From TimeML to $\mathcal{TPL}^*$

Ian Pratt-Hartmann[1]

School of Computer Science,
Manchester University,
Manchester, UK
ipratt@cs.man.ac.uk

Abstract. This paper describes a subset of the temporal mark-up language TimeML, and explains its relation to various formalisms found in the literature on interval temporal logic. The subset of TimeML we describe can be viewed as an interval temporal logic with a tractable satisfiability problem, but very limited expressive power. Most crucially, that logic does not permit quantification over events. The contribution of this paper is to point out that, by choosing an appropriate interval temporal logic, it is possible to introduce quantification into representations of event-structure without sacrificing decidability.

Keywords. Information Extraction, Interval temporal logic

1 Background

The temporal mark-up language *TimeML* (Pustejovsky *et al.* [1]) is a formalism developed for the purpose of automatically extracting information about the *event-structure* of narrative texts. The language consists of a collection of tags inserted into a text, intended to make explicit information about the events reported in the text and their temporal relations. Like all mark-up languages developed for information extraction, TimeML aims to steer a middle course between expressive complexity and computational simplicity: too much expressiveness, and the language is impossible to process; too little, and it cannot represent the information we hope to extract.

Because it concerns events and their temporal relations, TimeML strays into territory traditionally occupied by *interval temporal logic*—that branch of mathematical logic concerned with structures interpreted over collections of intervals on ordered sets. The purpose of this paper is to explain the relationship between TimeML and various systems of interval temporal logic that have been proposed in the literature. The main contention is that developments in interval temporal logic—some recent, others well-established—require a re-evaluation of the balance which the designers of TimeML have struck between expressive complexity and computational simplicity. Viewed from the vantage point of these developments, the expressive limitations on TimeML appear unnecessarily stringent.

Dagstuhl Seminar Proceedings 05151
Annotating, Extracting and Reasoning about Time and Events
http://drops.dagstuhl.de/opus/volltexte/2005/312

2 TimeML

The purpose of this section is to introduce the language TimeML. We present only that subset of the language which is purely temporal in character; in addition, we take the liberty of simplifying the syntax at various points. We focus on the purely temporal subset of TimeML because that is the part of the language for which formal semantics can be unproblematically given; and we simplify its syntax occasionally because doing so will help to clarify its relationship to existing systems of interval temporal logic.

With these caveats behind us, the principal features of TimeML can be quickly introduced by example. Consider the following 'text':

(1) After his talk with Mary, John drove to Boston. During the drive he ate a donut.

The first sentence in (1) identifies two events—John's talk with Mary, and his drive to Boston—and asserts that the former preceded the latter. The TimeML mark-up of that sentence makes this event-structure explicit as follows:

(2)
After <EVENT eid= talkJM> his talk with Mary </EVENT>
<EVENT eid= driveJB> John drove to Boston </EVENT>
<MAKEINST eid= talkJM eiid= I_1/>
<MAKEINST eid= driveJB eiid= I_2/>
<TLINK eventInst= I_1 relatedToEventInst= I_2 relType= BEFORE/>.

Here, each of the two events in the text is marked with a pair of TimeML tags: an <EVENT>-tag and a <MAKEINST>-tag. The <EVENT>-tag declares the type of event involved and assigns it a label called an *event-id*; the <MAKEINST>-tag then declares an instance of that event type and assigns it a label called an *event-instance-id*. Thus, in (2), John's talk with Mary is given the event-id talkJM and the event-instance-id I_1; likewise, John's drive to Boston is given the event-id driveJB and the event-instance-id I_2. The temporal relationship between these events is then recorded by means of a <TLINK>-tag. The <TLINK>-tag features a pair of event-instance-ids, corresponding to the events related, together with a mnemonic, called a *relation-type*, indicating the relation between them, with the triple interpreted as a predication in the obvious way. Thus, the TLINK-tag in (2) makes explicit the information that John's talk with Mary finished before his drive to Boston started.

The second sentence in (1) identifies the additional event of John's eating a donut, and asserts that it occurs during John's drive to Boston. Again, its TimeML mark-up makes this event-structure explicit as follows:

(3)
During the drive <EVENT eid= eatJd> he ate a donut </EVENT>
<MAKEINST eid= eatJd eiid= I_3/>
<TLINK eventInst= I_3 relatedToEventInst= I_2 relType= DURING/>.

