions, l'évaluation littérale et la fonction factorielle généralisée. Le système combiné peut vérifier des preuves en quelques étapes seulement au lieu d'un grand nombre. Un exemple d'un système hybride entièrement opérationnel est fourni.

Mots clés : raisonnement temporel, méthodes d'inférence spéciales, résolution de théories, représentation des connaissances.

[Traduit par la revue]

Comput. Intell. 6, 108–118 (1990)

1. Introduction

Given certain explicit relationships among a set of events or episodes, we would like to be able to infer additional relationships implicit in the ones given. For example, if the events are part of a narrative, they will frequently form sequences in which adjacent events are known to follow one another; for such a sequence, we would like to be able to infer, without much effort, that events earlier in the sequence precede later ones, regardless of the number of intervening events. In addition, if we are given information about the durations or absolute times of some of the events, we would like to be able to infer quantitative consequences of this information, such as minimum and maximum elapsed time between given events. These sorts of inferences are essential in several areas of artificial intelligence, including story understanding, causal reasoning, and planning (Allen 1984).

Since temporal orderings are transitive and durations cumulative, such inferences in a typical general theorem prover can be computationally expensive. To compensate for this, researchers have tried to develop special representations and efficient methods for temporal inference. However, much of this work concentrates on temporal inference in isolation, rather than on using such a mechanism in a more general environment.

In this paper, we start with an efficient temporal specialist based on Taugher and Schubert's model (Taugher 1983; Schubert *et al.* 1983, 1987[3]) which does temporal inference in isolation. This specialist is unusual in the way it exploits chains of events, such as are commonly found in narratives (or plans), so as to achieve constant-time determination of time order for many pairs of events. It does so without requiring all events to lie on chains, and without computing transitive closure. To further increase the effectiveness of the specialist, its capabilities are extended here to make it more complete and flexible; the major enhancement being the ability to handle both strict and nonstrict time ordering. Nonstrict ordering is often appropriate for the end of one event in relation to the beginning of the next in a narrative (where there may or may not be a delay between them), while strict ordering is often appropriate for the beginning of an event in relation to its end (when the event is of a type that cannot transpire instantaneously). This, as well as consistency and expressiveness considerations, required considerable expansion of the set of temporal predicates and argument patterns handled by the specialist.

We then incorporate the resulting specialist into a general inference system with a resolution-based core based on de Haan's theorem prover (de Haan and Schubert 1986), where it assists the theorem prover with function evaluation, literal evaluation, and generalized resolution and factoring. The temporal specialist bypasses the normal proof procedure for the operations it handles, and can cut out numerous proof steps that would otherwise be required. Temporal literal evaluation uses the specialist's timegraph representation to simplify assertions and resolvents generated by the theorem prover. Function evaluation simplifies a term by

---

[1]A shorter version of this paper appeared in the Proceedings of the Seventh Biennial Conference of the Canadian Society for the Computational Studies of Intelligence (CSCSI-88), Edmonton, Alberta.

[2]Current affiliation: Department of Computer Science, University of Rochester, Rochester, NY 14627, U.S.A.

[3]The Schubert *et al.* (1987) paper is a revision of the 1983 paper. Henceforth only the 1987 paper will be referred to.

evaluating it (for example, (*start-of-e*1) can be simplified to a constant, *e*1*start*). Generalized resolving and factoring make use of Stickel's partial theory resolution (Stickel 1983) to quickly determine incompatibility or subordination of one literal by another. This allows resolution and factoring to be done where they usually cannot. For example, if we have [*a before*-1 *b*] represented in the timegraph, we can factor [¯[*x during a*] *or* [*x during b*]] to ¯[*x during a*], even though the literals have different signs[4]. Similarly, if we know [*a before*-1 *b*], we can resolve [*x before a*] with [*x after b*] to the null clause, although the predicates are not identical and the signs are the same[5].

The hybrid system consisting of the resolution-based theorem prover and the temporal specialist has been implemented in Lucid Common Lisp and runs on a Sun 3/75. An example is included from the operation of the system, which is called ECONET[6]. This system and its descendent, ECOLOGIC (Schubert and Hwang 1989), are to the best of our knowledge the most powerful combinations of a general deductive mechanism with a temporal specialist built to date.

2. The temporal specialist

Most current research in temporal inference uses two basic types of representations — *time intervals* and *time points*. Time intervals represent a given time period of finite length. There are numerous simple relations (Allen 1983; Vilain and Kautz 1986) that can be defined between two intervals (*equal*, *before*, *meets* (i.e., immediately before), *during*, *starts*, *ends*, *overlaps*, and their inverses).

The other major representation used is time points. These are abstract "instants" of time and are assumed to be nondecomposable. Pairs of such time points can be used to represent intervals. Only a few ordering relations can exist between time points ($<$, $\leq$, $>$, $\geq$, $=$), and logical combinations of these can be used to define interval relations. Although we take for granted that sentences in a narrative typically involve implicit reference to events occupying *intervals* of time, we have found the time-point representation more convenient. Apart from being simpler, time-point relations are easily represented as graphs; a fact we have found very helpful in the design of efficient algorithms[7].

Various refinements of point-based representations have been proposed. For example, Ladkin (1986) allowed for *nonconvex* intervals, consisting of discontinuous segments (so that such "intervals" need not contain all "points" within). This gives rise to an extended algebra of interval operations and relations, including operations for forming nonconvex intervals from convex ones and vice versa. Leban *et al.* (1986) allowed for sequences of consecutive intervals called *calendars*, which could be combined recursively into *collections* and could be used to "dice" intervals, i.e., to segment them into units equal in size to the units of the calendar. They could also "slice" periodic subsequences out of collections, such as the fifth day of every week. Koomen (1989) develops an axiomatic theory of recurrence in which it is possible to reason about events repeated *n* times or until (or while) some conditions hold. Interesting as these refinements are, however, they fall largely outside the scope of current research on temporal specialists (including ours). To the extent that they are allowed for at all (e.g., by Koomen's TEMPOS), they are handled by traditional deductive methods — though they can, of course, benefit from rapid evaluation of simple temporal queries that arise in the course of a more complex temporal inference.

There is a trade-off between expressiveness and efficiency of temporal representations. In particular, the achievable efficiency depends on the extent to which *disjunctive* combinations of relations are allowed. Matters are further complicated by the fact that the effect of allowing disjunction depends on whether the representation is point-based or interval-based. Without disjunction, the two types of representation are equally expressive and allow computation of the closure of pairwise time-interval or time-point relations in $O(n^2)$ time, where n is the number of intervals or points. But if disjunctions of Allen's interval relations are allowed, the closure problem becomes NP-hard (Vilain and Kautz 1986). Disjunctions of point relations are less expressive but more tractable, allowing $O(n^3)$ closure. If arbitrary conjunctive/disjunctive combinations of constraints on sets of intervals or points are permitted, then, of course, the two representations are expressively equivalent, and closure is NP-hard in either case.

As a means of achieving acceptable efficiency in a system handling disjunctions of interval relations, Allen (1983) suggested that intervals be divided into groups, each subsumed under a *reference interval*. Koomen (1988, 1989) subsequently developed an algorithm for automatically organizing intervals into a hierarchy of groups, each group being within a reference interval. In practice, his TIMELOGIC system has yielded speedups in update efficiency (for adding a relationship) of around 2.5, compared to updates for an unorganized collection of interval relationships. Dean (1989) similarly uses multiple levels of representation (with time points at the lowest level) to achieve greater efficiency.

