Time in Automated Legal Reasoning

LLUÍS VILA

Department of Software (LSI), Technical University of Catalonia, Barcelona, Catalonia, Spain

HAJIME YOSHINO

Meiji Gakuin University, Tokyo, Japan

ABSTRACT *Despite the ubiquity of time and temporal references in legal texts, their formalization has often been either disregarded or addressed in an* ad hoc *manner. In this paper we address this issue from the standpoint of the research done on temporal representation and reasoning in AI. We identify the temporal requirements of legal domains and propose a temporal representation framework for legal reasoning which is independent of (i) the underlying representation language and (ii) the specific legal reasoning application. The approach is currently being used in a rule-based language for an application in commercial law.*

1. Introduction

Automated legal reasoning systems require a proper formalization of time and temporal information (McCarty, 1995; Sergot, 1995). Quoting L. Thorne McCarty (1995):

> ... time and action are both ubiquitous in legal domains....

Notions related to time are found in major legal areas such as labour law (e.g. the time conditions to compute benefit periods), commercial law (e.g. the time of the information used to establish the validity of agreements or to calculate damages (Blumsohn, 1991)),[1] criminal law (e.g. the temporal information known about the various elements involved in the analysis of a criminal case), patent law (e.g. the time constraints formulated in regulations for applying to patents). Moreover, many procedural codes associated with these statutes usually require the management of timetables based on some temporal representation.

We elaborate on two representative examples. The first example is taken from the *United Nations Convention for International Sale of Goods* (CISG) (Yoshino, 1994).

Example 1 (CISG) Article 15: *An offer becomes effective when it reaches the oferee. An offer, even if it is irrevocable, may be withdrawn if the withdrawal reaches the oferee before or at the same time as the offer.*

This article contains various temporal aspects that are common in legal texts. We find denotations for events that *happen at a certain time* (e.g. 'reach'), objects that have a certain *lifetime* (e.g. 'offer', 'withdrawal'), properties that *change over*

1360-0834/98/030173-25

time (e.g. 'an offer is effective') and *temporal relations* (e.g. 'before or at the same time').

We borrow our second example from Poulin *et al.* (1992).

Example 2 *Next two articles belong to the* Canadian Unemployment Insurance Law:
Section 9(1) *[...] A benefit period begins on the Sunday of the week in which (a) the interruption of earnings occurs, or (b) the initial claim for benefit is made, whichever the later.*
Section 7(1) *[...] the qualifying period of an insured person is the shorter of (a) the period of fifty-two weeks that immediately precedes the commencement of a benefit period under subsection 9(1), and*
(b) the period that begins on the commencement date of an immediately preceding benefit period and ends with the end of the week preceding the commencement of a benefit period under subsection 9(1).

In addition to denotations of temporal events (e.g. 'interruption of earnings', 'claim for benefits'), we find references to temporal units such as 'qualification period' and 'benefit period', and temporal relations such as 'begins', 'ends', 'period of fifty-two weeks', 'the period that precedes', 'the period that immediately precedes' and a rich variety of temporal operators such as 'the shorter of ...', 'the Sunday of the week ...', 'the later of ...'.

This work belongs to the tradition of formalizing law using logic. Despite the prominent presence of temporal references in legal texts, temporal representation and reasoning is an issue that legal reasoning projects have often either disregarded or addressed in an *ad hoc* manner. Furthermore, it is a surprising situation given the prolific research activity done on temporal reasoning in AI during the past 15 years (see Vila (1994) for a survey). This may be due to the fact that, quoting Marek Sergot (1995), 'it looks like a *huge* topic'. Another reason could be the utilization of techniques traditionally disconnected from legal reasoning such as constraint satisfaction.

Our goal here is to provide a representation framework well-suited to formalizing the temporal aspects of law in its different areas. We build upon results from the research area of *temporal reasoning in AI*.

We proceed by first identifying the requirements of legal domains (Section 2). Then we outline the features that characterize a temporal representation framework and point out some of the choices proposed for each feature (Section 3.1). After that we overview related work (Section 3.2) and, finally, we systematically discuss each feature and select the choice that best addresses the requirements (Section 4). We illustrate the adequacy of our proposal, called LTR, by revisiting the examples above. The guarantee of the applicability of LTR is conditioned to the validity of the requirement analysis we did.

The contribution of this paper is twofold: (i) as a reference for analysing the temporal representation in existing legal reasoning systems, and (ii) a, the foundation in building the 'temporal component' of a legal reasoning application. Temporal representation and reasoning is a very broad area and covering everything would be too ambitious for a single paper, even if its focus is on a particular application area. The following issues are out of the scope of this paper: (i) periodic occurrences, (ii) handling time associated with legal provisions, and (iii) non-monotonic temporal reasoning.

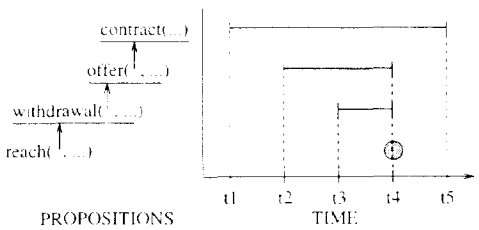

Figure 1. Repeated temporal reference example.

Terminology. Before going ahead we define a few terms common in the temporal reasoning literature used throughout this paper. By *temporal expression* we mean an expression whose denotation is naturally associated with a specific time. In the above examples, 'offer is effective' and 'interruption of earnings' are temporal expressions. We shall distinguish between *fluents* when they are expressions that describe the state of affairs in a given domain ('offer is effective'), and *events* when they represent occurrences that may change that state ('interruption of earnings').[2] A *temporal proposition* is a logical proposition representing a temporal expression. By *temporal relation* we mean a relation whose arguments are all temporal, and by *temporal function* a function whose range is temporal.[3]

2. Requirements

In this section we identify the requirements of a temporal representation language for formalizing law. The analysis is done at the two main general levels: *notational efficiency*, which comprises issues such as expressiveness, modularity, readability, compactness, flexibility, etc., and *computational efficiency*. Finally, we explain the issues that have not been considered in this work.

2.1. Notational efficiency requirements

Repeated temporal references. A *repeated temporal reference* is a temporal expression that includes a reference to another temporal expression. Repeated temporal references abound in legal texts. Let's have a look on a piece from example 1:

> An **offer**, even if it is **irrevocable**, may be **withdrawn** if the **withdrawal reaches** the **offeree** *before or at the same time* as the **offer**.

The 'reach' event makes reference to a 'withdrawal' of an 'offer' of a 'contract', all these being temporal objects with their own associated times of occurrence (see Figure 1). In addition, some implicit constraints may hold among these various times. For example, the 'reach' event cannot happen outside the lifetime interval of the offer.

Temporal operators. Legal texts with temporal references often involve a (sometimes large) number of temporal operators. Example 2, for instance, involves a function that returns 'the shorter of' two periods of a function that returns the 'the latest of' two dates.

Precise and indefinite temporal relations. In addition to exact times and dates (e.g. 3:15 p.m., 2 October 1996) many different classes of 'less precise' temporal relations appear in legal texts. The following are some examples: '... before or at the same time than ...', '... during ...', '... contains or overlaps ...', '... immediately precedes ...', '... in few days ...', '... between 2 or 3 days ...', '... either 2 or 3 days if ... or between 1 and 2 weeks if ...'. These relations are called *indefinite* since they represent a set (interpreted as a disjunction) of possible times. When the set is not convex we talk about *non-convex* or *disjunctive* relations.

Indefinite relations are often present in the description of legal cases (e.g. '... *few days later* the message was dispatched', 'the transaction took a *couple of weeks*', *'between 9:00 and 10:00* the suspect was seen at ...').

Several temporal levels. Some legal applications require distinguishing between different levels of temporal information (Sergot, 1995). A common distinction (often made in database systems (Tansel *et al.*, 1993)) is *real time* (in databases called *valid time*) versus *belief time* (i.e. *transaction time*).

Modularity. Since legal domains usually involve knowledge related to various notions such as evidence, belief, intention, obligation, permission and uncertainty, modularity is a central issue. A desirable feature of a temporal representation is that it allows for an orthogonal combination with other knowledge modalities.

2.2. *Computational efficiency requirements*

The ability to efficiently encode and process temporal relations may have a high impact on the performance of the overall procedure from both points of view: space and time.