The interpretation of these TimeML tags is completely analogous to (2). The donut-eating event is given the event-id eatJd and the event-instance-id I_3. The `TLINK`, which features the relation-type DURING, makes explicit the information that this donut-eating begins after John's drive to Boston begins, and ends before that drive ends. Note that TimeML recognizes a fixed collection of relation-types, each of which has a fixed interpretation, in a sense which we shall make precise below.

It is important to realise that we can make logical inferences in TimeML so as to recover information not explicit in the original text. To take a somewhat trivial example, the two sentences in (1) together entail that John's talk with Mary preceded the eating of the donut, an entailment which we can record by adding the further `TLINK`:

(4) `<TLINK eventInst=` I_1 `relatedToEventInst=` I_3 `relType=BEFORE/>`.

Detecting such implied TLINKs plays an important role both in improving the effectiveness of automatic annotation procedures and also in developing tools to ease the onerous task of hand-annotation. Current tools to carry out this task take the form of simple rule-chaining systems which are run to exhaustion on a given set of `TLINK` tags (e.g. Setzer *et al.* [2]). This simple approach is adequate because of the restricted set of relation-types which TimeML features. Indeed, as we remarked above, the need to keep such processing computationally manageable is one of the motivations for the limited expressive power of the language. In this respect, we can think of TimeML as a very inexpressive interval temporal logic: we develop this view of the language in detail below.

Perhaps the most striking manifestation of the limited expressive power of TimeML is its lack of quantifiers. Consider, for example:

(5) During each of John's drives to Boston he ate a donut.

Sentence (5) imparts information about the temporal relations between various drivings and eatings, while leaving open how many instances of each event-type occurred. But the system of tags introduced above cannot be indeterminate about the number of event-instances it recognizes; therefore it cannot adequately capture the content of Sentence (5). A more subtle—but also more commonly encountered—manifestation of the same problem concerns negative sentences. Consider, for example:

(6) John drove to Boston. During his drive he did not eat a donut.

The second sentence in (6) does not identify an event of donut-eating—rather, it denies that *any* such event occurred over a certain period. That is: it is quantificational in character. Again, the system of tags introduced above can only assert the existence of events; and no collection of such assertions can ever have the force of a denial.

Actually, the above characterization of TimeML is a little unfair. TimeML does in fact include tags purporting to represent the kind of information conveyed in (5) and (6). In particular, `MAKEINST`-tags have an optional 'polarity' argument to express negation; in addition, TimeML even features tags purporting to capture the meaning of quantifying adverbials such as every Monday. Unfortunately, these parts of the formalism lack properly-defined semantics, and are not incorporated into existing inferential mechanisms. As things stand, these aspects of TimeML are not susceptible to serious logical analysis.

3 From TimeML to interval constraint expressions

In the foregoing exposition of TimeML, we encountered *event-ids*, *relation-types* and *event-instance-ids*. Logically, event-ids function as unary predicates, since they denote types of events; relation-types function as binary predicates, since they denote relations between pairs of events; and event-instance ids function as variables ranging over events, since they form the arguments of event-ids and relation-types. Stripping away some notational clutter, then, we can transcribe the contents of the TimeML tags in (2)–(3) as a conjunctive formula of first-order logic thus:

(7) $\text{talkJM}(I_1) \wedge \text{driveJB}(I_2) \wedge \text{eatJd}(I_3) \wedge \text{BEFORE}(I_1, I_2) \wedge \text{DURING}(I_3, I_2)$.

Furthermore, since the only aspects of events we are concerned with here are entirely dependent on the time-intervals over which they occur, we may as well assume the event-instance variables in such a formula to range directly over time-intervals, re-interpreting the predicates accordingly. Thus, readers are encouraged to think of I as standing for 'time interval' rather than 'event-instance'. The logical sleight-of-hand involved here is, in fact, innocuous, and helps to clarify the connection to interval temporal logic explained below.

Following common practice, we identify time-intervals with convex, bounded, closed subsets of the real line having non-empty interiors. Any time-interval can thus be written in the standard way as $[a, b]$, where a, b are real numbers such that $a < b$. Denote the set of all time-intervals, under this identification, by $\mathcal{I}$. We can then give formal interpretations of the predicates mentioned in (7) as relations over $\mathcal{I}$ corresponding to the informal glosses given above. For example, we take the unary predicate talkJM to be satisfied by an interval I just in case there was an event of John's talking to Mary occurring exactly over I; similarly for the predicates driveJB and eatJd. Likewise, we take the binary predicate BEFORE to be satisfied by a pair of intervals $I = [a, b]$ and $J = [c, d]$ just in case $b < c$; and similarly the pair I, J satisfies DURING just in case $c < a$ and $b < d$.