However, neither approach seems capable of providing a humanlike ability to accept new relationships, and to determine implicit relationships on the basis of those previously given, in constant or near-constant time (as appears to happen in narrative understanding). Beginning with Taugher (1983), our strategy has been to determine empirically what classes of temporal structures are important in practice, and to design the specialist so as to handle as large a class of the practically important relationships as possible without sacrificing efficiency. Any inferences that fall outside the

---

[4]Intuitively, this is because the known fact [*a during b*] *subsumes* the given disjunction.

[5]*Before*-1 means strictly before.

[6]The ECO-prefix stand for English conversation, reflecting our larger goals (see de Haan and Schubert 1986).

[7]Allen (1984) argues that time-point representations require arbitrary assumptions about whether or not a property P holds at a point of transition from P to $\neg P$ (where P is some time-dependent property). However, we claim that no such assumptions are needed. For instance, our axiomatic theory of "sleeping" can simply remain agnostic about whether, if Mary slept from midnight till noon, Mary was still asleep at noon or no longer asleep. All it would guarantee is that she was asleep at all moments of time (strictly) *between* midnight and noon (and similarly, awake at all moments of time strictly within some interval right after noon). Of course, as long as we are using a standard logic, it will remain provable that Mary was either asleep or not asleep at noon, but that is very different from saying that it will either be provable that she was asleep at noon or provable that she was not. Such "agnosticism" about beginnings and end points seems in accord with intuition, and in no way hinders our ability to draw qualitative and quantitative conclusions about event relations of practical interest.

capabilities of the specialist can still be handled by the general deductive mechanism, albeit much less efficiently.

In particular, detailed analyses by Taugher of time relationships taken from newspapers, a book of history, and fairytales suggested that the time structure of narrative texts consists almost entirely of nondisjunctive relationships, forming substantial chains of consecutive events, with some side chains and parallel chains, and with relatively few interconnections between chains. Our aim has therefore been to find very fast (near-constant time) methods of inferring ordering and duration information for such structures. Temporal disjunctions, conjunctions, negations, and inequalities (which are not handled directly by the temporal specialist) may be expressed using the general inference mechanism. The specialist serves to simplify these when possible, and can also assist by comparing literals in such clauses during resolution.

Among the axioms available to the general deduction algorithm (and "built into" our specialist) is the transitivity axiom

$$[[t_1 \leq t_2] \,\&\, [t_2 \leq t_3]] \Rightarrow [t_1 \leq t_3]$$

where universal quantifiers are suppressed. Furthermore, if either of the relations in the antecedent is strict, then so is the consequent. Some other basic properties of the time ordering are

$$[t_1 \leq t_2] \Leftrightarrow [t_1 < t_2] \text{ or } [t_1 = t_2]$$
$$[t_1 < t_2] \Rightarrow \neg[t_2 \leq t_1]$$
$$[[t_1 \leq t_2] \,\&\, [t_2 \leq t_1]] \Rightarrow [t_1 = t_2]$$

where the last property is a consequence of the previous properties. Additional axioms defining the meaning of the predicates used by our temporal specialist in terms of $<$, $\leq$, and $=$ are included in a later section.

All of these axioms are satisfied by the real numbers. In fact, for the purpose of reasoning about the quantifiable aspects of time (dates and durations), it is natural to identify time with numbers, so that standard arithmetical operators can be used. Thus, the relationship between durations and times satisfies

$$\text{duration}(t_1, t_2) = t_2 - t_1$$
$$t_1 \leq t_2 \leq t_3 \Rightarrow \text{duration}(t_1, t_3) = \text{duration}(t_1, t_2) + \text{duration}(t_2, t_3)$$

We also have equations relating bounds on times and durations (Taugher 1983, p. 71). Given two consecutive time points t_1 and t_2 with lower and upper time bounds l_1, u_1, l_2, and u_2, respectively, and lower and upper bounds l and u on the duration between t_1 and t_2 ($t_2 - t_1$), the following hold:

$$l_1 \leq t_1 \leq u_1$$
$$l_2 \leq t_2 \leq u_2$$
$$l \leq t_2 - t_1 \leq u$$
$$u_1 \leq u_2$$
$$l_1 \leq l_2$$
$$t_1 \leq t_2$$

For updating the bounds, the following consequences are used:

$$l_2 - u \leq t_1 \leq u_2 - l$$
$$l + l_1 \leq t_2 \leq u + u_1$$
$$l_2 - u_1 \leq t_2 - t_1 \leq u_2 - l_1$$

Proof that propagation of absolute time bounds establishes the best possible bounds throughout a Taugher and Schubert timegraph can be found in an appendix of Taugher (1983).

2.1. Taugher and Schubert's representation

Taugher and Schubert's representation (Taugher 1983; Schubert *et al.* 1987) is based on time points. Besides being able to determine relations between time points quickly (often in constant time), it can represent and reason with durations and absolute times. The representation uses a partial order graph, called a timegraph, whose nodes represent time points. Directed links between points indicate the given relation between the two points ($<$ or $\leq$ depending on interpretation). The timegraph is partitioned into *chains*, which are defined as sets of points that are all linearly ordered with respect to each other, with possible transitive arcs[8]. Links between points in the same chain are *in-chain* links; between points in different chains, *cross-chain* links. Each point has a *pseudo-time* (a number) associated with it, which is arbitrary except that it respects the ordering relationship between it and other points on the same chain. Chain and pseudo-time information are calculated when the point is first entered into the timegraph and are stored directly with the point. Determining the relationship between any two points in the same chain may be done in constant time simply by comparing their pseudo-times, rather than following the in-chain links.

To determine the relationship between points in different chains, a search is required, but only the cross-chain links need to be examined explicitly. A *metagraph* keeps track of the cross-chain links effectively by maintaining a *metanode* for each chain, and using the cross-chain links for links between metanodes. As in-chain checking can be done in constant time, a graph search is dependent on the number of cross-chain links rather than the total number of time points. Creation of all supporting graph structures (including the metagraph) requires $O(n + e)$ space and $O(n + e)$ time, where n is the number of time points and e is the number of relations between them. Note that a timegraph amounts to a set of atomic inequalities. Disjunctions such as $[[a \leq b]$ or $[c \leq d]]$ or inequalities such as $[a \neq b]$ cannot be represented. This restriction, along with the exploitation of time chains, is what enables the specialist to operate efficiently.

Figure 1 shows an example timegraph and metagraph. Following the cross-chain links, we can get that triangle1 is before square3, and square1 is before circle3, but no information about triangle2 and square3. To see how new chains are started, consider adding a point p between square1 and square3. Since we don't know the relationship between p and square2, we must start a new chain for p to maintain the linear ordering of the chains.

Furthermore, an absolute-time (date) minimum and maximum are stored with each time point. These are six-tuples of the form (*year month day hour minute second*), where each element may be numeric or symbolic (e.g., (1987 04 *a* 12 *b c*) represents some time at or after 12 a.m. and before 1 p.m. of some day in April 1987). Absolute-time maxima

[8] Schubert *et al.* (1987) clarify this point, as Taugher's thesis was not clear on it. Transitive arcs are in-chain links which do not make up the "backbone" of the chain; that is, there are intervening nodes and links. Figure 1 shows a transitive arc between circle2 and circle4.

