Merging structured text using temporal knowledge

Anthony Hunter
Department of Computer Science
University College London
Gower Street
London WC1E 6BT, UK
a.hunter@cs.ucl.ac.uk

January 11, 2002

Abstract

Structured text is a general concept that is implicit in a variety of approaches to handling information. Syntactically, an item of structured text is a number of grammatically simple phrases together with a semantic label for each phrase. Items of structured text may be nested within larger items of structured text. Much information is potentially available as structured text including tagged text in XML, text in relational and object-oriented databases, and the output from information extraction systems in the form of instantiated templates. In a previous paper, we presented a framework for merging items of potentially inconsistent structured text [Hun00a]. In this paper, we extend that framework by considering the need to use temporal knowledge. The primary aim of this paper is to specify the language and types of axioms required in this temporal knowledge.

Keywords: Merging information, structured text, semi-structured text, logic-based inconsistency management techniques, logic-based fusion.

1 Introduction

Syntactically, an item of structured text is a data structure containing a number of grammatically simple phrases together with a semantic label for each phrase. The set of semantic labels in a structured text is meant to parameterize a stereotypical situation, and so a particular item of structured text is an instance of that stereotypical situation. Using appropriate semantic labels, we can regard a structured text as an abstraction of an item of text.

For example, news reports on corporate acquisitions can be represented as items of structured text using semantic labels including `buyer`, `seller`, `acquisition`, `value`, and `date`. Each semantic label provides semantic information, and so an item of structured text is intended to have some semantic coherence. Each phrase in structured text is very simple — such as a proper noun, a date, or a number with unit of measure, or a word or phrase from a prescribed lexicon. For an application, the prescribed lexicon delineates the types of states, actions, and attributes, that could be conveyed by the items of structured text. An example of structured text is given in Figure 1.

```
⟨bid report :
      ⟨bid date : 30 May 2000⟩
      ⟨buyer :
            ⟨company :
                  ⟨name : France Telecom⟩
                  ⟨capitalization : 150 Billion Euros⟩
                  ⟨headquarters : Paris, France⟩
            : company⟩
      : buyer⟩
      ⟨target :
            ⟨company :
                  ⟨name : Orange⟩
                  ⟨headquarters : Bristol, UK⟩
            : company⟩
      : target⟩
      ⟨bid type : agreed⟩
      ⟨bid value : 40 Billion Euros⟩
      ⟨report info :
            ⟨source : Orange website⟩
            ⟨URL : www.orange.co.uk⟩
            ⟨report date : 31 May 2000⟩
      : report info⟩
: bid report⟩
```

Figure 1: An example of a news report in the form of structured text.

Much material is potentially available as structured text. This includes items of text structured using XML tags, and the output from information extraction systems given in templates (see for example [CL96, Gri97, ARP98]). The notion of structured text also overlaps with semi-structured data (for reviews see [Abi97, Bun97]).

Whilst structured text is useful as a resource, there is a need to develop techniques to handle, analyse, and reason with it. In particular, we are interested in merging potentially inconsistent sets of news reports [Hun00a, Hun02a], and deriving inferences from potentially inconsistent sets of news reports [Hun00b, Hun00c, BH01, Hun02b].

1.1 The need for temporal knowledge in merging

In this paper, we want to consider the role of temporal knowledge in facilitating merging of structured news reports. To illustrate some of our needs, consider a set of news reports, in the following situations.

- The set of reports refer to the same time point of a subject. As an example, consider a report from a newspaper and a report from a radio station on the weather in London today. So the weather in London is the subject, and the time point is today. However, the TV report and radio reports have not necessarily been made at the same time. So for example, the newspaper report could have been published three days ago, whereas the radio report could have been broadcast today.

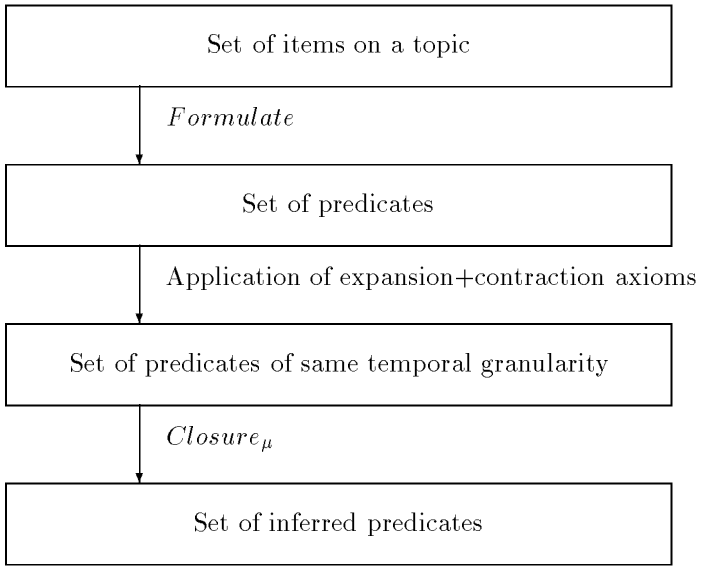

Figure 2: This figure summarizes the merging framework. $Formulate$ takes a set of structured reports and returns a set of logical predicates (covered in Section 2). The application of the expansion and contraction axioms allows for information to be inferred of common temporal granularity (covered in Section 4). $Closure_\mu$ takes a set of predicates and returns all inferred predicates that can be inferred using the domain knowledge and merging axioms in μ (covered in Section 5).

- The set of reports refer to the same time intervals[1] of a subject. As an example, consider a report from a TV station on the weather in London today and a report from a radio station on the weather in London today. So the weather in London is the subject, and the subject time interval is today. As with the example above, the TV report and radio reports have not necessarily been broadcast at the same time.
- The set of reports are broadcast at the same time point. As an example, consider a radio broadcast today with a long-range weather report for weather in Europe over the next month, and a TV broadcast today with a weather report for London today. Here, the granularity of the periods of the subjects of the reports is different.
- The set of reports are broadcast during the same time interval. As an example, consider a radio broadcast today with a weather report for weather in Europe over the next month, and a TV broadcast yesterday with a long-range weather report for weather in Europe over the next week. Here, the granularity of the periods of the subjects of the reports is different.

To handle these, and a number of related issues, we need to consider how we can enhance our approach to reasoning with the temporal aspects of news reports.