Note the difference in logical status between the treatment of event-ids on the one hand and relation-types on the other. The collection of predicates corresponding to event-ids is open-ended, and their interpretations represent contingent facts about the distribution of events in the world; by contrast, the

collection of predicates corresponding to relation-types is fixed, and their interpretations represent non-contingent facts about our underlying model of time. We shall recognize this difference by saying that the predicates corresponding to event-ids are *non-logical* primitives, whereas the predicates corresponding to relation-types are *logical primitives*. This distinction between logical and non-logical primitives is central to any logic, and will feature prominently in the sequel. We remark that the fact that the non-logical primitives here are *binary* predicates while the logical primitives are *unary* predicates is just a coincidence.

Let us re-examine the inference from the `TLINK`s in (2) and (3) to the `TLINK` in (4) in the light of this re-formulation. Suppose that the event-instance-ids in some marked-up text are $I_1, \ldots, I_n$. Evidently, the collection of `TLINK` tags in that text corresponds to a formula of the form

(8) $R_1(J_1, K_1) \wedge \cdots \wedge R_n(J_m, K_m)$,

where, for each i $(1 \leq i \leq m)$, J_i and K_i are chosen from among the $I_1, \ldots, I_n$, and R_i is chosen from among the relation-types recognized by TimeML. Expressions of the form (8) are familiar to temporal logicians as *interval constraint expressions*; and a set of such expressions forms an *interval constraint language*. An interval constraint expression of the form (8) is said to be *satisfiable* if there exists an assignment of elements of $\mathcal{I}$ to the variables $I_1, \ldots, I_n$ for which the conjuncts $R_i(J_i, K_i)$ all hold. The same expression is said to *entail* the constraint $R(J, K)$ if every assignment of elements of $\mathcal{I}$ to the variables $I_1, \ldots, I_n$ for which the conjuncts $R_i(J_i, K_i)$ all hold also makes $R(J, K)$ hold. For most interval constraint languages, the problem of determining entailments can be reduced in polynomial time to the problem of determining satisfiability. Hence, the fundamental question when dealing with such constraint languages is the computational complexity of the satisfiability problem.

The answer to this question depends, of course, on the set of interval relations which the R_h can be interpreted as. Call a binary relation on $\mathcal{I}$ *ordinal* if it is definable in terms of the relative temporal orders of the endpoints of the intervals in question. For example, it is obvious that the intended interpretations of BEFORE and DURING are ordinal relations. The set of all ordinal relations on $\mathcal{I}$ form a relation-algebra (see, e.g. Ladkin and Maddux [3], pp. 446 ff.), which has been intensively investigated. This relation-algebra has $2^{13} = 8192$ elements, and is generated by a set of 13 atoms. We call these 13 atoms the *simple interval relations*; they form the basis of the system described in Allen [4]. The relation-types in TimeML are (ignoring some needless clutter) exactly these 13 simple interval relations.

The satisfiability problem for constraint expressions featuring only the 13 simple interval relations can be determined in polynomial time (Vilain, Kautz and van Beek [5]). By contrast, the satisfiability problem for constraint expressions featuring all 8192 ordinal relations is NP-complete. This fact appears to vindicate the expressive limitations imposed by TimeML, because the set of simple interval relations recognized in `TLINK`-tags form a tractable subset of the full relation algebra. However, they are not the only such set. Nebel and Bürkert [6]

describe a (maximal) set of 868 ordinal interval relations whose satisfiability problem is tractable (actually: cubic); and Drakengren and Jonsson [7] characterize all maximal tractable sets of ordinal interval relations (subject to some technical conditions). Viewed in this light, the computational motivation behind the expressive limitations of TimeML are less convincing than they might at first have appeared.

Moreover, ordinal relations are not the only relations over $\mathcal{I}$ of relevance to the event-structure of texts. Particularly salient are *metric* interval relations. Consider for example:

(9) John ate the donut at least five hours before he drove to Boston.