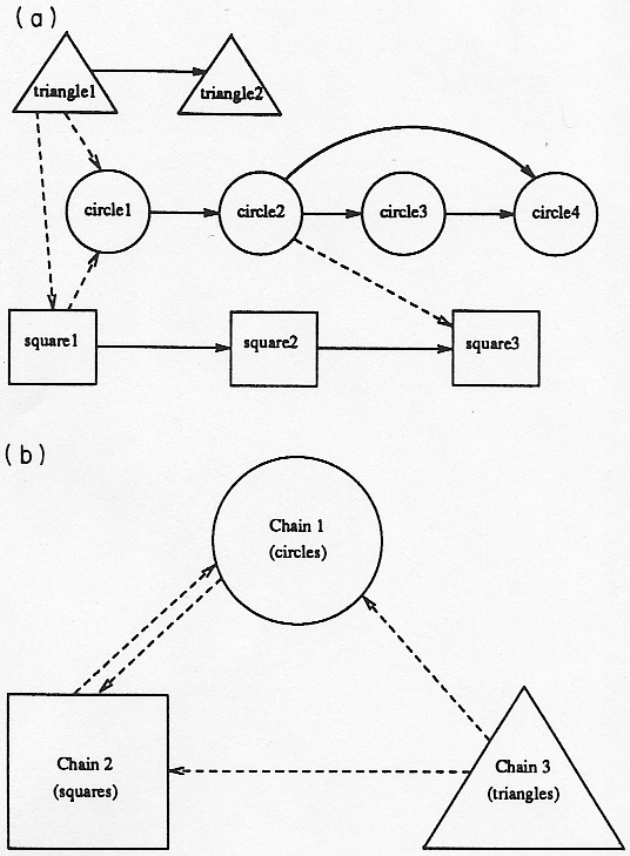

FIG. 1. Examples: (*a*) timegraph (each node represents a time point); (*b*) metagraph (each node represents a chain). (⟶) in-chain links; (- - ->) cross-chain links.

propagate back to points before the given point (in the chain or on other chains), and minima propagate forward. This ensures that each point has the best absolute-time information possible. Details may be found in Taugher (1983). Absolute-time comparisons can sometimes be used to get a relation in constant time between two points on different chains, avoiding a metagraph search. Duration minima and maxima (in seconds) are stored on the links between points. These may affect the absolute times around them, which are then propagated.

Insertion of a relation requires a constant amount of processing in most cases, except for propagation of absolute times. In the worst case, propagation may require going to every point in the graph, although this is highly unlikely. Occasionally, a chain may have to be renumbered, which involves all the points in a single chain. (This can only happen if temporal relations are inserted in radically nonchronological order.)

Intervals/events/episodes[9] are represented by their start and end time points. The intervals are kept in a separate table, which contains these end points. Interval relations are defined as combinations of time-point relations between the start and end points of the intervals.

[9]Interval, event, and episode are used interchangeably in this paper.

2.2. Enhancements to the Taugher and Schubert model

Although this time model is efficient and does most of the temporal reasoning we need, it requires several enhancements. First, confusion surrounding whether a point (*a*) could be inserted between two others (*b* and *c*) and remain on the same chain[10] was resolved by allowing this to take place only when *c* has the smallest pseudo-time of any in-chain descendant of *b* (i.e., is closest). Note that in this case the link from *b* to *c* will be a transitive arc (bypassing one intermediate node, namely *a*) of the resulting chain. Otherwise *a* must be placed on a new chain.

Second, the original representation can represent either strict relations ($<$, $>$, $=$) or nonstrict relations ($\leq$, $\geq$, $=$), depending on whether links between points are interpreted as $<$ or $\leq$, but not both. Our goal here was to extend the model to handle both, without losing the efficiency of the original. For cross-chain relations, a flag indicating strictness on the link itself is enough, since cross-chain links are explicitly examined during a search. For points within the same chain, however, examining each link for strictness would mean that we could no longer determine in-chain relationships in constant time. Attempts to handle this by adding another pseudo-time failed, because they either could not represent all possible combinations or were not constant time.

The successful method eventually found requires two extra numbers — a *maximum-pseudo* and a *minimum-pseudo*. The minimum-pseudo of a given point is the pseudo-time of the nearest predecessor on the chain that the point cannot be equal to; the maximum-pseudo, of the nearest successor on the chain that it cannot be equal to[11]. To determine the relation between two points, we first compare their pseudo-times to find out what order the points are in. Then we compare the pseudo-time of the first point with the range given by the minimum and the maximum of the second point (or vice versa). If it is properly within that range (i.e., greater than the minimum and less than the maximum), the relation given by the pseudo-times alone is nonstrict; otherwise, it is strict. Propagation of minima and maxima maintains strictness of relations throughout the chain. This propagation is similar to propagation of absolute times (minima propagate forward, maxima back), although the comparison at each propagation step may be shorter (only one number to compare instead of a possible six).

To see how this works, consider the following example. Suppose $a \leq b \leq c \leq d \leq e$, and then we assert $a < d$:

Point	*a*	*b*	*c*	*d*	*e*
Pseudo-time[12]	1	1000	2000	3000	4000
Minimum-pseudo	$-\infty$	$-\infty$	$-\infty$	1	1
Maximum-pseudo	3000	$+\infty$	$+\infty$	$+\infty$	$+\infty$

[10]A CMPUT 551 (Artificial Intelligence I) class at the University of Alberta discovered this in 1984 while reimplementing the algorithm as part of a project.

[11]This requires the addition of two additional pseudo-times to represent points before the beginning of the chain ($-\infty$) and beyond the end ($+\infty$). To make chain renumbering easier, pointers to the minimum and maximum points are kept, rather than their actual pseudo times.

[12]The pseudo-times are numbers generated by the system which reflect the ordering on the chain.

Note we have $a < d$, $a \leq b$, $b \leq c$, $a < e$, and so on, correctly. When comparing a and d, a's pseudo-time, 1, is less than d's (3000), indicating that a is before d in the time chain. Furthermore, a's pseudo-time does not fit properly within d's minimum and maximum (1, $+\infty$), so $a < d$. However, for c and e, c's pseudo-time (2000) is less than e's (4000), but now c's pseudo-time does fit properly within e's minimum and maximum (1 and $+\infty$), so $c \leq e$.

Now note what happens when we assert $c < e$:

Point	a	b	c	d	e
Pseudo-time	1	1000	2000	3000	4000
Minimum-pseudo	$-\infty$	$-\infty$	$-\infty$	1	2000
Maximum-pseudo	3000	4000	4000	$+\infty$	$+\infty$

Note that c's maximum-pseudo has changed so that it no longer includes e, and e's minimum-pseudo has changed so that it no longer includes c. Minimums propagate forward, and maximums back, so b's maximum was also changed to the same as c's. (If c cannot be greater than some point, neither can anything before c).

This method is flexible enough to handle any combination of relations, is easy to understand and implement, and maintains the constant-time in-chain checking.