1.2 Our approach to merging

In this paper, we consider a merging system for handling a collection of reports, such as weather reports, together with merging axioms (which specify how information should be combined) and domain knowledge. We assume that this collection of reports may be heterogeneous in structure. Furthermore, we assume the following:

- Each report incorporates an explicit time point at which the report was logged. We call this the **log time**. For example, a report in the 15/5/00 issue of the Financial Times would have 15/5/00 as the log time. As another example, a report in the December 2000 issue of the London Review of Books would have December 2000 as the log time.
- Each report can incorporate an explicit time point or time interval delineating the period of the event being covered by the report. We call this the **event time**. For example, a report on Microsoft's financial performance over the period from 1/1/00 to 31/3/00 would have the event time of 1/1/00 - 31/3/00. As another example, a report in the stock market may refer to the price of a stock at 14.00 hrs, and so 14.00hrs would be the event time.

We represent each news report as a set of monadic and binary predicates. We have assumed the words and phrases are sufficiently simple and restricted to not require natural language processing. We will represent each word or phrase by a constant symbol in the logic, and each semantic label as a relation symbol. A set of merging axioms and domain knowledge represented in classical logic is then used to derive a set of merged predicates. To do this we need to

1. Reduce all the information in the news reports to common granularities of time. For this, we present information in pointbased time, and this may involve rewriting information that is in intervalbased time.
2. Infer predicates that capture information merged from the predicates representing the individual news reports. For this, we need to conjoin information and to select the information from some news reports in preference to information from other reports if conflicts arise between them. To do this we will use various preference criteria including:

[1] In Section 2, we address differences between formalizing time points and time intervals.

- preferences over sources
- preference for more recently logged information
- preference for news that refers to a finer grained event time

3. Select the monadic and binary predicates from the inferences that capture the merged information.

Given these logical formulae, we show how we can construct a timeline, based on the natural numbers with the usual ordering, to which we can associate information from the news reports. This gives a set of linear discrete temporal models which correspond to tense or US temporal logic models. This correspondence with temporal logic provides various options for further temporal reasoning with the predicates.

To summarize the essence of our requirements, we want to reason with the different granularities of time points and time intervals, where there may be heterogeneous formats for the temporal information, and interchange between them where possible. To support this reasoning we will adapt and extend a formalization of time based on the natural numbers. We will present this formalization, and show how we can use it in a framework for merging, and reasoning with, structured news reports. We summarize the merging framework in Figure 2.

1.3 Other approaches to merging

Our logic-based approach differs from other logic-based approaches for handling inconsistent information such as belief revision theory (e.g. [Gar88, DP98, KM91, LS98]), knowledgebase merging (e.g. [KP98, BKMS92]), and logical inference with inconsistent information (e.g. [MR70, Bre89, CRS93, BCD$^+$93, BDP95]). These proposals are too simplistic in certain respects for handling news reports. Each of them has one or more of the following weaknesses: (1) One-dimensional preference ordering over sources of information — for news reports we require finer-grained preference orderings; (2) Primacy of updates in belief revision — for news reports, newest reports are not necessary the best reports; and (3) Weak merging based on a meet operator — this causes unnecessary loss of information. Furthermore, none of these proposals incorporate actions on inconsistency or context-dependent rules specifying the information that is to be incorporated in the merged information, nor do they offer a route for specifying how merged reports should be composed.

Other logic-based approaches to fusion of knowledge include the KRAFT system which uses constraints to check whether information from heterogeneous sources can be merged [PHG$^+$99, HG00]. If knowledge satisfies the constraints, then the knowledge can be used. Failure to satisfy a constraint can be viewed as an inconsistency, but there are no actions on inconsistency. Merging information is also an important topic in database systems. A number of proposals have been made for approaches based in schema integration (e.g. [PM98]) and conceptual modelling for information integration based on description logics [CGL$^+$98b, CGL$^+$98a, FS99]. These differ from our approach in that they do not seek an automated approach that uses domain knowledge for identifying and acting on inconsistencies. Heterogeneous and federated database systems also could be relevant in merging multiple news reports, but they do not identify and act on inconsistency in a context-sensitive way [SL90, Mot96, CM01], though there is increasing interest in bringing domain knowledge into the process (e.g. [Cho98, SO99]).

Our approach also goes beyond other technologies for handling news reports. The approach of wrappers offers a practical way of defining how heterogeneous information can be merged (see for example [HGNY97, Coh98, SA99]). However, there is little consideration of problems of conflicts arising between sources. Our approach therefore goes beyond these in terms of formalizing reasoning with inconsistent information and using this to analyse the nature of the news report and for formalizing how we can act on inconsistency.

1.4 The rest of the paper

As an overview of the rest of the paper, we will review a formalization of structured reports, and then present a new framework for representing and reasoning with structured reports using temporal knowledge. This framework is based on a first-order classical logic presentation of temporal reasoning. We present a temporal semantics for representing and reasoning with the merged information that is based on US temporal logic.

2 Formalizing structured news reports

In this section, we will formalize the notions of structured text. We will then explain how we can translate each item of structured text into a set of literals. Sections 2.1 and 2.2 are a review of [Hun00a, Hun00c]. Then in Section 2.3, we consider the format for capturing temporal information in structured news reports using timestamps. We then introduce, in Section 2.4, timestamp equivalence axioms to relate different formats for temporal information, and in Section 2.5 pointwise axioms for translating intervalbased information into pointbased information which is our common basis for representing temporal information in merging.

2.1 Structured text

Here we adopt some basic definitions that can be viewed as an adaptation of concepts used in a range of technologies including XML, relational and object-oriented databases, language engineering, and knowledgebased systems.

Definition 2.1 *A* **word** *is a string of alphanumeric characters, and a* **phrase** *is a string of one or more words. A* **text entry** *is either a phrase or a null value. A* **semantic label** *is a phrase. In this paper, we assume the set of semantic labels and the set of text entries are disjoint.*

Example 2.1 *Examples of words include* `John`, `France`, `drive`, `happy`, `23`, *and* `3i`, *and examples of phrases include* `University of London`, `John`, `23 April 1999`, *and* `warm and sunny`.

Definition 2.2 *If ϕ is a semantic label, and ψ is a text entry, then $\langle \phi : \psi \rangle$ is an* **atomic feature**.

Example 2.2 *An example of an atomic feature is* ⟨`today's weather: sunny and windy`⟩.

Definition 2.3 **Complex features** *are defined as follows: (1) if $\langle \phi : \psi \rangle$ is an atomic feature, then $\langle \phi : \psi \rangle$ is a complex feature; and (2) if ϕ is a semantic label and $\sigma_1, ..., \sigma_n$ are complex features, then $\langle \phi : \sigma_1, ..., \sigma_n : \phi \rangle$ is a complex feature. An* **item of structured text** *is just a complex feature.*

Example 2.3 *An example of a complex feature is:*

```
⟨weather report:
          ⟨date: 23 April 1999⟩,
          ⟨city: London⟩,
          ⟨today: cold and wet⟩,
          ⟨tomorrow: sunny and windy and cool⟩
          ⟨log: 23 April 1999⟩
:weather report⟩
```

Note, indentations and line-breaks make no difference to the structured text. They are just added to make the layout more readable.