For definiteness, let us suppose that (9) constrains the donut-eating to end at least five hours before the driving begins. That is, taking the basic unit of time to be 1 hour, the operative interval relation is given by $\{\langle [a,b],[c,d]\rangle \mid b < c-5\}$. Interval constraint expressions of the form (8) can of course feature binary predicates interpreted as metrical interval relations; and again, the crucial question is the computational complexity of the satisfiability problem. Such questions have in fact been addressed in the literature. For example, Dechter, Meiri and Pearl [8] define (in effect) just such a metrical constraint language, and show that it has a tractable satisfiability problem. Hence, from the point of view of tractability, there is no need to restrict the interpretation of the predicates in (8) to ordinal relations.

Again, the analysis of TimeML is complicated at this point by the fact that the language does in fact incorporate tags marking duration-denoting phrases, for example:

(10) `<TIMEX3 tid=` t_1 `type= DURATION value= P5H mod= GREATER>`
`5 hours </TIMEX3>`.

The meaning of (10) is probably best given by regarding t_1 as a variable ranging over elements of $\mathcal{I}$, and the rest of the tag as a unary predicate interpreted as the set of intervals: $\{[a,b] \in \mathcal{I} \mid b-a > 5\}$. Such tags can then be used, together with ordinary `TLINK`s, to express various metrical temporal relations. It is difficult to give a precise account of the underlying logic here, since the semantics of the relevant TimeML constructs is not completely specified. However, it is clear that this mechanism covers the relations definable in the system of Dechter *et al.* in a rather awkward way, incorporating considerable logical redundancy; in addition, there is again no standard régime within the TimeML framework for making logical inferences with temporal relations of this kind.

Summarizing, we have established that the collection of `TLINK`s in a TimeML-marked-up document can be regarded as as an interval constraint expression of the form (8), with binary predicates limited to the 13 simple interval relations. (If tags such as (10) are also included, metrical interval constraints can also be expressed.) All interesting logical questions concerning such constraints can be reduced to questions of satisfiability; and the computational complexity of

interval constraint satisfiability problems has been thoroughly investigated in the literature. The constraints expressed by TLINKs belong to a set for which the corresponding satisfiability problem is tractable; however, it is by no means the only such set.

4 From interval constraint expressions to first-order interval logic

We referred to Formula (8) above as an interval constraint expression; but of course, that is just another name for a formula of first-order logic over a certain signature of binary predicates in which the only logical connective is $\wedge$. The question naturally arises: what happens if we liberalize the syntax to allow arbitrary first-order formulas, and include the unary predicates corresponding to event-ids in our signature?

Consider again Sentence (5). This sentence asserts that, within some given temporal context, every interval over which John drove to Boston includes some interval over which he ate a donut. Interpreting the predicates driveJB, eatJd and DURING as above, the meaning of (5) can be represented by the first-order formula

$$(11)\quad \forall I_1(\text{driveJB}(I_1)\wedge\text{DURING}(I_1, I) \rightarrow \exists I_2(\text{eatJd}(I_2)\wedge\text{DURING}(I_2, I_1))).$$

Here, the temporal context to which the quantification in (5) is implicitly limited is represented by the free variable I. Of course, there is no reason why first-order formulas need have any free variables at all: however, for reasons which will emerge later, we shall be particularly interested in formulas having exactly one free variable.

Likewise, consider again the pair of sentences in (6). These assert that there is an interval over which John drove to Boston, and that this interval includes no interval over which he ate a donut. Actually, they arguably assert something more: the phrase his drive in the second sentence suggests that, within the given temporal context, the event of John's driving to Boston is realized uniquely. Let us suppose that we want to record that information. Helping ourselves to the definite quantifier $\iota x(\phi, \psi)$ with the standard (Russellian) semantics, then, we may do so using the first-order formula

$$(12)\quad \iota I_1(\text{driveJB}(I_1) \wedge \text{DURING}(I_1, I), \neg\exists I_2(\text{eatJd}(I_2) \wedge \text{DURING}(I_2, I_1))).$$

Note incidentally that we can now improve on our earlier formulation of the first sentence (1) so as to take account—if we like—of the implicated uniqueness of John's talk with Mary within the relevant temporal context. Again using the free variable I to represent that temporal context, we can replace the formalization (7) with

$$(13)\quad \begin{array}{l}\iota I_1(\text{talkJM}(I_1) \wedge \text{DURING}(I_1, I),\\ \qquad \exists I_2(\text{driveJB}(I_2) \wedge \text{DURING}(I_2, I) \wedge \text{BEFORE}(I_1, I_2))).\end{array}$$

The above formulas feature unary predicates corresponding to event-types, binary predicates corresponding to the 13 simple interval relations, and the full syntax of first-order logic, with variables assumed to range over the set $\mathcal{I}$. Let us call the logic comprising these resources *first-order interval logic*.