To evaluate whether a given relation holds, or to determine the strongest relation constraining two points, we now need to consider that there may be numerous paths between the points, some of which may represent strict relations, and some nonstrict. To determine the relation between two points, one must continue finding paths until a strict path is found or all paths between the points have been found. Within each chain, the strictness of the relation between points is easily calculated in constant time (the range comparison) and used in the path determination. It is only where there are several possible cross-chain links leaving a point that there is a possibility for paths of different strictness. Using a modified search algorithm, the time required is still $O(m)$.

With the additional capabilities for handling strict and nonstrict ordering, we now need some way of telling the temporal specialist which ordering to use. This was one reason, among others, for some changes to the set of temporal predicates recognized by the specialist. In Taugher's thesis, the following predicates were used for entry of intervals/events:

a before b [*c*] ⇒ *a end* ≤ *b start*
(both within time frame *c* if specified)
a after b [*c*] ⇒ *a start* ≥ *b end*
(both within time frame *c* if specified)
a equal b ⇒ *a start* = *b start* and *a end* = *b end*
a during b [*c*] ⇒ if *c* specified, *a between b and c*;
otherwise *a within b*

During evaluation, some additional relations could be determined: *overlaps*, *overlapped-by*, *contains*, as well as some weaker ones: *starts-before*, *ends-before*, *starts-after*, *ends-after*, *starts-equal*, and *ends-equal*.

The predicates used in the temporal specialist implemented in this paper are essentially the same ones that Taugher used, with some exceptions. The weaker predicates have not been implemented. *During* is used only in the sense of "within," enabling the option of a timeframe argument. *Between* was introduced to handle the case where one argument is to be inserted "during" two others (see above). Except for *between*, all predicates now have the optional third argument representing a time frame (instead of just *before* and *after*). Usage of this argument leads to fewer chains being built in the timegraph, resulting in a more nearly optimal timegraph. All the predicates may be used for both entry (assertion) and evaluation for consistency.

The original predicates are now considered "stems" and may have one or two strictness values appended (stem [−strict1[−strict2]]). Strictness values are

Strictness Value	Meaning
−1	Strict (< or >)
−0	Meets (end points abut, i.e., are equal)
nil	Nonstrict (≤ or ≥)

In terms of the temporal ordering relations $<$, $\leq$, and $=$, the following axioms define the temporal predicates used by our specialist (arguments beginning with e represent intervals (events); with t, time points):

$[e_1$ PRED $e_2 e_3]$ ⇔ $[e_1$ PRED $e_2]$ &
PRED is one of: $[(start\text{-}of\ e_3) \leq (start\text{-}of\ e_1)]$ &
before, *after*, *during*, $[(end\text{-}of\ e_1) \leq (end\text{-}of\ e_3)]$ &
contains, *overlaps*, $[(start\text{-}of\ e_3) \leq (start\text{-}of\ e_2)]$ &
overlapped-by $[(end\text{-}of\ e_2) \leq [(end\text{-}of\ e_3)]$

$[e_1\ equal\ e_2]$ ⇔ $[(start\text{-}of\ e_1) = (start\text{-}of\ e_2)]$ &
$[(end\text{-}of\ e_1) = (end\text{-}of\ e_2)]$
$[t_1\ equal\ t_2]$ ⇔ $[t_1 = t_2]$

$[e_1\ before\ e_2]$ ⇔ $[(end\text{-}of\ e_1) \leq (start\text{-}of\ e_2)]$
$[e_1\ before\text{-}1\ e_2]$ ⇔ $[(end\text{-}of\ e_1) < (start\text{-}of\ e_2)]$
$[e_1\ before\text{-}0\ e_2]$ ⇔ $[(end\text{-}of\ e_1) = (start\text{-}of\ e_2)]$
$[t_1\ before\ t_2]$ ⇔ $[t_1 \leq t_2]$
$[t_1\ before\text{-}1\ t_2]$ ⇔ $[t_1 < t_2]$
$[t_1\ before\text{-}0\ t_2]$ ⇔ $[t_1 = t_2]$

$[e_1\ after\ e_2]$ ⇔ $[(end\text{-}of\ e_2) \leq (start\text{-}of\ e_1)]$
$[e_1\ after\text{-}1\ e_2]$ ⇔ $[(end\text{-}of\ e_2) < (start\text{-}of\ e_1)]$
$[e_1\ after\text{-}0\ e_2]$ ⇔ $[(end\text{-}of\ e_2) = (start\text{-}of\ e_1)]$
$[t_1\ after\ t_2]$ ⇔ $[t_2 \leq t_1]$
$[t_1\ after\text{-}1\ t_2]$ ⇔ $[t_2 < t_1]$
$[t_1\ after\text{-}0\ t_2]$ ⇔ $[t_2 = t_1]$

$[e_1\ during\ e_2]$ ⇔ $[(start\text{-}of\ e_2) \leq (start\text{-}of\ e_1)]$ &
$[(end\text{-}of\ e_1) \leq (end\text{-}of\ e_2)]$
$[e_1\ during\text{-}1\ e_2]$ ⇔ $[(start\text{-}of\ e_2) < (start\text{-}of\ e_1)]$ &
$[(end\text{-}of\ e_1) \leq (end\text{-}of\ e_2)]$
$[e_1\ during\text{-}0\ e_2]$ ⇔ $[(start\text{-}of\ e_2) = (start\text{-}of\ e_1)]$ &
$[(end\text{-}of\ e_1) \leq (end\text{-}of\ e_2)]$
$[e_1\ during\text{-}1\text{-}1\ e_2]$ ⇔ $[(start\text{-}of\ e_2) < (start\text{-}of\ e_1)]$ &
$[(end\text{-}of\ e_1) < (end\text{-}of\ e_2)]$
$[e_1\ during\text{-}0\text{-}1\ e_2]$ ⇔ $[(start\text{-}of\ e_2) = (start\text{-}of\ e_1)]$ &
$[(end\text{-}of\ e_1) < (end\text{-}of\ e_2)]$
$[e_1\ during\text{—}1\ e_2]$ ⇔ $[(start\text{-}of\ e_2) \leq (start\text{-}of\ e_1)]$ &
$[(end\text{-}of\ e_1) < (end\text{-}of\ e_2)]$
$[e_1\ during\text{-}1\text{-}0\ e_2]$ ⇔ $[(start\text{-}of\ e_2) < (start\text{-}of\ e_1)]$ &
$[(end\text{-}of\ e_1) = (end\text{-}of\ e_2)]$
$[e_1\ during\text{-}0\text{-}0\ e_2]$ ⇔ $[(start\text{-}of\ e_2) = (start\text{-}of\ e_1)]$ &
$[(end\text{-}of\ e_1) = (end\text{-}of\ e_2)]$

```
==> ?(E x_episode (W alive x) & ((x after all-talk) or (x after-1 (end-of eat-goodies))))
Entering disproof clauses:
(W ALIVE SCON-380) (depth 1)
((SCON-380 AFTER ALL-TALK) or (SCON-380 AFTER-1 (END-OF EAT-GOODIES))) (depth 1)

  Time Specialist: END-OF (EAT-GOODIES) evaluated to EAT-GOODIESEND
((SCON-380 AFTER ALL-TALK) or (SCON-380 AFTER-1 EAT-GOODIESEND)) (depth 1)

  Time Specialist: Trying to factor (SCON-380 AFTER ALL-TALK) and (SCON-380 AFTER-1 EAT-GOODIESEND)
  Time Specialist: Factored to (SCON-380 AFTER ALL-TALK)
(SCON-380 AFTER ALL-TALK) (depth 1)

  Resolved (W ALIVE SCON-380) in the disproof clause
    (W ALIVE SCON-380)
    against (~ U-VAR-2 ALIVE EPISODE-VAR-1) in ((~ U-VAR-1 KILL U-VAR-2 EPISODE-VAR-2)
      or (~ EPISODE-VAR-1 AFTER EPISODE-VAR-2) or (~ U-VAR-2 ALIVE EPISODE-VAR-1))
    yielding ...
((~ SCON-380 AFTER EPISODE-VAR-1) or (~ U-VAR-1 KILL W EPISODE-VAR-1)) (depth 2)

  Resolved (~ U-VAR-1 KILL W EPISODE-VAR-1) in the disproof clause
    ((~ SCON-380 AFTER EPISODE-VAR-1) or (~ U-VAR-1 KILL W EPISODE-VAR-1))
    against (WOODCUTTER KILL W WOLF-DEMISE) in (WOODCUTTER KILL W WOLF-DEMISE)
    yielding ...
(~ SCON-380 AFTER WOLF-DEMISE) (depth 3)

  Time Specialist: Trying to resolve (~ SCON-380 AFTER WOLF-DEMISE) against (SCON-380 AFTER ALL-TALK)
  Time Specialist: Resolved with residues null
  Resolved (~ SCON-380 AFTER WOLF-DEMISE) in the disproof clause
    (~ SCON-380 AFTER WOLF-DEMISE)
    against (SCON-380 AFTER ALL-TALK) in (SCON-380 AFTER ALL-TALK)
    yielding the null clause.
NO
```