Definition 2.4 *Let $\langle \phi : \sigma_1, .., \sigma_n : \phi \rangle$ be a complex feature. The* **sub** *function is defined as follows:*

$$Sub(\langle \phi : \sigma_1, .., \sigma_n : \phi \rangle) = \{\sigma_1, .., \sigma_n\} \cup Sub(\sigma_1) \cup .. \cup Sub(\sigma_n)$$

$$Sub(\langle \phi : \psi \rangle) = \{\langle \phi : \psi \rangle\}$$

For complex features α, β, α is a complex feature in β iff $\alpha \in Sub(\beta)$.

We can consider a complex feature as a tree where the semantic labels are non-leaf nodes and the text entries are leaves.

Definition 2.5 *Let $\langle \phi : \sigma_1, .., \sigma_n : \phi \rangle$ be a complex feature. The semantic label ϕ is the* **parent** *of the complex features $\sigma_1, .., \sigma_n$. The elements of $Sub(\langle \phi : \sigma_1, .., \sigma_n : \phi \rangle)$ are the* **offspring** *of ϕ.*

Definition 2.6 *Let $\langle \phi : \sigma_1, .., \sigma_n : \phi \rangle$ be a complex feature. The function Root is defined as follows:*

$$Root(\langle \phi : \sigma_1, .., \sigma_n : \phi \rangle) = \phi$$

Definition 2.7 *The complex features, $\langle \phi : \sigma_1, .., \sigma_n : \phi \rangle$ and $\langle \phi : \psi_1, .., \psi_n : \phi \rangle$ have the* **same structure** *iff σ_1 and ψ_1 have the same structure, .., and σ_n and ψ_n have the same structure. The atomic features $\langle \phi : \alpha \rangle$ and $\langle \phi : \beta \rangle$ have the same structure.*

We assume that for an application, some complex features will be classified as **structured reports**. These will be complex features with some minimum structure. As an illustration, we may choose to regard Example 2.3 as a structured report since it contains some of the atomic features necessary for a weather report. We do not however assume any general conditions for classifying items of structured text as structured reports.

Definition 2.8 *If θ is a structured report, then θ has a* **unique reference label**.

All we require for Definition 2.8 is that if we have a set of structured reports, then we can refer to individual members by their reference label.

2.2 Representing structured reports as logical formulae

To represent structured reports as logical formulae, we adopt classical first-order logic. For logical reasoning, we assume the usual language of classical first-order logic using the usual symbols $\forall$ and $\exists$ for quantification and the usual symbols $\wedge, \vee, \rightarrow$ and $\neg$ for logical connectives. We represent the classical logical consequence relation by $\vdash$, and logical inconsistency by $\perp$.

Now we consider how we can represent a structured report as a set of classical logic formulae. To do this, we harness the tree structure implicit in an item of structured text with the semantic labels as non-leaf nodes and the text entries as leaves.

Definition 2.9 *Let $\langle \phi : \sigma_1, .., \sigma_n : \phi \rangle$ be a structured report that we want to represent by logical formulae. We call ϕ the root of the item. We assume each semantic label in the complex feature has a reference label. If a semantic label occurs more than once in a complex feature, then each occurrence will have a different reference label. The set of formulae that can be formed from this complex feature is obtained by exhaustively applying the following rules:*

1. *If X is a semantic label of a complex feature, and Y is the reference label for the parent of X, and Z is the reference label for X, then $X(Y, Z)$ is a formula.*
2. *If X is a semantic label of an atomic feature, and Y is the reference label for the parent of X, and Z is the text entry for the atomic feature, then $X(Y, Z)$ is a formula.*
3. *If X is the root of the tree and X is not an atomic semantic label, and Y is the reference label for X, then $X(Y)$ is a formula.*
4. *If X is the root of the tree and X is an atomic semantic label, and Y is the reference label for X, and Z is the text entry, then $X(Y, Z)$ is a formula.*

If Y is the reference label for a semantic label for a complex feature F and Y' is the reference label for a semantic label for a complex feature F', and F' is nested in F, then Y' is an offspring reference label for label for Y.

In Definition 2.9, Rule 1 gives the binary relations defining the path from the root to any other semantic label, Rule 2 gives the text entry for each branch, Rule 3 gives the reference label for the tree, and Rule 4 gives the text entry when the structured report is an atomic feature.

Example 2.4 *Consider the following item of structured text with the reference label* `e`.

$$
\begin{array}{l}
\langle \texttt{auctionreport} : \\
\quad \langle \texttt{buyer} : \langle \texttt{name} : \langle \texttt{firstname} : \texttt{John} \rangle, \langle \texttt{surname} : \texttt{Smith} \rangle : \texttt{name} \rangle : \texttt{buyer} \rangle, \\
\quad \langle \texttt{seller} : \langle \texttt{name} : \langle \texttt{firstname} : \texttt{Mary} \rangle, \langle \texttt{surname} : \texttt{Jones} \rangle : \texttt{name} \rangle : \texttt{seller} \rangle \\
\quad \langle \texttt{property} : \texttt{Lot37} \rangle \\
: \texttt{auctionreport} \rangle
\end{array}
$$

From this item, we obtain the following set of formulae using Definition 2.9, where `b`, `b1`, `s`, `s1`, *and* `p` *are reference labels.*

$$
\begin{array}{c}
\texttt{auctionreport(e)}, \texttt{buyer(e,b)}, \texttt{seller(e,s)}, \texttt{property(e,p)}, \\
\texttt{name(b,b1)}, \texttt{firstname(b1,John)}, \texttt{surname(b1,Smith)}, \\
\texttt{name(s,s1)}, \texttt{firstname(s1,Mary)}, \texttt{surname(s1,Jones)}, \\
\texttt{property(p,Lot37)}
\end{array}
$$

Definition 2.10 *The* **formulate** *function, denoted $Formulate$, is defined as follows: For a structured report θ, let $Formulate(\theta)$ denote a set of formulae obtained by exhaustive application of Definition 2.9 to θ. For a set of structured reports $\Theta = \{\theta_1, .., \theta_n\}$, let $Formulate(\Theta) = Formulate(\theta_1) \cup .. \cup Formulate(\theta_n)$.*

For a structured report θ, $Formulate(\theta)$ is not unique, since the reference labelling is not unique. If we assume appropriate equality axioms for the labels, then $Formulate(\theta)$ is unique.

2.3 Timestamps

In this paper, we will assume that each of our news reports has a timestamp that gives the event time for the report, and a timestamp that gives the log time for the report. There are four types of timestamp that we will consider, namely an atomic pointbased timestamp, an atomic intervalbased timestamp, a complex pointbased timestamp, and a complex intervalbased timestamp. We define these below.