Recall our earlier distinction between logical and non-logical primitives: the binary predicates corresponding to relation-types, such as BEFORE and DURING were designated as logical primitives, whereas the unary predicates corresponding to event-ids, such as talkJM, driveJB and eatJd etc. were designated as non-logical primitives. Following standard logical practice, we take a *structure* $\mathfrak{A}$ to be an assignment of interpretations to the non-logical primitives. Thus, in the present case, a structure amounts to a specification, for each event-id, of precisely which time-intervals an event of the corresponding type occurred over.

Let $\mathfrak{A}$ be a structure and $\phi(I_1, \ldots, I_n)$ a first-order formula (over the above signature) with $I_1, \ldots, I_n$ as its only free variables. Together, the interpretations of the non-logical primitives provided by $\mathfrak{A}$ and the fixed interpretations of the logical primitives determine, via the usual semantics of first-order logic, precisely which n-tuples from $\mathcal{I}$ satisfy $\phi(I_1, \ldots, I_n)$. The formula $\phi(I_1, \ldots, I_n)$ is said to be *satisfiable* if there exists a structure $\mathfrak{A}$ in which it is satisfied by some n-tuple from $\mathcal{I}$. It is easy to see that satisfiability of interval constraint expressions, as defined in Section 3, is a special case of satisfiability of formulas in first-order interval logic, as defined here. Thus, the logical analysis of interval constraint expressions fits smoothly into the standard logical account of first-order interval logic. As with interval constraint expressions, so too with first-order interval logic, most interesting logical questions can be reduced to the problem of determining satisfiability.

Unfortunately, the satisfiability problem for first-order interval logic is undecidable. (This actually follows from stronger results reported in the next section.) But in choosing a mark-up language to record event-structure in texts, a logic in which the satisfiability problem is at least decidable seems the very minimum we should require. It follows that, in seeking to extend the expressive power of TimeML, we need to examine formalisms of lower expressive power. To this end, we employ the syntactic apparatus of modal logic.

5 From first-order interval logic to modal interval logic

Consider a unary predicate corresponding to an event-id—for example, driveJB—and a structure $\mathfrak{A}$ which interprets it. This predicate can be equivalently regarded as a proposition-letter to which $\mathfrak{A}$ assigns a truth-value relative to any given interval I: true if I satisfies driveJB in $\mathfrak{A}$; false otherwise. By combining proposition-letters using Boolean connectives in the usual way, we can build formulas to which $\mathfrak{A}$ assigns a truth value with respect to any given interval I. We write $\mathfrak{A} \models_I \phi$ if the (propositional) formula ϕ is true at I in $\mathfrak{A}$. Now consider a binary predicate corresponding to some relation-type—for example, DURING—and let us add the modal operators [DURING] and $\langle$DURING$\rangle$ to our proposi-

tional language. We can give the semantics of these operators by declaring that, for any structure $\mathfrak{A}$, any interval I and any formula ϕ: (i) $\mathfrak{A} \models_I$ [DURING]ϕ just in case, for all J such that the pair $\langle J, I\rangle$ satisfies the predicate DURING, $\mathfrak{A} \models_J \phi$; and (ii) $\mathfrak{A} \models_I \langle$DURING$\rangle\phi$ just in case, for some J such that the pair $\langle J, I\rangle$ satisfies the predicate DURING, $\mathfrak{A} \models_J \phi$. Applying these modal operators freely to propositions, we obtain a propositional modal language whose formulas have an obvious translation into formulas of first-order interval logic with exactly one free variable. In particular, the modal formula

(14) [DURING](driveJB $\rightarrow \langle$DURING$\rangle$eatJd)

is easily seen to translate (up to logical equivalence) to the first-order interval logic formula (11).

By introducing modal operators for the other simple interval relations in the same way, we arrive essentially at the system known as $\mathcal{HS}$, after its inventors, Halpern and Shoham [9]. A formula ϕ of $\mathcal{HS}$ is said to be *satisfiable* if there exists some structure $\mathfrak{A}$ and some interval I such that $\mathfrak{A}, \models_I \phi$. Again, most interesting logical questions can be reduced to the problem of determining satisfiability. The logic $\mathcal{HS}$ is strictly less expressive than the full first-order language (Venema [10]). Nevertheless, Halpern and Shoham show that the satisfiability problem for $\mathcal{HS}$ is undecidable. Until recently, very little was known about decidable fragments of $\mathcal{HS}$; for a survey, see Goranko *et al.* [11].