FIG. 2. Example of ECONET in operation.

$$[e_1\ during\text{—}0\ e_2] \Leftrightarrow [(start\text{-}of\ e_2) \le (start\text{-}of\ e_1)]\ \&\ [(end\text{-}of\ e_1) = (end\text{-}of\ e_2)]$$

$$[e_1\ contains\ e_2] \Leftrightarrow [(start\text{-}of\ e_1) \le (start\text{-}of\ e_2)]\ \&\ [(end\text{-}of\ e_2) \le (end\text{-}of\ e_1)]$$

$$[e_1\ overlaps\ e_2] \Leftrightarrow [(start\text{-}of\ e_1) \le (start\text{-}of\ e_2)]\ \&\ [(end\text{-}of\ e_1) \le (end\text{-}of\ e_2)]$$

$$[e_1\ overlapped\text{-}by\ e_2] \Leftrightarrow [(start\text{-}of\ e_2) \le (start\text{-}of\ e_1)]\ \&\ [(end\text{-}of\ e_2) \le (end\text{-}of\ e_1)]$$

$$[e_1\ between\ e_2 e_3] \Leftrightarrow [(end\text{-}of\ e_2) \le (start\text{-}\ of\ e_1)]\ \&\ [(end\text{-}of\ e_1) \le (start\text{-}of\ e_3)]$$

Note that this assumes the existence of functions *start-of*, which returns the time point corresponding to the start of an interval, and *end-of*, which returns the time point corresponding to the end of an interval. For brevity, not all predicates are represented here: *contains*, *overlaps*, *overlapped-by*, and *between* have stem endings like *during*. Axioms for these predicates with strictness indicators can be determined by noting the effect of the strictness indicators on the *during* axiom.

The new specialist is quite flexible, accepting episodes, time points, or absolute times in any combination as arguments for the predicates (except the timeframe, which must be an episode), where Taugher's programs were quite rigid in the argument patterns accepted. On assertion, if an argument is an absolute time instead of a named time point or episode, the appropriate absolute-time bound of the other argument is updated. For example,

(*a before* (*date* '1987 '04 '01 '00 '00 '00))

would update the upper bound of *a*'s absolute time (the maximum). For evaluation, the appropriate bound is compared against the absolute time, or two absolute times may be compared.

In addition, some new predicates have been introduced to handle durations: *at-most-before*, *at-most-after*, *at-least-before*, *at-least-after*, *exactly-before*, and *exactly-after*. These take three arguments, of which the first two may be events or time points (no absolute times) and the third denotes the duration between the two in seconds. *At-most* implies maximum duration, *at-least-* implies minimum duration, and *exactly-* involves both. To determine the duration between any two points, an exhaustive search must be done between those points, following both in-chain and cross-chain links, to get the best duration bounds (the greatest minimum and the smallest maximum). Duration information on arcs and implicit in absolute times is used. Details are in Miller (1988).

3. Use of the temporal specialist in a more general environment

The main system for representing knowledge and making inferences uses a resolution-based theorem prover featuring automatic classification of propositions, topical access of clauses for resolution, and type inheritance through a type hierarchy (de Haan and Schubert 1986). This system is designed to handle large, diverse bodies of knowledge efficiently. Although the topical access and type hierarchy improve the performance of the theorem prover dramatically, it can still suffer from the computational explosions to which all theorem provers are prone. This is especially true when working with the transitive relations involved in temporal inference, so that it is desirable to delegate temporal inference to the temporal specialist as far as possible.

Interesting proofs we can ask the system to do involve "mixed" inference (i.e., involving both temporal relations and others). For example, in the story of Little Red Riding Hood, deciding whether the wolf was alive when everyone was eating the goodies in the basket requires using knowledge that one can only be alive before one is killed, and temporal inference to determine that the episode of eating the goodies came after the episode of killing the wolf. Other examples of the uses of such "mixed" reasoning in planning and problem-solving can be found in Allen and Kautz (1985).

Further extensions were necessary to integrate the specialist into the main system. The main system organizes modal propositions into subnets, with one subset for each person's mental world. As the subnets may contain contradictory information, the temporal specialist also maintains a separate timegraph (and metagraph) for each subnet[13]. Simple temporal evaluations may be done within the subnets, although at this stage no real modal inference is done by either the temporal specialist or the main theorem prover. Upon completion of proofs by contradiction, the main system retracts all changes made. To remain consistent, the temporal specialist had to be extended to have this capability as well, which is used extensively in generalized resolving and factoring. Changes are stored at an atomic level and retracted in reverse order.

Schubert *et al.* (1987) suggest that a specialist can assist a theorem prover in literal evaluation and generalized resolution and factoring. In addition, the specialist can be used to simplify literals by evaluating functional terms. Assertions must also be entered into the specialist representation for use in future evaluations (only positive — non-negated — literals are entered). Most of the requirements for simple literal evaluation and entry have already been discussed. Generalized resolution and factoring make use of both the entry and evaluation phases to check for resolving or factoring actions. Although during the entry and evaluation phases, only literals with constant arguments are considered, during generalized resolution, this is extended to include variables as well (so one might temporarily get a time point for the variable x in the timegraph).

[13]This assumes that all temporal propositions within a mental world are consistent.