Definition 2.11 *The set of* **temporal semantic labels** *is a subset of the set of semantic labels used for structured news reports.*

Example 2.5 *The set of temporal semantic labels includes* `time`*,* `date`*,* `publicationdate`*, and* `year`*.*

Definition 2.12 *The set of* **temporal text entries** *is a subset of the set of text entries used for structured news reports. A temporal text entry may refer to a point or interval in a clock and/or calendar. A temporal text entry is called a* **pointbased text entry** *if it refers to a point in a clock and/or calendar. And a temporal text entry is called an* **intervalbased text entry** *if it refers to an interval in a clock and/or calendar.*

Example 2.6 *Temporal text entries include* `14.00hrs`*,* `19 April 2000`*, and* `19/4/00`*. Temporal text entries may or may not include the units of time used. For example, both* `14.00` *and* `14.00hrs` *are temporal text entries.*

We will look more closely at the nature of points and intervals in the following.

Definition 2.13 *An* **atomic pointbased timestamp** *is an atomic feature $\langle \alpha, \beta \rangle$ where α is a temporal semantic label referring to a particular clock and/or a particular calendar and β is a pointbased text entry with a value denoting a point in that clock and/or calendar.*

Example 2.7 *Examples of atomic pointbased timestamps include:*

$\langle \texttt{time} : \texttt{14.00hrs} \rangle$ $\langle \texttt{date} : \texttt{19 April 2000} \rangle$ $\langle \texttt{publicationdate} : \texttt{19/4/00} \rangle$ $\langle \texttt{year} : \texttt{2004} \rangle$

In Section 2.4, we consider how we can make different formats for temporal text entries interchangeable, so that for example `19 April 2000` is equivalent to `19/4/2000`.

Definition 2.14 *An* **atomic intervalbased timestamp** *is an atomic feature $\langle \alpha, \beta \rangle$ where α is a temporal semantic label referring to a particular clock and/or a particular calendar and β is a intervalbased text entry with a value denoting an interval in that clock and/or calendar.*

We view time intervals as being either implicitly given as an interval with the start and end points being inferred, or explicitly given in terms of a start point and an end point.

Definition 2.15 *An* **explicit intervalbased text entry** *is of the form* `X-Y`*, where* `X` *and* `Y` *denote points, and an* **implicit intervalbased text entry** *of the form* `X` *where* `X` *describes a period of time without using explicit end points.*

Example 2.8 *Examples of atomic intervalbased timestamps with explicit intervalbased text entries include:*

$$\langle \texttt{reportingperiod} : \texttt{1/1/00-31/3/00} \rangle \qquad \langle \texttt{openhours} : \texttt{09.00-12.00} \rangle$$

Example 2.9 *Examples of atomic intervalbased timestamps with implicit intervalbased text entries include:*

$$\langle \texttt{reportingperiod} : \texttt{2004} \rangle \qquad \langle \texttt{holidayperiod} : \texttt{Easter} \rangle$$

So for `2004`*, the inferred start point in days is* `1/1/2004` *and the inferred endpoint in days is* `31/12/2004`*.*

It can appear difficult to distinguish some implicit intervalbased text entries from pointbased text entries. We address the problem of handling implicit intervals in Section 2.5 by reducing each intervalbased text entries to pointbased text entries.

Definition 2.16 *A* **complex pointbased timestamp** *is either an atomic pointbased timestamp or a complex feature $\langle \psi : \phi_1, .., \phi_n : \psi \rangle$ where ψ is a temporal semantic label referring to a particular clock and/or a particular calendar and $\phi_1, .., \phi_n$ are complex pointbased timestamps that describe the point in that clock and/or calendar.*

In this paper, we will assume atomic and complex pointbased timestamps should be inter-changeable. In Section 2.4, we show how we support this by appropriate axioms in the temporal knowledge.

Example 2.10 *An example of a complex pointbased timestamp is:*

$$\begin{array}{l} \langle \texttt{date:} \\ \quad\quad \langle \texttt{day: 23} \rangle, \\ \quad\quad \langle \texttt{month: April} \rangle, \\ \quad\quad \langle \texttt{year: 2000} \rangle, \\ \texttt{:date} \rangle \end{array}$$

So we can assume this complex pointbased timestamp is equivalent to $\langle \texttt{date: 23 April 2000} \rangle$*. We formalize this equivalence in Example 2.13.*

Definition 2.17 *A* **complex intervalbased timestamp** *is either an atomic intervalbased timestamp or a complex feature $\langle \psi : \phi_1, .., \phi_n : \psi \rangle$ where ψ is a temporal semantic label referring to a particular clock and/or a particular calendar and $\phi_1, .., \phi_n$ are complex pointbased timestamps that describe the interval in that clock and/or calendar.*

Also, in this paper, we will assume atomic and complex intervalbased timestamps should be interchangeable. In Section 2.4, we show how we support this by appropriate axioms in the temporal domain knowledge.

Example 2.11 *An example of a complex intervalbased timestamp is:*

```
⟨university term:
      ⟨first day of term: 10/1/2000⟩,
      ⟨last day of term: 29/3/2000⟩,
:university term⟩
```

So we assume this complex intervalbased timestamp is equivalent to

```
⟨university term: 10/1/2000-29/3/2000⟩
```

We formalize this equivalence in Example 2.14.

In the rest of this, paper we assume each news report has an atomic feature for log time of the form ⟨`log: T`⟩ where `T` is a pointbased text entry. Hence, this will be represented by the predicate `log(X,T)` in the logical formalization where `X` is a reference label.

We also assume a timestamp for the event time of the report. This may be pointbased or intervalbased, and complex or atomic. If the temporal semantic label for this timestamp is `P`, then there is a binary predicate `P(X,Y)` that will be in the logical formalization, where `X` is a reference label and `Y` is either a reference label or a temporal text entry. We will call this predicate the **event time predicate**.

Whilst we take a restricted position on timestamps in this paper, we believe that the approach presented here can be extended to further types and combinations of timestamps in structured text.

2.4 Timestamp equivalence axioms

Since a set of news reports may be heterogeneous in structure, we require axioms to identify equivalences between different kinds of timestamp. Our approach is to define these axioms in classical logic to manipulate the logical predicate form of news reports. Axioms may be required to: (1) relate timestamps where synonyms are used for semantic labels; (2) relate timestamps where synonyms are used for text entries; (3) relate atomic timestamps with equivalent complex timestamps; and (4) break down timestamps into constituent items of temporal information. We give some examples below.