At this point, the elimination of quantification from TimeML appears to be vindicated. Not only is first-order interval logic undecidable; so too is the less expressive modal interval logic $\mathcal{HS}$. As we shall see in the next section, however, closer analysis shows that such a conclusion would be hasty.

6 From modal interval logic to $\mathcal{TPL}^*$

In this section, we define a formalism for recording the event-structure of texts which combines considerable expressive power with a decidable satisfiability problem. The following modal interval logic is a subset of the logic $\mathcal{TPL}$ introduced in Pratt-Hartmann [12]. For want of a better name, let us call it $\mathcal{TPL}^*$. In the sequel, let E be a fixed set of event-ids.

Definition 1. *Let e range over the set E. We define a $\mathcal{TPL}^*$-formula ψ by the syntax:*

$$\psi := \top \mid \bot \mid \langle e\rangle\psi \mid [e]\psi \mid \{e\}\psi \mid \{e\}_>\psi \mid \{e\}_<\psi \mid \neg\psi \mid \psi \wedge \psi' \mid \psi \vee \psi' \mid \psi \rightarrow \psi'.$$

The following notation will be used to present the semantics of $\mathcal{TPL}^*$. Let $I = [a, b]$ and $J = [c, d]$ be intervals. If $a < c < d < b$, we write DURING(J, I). Further, if DURING(J, I), we let the terms init(J, I) and fin(J, I) denote the intervals $[a, c]$ and $[d, b]$, respectively. In other words, whenever DURING(J, I) is true, we take init(J, I) to denote the initial segment of I up to the beginning

of J, and $\mathrm{fin}(J, I)$ to denote the final segment of I from the end of J. Again, we take an *interpretation* $\mathfrak{A}$ simply to be an assignment, to each event-id in E, of a subset of $\mathcal{I}$, except that we impose the added condition that, for each $e \in E$, the set $\mathfrak{A}(e)$ of intervals at which e holds in $\mathfrak{A}$ is *finite*. The motivation here is simply that we have in mind situations in which event-atoms denote everyday event-types instantiated in finite contexts.

Definition 2. *Let ϕ be a formula, $\mathfrak{A}$ an interpretation, and $I \in \mathcal{I}$. We define $\mathfrak{A} \models_I \phi$ recursively as follows:*

1. $\mathfrak{A} \models_I \langle e \rangle \psi$ *iff for some* J *such that* $\mathrm{DURING}(J, I)$, $J \in \mathfrak{A}(e)$ *and* $\mathfrak{A} \models_J \psi$;
2. $\mathfrak{A} \models_I [e]\psi$ *iff for all* J *such that* $\mathrm{DURING}(J, I)$, $J \in \mathfrak{A}(e)$ *implies* $\mathfrak{A} \models_J \psi$;
3. $\mathfrak{A} \models_I \{e\}\psi$ *iff there is a unique* J *such that* $\mathrm{DURING}(J, I)$ *and* $J \in \mathfrak{A}(e)$, *and for that* J, $\mathfrak{A} \models_J \psi$;
4. $\mathfrak{A} \models_I \{e\}_<\psi$ *iff there is a unique* J *such that* $\mathrm{DURING}(J, I)$ *and* $J \in \mathfrak{A}(e)$, *and for that* J, $\mathfrak{A} \models_{\mathrm{init}(J,I)} \psi$;
5. $\mathfrak{A} \models_I \{e\}_>\psi$ *iff there is a unique* J *such that* $\mathrm{DURING}(J, I)$ *and* $J \in \mathfrak{A}(e)$, *and for that* J, $\mathfrak{A} \models_{\mathrm{fin}(J,I)} \psi$;
6. *the usual rules for* $\top$, $\bot$, $\neg$, $\wedge$, $\vee$ *and* $\rightarrow$.

A formula ϕ is said to be satisfiable *if, for some $\mathfrak{A}$ and I, $\mathfrak{A} \models_I \phi$.*

As usual, most interesting logical questions can be reduced to the problem of determining satisfiability.