The *size* of the temporal representation is polynomial in the number of temporal propositions and the number of possible temporal relations which, in turn, depends on the model of time adopted (bounded, dense/discrete, etc.).

The *time* performance of answering temporal queries can be strongly influenced by the class of temporal relations supported. The worst-case time complexity of checking consistency of a set of temporal constraints can at best be linear in the number of relations, but if the indefiniteness of temporal relations is *non-convex* it is unlikely that the problem is tractable (Vilain *et al.*, 1989, Dechter, 1991). In most legal scenarios the ratio *number of temporal relations versus number of temporal propositions* is relatively low and the amount of non-convex indefiniteness is small. However, some cases are found in specific domains (such as in some criminal cases) or some tasks (e.g. *legal planning*) where multiple temporal possibilities need to be taken into consideration.

In both easy and hard cases, the capability of efficiently answering queries about temporal relations is an important issue: in the easy case because the number of temporal propositions involved in legal scenarios may be large; in the hard case because of the potential dramatic performance degradation due to the combinatorial nature of non-convex relations.

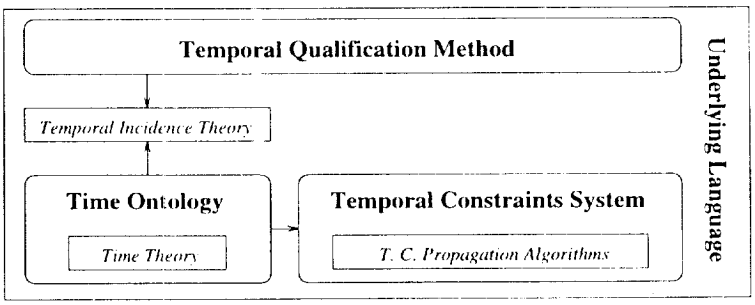

Figure 2. The features of a temporal representation framework.

2.3. *Issues not addressed*

Periodic occurrences. Although not very common, some legal norms and cases require the expression of periodic events such as 'pay X once every month' or 'get a supply twice a week from 1/1/95 to 1/1/96'. This is an issue of current research (Morris *et al.*, 1994; Wetprasit *et al.*, 1996) that we shall not address here.

The time of law. Law changes over time. New norms are introduced and some existing ones are derogated over time. A proper account of these changes is obviously important to correctly interpret the law (Bulygin, 1982; Chemilieu-Gendrau, 1987). This is a fairly open issue in automated legal reasoning, which could be handled by means of a temporal representation that associates time with objects more complex than atomic propositions such as rules or contexts. Our investigation here is restricted to time associated to atomic propositions.

Non-monotonic temporal reasoning. Rescinding agreements, withdrawing decisions, handling retro-active provisions,[4] etc., all require non-monotonic reasoning capabilities. It can be considered a 'temporal' issue since non-monotonic assumptions and inference rules can be formulated using the underlying temporal language. Moreover, there is a non-monotonic reasoning specifically temporal: the one that concerns assumptions about temporal relations. For instance, we may want to assume that a fluent over time as long as it is consistent with the rest of the information. This matter is out of the scope of this paper.

3. Temporal representation: background

3.1. *Features*

A temporal reasoning formalism is defined by a set of features that we survey in this section. They are graphically[5] presented in Figure 2.

Time ontology. The most basic feature is the *ontology* of time, namely the set of *primitive temporal units* and *primitive temporal relations*. The two classical approaches are *instants* (or time points) and *periods* (or time intervals). As instant primitive relations, for example, one can take the three simple qualitative relations between two points in a line $<, =$ and $>$. When temporal relations

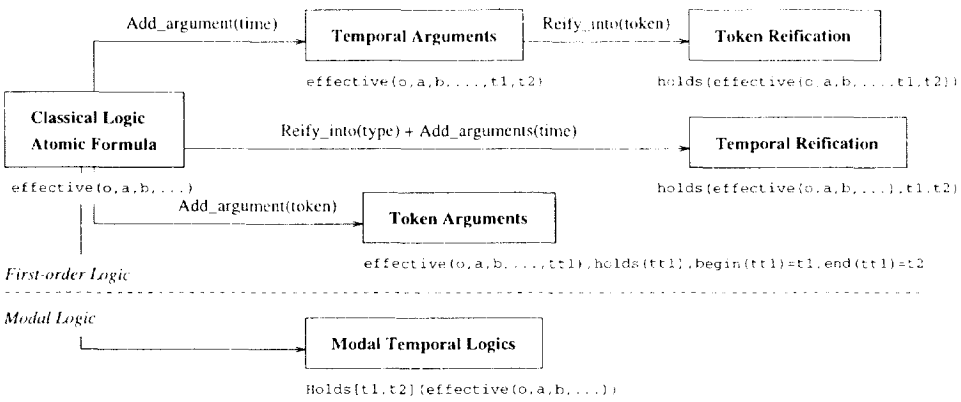

Figure 3. Temporal qualification methods in AI.

involve numeric information, an additional ontological unit is needed: the *duration*. A duration is the distance between two time points.[6]

A related ontological issue is *granularity*. From a semantic point of view, granularity is defined as the primitive unit of 'real time'[7] over which the primitives of our time ontology are interpreted. From a practical point of view, the granularity is determined by the smaller unit used to specify durations in a given context. Different contexts may require different granularities and systems dealing with different contexts may require a mechanism to switch from one granularity to another.

The intuitions about the structure of time (such as the type of ordering, bound/unbound, discrete/dense, etc.) are specified by a set of axioms called the **theory of time**. A lot of work has been done on the study of theories based on instants (van Benthem, 1991) and periods (Walker, 1948; Hamblin, 1972; Newton-Smith, 1980; Allen & Hayes, 1985), on deriving one primitive from the other, and on defining ontologies that combine them (Tsang, 1987; van Benthem, 1991; Allen & Hayes, 1989; Bochman, 1990; Galton, 1990; Vila & Schwalb, 1996).

Temporal constraints. The primitive temporal relations and (logical) combinations of them are naturally regarded as constraints. For example, 'the time point p is before or after the time point p'' is a constraint that restricts the set of possible values for the relative temporal distance between p and p'. When the set is non-convex we talk about non-convex constraints. This together with the temporal units and the allowed temporal constraints determine the *temporal constraint class*. For instance, the constraint in the above example is a *non-convex qualitative point* constraint. A temporal constraint formalism must be provided with a set of specialized **temporal constraint satisfaction algorithms** (van Beek, 1992; Gerevini & Schubert, 1995; Schwalb & Dechter, 1997).

Temporal qualification. A central feature is the method employed to abscribe time to temporal propositions. It usually involves a number of newly defined predicates such as Allen's *Holds* or Shoham's *True*, which express that a given proposition is true at a certain time. These are called *temporal incidence predicates*

(TIP). Figure 3 presents a scheme of the various temporal qualification methods proposed in the literature.

The most straight forward approach, called *temporal arguments* (Haugh, 1987; Bacchus *et al.*, 1991), proposes introducing time as one or more additional arguments (e.g. `effective(o,a,b, ..., t1,t2)`). A variation called *token arguments* (Dean & McDermot, 1987; Galton, 1991) uses a third element, the *temporal token* or *token* which links propositions with their relative times (e.g. `effective(o,a,b, ..., tt1)`, `begin(tt1) = t1`). A token represents a particular temporal instance of a given temporal proposition.

The *temporal reification* approach (McDermott, 1982; Allen, 1984) models temporal propositions as logical terms called *propositional terms*. A propositional term is associated with its times by making them all the arguments of a TIP (e.g. `Holds(effective(o,a,b, ...),[t1,t2])`). A variant called *token reification* (Vila & Reichgelt, 1996) proposes first adding time as argument and then reifying (e.g. `holds(effective(o,a,b, ..., t1,t2))`). In this case the propositional term denotes a temporal token.

Finally, modal approaches introduce a number of temporal modal operators that qualify propositions. Classically temporal modal operators are *relative*. For instance, given a proposition Φ, `F`Φ means Φ is true in some future, `G`Φ means Φ is true in every future time. `N`Φ means Φ is true at next time. More general, *absolute* operators are formed by using time as an index (e.g. `Holds[t1,t2](effective(o,a,b, ...))`). Modal approaches are attractive for their expressiveness, notational compactness and modularity. Although it is an appealing choice, in this paper we only consider methods based on first-order logic since it is a more standard and widely used language, which turns out to be expressive enough for our requirements.