Figure 2 shows an example of the system in operation[14], trying to answer a question about the story of Little Red Riding Hood, similar to the one mentioned earlier. The question the system tries to answer is

Was the wolf alive at some time after everyone talked or after every one ate the goodies?[15]

which we translate to[16]

?[*E x__episode* [*W alive x*] & [[*x after all-talk*]
or [*x after*-1 (*end-of eat goodies*)]]]

In this translation, it is taken for granted that there is a noun phrase referent determination process which has selected the episode here called *all-talk* as the referent of *everyone talked*, and *eat-goodies* for *everyone ate the goodies*. Similarly, *W* is the individual referred to as *the wolf*. It is sometimes possible to handle this with existential quantification (e.g., [*E x__episode* [*everyone eat goodies x*] & [*x* ...]]), but this only works satisfactorily when a "yes" answer is expected. If a "no" answer is expected, the system currently answers "unknown," as there may be events fitting that description it doesn't know about (default reasoning would be required otherwise).

The example in Fig. 2 required about 57.5 s — just under a minute. This was with a knowledge base consisting of about 150 propositions (which are normalized into over 450 clauses), some for general knowledge and some for the simplified version of the Little Red Riding Hood story we use for testing (it consists of over 70 propositions, 20 of which temporally relate episodes in the story). The number of steps required for the proof was small, although the operations accelerated by the specialist would normally have required many applications of temporal axioms.

Inferences made by the specialist are essentially applications of axioms of time theory. For example, the ordering inferences are based on the transitivity axiom stated earlier. As we shall see shortly, generalized resolution and factoring embody some other useful axioms. Because the inferences are basically resolution inferences with the temporal axioms, the specialist inherits the soundness of resolution.

Completeness could be also proven for certain classes of premises for this specialist; for example, it is clearly complete for $\leq$ and $<$ for arbitrary sets of ground inequalities.

[14]This example shows actual output of the system, edited for clarity and brevity. Bold print is user input; the rest, system output. The existentially quantified variable x in the question has been converted to the skolem constant SCON-380 used in the proof. The timegraph contained, among other relations, that *all-talk* is during *eat-goodies*, and that *eat-goodies* is after *wolf-demise*, the episode corresponding to the woodcutter killing the wolf.

[15]Although somewhat awkard, this question illustrates all the areas of temporal specialist assistance.

[16]The question is phrased using *after*-1, and specifying *end-of* to show that the system can actually handle predicates of different strictness, and combinations of time points and events. The questions could just as easily have been phrased as [*E x__episode* [*W alive x*] & [[*x after all-talk*] or [*x after eat-goodies*]]]. Also note that a sort tag (*__episode*) is attached to some terms; this is used to differentiate terms so that the temporal specialist is called only when appropriate.

These ordering inferences are the main function of the temporal specialist. In outline, the proof is as follows. The algorithm confirms $t_0 \leq t_n$ for any two of the given time points t_0 and t_n, just in case there is a path from t_0 to t_n. By the transitivity axiom, this inference is sound. Moreover, suppose that this procedure is not complete, so that the ground inequalities represented by the graph along with the time axioms logically entail $t_0 \leq t_n$, but there is no path from t_0 to t_n. But it is an elementary property of acyclic directed graphs that for any two vertices not connected by a directed path, one can insert an edge from either of these vertices to the other, without creating any cycles. Since any acyclic graph is consistent with the time axioms (e.g., there is always a real number interpretation of the vertices satisfying the ordering constraints, even when all of them are interpreted strictly), it follows that $t_n < t_0$ can be consistently added to the premises, and hence $t_0 \leq t_n$ cannot be a consequence.

Furthermore, the algorithm confirms $t_0 < t_n$ just in case there is a path from t_0 to t_n containing a strict ordering link. Again soundness follows from (the strict version of) the transitivity axiom. If there is no path from t_0 to t_n, completeness follows as above. If there are paths, but they consist entirely of $\leq$ links, then all these paths can be collapsed into a single vertex without creating any cycles in the timegraph as a whole (again a property of acyclic graphs). Hence, t_0 can be interpreted as equal to t_n without inconsistency (e.g., using real numbers as a model), and so $t_0 < t_n$ cannot be a consequence.

We will not attempt to establish any further completeness claims here, as this is not the emphasis of this research. The objective is to *accelerate* inference in cases that frequently arise in practice (such as transitive inferencing across ground inequalities), and for this objective, the issue of completeness with respect to certain restricted classes of premises is somewhat irrelevant.

In the next sections, we gloss over the interface mechanism between the theorem prover and the temporal specialist, and concentrate on the operation of the temporal specialist. Details on the interface itself may be found in Miller and Schubert (1988*b*) and Miller (1988).

3.1. Function evaluation

During literal simplification, the temporal specialist may evaluate temporal functional terms to a more usable entity than the original term. The functions most commonly used are *start-of* and *end-of*, which return the time point for the start and end of an episode, respectively, and *date*, which returns an absolute-time representation recognized by the temporal specialist, given arguments for the year, month, day, hour, minute, and second (not all of which will necessarily be numbers — some may be symbols representing unknown or inexact values).

3.2. Literal evaluation

When evaluating whether a literal is true, one or more of the evaluation techniques already discussed are used. If the predicate is a duration predicate (e.g., *at-most-before*), the duration is calculated and compared to the one given. For other predicates, if any arguments are absolute times, an absolute-time comparison is done. Otherwise, the question is split into several time-point-evaluation questions, each of which uses the following:

1. If the points are on the same chain, a pseudo-time comparison is done.
2. For points on different chains, if both have absolute times associated with them, the absolute times are compared.
3. If this fails, a metagraph search, as described earlier, is done. If a path cannot be found from the first point to the second, a search is done to find a path from the second to the first.

The example shown in Fig. 2 does not overtly show incidences of literal evaluation, but it was used during the generalized factoring and resolving steps there.

3.3. Generalized resolution and factoring

Schubert *et al.* (1987) show some examples of when generalized resolving and factoring can be done by a temporal specialist. Although some of the inferences made for generalized resolution and factoring are not as "obvious" as simple evaluations are, they do reflect the properties of temporal relations mentioned earlier, such as transitivity. Some examples of generalized resolving (Schubert *et al.* 1987):

$$[t \leq t'] \text{ against } [t' \leq t]$$

gives a residue[17] of $[t = t']$. This inference is equivalent to two ordinary resolution steps based on the antisymmetry axiom

$$[t_1 \leq t_2] \,\&\, [t_2 \leq t_1] \Rightarrow [t_1 = t_2]$$

Another example, using additional information from the timegraph:

$$\neg[c \leq t] \text{ against } [c' \leq t]$$

along with timegraph information $[c \leq c']$, gives the null clause. This can be determined by resolution inferences with the transitivity axiom, $[c \leq c']$ and $[c' \leq t]$, giving $[c \leq t]$, which directly contradicts $\neg[c \leq t]$, giving the null clause. An example of generalized factoring, using timegraph information:

$$[c \leq t] \text{ or } [c' \leq t]$$

again with timegraph knowledge that $[c \leq c']$, gives generalized factor $[c \leq t]$, which can be obtained in two ordinary resolution steps and one ordinary factoring step by applying the transitivity axiom. (However, note that the preceding examples of generalized resolution and factoring may in a sense condense arbitrarily many ordinary resolution steps, since the relationship between c and c' may be a consequence of arbitrarily many facts stored in the timegraph.)