Example 2.12 *Consider the following axioms. We might assume axioms like this for every date in the domain knowledge.*

$$\forall \mathtt{X}\ \mathtt{date}(\mathtt{X}, 12.01.99) \leftrightarrow \mathtt{day}(\mathtt{X}, 12) \wedge \mathtt{month}(\mathtt{X}, \mathtt{January}) \wedge \mathtt{year}(\mathtt{X}, 1999)$$

$$\forall \mathtt{X}\ \mathtt{date}(\mathtt{X}, 12\ \mathrm{Jan}\ 99) \leftrightarrow \mathtt{day}(\mathtt{X}, 12) \wedge \mathtt{month}(\mathtt{X}, \mathtt{January}) \wedge \mathtt{year}(\mathtt{X}, 1999)$$

Now consider the atomic features ⟨`date: 12.01.99`⟩ *and* ⟨`date: 12 Jan 99`⟩. *Here,* `12.01.99` *and* `12 Jan 99` *are not identical terms. However, we can infer from the above axioms that they are equivalent.*

In Section 2.3, we expressed a need for a form of equivalence between atomic and complex pointbased timestamps. To support this, we require further timestamp equivalence axioms such as given in Example 2.13.

Example 2.13 *Consider the following axioms which can be used to reason with the complex timestamps in Example 2.10. We might assume axioms like this for every date in the domain knowledge and news reports.*

$$\forall \mathtt{X} \exists \mathtt{X1}, \mathtt{X2}, \mathtt{X3}\ \mathtt{date}(\mathtt{X}, \mathtt{23.04.00}) \leftrightarrow \\ \mathtt{day}(\mathtt{X1}, 23) \land \mathtt{month}(\mathtt{X2}, \mathtt{April}) \land \mathtt{year}(\mathtt{X3}, 2000) \\ \land\ \mathtt{day}(\mathtt{X}, \mathtt{X1}) \land \mathtt{month}(\mathtt{X}, \mathtt{X2}) \land \mathtt{year}(\mathtt{X}, \mathtt{X3})$$

$$\forall \mathtt{X} \exists \mathtt{X1}, \mathtt{X2}, \mathtt{X3}\ \mathtt{date}(\mathtt{X}, \mathtt{23\ April\ 00}) \leftrightarrow \\ \mathtt{day}(\mathtt{X1}, 23) \land \mathtt{month}(\mathtt{X2}, \mathtt{April}) \land \mathtt{year}(\mathtt{X3}, 2000) \\ \land\ \mathtt{day}(\mathtt{X}, \mathtt{X1}) \land \mathtt{month}(\mathtt{X}, \mathtt{X2}) \land \mathtt{year}(\mathtt{X}, \mathtt{X3})$$

Here the predicates `day(X,X1)`, `month(X,X2)`, *and* `year(X,X3)` *give the structure for the complex timestamp.*

We also need a form of equivalence between atomic and complex intervalbased timestamps. To support this, we require further timestamp equivalence axioms such as given in Example 2.14.

Example 2.14 *Consider the following axioms which can be used to reason with the complex intervalbased timestamp in Example 2.11. We might assume axioms like this for every time and date used in the domain knowledge and news reports.*

$$\forall \mathtt{X} \exists \mathtt{X1}, \mathtt{X2}\ \mathtt{universityterm}(\mathtt{X}, \mathtt{10/1/00\text{-}29/3/00}) \leftrightarrow \\ \mathtt{firstdayofterm}(\mathtt{X}, \mathtt{X1}) \land \mathtt{lastdayofterm}(\mathtt{X}, \mathtt{X2}) \\ \land\ \mathtt{firstdayofterm}(\mathtt{X1}, 10/1/2000) \land \mathtt{lastdayofterm}(\mathtt{X2}, 29/3/2000)$$

The actual choice of timestamp equivalence axioms depends on the application area, since this will dictate what syntax is used for the temporal semantic labels, temporal text entries, and the structure of the complex features used, in the timestamps. Furthermore, the axioms chosen will need to capture the intended meaning for this syntax. To support this kind of issue, we look at semantics in Section 3.

In practice, we would require many timestamp equivalence axioms. However, given the regularity of these axioms, it is possible to use succinct algorithms to produce these axioms on demand.

2.5 Pointwise axioms

Even though timestamps can be either pointbased or intervalbased, we require a common format for temporal information in order to support the actual merging process described in Section 5. The common format will be pointbased. This means that we need to be able to rewrite information in intervalbased timestamps into information in pointbased timestamps. In other words, for each intervalbased timestamp, we should be able to infer a set of pointbased timestamps. We give examples below of axioms to rewrite temporal intervals into one or more timepoints.

Example 2.15 *Consider the following atomic intervalbased timestamp, with the explicit intervalbased text entry.*

$$\langle \mathtt{taxyear} : \mathtt{1\ April\ 2000\ -\ 31\ March\ 2001} \rangle$$

We have a number of choices for the set of pointbased timestamps that correspond to this. For example, we can consider the interval being captured by the point time 2000 in the calendar of years, and so giving the atomic pointbased timestamp.

$$\langle \mathtt{taxyear} : \mathtt{2000} \rangle$$

This means all the information associated with the above intervalbased timestamp is now associated with this pointbased timestamp. In other words, we can consider the interval `1 April 2000 - 31 March 2001` *being replaced in the structured news report by* `2000`*. Alternatively, we may choose to use the sequence of days to represent the tax year:*

$$\langle \texttt{taxyear} : \texttt{1/4/2000} \rangle, ..., \langle \texttt{taxyear} : \texttt{31/3/2001} \rangle$$

So in effect the information that is associated with the above intervalbased timestamp, we associate with every one of these pointbased timestamps. To illustrate, this later choice can be captured by the following axiom:

$$\texttt{taxyear(X, 1 April 2000 - 31 March 2001)} \leftrightarrow \texttt{taxyear(X, 1/4/00)} \wedge ... \wedge \texttt{taxyear(X, 31/3/01)}$$

There are numerous other choices we could make in this situation.

Example 2.16 *Consider the atomic intervalbased timestamp* $\langle \texttt{holiday} : \texttt{Easter2000} \rangle$ *which incorporates an implicit intervalbased text entry. Here we may choose the sequence of days to represent the Easter period in 2000.*

$$\langle \texttt{holiday} : \texttt{10/4/2000} \rangle, ..., \langle \texttt{holiday} : \texttt{16/4/2000} \rangle$$

This therefore requires the following axiom:

$$\texttt{holiday(X, Easter2000)} \leftrightarrow \texttt{holiday(X, 10/4/2000)} \wedge ... \wedge \texttt{holiday(X, 16/4/2000)}$$

In practice, the exact choice of axiom for presenting equivalence really needs to involve more careful consideration of the underlying semantics of the language used in a particular application. This is the subject of the next section. Also in practice, we would require many timestamp equivalence axioms. However, given the regularity of these axioms, it is possible to use succinct algorithms to produce these axioms on demand.