The trade-off among the various first-order approaches is increased expressive power (which is limited in *temporal arguments*) versus keeping the language simple, standard and ontologically clear (which are common objections to reification).

Temporal incidence. The general properties of the TIPs are specified by the *temporal incidence theory*. A classical example of temporal incidence axiom is *homogeneity* of `Holds` : if a proposition holds over a period it holds over any of its subtimes.

The underlying language. Finally all previous temporal elements are integrated within a language which we refer to as the underlying language.

As an example, Table 1 shows how the influential Allen's temporal representation approach (Allen, 1984) is described using the previous set of features.

3.2. *Related work*

In legal reasoning systems, time is usually represented as any other attribute. Some systems are provided with an *ad hoc* temporal representation which may range from few built-in functions to a whole temporal subsystem.

Gardner (1987), for instance, proposes a system for analysis of contract formation which includes a temporal component. The ontology is composed of time points and time intervals. A distinction is made between events and states (i.e. fluents). Time is treated as another argument. All the arguments are

Table 1. Description of Allen's temporal logic

Time ontology	Units: Interval Relations: {13 Qualitative interval relations}
Time theory	Interval existence Interval relations exclusivity Interval transitivity axioms
Temporal constraints	Formalism: Interval algebra (**IA**) Algorithm: **IA** path-consistency
Temporal qualification	Temporal reification
Temporal incidence theory	TIPs: {`holds`, `occurs`, `occurring`} Axioms: fluents homogeneity, events solidness
Underlying language	First-order logic

expressed through a proposition identifier, time among them, therefore the temporal qualification method here is a sort of token arguments method. Some relevant features, however, are less developed due to the bias towards the specific application: the time unit is fixed to days, only a few point-to-point relations are considered (some temporal relations such as 'follows' or 'immediately' are mentioned but not supported), and issues such as temporal constraints and temporal incidence are not considered at all.

KRIP-2 (Nitta *et al.*, 1988) is a system for legal management and reasoning in patent law whose language supports temporal representation. The ontology is also based on instants and periods, and includes both convex metric and qualitative interval temporal constraints. Events are qualified with time by using the form

event(Id, class, conditions, time)

Although *Id* looks like a token symbol, it is not used for temporal qualification since *time* is also an argument.

These temporal representation approaches turn out to be adequate for the purposes of the system they are defined in. However as a general approach to temporal representation in law they lack some of the following: (i) an explicit identification of requirements from legal domains; (ii) a consideration of the results in temporal reasoning in AI; and (iii) a rational decision on each of the issues involved in a temporal representation framework. In previous sections we have already gone over (i) and (ii). In next section we go over (iii) but, before that, we analyse two pieces of work that do take care of these three issues.

The first is the *event calculus* (EC) (Kowalski & Sergot, 1986), a temporal database management framework specified in PROLOG. Although not specifically intended for legal reasoning, EC has been used in several legal formalizations (Bench-Capon, 1988; Sergot, 1988). According to the above features, EC is described in Table 2.

The second is presented in the context of the *Chomexpert* system (Mackaay *et al.*, 1990; Poulin *et al.*, 1992), an application on the *Canadian Unemployment Insurance Law*. The features of the temporal representation language, called EXPERT/T, are summarized in Table 3.

Table 2. Event calculus

Time ontology	Units:	Instant, period
	Relations:	{<, =, >}
Time theory	Not defined	
Temporal constraints	Not defined	
Temporal qualification	For fluents:	*Temporal reification*
	For events:	*Token arguments*
Temporal incidence theory	TIPs:	{holds, holds_at}
	Axioms:	holds homogeneity
Underlying language	PROLOG	

Although both Tables 2 and 3 start from an analysis of temporal representation requirements, neither identifies *repeated temporal references*, *multiple time levels* and *modularity* as relevant issues to address. This is the reason why some of the decisions made on the temporal features are not the most well-suited for formalizing legal texts. Both proposals (in EC only for fluents) use temporal reification as temporal qualification method. In the next section we give a number of reasons to prefer the token arguments approach. Both use PROLOG as the underlying language. A shortcoming of languages purely based on logic (logic programming among them) is their inefficiency in handling constraints. Proof-driven inference procedures are discovered to perform poorly in constraint processing. The integration of a constraint specialist seems the natural way to overcome this problem. EC does not provide any 'machinery' for processing temporal constraints. Although the period primitive is part of the time ontology, period relations and interval algebra constraints (*à la* Allen) are not supported. EXPERT/T processes qualitative constraints using Allen's path-consistency propagation algorithm (Poulin *et al.*, 1992), but no type of metric constraints is supported.

Our approach here is based on integrating temporal constraints and the appropriate temporal qualification method into a logic-based language.

Table 3. EXPERT/T

Time ontology	Units:	Instant, Period
	Relations:	Qualitative point, qualitative interval, Qualitative point-interval, absolute dates
Time theory	Not defined	
Temporal constraints	Point and interval algebras Unary metric (absolute dates)	
Temporal qualification	*Temporal reification*	
Temporal incidence theory	TIPs:	{holds_on, occurs_at}
	Axioms:	Not defined
Underlying language	PROLOG	

4. Legal temporal representation

In this section we present our proposal called LTR. We analyse each of the features of Section 3.1: for each feature we select the choice that best fits the requirements identified in Section 2.

4.1. Time ontology: instants, periods and durations as dates

Primitive units. Most temporal expressions in legal domains are associated with a period of time (e.g. 'an offer being effective' in example 1, or the 'qualifying' and 'benefit' periods in example 2). Moreover, these expressions are often related by period relations such as 'a period of validity of an offer happens *during* its period of existence' or 'the qualifying period *immediately precedes* the benefit period'. Hence, it is natural to include the period as a time primitive. Do we also need instants? A brief analysis of legal texts yields several cases where the notion of instant appears:

1. The endpoints of the periods above are naturally associated with instants such as the moment where 'the offer becomes effective' or the time as of which 'the contract is no longer valid'.
2. Some *events* such as 'the offer *reaches* the offeree' are viewed as instantaneous. These are called *instantaneous events*.
3. Norms often involve conditions about the state of a certain fluent at a certain instant. For example, 'If ... and the offer is not withdrawn *at the moment* when it reaches the offeree and ... then ...'. Notice that, even if the 'reach' event is modelled as durable, the condition may still refer to the instant at the end of that period.
4. Whenever metric temporal relations are involved, they are often stated as constraints between instants, (e.g. 'a document sent by mail reaches its destination between 3 and 5 days later').

Besides instants and periods, since legal domains involve numeric relations the *duration* unit is also needed.

In practice, time in legal domains is expressed in *clock/calendar units*. Accordingly we define our instant, period and duration constants in terms of *dates*, where a *date* is defined as an indexed sequence of values for clock/calendar units:

$$date ::= [second''][minute'][hourh][dayd][weekw][monthm][yeary]]$$

For example, `00"15'21h2d10m96y`, `00"15'21h`, `21h2d10m96y`, `10w96y`, `96y` are well-formed dates. Some convenient shorthands are clock times (e.g. `00:15:21`) and calendar dates (e.g. `2/10/96`). Dates are used as both instant and duration constants. Period constants are defined as ordered pairs of dates. We use the conventional notation ()/[] to specify open/closed intervals. In addition, a set of indexed symbolic constants (`i1, i2, ... p1, p2, ...`) is included for each unit to express times not associated to any specific temporal proposition.

Granularity. The adequate time granularity may vary from one legal context to another, yet the basic structure of time and the properties of temporal constraints do not change. We address this issue by allowing the user to select the

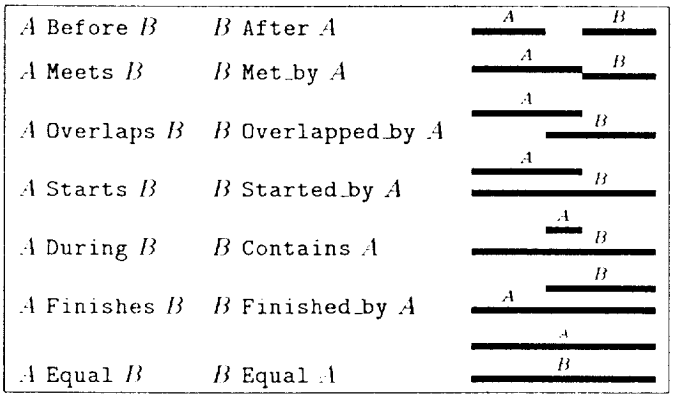

Figure 4.

appropriate granularity. Date constants will be interpreted as either an instant or a period according to what is specified by the directive `Granularity()`, which takes a clock/calendar unit as its only argument. The issues of combining various granularities or dynamically changing among from one granularity to another are not addressed.