There are many more cases where such operations can be useful, dependent mainly on information already asserted in the timegraph. Determining whether two literals are resolvable or factorable involves similar methods, so they will be described together. The temporal specialist determines posible unifications that may lead to resolving or fac-

[17] A residue (from Stickel's partial theory resolution (Stickel 1983)) is a literal (or set of literals) whose negation would make the two literals incompatible (resolvable in one or more steps to the null clause).

toring actions, and uses the timegraph as a medium in which to compare the literals after substitution.

When resolving or factoring, unification of the arguments of the two literals is required. Since the predicates in the two literals are not necessarily identical, there is no restriction that the arguments be unified in the typical order (i.e., first from literal1 with first from literal2, and so on). The temporal specialist tries all possible unifications, testing after each to see if there is a resolving or factoring action that can be taken. Note that the two literals do *not* need to have the same number of arguments[18]. For example,

Literals	Unifications
(x *after* y)	(x/e_1, y/e_2), (x/e_1, y/e_3),
vs.	(x/e_2, y/e_1), (x/e_2, y/e_3),
(e_1 *between* e_2 e_3)	(x/e_3, y/e_1), (x/e_3, y/e_2)

These can be obtained by using the definitions of the predicates and axioms about temporal predicates. Using the definitions, we get

$$[(\textit{start-of}\ x) \geq (\textit{end-of}\ y)]$$

vs.

$$[[(\textit{start-of}\ e_1) \geq (\textit{end-of}\ e_2)] \text{ and } [(\textit{end-of}\ e_1) \leq (\textit{start-of}\ e_3)]]$$

The unification (x/e_1, y/e_2) is available by unifying the first literal (the one with x and y) with the first literal in the conjunction. The second literal in the conjunction can be written as

$$[(\textit{start-of}\ e_3) \geq (\textit{end-of}\ e_1)]$$

using the axiom

$$(AxAy\ [x \leq y] \Rightarrow [y \geq x])$$

which unifies in the traditional way against the first literal, giving us unification (x/e_3, y/e_1). Applying the axiom

$$(Ax[(\textit{start-of}\ x) \leq (\textit{end-of}\ x)])$$

plus transitivity, gives us the relations

$$(\textit{start-of}\ e_2) \leq (\textit{end-of}\ e_2) \leq (\textit{start-of}\ e_1) \leq (\textit{end-of}\ e_1) \leq (\textit{start-of}\ e_3) \leq (\textit{end-of}\ e_3)$$

From these inferences, $[(\textit{end-of}\ e_2) \leq (\textit{start-of}\ e_3)]$ will give us unification (x/e_3, y/e_2). The remaining three unifications can be obtained by applying the axiom

$$(AxAy\ [x \geq y] \Rightarrow \neg[y < x])$$

to $[(\textit{start-of}\ x) \geq (\textit{end-of}\ y)]$, yielding

$$\neg[(\textit{start-of}\ x) < (\textit{end-of}\ y)]$$

By separating the disjunctions into $<$ and $=$ literals, for example,

$$[[(\textit{start-of}\ e_2) < (\textit{end-of}\ e_1)] \text{ or } [(\textit{start-of}\ e_2) = (\textit{end-of}\ e_1)]]$$

and unifying against the negated literal, we get the final three unifications, (x/e_2, y/e_1), (x/e_1, y/e_3), and (x/e_2, y/e_3).

For each unification, we try to enter one literal into the timegraph and compare the other one to it within the timegraph. Use of the timegraph allows comparisons against the newly entered literal (what we started out to do), as well as other temporal relations asserted earlier (which enables the temporal specialist to shortcut the proof so drastically). During a generalized resolving attempt, we are looking for a unification that makes the two literals incompatible, or incompatible with the negation of a residue. During a generalized factoring attempt, we look for a unification that makes one literal unnecessary. Details of the algorithm may be found in Miller (1988).

4. Conclusions

We have incorporated an efficient temporal reasoner into a general reasoning environment, handling most of the temporal inferences needed for story understanding. As discussed below, other applications are expected as well.

The extensions to Taugher and Schubert's temporal specialist itself enabled it to handle more temporal inferences, including reasoning with both strict and nonstrict relations, at modest computational expense. The specialist accelerates the main theorem prover by performing literal evaluation and generalized resolution and factoring, as well as simplifying literals by functional term evaluation. The resulting hybrid can do some proofs that would normally require numerous steps in just a few. Although the temporal specialist slows down single steps of the system (because of the greater complexity of tests for resolvability or factorability), the drastic reduction in the lengths of proofs involving time relationships allows many previously infeasible deductions to be completed in reasonable time.

One drawback in such a system is that some of the inferences are hidden within the temporal specialist and are thus "invisible." However, since the type of temporal inference performed by the specialist is well understood and almost "obvious," justification is not essential, provided bugs and "holes" have been eliminated.

Since the completion of the temporal specialist, several other specialists have been added to the system (namely, a number/arithmetic specialist, a color specialist, and a set/list specialist), using the same interface method. Effective, uniform communication between specialists has also been added. The interface used for this hybrid in described in Miller (1988) and Miller and Schubert (1988*b*).

As a general reasoning system with a fully integrated temporal specialist, our system appears to be something of a rarity. For example, the lengthy bibliography by Frisch and Scherl (1988) on hybrid systems has almost no entries on systems of this type. Perhaps the most similar system is Koomen's TEMPOS (Koomen 1989), which is an extension of TIMELOGIC layered on top of the Rhet (or rhetorical) system of Allen and Miller (1988). (Rhet is a Horn-clause theorem prover with taxonomic and equality specialists, besides the temporal one. It is the successor to the HORNE

[18]The only restriction on the unifications is that the timeframe argument, if present, not be unified with anything. That argument is used on entry only to generate a more nearly optimal timegraph and is not essential for later reasoning; the relation given by the first two arguments with the predicate is the important one. This means fewer unifications to be tried for factoring or resolution, and therefore these attempts will be faster.

system of Allen *et al.* (1984).) The tasks carried out by TIMELOGIC for TEMPOS are mainly of the type, "Find all instances of temporal relation $x(y, z)$," where y and z may be interval constants or variables. Naturally, verification of particular relationships is a special case. The main difference from ECONET is that TEMPOS uses closure operations to maintain time relations in rapidly retrievable form, with considerable costs in update time and storage, while the ECONET time specialist obtains near-constant time operation without closure. This, of course, holds only for the kinds of story-based data we have focused on. TIMELOGIC is certainly more expressive, in its handling of disjunctions, retrieval requests with variables, and recurrence. ECONET handles disjunctions and variables through the general reasoner, and it remains unclear whether in the applications that were of interest to Koomen it handles them more effectively or less effectively than TEMPOS. Our time specialist offers more general types of assistance to ECONET in its pursuit of refutation proofs, through generalized resolving and factoring. These are dependent on efficient assertion and retraction of time information (since clauses generated in a refutation proof may be false and have to be retracted), and thus would be very hard to implement in a closure-based system.