3 Modelling the flow of time

In order to relate pieces of temporal information from a collection of structured reports, we assume a timeline for our reasoning. This is based on the natural numbers with the usual ordering. This gives us a discrete linear view on time.

3.1 Time points

Time points are instantaneous. In other words, a time point has no duration. This allows us to equate a time point here with the notion of a time point used in a model in temporal logic. We discuss this relationship in Section 3.6.

There are two views on the information associated with timepoints:

Snapshot time points We can regard a point in time as having an associated instantaneous snapshot of the world we are interested in. So if we have a timeline of days, then for each day we have an associated snapshot of the world taken during that day. For example, in a weather report for London for the day 1/7/00, we could have snapshot, say taken at midday where the temperature is 20C, the wind speed is 5mph, and the humidity is 60%.

Representative time points We can regard a point in time as having an associated representative picture of the world we are interested in. So if we have a timeline of days, then for each day we have an associated description of the world that reflects that day. For example, in a weather report for London for the day 1/7/00, we could have that the temperature is 20C, the wind speed is 5mph, and the humidity is 60%, and yet there might be no point in the day where these three factors have these values together. In this way the combination of values are representative of the values of the day.

We can regard snapshots as a special kind of representative time point where the snapshot is used as the representative. For example, using share prices at market close is a snapshot that can be used as the representative for the day's share prices.

However, in general, a snapshot is not as good as a representative. For example, if we take a snapshot of the traffic in central London at 3am every day, then that snapshot is not a good representative of the traffic that day since there is little relationship between traffic at 3am and traffic during the rush hours. It would also be no better to use a randomly selected hour in the day to take a snapshot.

Let Ω_{days} denote the set of days. In other words, each $x \in \Omega_{days}$ is a time point referring to a particular day. Since we assume time points are instantaneous, each element of Ω_{days} has only static information about that day. So for example 27/3/2000, where $27/3/2000 \in \Omega_{days}$, could be a snapshot or representative taken at midday, and 28/3/2000 could be a snapshot taken at 3pm.

In the rest of this paper, we will focus our attention on representative time points.

Definition 3.1 *Let Ω_i denote a set of representative time points. Each $x \in \Omega_i$ is a time point of granularity i.*

In the next subsection, we develop the notion of representative time points and consider how they can be used for capturing temporal information represented in structured text.

3.2 Timelines

In this paper, we consider a number of kinds of timeline based on representative time points including $(\Omega_{years}, \leq_{years})$, $(\Omega_{months}, \leq_{months})$, $(\Omega_{weeks}, \leq_{weeks})$, $(\Omega_{days}, \leq_{days})$, $(\Omega_{hours}, \leq_{hours})$, and $(\Omega_{minutes}, \leq_{minutes})$, denoting years, months, weeks, days, hours, and minutes with the usual ordering. So for example in $(\Omega_{years}, \leq_{years})$ each element of Ω_{years} denotes a year and $\leq_{years}$ is the linear ordering over these years. We can also for example use $\Omega_{middays}$ and $\Omega_{marketclose}$ for middays and stockmarket daily market close respectively.

Definition 3.2 *A* **timeline** *is a pair $(\Omega_i, \leq_i)$ where Ω_i be a set of time points of type i, and $i \in \{years, months, weeks, days, hours, minutes\}$ and the following conditions hold for the ordering relation $\leq_i$:*

$$\begin{array}{rl} \forall x \in \Omega_i \; x \leq_i x & \textit{Reflexivity} \\ \forall x, y \in \Omega_i \; x \leq_i y \wedge y \leq_i x \rightarrow x = y & \textit{Antisymmetry} \\ \forall x, y, z \in \Omega_i \; x \leq_i y \wedge y \leq_i z \rightarrow x \leq_i z & \textit{Transitivity} \end{array}$$

We assume each set $(\Omega_i, \leq_i)$ is isomorphic to the natural numbers with the usual ordering $(\mathbb{N}, \leq)$. We also assume that each time point in a timeline is instantaneous and so it has no duration.

Note, for each granularity of time point, there are an infinite number of variants. For example, consider $(\Omega_{years}, \leq_{years})$. Here each variant differs because of the choice of the year for the first year in the sequence. For example, we have one variant starting at year 1900, another variant starting at 1901, yet another at 1902, and so on.

To simplify this situation, let us suppose for this paper that we restrict consideration to timelines that start at the same point in time. Since we are considering news reports in this paper, we adopt the fair timelines given below.

Definition 3.3 *A* **fair timeline** *is a timeline where the first item in the ordering is fixed as follows: We choose* $(\Omega_{years}, \leq_{years})$ *to start at year 0,* $(\Omega_{months}, \leq_{months})$ *to start at January of year 0,* $(\Omega_{weeks}, \leq_{weeks})$ *to start at the first week of January of year 0,* $(\Omega_{days}, \leq_{days})$ *to start at the first day of January of year 0,* $(\Omega_{hours}, \leq_{hours})$ *to start at the first hour of year 0, and* $(\Omega_{minutes}, \leq_{minutes})$ *to start at the first minute of year 0.*

We may also consider further timelines that capture other kinds of time points. For instance, $(\Omega_{holidays}, \leq_{holidays})$ that could include public holidays such as New year 2000, Easter 2000, May day 2000, Christmas 2001, New year 2001, Easter 2001, and so on, with the expected ordering over them.

3.3 Relationship between timelines

We can relate different kinds of timelines using a series of mappings. These mappings constrain the relationship we expect between the timelines and thereby enhance the semantics.

First we consider synchronization functions. As the name suggests, these capture the synchronization of timelines, so that for example, 2000 and January 2000 are linked and 2001 and January 2001 are linked.

Definition 3.4 *A* **synchronization function***, denoted* s^i_j*, is an order-preserving injection from the timeline* Ω_i *to the timeline* Ω_j*.*

The basic definition is relatively unconstrained, and so for particular synchronization functions, further intuitive conditions are applied.

Definition 3.5 *Particular synchronization functions that we use in this paper are further delineated as follows:*

$$s^{year}_{month}(x) = y \text{ where } y \text{ is the first month of year } x$$

$$s^{month}_{day}(x) = y \text{ where } y \text{ is the first day of month } x$$

$$s^{day}_{hour}(x) = y \text{ where } y \text{ is the first hour of day } x$$

$$s^{hour}_{minute}(x) = y \text{ where } y \text{ is the first minute of hour } x$$

Further synchronization functions with the obvious definitions include s^{year}_{week} *and* s^{week}_{day}*. We can also combine the functions so that for example, we can obtain a synchronization function* s^{year}_{day} *by combining* s^{year}_{month} *and* s^{month}_{day}*.*

Note, we do not have s^{months}_{weeks}, since there is no mapping that discretely assigns weeks to months.