Primitive relations. Our proposal is based on the following primitive temporal relations: the 3 qualitative point relations $\prec, =$ and $\succ$, the 5 qualitative point-interval relations `Before`, `Begin`, $\in$, `End`, `After`, the 13 qualitative interval relations (see Figure 4) and the duration relations $=$ and $\in$ used to express unary constraints only[9] (e.g. `duration(tt1) = 52w`, `begin(tt2) - end(tt1)` $\in$ `[3w,4w]`). Binary duration constraints are an issue of current research (Navarette & Marin, 1997).

Primitive functions. We define a set of logical functions between temporal units. Some of them are just the functional version of a temporal relation above:

`Begin, End:`	*period*	$\mapsto$ *instant*
`[], (), (], [):`	*instant* $\times$ *instant*	$\mapsto$ *period*
`Duration:`	*period*	$\mapsto$ *duration*

Besides, a set of *interpreted*[10] temporal functions is required in practice. These functions are not involved in the term unification process but they are computed at inference time. This set includes functions such as the following:

- *Date arithmetics*, e.g. `+`: *date* $\times$ *date* $\mapsto$ *date*
- *Date predicates*, e.g. `is_holiday`: *date* $\mapsto$ {*true*/*false*}
- *Date operations*, e.g. *next_holiday*: *date* $\mapsto$ *day*
- *Date transformations*, e.g. `week of`: *date* $\mapsto$ *week*
- *Date set operations*, e.g. `nth`, `latest`, `shorter_of`: *date-set* $\mapsto$ *date*

A list of them is given in Vila & Yoshino (1996).

Time theory. Provided with the set of dates as our underlying model of time. the only structural property of time that demands a specific discussion here is the dense/discrete one. Dense models are required in domains where *continuous*

change needs to be modelled such as *qualitative physics*. This is not the case of legal domains where the relevant changes are (viewed as) discrete (e.g. 'signing a contract' 'receiving an offer', 'interruption of earnings', etc.) and the dates set has a basic, indivisible granularity. Therefore we adopt a *discrete* model of time, which has two consequences. At the ontological level, we add two instant relations that are exclusive of discrete models: `Previous,Next`: *instant* × *instant*.[11] At the axiomatics level, we take a discrete time theory. It is based on *IP* (Vila, 1994b), a simple instant-period theory that accepts both discrete and dense models, plus few *discreteness* axioms. Both sets of axioms are given in Appendix A.

The 'immediate' relation. Immediate is a difficult temporal term to characterize because its meaning may vary from one context to another. It may mean 'in few seconds' or 'in few hours'. Even in a fixed context, it may not have a precise interpretation. Our proposal is based on regarding immediate as a qualitative relation somewhere between `Previous(Next)` and $\prec(\succ)$. This loose connection is formally specified by the following axioms over instants:

$$\mathbf{Im}_1\ i\ \texttt{ImmediateAfter}\ i' \Rightarrow i' \prec i$$
$$\mathbf{Im}_2\ i\ \texttt{ImmediateBefore}\ i' \Rightarrow i \prec i'$$
$$\mathbf{Im}_3\ i\ \texttt{Previous}\ i' \Rightarrow i\ \texttt{ImmediateBefore}\ i'$$
$$\mathbf{Im}_4\ i\ \texttt{Next}\ i' \Rightarrow i\ \texttt{ImmediateAfter}\ i'$$

When `Immediate` is adjoined to period relations, it is interpreted as one of the following two:

1. The period relation `Meets(Met_by)`.
2. The first (last) of the *set* of periods that follow (precede) the current period.

The appropriate choice will depend on the context. It is left to the responsibility of the language user. We formalize some instances of immediate relations in the examples below.

4.2. *Temporal constraints*

Given the indefiniteness of temporal relations in some legal domains[12] and the fact that existing temporal constraint algorithms scale down well in general, our framework includes almost all kinds temporal constraints:

- Qualitative constraints between instants (e.g. `begin(tt1)` $\leqslant$ `begin(tt2)`)
- Metric constraints over instants (e.g. `begin(tt2)-begin(tt1)` $\in$ `{[2d,3d] [1w,2w]}`)
- Qualitative constraints between periods (e.g. `period(tt3) Contains Overlaps period(tt2)`)
- Qualitative constraints between an instant and a period (e.g. `instant(tt2)` $\in$ `1/Oct/95`)
- Unary metric constraints over durations (e.g. `duration(P1)` = `52w`)

Besides representing indefinite temporal relations, temporal constraints can be used to maintain a partial representation over time. Consider, for instance, a fluent `f` that is holding now. Unless we have specific information, it may cease holding any time as of the current time. It can be expressed by a constraint similar to `end(f)` $\in$ `[now, +inf]`.

Temporal constraints are either unary or binary and in both cases the syntax has the form

time-term temporal-relation time-term

where the types of the time terms agree with the signature of the temporal relation. In unary constraints, one of the time terms is always ground. The formal syntax of the constraints is given in Vila & Yoshino (1996).

Temporal constraints are processed by representing them in a constraint network and applying the available efficient techniques for processing different classes of constraints: qualitative point (Ghallab & Mounir, 1989; van Beek, 1992; Gerevini & Schubert, 1995; Delgrande & Gupta, 1996), qualitative interval (van Beek, 1992) and metric point (Dechter *et al.*, 1991; Schwalb & Dechter, 1997). Also some progress has been achieved in combining metric-point and interval algebra constraints (Kautz & Ladkin, 1991; Meiri, 1991). This currently is an area of active research and forthcoming results can be straightforwardly integrated within our framework.

4.3. *Temporal qualification: token arguments*

Since *repeated temporal references* are pervasive in legal domains, temporal qualification methods based on tokens are more adequate. Among the two token-based methods proposed in the literature, *token arguments* is better suited to our needs here as we shall see in a moment. In *token arguments*, something like an offer of the contract `c` from `a` to `b` is formalized as `offer(c,a,b, ...,tt1)` where `tt1` is a constant symbol of the new *token* sort.[13] We call these atomic formula *token atoms*. To improve *readability* we emphasize the role of the token argument with some syntactic sugar: instead of `offer(c,a,b, ..., tt1)` (where `tt1` is a token term) we shall write

```
tt1 : offer(c,a,b, ...)
```

A set of functions, called *token temporal* fuctions,[14] which map tokens to their relevant times, is defined. For example, `begin(tt1)` denotes the initial instant of the token denoted by `tt1` and `period(tt1)` its period. TIPs are used to express that the temporal proposition is true at its associated time(s) as discussed in Section 4.4.

The token arguments method has several advantages:

1. Token symbols can be directly used as an argument of other predicates. In the above example, `tt1` can be used in `dispatch(tt1,a,b, ...)` to express that the offer `tt1` is dispatched from `a` to `b`.
2. Different levels of time are supported by diversifying the token temporal functions. For instance, we may have `begin_v(tt1)` to refer to *valid time* and `begin_t(tt1)` to refer to *transaction time*. At the implementation level a different temporal constraint network instance is maintained for each time level.
3. Token symbols can be used as the link to other knowledge modalities. For instance, in a multiple agents domain, the degree of belief of a proposition `p(...)` by an agent `a` can be represented by `belief(a,tt1)` where `tt1` is a token from `tt1:p(...)`. *Deontic modalities* can be represented by predicates (such as `O` for obligation and `P` for permission) that take a token as an

argument. Furthermore, we can distinguish between the time where the deontic relation holds and the time of the object in the relation. For example, consider that a legal person `a` is obligated to offer a contract `c` to `b`. We represent the offer by `tt1:offer(c,a,b, ...)`, its relevant instants by `begin(tt1)` and `end(tt1)`, the obligation by `tt2:O(a,tt1)` and the beginning and end instants of the obligation by `begin(tt2)` and `end(tt2)`.