Under the above semantics, formulas of $\mathcal{TPL}^*$ translate naturally into formulas of first-order interval logic having one free variable, just as for the modal interval logic $\mathcal{HS}$ considered in Section 5. We can get an idea of the expressive power of $\mathcal{TPL}^*$ by considering the sentences (5)–(6), whose truth-conditions were given by the first-order interval logic formulas (11)–(12). A little checking shows that the following $\mathcal{TPL}^*$-formulas translate to (11)–(12), respectively:

(15) $[\mathrm{driveJB}]\langle \mathrm{eatJd} \rangle \top$
(16) $\{\mathrm{driveJB}\}[\mathrm{eatJd}]\bot$.

Moreover, consider again Sentence (1) and its 'improved' first-order interval logic rendition (13). Again, it is easy to verify that the same satisfaction-conditions are given by the $\mathcal{TPL}^*$-formula

(17) $\{\mathrm{talkJM}\}_>\langle \mathrm{driveJB} \rangle \top$.

Thus, $\mathcal{TPL}^*$ is an expressive formalism. Nevertheless, its satisfiability problem is decidable. The following fact follows easily from Pratt-Hartmann [12], Theorem 1.

Fact 1 *The satisfiability problem for $\mathcal{TPL}^*$ is in NEXPTIME.*

Moral: it is possible to have a quantified representation language for event-structure in texts and yet retain a decidable satisfiability problem.

7 Conclusion

In this paper, we have described a subset of the temporal mark-up language TimeML and explained its relation to some other formalisms found in the literature on interval temporal logic: temporal constraint expressions, first-order interval logic and the modal interval logics $\mathcal{HS}$ and $\mathcal{TPL}^*$. We have seen how the subset of TimeML we examined forms a very inexpressive interval temporal logic, with a tractable satisfiability problem. Unfortunately, that subset of TimeML is too inexpressive for much of the information we want to record about event-structure in texts: most crucially, it does not permit quantification over events. The contribution of this paper is to point out that, by choosing an appropriate interval temporal logic—such as $\mathcal{TPL}^*$, for example—it is possible to introduce quantification into representations of event-structure without sacrificing decidability. To be sure, we do not claim that $\mathcal{TPL}^*$ as it stands is a suitable formalism for marking up event-structure in texts, not least because of its uncomfortably high complexity. Nevertheless, it does, we contend, point the way for future research in this area.

References

1. Pustejovsky, J., Castaño, J., Ingria, R., Surí, R., Gaizauskas, R., Setzer, A., Katz, G.: Timeml: Robust specification of event and temporal expressiveness in text. In: Proceedings, IWCS-5, Tilburg (2003)
2. Setzer, A., Gaizauskas, R., Hepple, M.: Using semantic inference for temporal annotation comparison. In: Proceedings, ICoS-4, Nancy (2003) 185–196
3. Ladkin, P.B., Maddux, R.D.: On binary constraint problems. Journal of the ACM **41** (1994) 435–469
4. Allen, J.F.: Maintaining knowledge about temporal intervals. Communications of the ACM **26** (1983) 832–843
5. Vilain, M., Kautz, H., van Beek, P.: Constraint propagation algorithms for temporal reasoning: a revised report. In Weld, D.S., de Kleer, J., eds.: Readings in qualitative reasoning about physical systems. The Morgan Kaufman series in representation and reasoning. Morgan Kaufman, San Mateo, CA (1990) 373–381
6. Nebel, B., Bürckert, H.J.: Reasoning about temporal relations: a maximal tractable subclass of Allen's interval algebra. J. ACM **42** (1995) 43–66
7. Drakengren, T., Jonsson, P.: A complete classification of tractability in Allen's algebra relative to subsets of basic relations. Artificial Intelligence **106** (1998) 205–219
8. Dechter, R., Meiri, I., Pearl, J.: Temporal constraint networks. Artificial Intelligence **49** (1991) 61–95
9. Halpern, J.Y., Shoham, Y.: A propositional modal logic of time intervals. Journal of the ACM **38** (1991) 935–962
10. Venema, Y.: Expressiveness and completeness of an interval tense logic. Notre Dame Journal of Formal Logic **31** (1990) 529–547
11. Goranko, V., Montanari, A., Sciavicco, G.: A road map of interval temporal logics and duration calculi. Journal of Applied Non-classical Logics **14** (2004) 9–54
12. Pratt-Hartmann, I.: Temporal prepositions and their logic. Artificial Intelligence (2005) (article in press).