Example 3.1 *Let* s^{year}_{month} *be a synchronization function. So* $s^{year}_{month}(2001)$ = *January 2001*

Synchronization functions give a particular way of lining up different timelines. They associate time points in one timeline with time points in another timeline.

Definition 3.6 *For the timeline* Ω_i *and the timeline* Ω_j, j *is* **finer grained** *than* i *if there is a synchronization function* s^i_j. *Also* i *is* **coarser grained** *than* j *if* j *is* **finer grained** *than* i.

Further timelines using the fair timelines together with a temporal shift as follows: A **shifted timeline** is a timeline $(\Omega_j, \leq_j)$ defined in terms of an fair timeline $(\Omega_i, \leq_i)$ and a temporal shift x, where x is some time period, as follows: For each time point $t \in \Omega_i$, there is a time point $t' \in \Omega_j$, where $t' = t + x$.

Example 3.2 *Let* $(\Omega_{middays}, \leq_{middays})$ *be a shifted timeline defined by using the fair timeline* $(\Omega_{days}, \leq_{days})$ *and the temporal shift of 12 hours. Similarly, let* $(\Omega_{marketclose}, \leq_{marketclose})$ *be a shifted timeline defined using the fair timeline* $(\Omega_{days}, \leq_{days})$ *and the temporal shift of 16 hours, where marketclose is intended to denote the close of the stock exchange.*

Another useful function is the elapse function e^i_j which gives the timepoints of granularity j that should be associated with each time point of granularity i. For example, if i is $years$ and j is $months$ then the set {January 2000,......., December 2000} is associated with the time point 2000.

Definition 3.7 *Let* Ω_i *and* Ω_j *be timelines. An* **elapse function**, *denoted* e^i_j, *is function from* Ω_i *to* $\wp(\Omega_j)$ *defined as follows, where* s^i_j *is a synchronization function:*

$$e^i_j(x) = \{y \in \Omega_j \mid s^i_j(x) \leq y < s^i_j(x+1)\}$$

Example 3.3 *Let* e^{year}_{day} *be an elapse function.*

$$e^{year}_{day}(2001) = \{1/1/2001, ..., 31/12/2001\}$$

We call this function an elapse function because it gives the set of time points of granularity j that elapse before the next time point of granularity i occurs according to the synchronization function.

Our third function on timelines is the harmonization function. Essentially, for each time point in a finer grained timeline, it associates a time point in a coarser grained timeline. For example, March 1999 would be associated with 1999.

Definition 3.8 *A* **harmonization function**, *denoted* h^i_j *is an order-preserving surjection from the timeline* Ω_i *to the timeline* Ω_j.

The basic definition is relatively unconstrained, and so for particular harmonization functions, further intuitive conditions are applied.

Definition 3.9 *Particular harmonization functions that we use in this paper are further delineated as follows:*

$$h_{year}^{month}(x) = y \text{ where } y \text{ is the year in which month } x \text{ occurs}$$

$$h_{month}^{day}(x) = y \text{ where } y \text{ is the month in which day } x \text{ occurs}$$

$$h_{day}^{hour}(x) = y \text{ where } y \text{ is the day in which hour } x \text{ occurs}$$

$$h_{hour}^{minute}(x) = y \text{ where } y \text{ is the hour in which minute } x \text{ occurs}$$

Further harmonization functions with the obvious definitions include h_{week}^{year} and h_{day}^{week}. We can also combine the functions so that for example, we can obtain a harmonization function h_{day}^{year} by combining h_{month}^{year} and h_{day}^{month}.

There are a number of relationships that we can now identify between these definitions such as the following which says that for every time point $y \in \Omega_j$ that has elapsed since $x \in \Omega_i$, the harmonization function from Ω_j to Ω_i maps y to x.

$$y \in e_j^i(x) \text{ iff } h_i^j(y) = x$$

Since the harmonization functions are order-preserving, we have relationships such as the following:

$$\forall x, y \in \Omega_{months}\ (x \leq_{months} y) \rightarrow (h_{years}^{months}(x) \leq_{years} h_{years}^{months}(y)$$

$$\forall x, y \in \Omega_{days}\ (x \leq_{days} y) \rightarrow (h_{weeks}^{days}(x) \leq_{weeks} h_{weeks}^{days}(y)$$

$$\forall x, y \in \Omega_{hours}\ (x \leq_{hours} y) \rightarrow (h_{days}^{hours}(x) \leq_{days} h_{days}^{hours}(y)$$

Similarly, synchronization functions are order-preserving, with relationships such as the following:

$$\forall x, y \in \Omega_{years}\ (s_{months}^{years}(x) \leq_{months} s_{months}^{years}(y) \rightarrow (x \leq_{years} y)$$

$$\forall x, y \in \Omega_{weeks}\ (s_{days}^{weeks}(x) \leq_{days} s_{days}^{weeks}(y) \rightarrow (x \leq_{weeks} y)$$

$$\forall x, y \in \Omega_{days}\ (s_{hours}^{days}(x) \leq_{hours} s_{hours}^{days}(y) \rightarrow (x \leq_{days} y)$$

Example 3.4 *Consider the synchronization function s_{days}^{months}. Here we have $s_{days}^{months}(05/2000) = 01/05/2000$ and $s_{days}^{months}(06/2000) = 01/06/2000$. So $05/2000 \leq_{months} 06/2000$ and $01/05/2000 \leq_{days} 01/06/2000$.*

Appropriate definitions for timelines, and for the synchronization, elapse, and harmonization functions, ensure the basis of an intuitive and well-behaved semantics for modelling time of different granularities in a given application.

In this paper, our reasoning with granularity is tailored for our requirements for merging structured text. For alternative and more comprehensive representation and reasoning with granularity see [BJW00, GLO+01].

3.4 Time intervals

Whilst time points are durationless, we can model duration using time points. Explicit time intervals are represented by pairs of time points.

Definition 3.10 *Let* $(\Omega_i, \leq_i)$ *be a timeline. For* $x, y \in \Omega_i$*, where* $x \leq_i y$*,* (x, y) *is an* **explicit time interval** *in the timeline* $(\Omega_i, \leq_i)$*.*

Example 3.5 *Let* $14/2/2001, 28/2/2001 \in \Omega_{days}$*. So* $(14/2/2001, 28/2/2001)$ *is an explicit time interval in* $(\Omega_{days}, \leq_{days})$

Definition 3.11 *An* **implicit time interval** *of granularity of* i *is a synonym for an explicit time interval in the time line* $(\Omega_i, \leq_i)$*.*

So for example, we can say the implicit time interval May 2000, is the interval of days given by 01/05/2000 as the start time point and 31/05/2000 as the end time point.