To increase notation *conciseness* we define syntactic sugar that allows omitting token symbols whenever they are not strictly necessary (i.e. whenever there are no references to them). There are two cases. In the first case two or more token atoms are collapsed into one. For instance, the facts

```
tt1: offer(c,a,b, ...)
tt2: withdrawal(tt1)
tt3: reach(tt2,b)
```

in a rule that does not contain other references to `tt2`, can be rewritten as

```
tt1: offer(c,a,b, ...)
tt3: reach(withdrawal(tt1),b)
```

The second case is related with temporal incidence expressions and is explained in the next subsection.

4.4. Temporal incidence

We introduce the TIP holds to express holding of fluents (e.g. `holds(tt1)`) and occurs to express occurrence of events. We call these atomic formulas *incidence atoms*.

Holds incidence. There is a common agreement in the literature about the *homogeneity* of holding of fluents (McDermott, 1982; Allen, 1984; Shoham, 1987). Since our ontology includes both instants and periods, the holding of a fluent over a period should not constrain its holding at the period endpoints to avoid the *dividing instant problem* (Vila & Schwalb, 1996). These properties are captured by a simple axiom which, expressed in temporal reification form, is as follows:

$\forall f$ :: fluent, p :: period (`holds_on`(f,p) $\Rightarrow \forall i$ :: instant (`Within`(i,p) $\Rightarrow$ `holds_at`(f.i)))

An important convention we make at this point is what we call *token holds maximality*:

A fluent token denotes a maximal piece of time where that fluent is true.

A consequence of this convention is the following *Event Calculus* axiom:

Any two periods associated with the same fluent are either identical or disjoint.

In practice, one is interested in knowing whether the current token database entails that a certain fluent is true at a certain time. To this purpose we define the following TIPs:

`holds_on` (*fluent*, *period*)
`holds_at` (*fluent*, *instant*)

Notice that these are neither syntactic sugar of the above nor temporal reification TIPs, but they are new TIPs with the following existential meaning. Given a fluent f, a period p and an instant i:

$$\begin{aligned}&\texttt{holds\_on}(f,p) \equiv\\ &\quad \exists TT\ TT{:}f \wedge \texttt{holds\_on}(TT) \wedge\\ &\quad p\ \texttt{During Starts Finishes Equal period}(TT)\\ &\texttt{holds\_at}(f,i) \equiv\\ &\quad \exists TT\ TT{:}f \wedge (\texttt{holds\_on}(TT) \wedge i\ \texttt{Within period}(TT) \vee\\ &\quad \texttt{holds\_at}(TT) \wedge i = \texttt{instant}(TT))\end{aligned}$$

where TT is a variable of the *fluent token* sort.

Occurs incidence. There is no common agreement on the characterization of the occurrence of events (Allen, 1982; Shoham, 1982; Galton, 1991). As a matter of fact, no evidence on the need for any specific theory of events is found in practice. However, we keep `occurs` TIP to express the actual occurrence of an event and, thus, to allow describing events whose occurrence is unknown (e.g. to express the possibility or the obligation for that event to occur).

Some syntactic sugar for incidence expressions is defined to omit token symbols. The expression

```
TT:become-effective(...)
Occurs(TT)
instant(TT) = I
```

will be written as

```
Occurs(become-effective(...),I)
```

The formal syntax for incidence atoms is given in Vila & Yoshino (1996).

4.5. *Underlying language*

Our proposal is independent of the underlying language, as long as it is a many-sorted language. The sorts set must include our three temporal sorts (namely instants, periods and durations), and the two tokens sorts (namely fluent and event tokens).

In this section we address few additional relevant features:

Negation. Negation of token and incidence atoms will be handled by the standard mechanism of the underlying language. Negation of temporal constraints is less problematic since temporal constraints exhibit the following property:

Proposition 1 *In a constraint language that does not restrict non-convex constraints, any negated constraint can be expressed as an equivalent non-negated constraint form.*

For example $\neg(t \leq t') \equiv t > t'$, or $\neg(t - t' \in \{[3,5]\}) \equiv t - t' \in \{[-\infty, 3),(5, +\infty]\}$. Hence negated constraints will be asserted and queried by regular constraint propagation and entailment.

Token sets. Some applications require dealing with sets of temporal elements.[15] For instance, let us consider the following piece of text from example 2:

> ... (b) the period that begins on the commencement date of an *immediately* preceding **benefit period** and ends with the end of the week preceding the commencement of a benefit period under subsection 9(1).

Since for a given person there might be several *benefit periods*, a possible interpretation for 'immediately preceding benefit period ...' is, as noted in Section 4.1, 'the last of all benefit periods before ...'. Thus, we need to refer to the set of all those 'benefit period' tokens that are `Before` ... Coping with the notion of set requires higher order expressiveness. Some research has been done on extending first order languages in this direction (Maier, 1986; Kuper, 1987; Abiteboul & Grumbach, 1988; Chen & Warren, 1989; Kifer & Lausen, 1989; Chimenti, 1990). We restrict the development here to the context of a token-based approach where the set notion is used to specify sets of temporal tokens that satisfy a certain condition. The syntax we propose is as follows.[16]

`token set(`[*temporal atom*]$^+$`)`

where *temporal atom* can be either a token atom, an incidence atom or a temporal constraint. The token set operator binds the token variables appearing in the token atoms (e.g. the variable `TT3` in `TT3: benefit-period(TT1)`) to all those tokens of that relation that satisfy all the conditions inside the form. For instance, the example above is formalized as

```
token_set(TT3: benefit-period(TT1)
          period(TT3) Before Meets period(TT2))
```

We define a number of practical operators on sets of tokens. For instance, `latest` denotes the last token of that set according to the temporal ordering. These operators can be applied on token set variables (e.g. `latest(TT3)`). Some of these operators admit an alternative first-order formulation by splitting the conditions into different rules and using negation, however, this approach is clearly impractical.[17]

Token attributes. The token arguments method allows one to detach time from its temporal proposition. The same can be done for the remaining attributes of the proposition to enhance language flexibility. For example, we can refer to the offeror of `tt1: offer(c,a,b, ...)` by `offeror(tt1)`. Now attribute names are represented explicitly. It requires (i) *declaring* the attributes for each predicate.

```
Attribute(what,offer)
Attribute(offeror,offer)
Attribute(offeree,offer)
...
```

for which we shall use the shorthand

```
Attributes(offer,{what,offeror,offeree, ...})
```

Table 4. LTR

Time ontology	Units:	Instants, periods, durations with clock/calendar forms as constants.
	Relations:	{<, =, >, Before, After, Begin, End, E 13 Allen relations Next, Previous, Immediate Before, Immediate After}
Time theory	$\mathcal{IP}$ axioms + discreteness axioms + $\mathbf{Im}_{1:4}$ axioms (The axioms are given in appendices A.1 and A.2)	
Temporal constraints	Combined (metric) point-interval constraints	
Temporal qualification	*Token arguments*	
Temporal incidence theory	TIPs:	{holds, occurs, holds_at, holds_on}
	Axioms:	holds and holds_on homogeneity

and (ii) referring to the attributes of a particular token. Our `tt1: offer (c,a,b, ...)` can be regarded as a shorthand[18] for

```
what(tt1) = c
offeror(tt1) = a
offeree(tt1) = b
...
```

Summary. The set of choices that defines our proposal is summarized in Table 4.

5. Examples

In this section we illustrate the application of our approach as we revisit the two examples introduced in Section 1. We take a rule-based language as the underlying language without making any assumption about the inference regime. A set of facts in both the body and the head of a rule is interpreted as a conjunction. The marks [[...]] indicate pieces of text that have not been formalized either because their meaning is not clear, their main emphasis is not temporal or they are merely redundant. The mark `% Implicit` indicates pieces of formal knowledge that are not directly derived from the legal text. Ontological elements resulting from a conceptualization process are emphasized in **bold**. Temporal relations are underlined.

5.1. Formalizing the CISG example

The CISG is intended to provide a normative frame for international commerce. Part II of the law is devoted to the formation of contracts. For instance, it is used to determine when a contract is concluded. Queries like this can be answered in the LTR formalization we present next.