Dean's system TMM (Dean 1989) is expressively nearly equivalent to ours, allowing a (nondisjunctive) set of constraints in the form of dates and bounds on temporal distances. One difference is that our system allows for (partially or completely) symbolic dates; on the other hand, TMM does a certain amount of cause–effect reasoning, based on rules connecting the occurrence of one type of event with another. (In our approach, cause–effect reasoning is handled by the general reasoner, which transmits times associated with inferred causes or effects to the time specialist.) In its organization, TMM is much more similar to Koomen's system than to ours; i.e., it uses a hierarchical organization to limit the complexity of constraint propagation (partial closure), and its main function is rapid retrieval of sets of events satisfying given temporal constraints. Time relationships are determined by tree search and heuristic search, with no use of anything analogous to our constant-time "pseudo-time" comparisons. It is described as Prolog-compatible, and as such can presumably be interfaced to a more general reasoning system in much the same way as TEMPOS, though no details are given. Like TEMPOS, it is unlikely to lend itself to more general support of a theorem prover through generalized resolving and factoring.

Plaisted (1986) describes a method for combining a temporal logic with specialized theories, but it does not easily extend to quantified theories (while our hybrid includes a general method which does handle quantifiers).

Our temporal specialist was mainly inspired by the requirements of natural language understanding. One interesting direction for further research concerns possible applications to other domains, such as planning. In planning (as opposed to story understanding), temporal constraints may accumulate in a highly nonchronological order. In such a situation, the timegraphs built by the temporal specialist may be far from optimal, containing far more chains and cross-links than necessary, i.e., large m. Therefore, it would be appropriate to perform occasional "optimizations" which restructure the graph to reduce m. This could include "merging" chains where the end of one is before the beginning of the other — either because of an actual link from one to the other or because the absolute-time bounds of the respective end and beginning don't overlap.

Another issue relevant to the planning domain concerns the *practical* utility of disjunction. Allen and Koomen (1983) provide interesting examples of how disjunctive domain and task constraints can be used in the development of plans. In these examples, constraint propagation suffices to determine plans to achieve given disjunctive goals, which would otherwise require STRIPS-like search. On the other hand, there is a very substantial computational price paid for general disjunction-handling, and as stated above, it remains unclear whether shifting disjunction-handling from the general reasoner to the specialist yields significant overall improvements (especially if the general reasoner — like that of ECOSYSTEM — employs a better proof strategy than simple backtrack search). Dean (1989) states that his representation emerged from a series of attempts to provide a general, practical tool for temporal information management, with apparent emphasis on planning and scheduling problems. Since this quest did not lead him to a disjunctive representation, but rather one expressively similar to ours, there seems to be cause for optimism about planning applications of our approach. Also, planning requires the generation of alternative time graphs even when disjunctive relationships are allowed (Koomen 1988), and this is likely to be easier for our representation than for Koomen's or Dean's, because of its simple, spatially economic structure and the speed of updates. However, a definite judgement must await further research, especially into the question of "optimizing" timegraphs. In the meantime, our temporal specialist is serving as a very stable and useful component of our successive inference systems (ECONET and ECOLOGIC) aimed at story understanding.

Acknowledgements

We are grateful to Hans Koomen for helpful comments on the relation between temporal reasoning and planning, and to the referee for useful suggestions for improvements. The research was supported by Boeing Co. (under Purchase Contract W-278258), the Natural Sciences and Engineering Research Council of Canada (under Operating Grant A8818 to the second author), and a Province of Alberta scholarship (to the first author).

ALLEN, J.F. 1983. Maintaining knowledge about temporal intervals. Communications of the ACM, **26**: 832–843.

———1984. Towards a general theory of action and time. Artificial Intelligence, **23**: 123–154.

ALLEN, J.F., and KAUTZ, H.A. 1985. A model for naive temporal reasoning. *In* Formal theories of the commonsense world. *Edited by* J.R. Hobbs and R.C. Moore. Ablex, Norwood, NJ. pp. 251–268.

ALLEN, J.F., and KOOMEN, J.A. 1983. Planning using a temporal world model. Proceedings of the 8th International Joint Conference on Artificial Intelligence, Karlsruhe, West Germany, pp. 741–747.

ALLEN, J.F., and MILLER, B.W. 1988. The rhetorical knowledge representation system: a user's manual. Technical Report 238, Department of Computer Science, University of Rochester, Rochester, NY. (Rerevised March 1989).

ALLEN, J.F., GIULIANO, M., and FRISCH, A.M. 1984. The HORNE Reasoning System. Technical Report 126, Computer Science Department, University of Rochester, Rochester, NY.

DEAN, T. 1989. Using temporal hierarchies to efficiently maintain large temporal databases. Journal of the Association for Computing Machinery, **36**: 687–718.

DE HAAN, J., and SCHUBERT, L.K. 1986. Inference in a topically organized semantic net. Proceedings of the 5th National Conference on Artificial Intelligence, Philadelphia, PA, pp. 334–338.

FRISCH, A., and SCHERL, R. 1988. Bibliography on hybrid reasoning. Proceedings of the Workshop on Principles of Hybrid Reasoning, St. Paul, MN, pp. 1–8.

KOOMEN, J.A. 1988. The TIMELOGIC temporal reasoning system. Technical Report 231, Department of Computer Science, University of Rochester, Rochester, NY.

———1989. Localizing temporal constraint propagation. Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning, Toronto, Ont., pp. 198–202.

LADKIN, P. 1986. A taxonomy of interval relations. Proceedings of the 5th National Conference on Artificial Intelligence, Philadelphia, PA, pp. 360–366.

LEBAN, B., MCDONALD, D.D., and FOSTER, D.R. 1986. A representation for collections of temporal intervals. Proceedings of the 5th National Conference on Artificial Intelligence, Philadelphia, PA, pp. 367–371.

MILLER, S.A. 1988. Time revisited. M.Sc. thesis, Department of Computing Science, University of Alberta, Edmonton, Alta.

MILLER, S.A., and SCHUBERT, L.K. 1988*a*. Time revisited. Proceedings of the 7th Biennial Conference of the Canadian Society for Computational Studies of Intelligence (CSCSI'88), Edmonton, Alta., pp. 39–45.

———1988*b*. Using specialists to accelerate general reasoning. Proceedings of the 7th National Conference on Artificial Intelligence, St. Paul, MN, pp. 161–165.

PLAISTED, D.A. 1986. A decision procedure for combinations of propositional temporal logic and other specialized theories. Journal of Automated Reasoning, **2**: 171–190.

SCHUBERT, L.K., and HWANG, C.H. 1989. An episodic knowledge representation for narrative texts. Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning (KR'89), Toronto, Ont., pp. 444–458.

SCHUBERT, L.K., PAPALASKARIS, M.A., and TAUGHER, J. 1983. Determining type, part, color, and time relationships, Computer, **16**: 53–60.

———1987. Accelerating deductive inference: special methods for taxonomies, colours and times. *In* The knowledge frontier. *Edited by* N. Cercone and G. McCalla. Springer-Verlag, New York, NY. pp. 187–220.

STICKEL, M.E. 1983. Theory resolution: building in nonequational theories. Proceedings of the 3rd National Conference on Artificial Intelligence, Washington, DC, pp. 391–397.

TAUGHER, J. 1983. An efficient representation for time information. M.Sc. thesis, Department of Computing Science, University of Alberta, Edmonton, Alta.

VILAIN, M., and KAUTZ, H. 1986. Constraint propagation algorithms for temporal reasoning. Proceedings of the 5th National Conference on Artificial Intelligence, Philadelphia, PA, pp. 377–382.