The predicate attributes used in the example are:

```
Attributes(contract,{offeror,offeree,class,type,qp-
  provision})
Attributes(offer,{what,offeror,offeree,is-irrevocable,
  offer-begin,offer-end})
```

```
Attributes(acceptance,{what})

Attributes(effective,{what})
Attributes(concluded,{what})
Attributes(withdrawn,{what})
Attributes(accepted,{what})

Attributes(become-effective,{what})
Attributes(become-concluded,{what})

Attributes(reach,{what,who})
Attributes(dispatch,{what,who,to-whom,type,stamped-date})
```

A granularity of days might seem fine enough for this example; however, some occurrences of the 'immediate' relation require moving to a finer granularity:

```
Granularity(second)
```

A law article is formalized as (a number of) rules that express the relations between occurrence of events under certain conditions and their effects in terms of the holding of derived fluents. For instance in example 1, '**Article 15(1)**. An **offer** becomes **effective** when it **reaches** the **offeree**' is formalized as

```
If    TT1: offer(C,OR,OE, …)
      TT2: reach(TT1,OE)
      Occurs(TT2)
      ¬Holds_at(withdrawn(TT1),instant(TT2))      % Implicit
then  Occurs(become-effective(TT1),instant(TT2))

If    TT2: become-effective(TT1)% Implicit
      Occurs(TT2)
then  Holds(effective(TT1),(instant(TT2),_))
```

Next we include few additional interesting articles also from CISG part II.

Article 18(2). An **acceptance** of an **offer** becomes **effective** at the moment the **indication of assent reaches** the **offeror**. An **acceptance** is **not effective** if the **indication of assent** does not **reach** the **offeror** within the time he has fixed [[or, if no time is fixed, within a reasonable time, due account being taken of the circumstances of the transaction. including the rapidity of the means of communication employed by the offeror.]]

```
If    TT1: offer(_,OR,_,_,OBegin,OEnd)
      TT2: acceptance(TT1)
      TT3: reach(TT2,OR)
      Occurs(TT3)
      instant(TT3) ∈ [OBegin,OEnd]
      Holds_at(accepted(TT1),instant(TT3))    % Implicit
then  Occurs(become-effective(TT1),instant(TT3))
```

Implicit from Article 18(2). When an acceptance of an offer of a contract becomes effective the contract becomes concluded.

```
if    TT2: become-effective(acceptance(offer(TT1, …)))
      Occurs(TT2)
```

```
then  Occurs(become-concluded(TT1), instant(TT2))

If    TT2: become-concluded(TT1) % Implicit
      Occurs(TT2)
then  Holds(concluded(TT1),(instant(TT2),_))
```

Article 18(2) (cont.). An **oral offer** must be[19] **accepted** immediately [[unless the circumstances indicate otherwise.]]

```
If      TT1: offer(…)
        TT2: dispatch(TT1,_,_,oral,_,_)
        Occurs(TT2)
then    offer-begin(TT1)←instant(TT2)
        offer-end(TT1)←ImmediateAfter(instant(TT2))
```

Article 20(2). Official **holidays** or non-business days occurring during the **period for acceptance** are included in calculating the **period**. However, if a notice of **acceptance** cannot be **delivered** at the address of the **offeror** on the last day of the period because that day falls on an official holiday or a non-business day at the place of business of the **offeror**, the **period** is extended until the first business day which follows.

```
If      TT2: offer(…)
        Is_holiday(offer-end(TT2))
then    offer-end(TT2)←next_holiday(offer-end(TT2))
```

Temporal database projection[20] would be sufficient to answer the intended queries. The bottom-up inference procedure would make intensive use of the specialized modules for (i) constraint processing and (ii) token management. The result will be a temporal map composed of instants and periods for the instances of events and fluents, together with the temporal constraints holding among them. For example, given the input formalized by the following facts

```
tt1:  contract(a,b,sale,machine,_)
tt2:  offer(tt1,a,b,_,[−;nf,+;nf],[−;nf,+;nf]),
      instant(tt2) ∈ 1/Oct/95
tt3:  dispatch(tt2)
tt4:  reach(tt2,b), instant(tt4) ∈ 8/Oct/95
tt5:  withdrawal(tt2)
tt6:  dispatch(tt5,a), instant(tt6) ∈ 7/Oct/95
tt7:  reach(tt5,b), instant(tt7) ∈ 11/Oct/95
tt8:  acceptance(tt2)
tt9:  dispatch(tt8,b), instant(tt8) ∈ 10/Oct/95
tt10: reach(tt8,a), instant(tt10) ∈ 12/Oct/95
```

The time map shown in Figure 5 would be generated. The query 'Is the contract concluded' will be affirmatively answered by `YES, as of October 12 '95`. The sequence of rules involved in deriving token `tt1.2: concluded(tt2)` can be easily recorded and returned as justification.

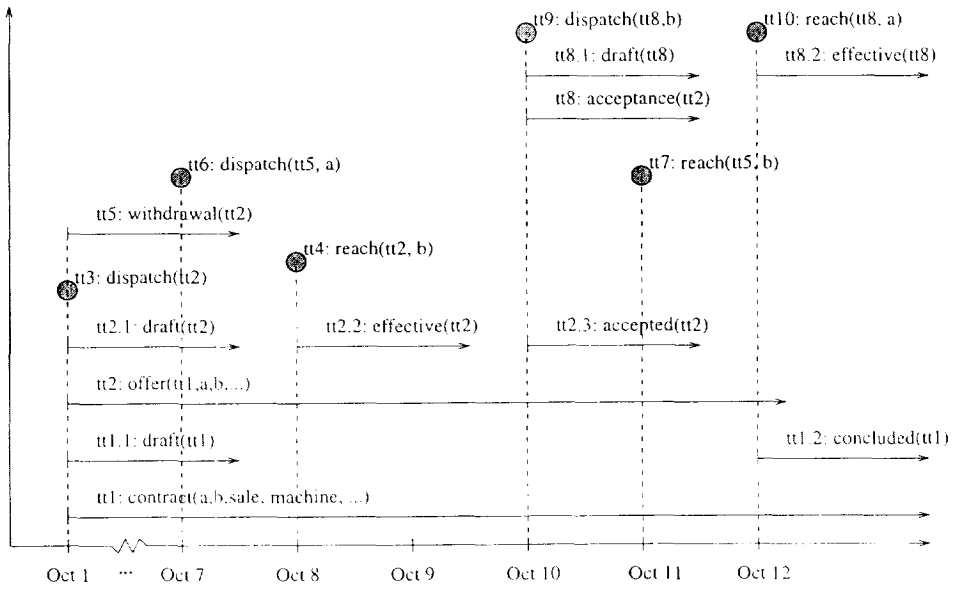

Figure 5. CISG example.

5.2. *Formalizing the Canadian Unemployment Insurance Law example*

A key section of the *Canadian Unemployment Insurance Law* (Poulin *et al.*, 1992) is intended to determine whether a person is eligible for benefits or not. It involves determining a *qualifying period* (the period during which the person has been employed) and a *benefit period* (the period during which the person should receive benefits).

The following predicate attributes need to be declared:

```
Attributes(insured-person,{...})
Attributes(benefit-period,{whom})
Attributes(qualifying-period,{whom})

Attributes(interruption-of-earnings,{what})
Attributes(initial-claim,{what})
```

For a proper formalization of the temporal aspects of this act, a granularity of days is fine enough.

```
Granularity(day)
```

Next we show the sections that address the assessment of the benefit and qualifying periods and their formalization in **LTR:**

Section 7(1). [...] the **qualifying period** of an **insured person** is the shorter of (a) the period of fifty-two weeks that immediately precedes the commencement of a **benefit period** under subsection 9(1), and
(b) the period that begins on the commencement date of an immediately preceding **benefit period** and ends with the end of the week preceding the commencement of a **benefit period** under subsection 9(1).

Table 5. LTR

Time ontology	Units:	Instants, periods, durations with clock/calendar forms as constants.
	Relations:	{<, =, >, Before, After, Begin, End, E 13 Allen relations Next, Previous, Immediate Before, Immediate After}
Time theory	$\mathcal{IP}$ axioms + discreteness axioms + $\mathrm{Im}_{1 \leq 4}$ axioms (The axioms are given Appendices A1 and A2)	
Temporal constraints	Combined (metric) point-interval constraints	
Temporal qualification	*Token arguments*	
Temporal incidence theory	TIPs:	{holds, occurs, holds_at, holds_on}
	Axioms:	holds and holds_on homogeneity

```
If    TT1: insured-person()
      TT2: benefit-period(TT1)
      duration(P1) = 52w
      P1 Meets period(TT1)
      token_set(TT3: benefit-period(TT1)
        period(TT3) Before Meets period(TT2))
      begin(P2) = begin(latest(TT3))
      end(P2) ← end_of_week(week_before(week_of(begin(TT2))))
then  TT5: qualifying-period(TT1)
      period(TT5) ← shorter_of({P1,P2})
```

Section 9(1). [...] A benefit period begins on the Sunday of the week in which
(a) the **interruption of earnings** occurs, or
(b) the **initial claim** for benefit is made,
whichever the later.

```
If    TT1: insured-person()
      TT2: interruption-of-earnings(TT1)
      Occurs(TT2)
      TT3: initial-claim(TT1)
      Occurs(TT3)
then  TT4: benefit-period(TT1)
      begin(TT4) ← sunday_of(week_of(latest
        of(instant(TT2),instant(TT3))))
```

6. Conclusions

We explored the representation of time and temporal information in legal domains in the tradition of using logic to formalize law. We propose **LTR**, a temporal representation framework described by the following choices on the temporal reasoning features shown in Table 5.

Our approach is independent of the underlying representation language and the specific legal reasoning application. We discussed its adequacy with respect to the requirements identified in legal domains. **LTR** is currently being used

within a rule-based language in the formalization of the *Convention for International Sale of Goods*.

In this work we did not address the issues of (i) representing periodic occurrences, (ii) temporal non-monotonic reasoning, and (iii) handling time of legal statutes. For instance, tasks that involve meta-reasoning about the validity of statutes and laws over time are out of the scope of our approach. This is matter of our current research.

Acknowledgements

We are thankful to Trevor Bench-Capon, Eddie Schwalb, Marek Sergot and anonymous referees for helpful discussions. Lluís Vila was partially supported by MEC of Spain (grant EX-94 77909683) and IIIA-CSIC.

Notes

1. It can either be before the tort, at the tort time, before the trial, until damages have been paid or even after that.
2. 'Offer' can be modelled as an event, if we refer to the offer object, or as a fluent if we refer to the 'existence of the offer'.
3. As opposed to a function whose interpretation is time-dependent.
4. Retro-active effects are also related to the issue of law change.
5. Labels in **bold** indicate framework components and those in *italics* are either a set of axioms or a set of algorithms based on a set of axioms. Imbricated blocks denote *part-of* dependencies whereas arrows represent a *design* dependency, i.e. the design of the source has implications on the design of the target.
6. Note that instant-to-instant numeric relations, period lengths and absolute times (such as dates) can all be regarded as durations.
7. 'Real time' here means time that can be measured by an existing device.
8. This is the approach classically used in databases.
9. Although the relations are binary, only one of the arguments will be a duration variable.
10. Interpreted functions are also referred as *built-in* functions or *operators*.
11. These relations will also be used in their functional form as time operators (e.g. `begin(tt1) = Next(end(tt2))`).
12. Although in most legal applications only some specific classes of temporal constraints are involved, different applications require different types of constraints. Moreover, some few domains (such as labour law) where the temporal issue is paramount and data may be imprecise, involve all kinds of temporal constraints.
13. The idea behind token arguments is similar to the *Compound Predicate Formula* approach (Yoshino, 1994) when applied to temporal pieces of information.
14. To be distinguished from the temporal functions in Section 4.1 with similar names but different signature.
15. This issue is not included in the requirements list (Section 2) because the notion of set is not strictly a temporal representation feature, but the notion of set of temporal elements is relevant here as we discuss it in this section.
16. We are not particularly happy with this syntax since it does not follow a pure declarative style, but it turns out to be adequate in practice.
17. As an exercise, you may try to use this approach to specify the operator `4th` which selects the 4th token that satisfies certain conditions.
18. The translation will take the order of the attributes from an explicit declaration supported by the underlying language.
19. Notice that 'must be' here does not denote obligation but a temporal constraint.
20. As in the TMM system (Dean & McDermott, 1987; Schrag & Boddy, 1991) for example.
21. Notice that $\mathbf{IP}_1$ is actually redundant since it can be derived from $\mathbf{IP}_2$. We include it for clarity.

References

Abiteboul, S. & Grumbach, S. (1988) A logic-based language for complex objects, in *Proc. of the Int. Conf. on Extending Database Technology.*

Allen, J. (1984) Towards a general theory of action and time, *Artificial Intelligence*, 23, pp. 123–154.

Allen, J. & Hayes, P. (1985) A common-sense theory of time, in *Proc. IJCAI'85*, pp. 528–531 (Morgan Kaufman).

Allen, J. & Hayes, P. (1989) Moments and points in an interval-based temporal logic, *Computational Intelligence*, 5, pp. 225–238.

Bacchus, F., Tenenberg, J. & Koomen, J. (1991) A non-reified temporal logic, *Artificial Intelligence*, 52, pp. 87–108.

Bench-Capon, T., Robinson, G., Routen, T. & Sergot, M. (1988) Logic programming for large scale applications in law: a formalization of supplementary benefit legislation, in Hayes, Michie, and Richards (Eds), *Machine Intelligence*, pp. 209–260 (Oxford, Oxford University Press).

Blumsohn, A. (1991) *Three Essays on Law and Information in the Law of Damages.* Dissertation Information Service, UMI.

Bochman, A. (1990) Concerted instant-interval temporal semantics i: Temporal ontologies, *Notre Dame Journal of Formal Logic*, 31(3), pp. 403–414.

Bulygin, E. (1982) Time and validity, in A. Martino (Ed.) *Deontic Logic, Computational Linguistics and Information Systems*, Vol. II, pp. 65–81 (North-Holland).

Chemilieu-Gendrau, M. (1987) *Le Role du Temps dans la Formation du Droit International* (Droit International, Editions Pedone).

Chen, W. & Warren, D. (1989) C-logic of complex objects, in *Proc. of 8th ACM-SIGACT-SIGMOD-SIGART Symposium of Principles of Database Systems.*

Chimenti, D. (1990) The IDL system prototype, *IEEE Transactions on Knowledge and Data Engineering*, 2(1), pp. 78–90.

Dean, T. & McDermott, D. (1987) Temporal database management, *Artificial Intelligence*, 32, pp. 1–55.

Dechter, R., Meiri, I. & Pearl, J. (1991) Temporal constraint networks, *Artificial Intelligence*, 49, pp. 61–95.

Delgrande, J. & Gupta, A. (1996) A representation for efficient temporal reasoning, in *Proc. AAAI'96*, Menlo Park, 1996, pp. 381–388 (AAAI Press/The MIT Press).

Galton, A. (1991) A critical examination of Allen's theory of action and time, *Artificial Intelligence*, 42, pp. 159–188.

Galton, A. (1991) *Reified temporal theories and how to unreify them, In Proc. IJCAI'91*, pp. 1177–1182 (Morgan Kaufman).

Gardner, A. (1987) *An Artificial Intelligence Approach to Legal Reasoning* (Cambridge, MA, The MIT Press).

Gerevini, A. & Schubert, L. (1995) Efficient algorithms for qualitative reasoning about time, *Artificial Intelligence*, 74(3), pp. 207–248.

Ghallab, M. & Mounir Alaoui, A. (1989) Managing efficiently temporal relations through indexed spanning trees, in *Proc. IJCAI'89*. pp. 1297–1303 (Morgan Kaufman).

Hamblin, C. (1972) Instants and intervals, in J. Fraser (Ed.), *The Study of Time*, pp. 325–331 (Springer-Verlag).

Haugh, B.A. (1987) Non-standard semantics for the method of temporal arguments, in *Proc. IJCAI'87*, pp. 449–454 (Morgan Kaufman).

Kautz, I. & Ladkin, P. (1991) Integrating metric and qualitative temporal reasoning, in *Proc. AAAI'91*, pp. 241–246 (AAAI Press).

Kifer, M. & Lausen, G. (1989) F-logic: a higher-order language for reasoning about objects, in *Proc. of 8th ACM-SIGACT-SIGMOD-SIGART Symposium of Principles of Database Systems* (Morgan Kaufman).

Kowalski, R. & Sergot, M. (1986) A logic-based calculus of events, *New Generation Computing*, 3.

Kuper, G. (1987) Logic programming with sets, in *Proc. of 6th ACM-SIGACT-SIGMOD-SIGART Symposium of Principles of Database Systems* (Morgan Kaufman).

Mackaay, E., Poulin, D., Fremont, J., Bratley, P. & Deniger, C. (1990) The logic of time in law and legal expert systems, *Ratio Juris*, 3(2), pp. 254–271.

Maier, D. (1986) A logic for objects, in *Proc. of the Workshop on Foundations of Deductive Database and Logic Programming* (Morgan Kaufman).

McCarty, L.T. (1995) Some requirements on an action language for legal discourse (position paper), in *Spring Symposium Series '95: Extending Theories of Action*, pp. 136–138 (AAAI Press).

McDermott, D. (1982) A temporal logic for reasoning about processes and plans, *Cognitive Science*, 6, pp. 101–155.

Meiri, I. (1991) Combining qualitative and quantitative constraints in temporal reasoning, in *Proc. AAAI'91* (AAAI Press).
Morris, R., Shoaf, W. & Khatib, L. (1997) Domain independent temporal reasoning with recurring events, *Computational Intelligence*, 12, pp. 450–477.
Navarrete, I. & Marin, R. (1997) Qualitative temporal reasoning with points and durations, in *Proc. IJCAI'97*, pp. 1454–1459.
Newton-Smith, W. (1980) *The Structure of Time* (London, Routledge & Kegan Paul).
Nitta, K., Nagao, J. & Tetsuya, M. (1988) A knowledge representation and inference system for procedural law, *New Generation Computing*, 5, pp. 319–359.
Poulin, D., Mackaay, E. Bratley, P. & Fremont, J. (1992) Time server—a legal time specialist, in A. Martino (Ed.) *Expert Systems in Law*, pp. 295–312.
Schrag, B. & Boddy, M (1991) β-tmm functional description. Technical report, Honeywell SRC.
Schwalb, E. & Dechter, R. (1997) Processing temporal constraint networks, *Artificial Intelligence*, in press. Also available as UCI technical report (1995).
Sergot, M. (1988) Representing legislation as logic programs, in Hayes, Michie & Richards (Eds), *Machine Intelligence*, pp. 209–260 (Oxford, Oxford University Press).
Sergot, M. (1995) Tutorial notes: using logic for knowledge representation in legal knowledge based system. 5th International Conference on Artificial Intelligence in Law, Maryland, May 1995.
Shoham, Y. (1987) Temporal logics in AI: semantical and ontological considerations, *Artificial Intelligence*, 33, pp. 89–104.
Tansel, A., Clifford, J., Gadia, S., Jajodia, S., Segev, A. & Snodgrass, R. (1993) *Temporal Databases* (Benjamin/Cummings Publishing).
Tsang, E. (1987) Time structures for AI, In *Proc. IJCAI'87*, pp. 456–461 (Morgan Kaufman).
van Beek, P. (1992) Reasoning about qualitative temporal information, *Artificial Intelligence*, 58, 297–326.
van Benthem, J. (1991) *The Logic of Time* (Dordrecht, Kluwer Academic, 2nd edn).
Vila, L. (1994a) A survey on temporal reasoning in artificial intelligence, *AI Communications*, 7(1), pp. 4–28.
Vila, L. (1994b) *IP*: An instant-period based theory of time, in R. Rodriguez (Ed.) *Proc. ECAI'94 Workshop on Spatial and Temporal Reasoning* (IEEE Computer Society Press).
Vila, L. & Reichgelt, H. (1996) The token reification approach to temporal reasoning, *Artificial Intelligence*, 83(1), pp. 59–74.
Vila, L. & Schwalb, E. (1996) A theory of time and temporal incidence based on instants and periods, in *Proc. of the Intl. Workshop on Temporal Representation and Reasoning (TIME'96)*, pp. 21–28 (IEEE Computer Society Press).
Vila, L. & Yoshino, H. (1996) Time in Automated legal reasoning (the long report). Technical Report 96-57, UC Irvine.
Vilain, M., Kautz, H. & van Beek, P. (1989) Constraint propagation algorithms for temporal reasoning: a revised report, in D.S. Weld & J. de Kleer (eds), *Readings on Qualitative Reasoning about Physical Systems*, pp. 373–381 (Morgan Kaufman).
Walker, A. (1948) Durées et instants, *Revue Scientifique*, 85, pp. 131–134.
Wetprasit, R., Sattar, A. & Khatib, L. (1996) Reasoning with sequences of events (an extended abstract), in *Proc. of the Intl. Workshop on Temporal Representation and Reasoning (TIME'96)*, pp. 36–38 (IEEE Computer Society Press).
Yoshino, H. (1994) Representation of legal knowledge by compound predicate formula, in D.T.C. Biagioli & G. Sartor (Eds) *Proc. of the Workshop on Legal Application of Logic Programming, ICLP'94*, pp. 128–137 (MIT Press).
Yoshino, H. (1994) Representation of legal knowledge by legal flowchart and compound predicate formula. Technical Report TM-1298. ICOT.

Appendix: a discrete theory of time

A.1. IP theory

$\mathcal{IP}$ is defined upon a structure composed of two sorts of symbols instants (I) and periods (P), which are formed by two infinite disjoint sets of symbols, and three primitive binary relation symbols, $\prec : \mathcal{I} \times \mathcal{I}$ and $\texttt{begin}, \texttt{end}: \mathcal{I} \times \mathcal{P}$.

The first-order axiomatization of $\mathcal{IP}$ is as follows:

$\mathbf{IP}_1$ $\neg(i \prec i)$
$\mathbf{IP}_2$ $i \prec i' \Rightarrow \neg(i' \prec i)$
$\mathbf{IP}_3$ $i \prec i' \wedge i' \prec i'' \Rightarrow i \prec i''$
$\mathbf{IP}_4$ $i \prec i' \vee i \prec i' \vee i = i'$
$\mathbf{IP}_{5.1}$ $\exists i'\ (i' \prec i)$
$\mathbf{IP}_{5.2}$ $(\exists i'(i \prec i')$
$\mathbf{IP}_6$ $\mathtt{begin}(i,p) \wedge \mathtt{end}(i',p) \Rightarrow i \prec i'$
$\mathbf{IP}_{7.1}$ $\exists i\ \mathtt{begin}(i,\ p)$
$\mathbf{IP}_{7.2}$ $\exists i\ \mathtt{end}(i,\ p)$
$\mathbf{IP}_{8.1}$ $\mathtt{begin}(i,p) \wedge \mathtt{begin}(i',p) \Rightarrow i = i'$
$\mathbf{IP}_{8.2}$ $\mathtt{end}(i,\ p) \wedge \mathtt{end}(i',p) \Rightarrow i = i'$
$\mathbf{IP}_9$ $i \prec i' \Rightarrow \exists p\ (\mathtt{begin}(i,p) \wedge \mathtt{end}(i',p))$
$\mathbf{IP}_{10}$ $\mathtt{begin}(i,p) \wedge \mathtt{end}(i',p) \wedge$
$\wedge \mathtt{begin}(i,p') \wedge \mathtt{end}(i',p') \Rightarrow p = p'$

$\mathbf{IP}_1 \div \mathbf{IP}_4$ are the conditions for $\prec$ to be a *strict linear order*—namely irreflexive, asymmetric, transitive and linear—relation over the instants.[21] $\mathbf{IP}_5$ imposes unboundness on this ordered set. $\mathbf{IP}_6$ orders the extremes of a period. This axiom rules out durationless periods that are not necessary since we have instants as a primitive. The pairs of axioms $\mathbf{IP}_{7\_}$ and $\mathbf{IP}_{8\_}$ formalize the intuition that the beginning and end instants of a period always *exist* and are *unique*, respectively. Conversely, axioms $\mathbf{IP}_9$ and $\mathbf{IP}_{10}$ close the connection between instants and periods by ensuring the *existence* and *uniqueness* of a period for a given ordered pair of instants.

See Vila (1994b) for a characterization of the models and relation with other time theories.

A.2. Discreteness axioms

The discreteness axioms under an unbounded time are as follows:

$\mathbf{IP}_{di1}$ $i\mathtt{Previous}\ i' \Leftrightarrow i'\mathtt{Next}\ i$
$\mathbf{IP}_{di2}$ $i\mathtt{Previous}\ i' \Rightarrow i \prec i'$
$\mathbf{IP}_{di3}$ $\exists i'\ i\ \mathtt{Previous}\ i'$
$\mathbf{IP}_{di3}$ $\exists i'\ i\ \mathtt{Next}\ i'$
$\mathbf{IP}_{di4}$ $i\mathtt{Previous}\ i' \Rightarrow \neg\exists\ i''(i \prec i'' \prec i')$