Definition 3.12 *A* **start function***, denoted* $start_j$ *takes an interval and gives the start time point of type* j*. An* **end function***, denoted* end_j *takes an interval and gives the end time point of type* j*. The start time point and end time point bound the corresponding time interval inclusively.*

Example 3.6 *Consider the period* 3/9/2000-16/9/2000 *for which we get the following.*

$$start_{days}(3/9/2000\text{-}16/9/2000) = 3/9/2000 \qquad end_{days}(3/9/2000\text{-}16/9/2000) = 16/9/2000$$

Example 3.7 *Consider the month* May 2000 *for which we get the following.*

$$start_{days}(\text{May2000}) = 01/05/2000 \qquad end_{days}(\text{May2000}) = 31/05/2000$$

So for explicit time intervals, $start_j$ and end_j are completely delineated by Definition 3.10. For implicit time intervals, the associated explicit time interval needs to also be known.

We can also view coarser grained time points as implicit time intervals. In other words, if i is a coarser grained than j, then any $x \in \Omega_i$ can be viewed as an implicit time interval of granularity j. The explicit time interval that defines x is then $(min(e^i_j(x)), max(e^i_j(x))$. So for example the time point January 2000 can be viewed as the time interval 1/1/2001-31/1/2001.

In this paper, our reasoning with intervals is indirect. We handle intervals in terms of time points. This has advantages in terms of having a homogeneous framework for handling the temporal information. For more details on handling intervals using time points see [GHR94]. However, there are a number of alternative proposals that have intervals as the basic unit in the semantics [All84]. Interval logics such as this can incorporate sophisticated axioms for reasoning about the relationships between intervals. We do not support comparable reasoning here. We discuss this in more detail in the next section.

3.5 Correspondence of timestamps with time points

Timestamps are part of the syntax of representing and reasoning with news reports. This is the case in the original form of structured text and in the form of logical predicates formed from the structured text using *Formulate*. In contrast, we regard timelines as providing the semantics for representing and reasoning with news reports.

To use the semantics to help understand the syntax, and in particular to help in the formalization of further axioms for handling the temporal information in structured news reports, we need to consider the relationship between timestamps and timelines. First, we need to associate individual time points and timestamps.

Definition 3.13 *Let T_i be the set of pointbased text entries of granularity i, and let Ω_i be the set of time points of granularity i. A* **correspondence function** *of granularity i, denoted λ_i, is a surjection from T_i to Ω_i.*

We define a correspondence function for each granularity i so that for each $x \in T_i$, if the value of x is y, then $\lambda_i(x) = y$. In other words, we define a correspondence function to give the meaning for each pointbased text entry that we use in the domain knowledge and news reports.

Example 3.8 *Suppose* `15 Feb 2001` *and* `15/2/01` *are in T_{days}, and $15/2/01 \in \Omega_{days}$. Then, we could choose to define the correspondence function λ_{days} so that the following hold:*

$$\lambda_{days}(\texttt{15 Feb 2001}) = 15/2/2001 \qquad \lambda_{days}(\texttt{15/2/01}) = 15/2/2001$$

So the meaning of `15 Feb 2001` *and* `15/2/01` *is given by* $15/2/2001$ *in the timeline* $(\Omega_{days}, \leq_{days})$

We deal with intervalbased text entries indirectly. First, we consider explicit intervalbased text entries. Let R_i be the set of explicit intervalbased text entries of granularity i. So for each $v \in R_i$, v is of the form $x - y$ and $x, y \in T_i$. So the meaning of v is given by the time interval $(\lambda_i(x), \lambda_i(y))$ in the timeline $(\Omega_i, \leq_i)$.

Example 3.9 *Suppose* `15/2/01-21/2/01` $\in R_{days}$*. Also suppose* $\lambda_{days}(\texttt{15/2/01}) = 15/2/2001$ *and* $\lambda_{days}(\texttt{21/2/01}) = 21/2/2001$*. So the meaning of* `15/2/01-21/2/01` *is given by the pair*

$$(\lambda_{days}(\texttt{15/2/2001}), \lambda_{days}(\texttt{21/2/2001}))$$

in the timeline $(\Omega_{days}, \leq_{days})$*.*

Now, we consider implicit intervalbased text entries. Let S_i be the set of implicit intervalbased text entries of granularity i. So for each $v \in S_i$, there is an explicit time interval of granularity i which defines v. To clarify we need the following definition.

Definition 3.14 *A* **recovery function** *of granularity i, denoted ρ_i, is a surjection from S_i to R_i.*

Example 3.10 *Let* `Easter 2000` *be an implicit intervalbased text entry. Then we could chose to define the recovery function so that:*

$$\rho_i(\texttt{Easter2000}) = \texttt{10/4/2000-16/4/2000}$$

So as with the correspondence function, we are free to define the recovery function to give the meaning we require for the temporal text entry.

3.6 Using temporal logic

In this section, we consider the semantics for the predicates using timelines. To do this, we harness temporal logic. Essentially, we take a set of news reports Θ, and identify a set of temporal models that satisfies $Formulate(\Theta)$. First we consider the temporal models we are interested in. Essentially, a temporal model is a timeline together with a classical model associated with each time point.

Definition 3.15 *A temporal model is a triple $(\Omega_i, \leq_i, \pi)$ where $(\Omega_i, \leq_i)$ is a timeline and π is a mapping from time points to classical models so that if $t \in \Omega_i$, $\pi(t)$ is a classical propositional model. The granularity of $(\Omega_i, \leq_i, \pi)$ is i.*

The notion of a temporal model is just a linear discrete model used in tense logics [McA76, Bur82]. Assuming this notion of a temporal model we can adopt all the usual tense logic notions of satisfaction and validity.

Definition 3.16 *A temporal model $(\Omega_i, \leq_i, \pi)$ satisfies a formula α, denoted $(\Omega_i, \leq_i, \pi) \models \alpha$, iff for all $t \in \Omega_i$, $\pi(t) \models \alpha$, where $\models$ is the classical satisfaction relation.*

For our purposes, we require a particular type of temporal model called a calibrated temporal model. This ensures that the temporal knowledge (which includes timestamp equivalence axioms and pointwise axioms) is satisfied at each time point and it ensures the event time predicate is true at the appropriate time point for each news report. However, since we want to stay in a propositional framework, we assume the grounded version of the temporal knowledge, where the ground terms come from the news reports being handled.

Definition 3.17 *A* **calibrated temporal model** *is a temporal model $(\Omega_i, \leq_i, \pi)$ with the following condition holding, where Σ is the grounded form of the temporal knowledge:*

$$\textit{If } \alpha \in \Sigma \textit{ and } t \in \Omega_i, \textit{ then } \pi(t) \models \alpha$$

Definition 3.18 *A calibrated temporal model $(\Omega_i, \leq_i, \pi)$, with correspondence function λ_i, satisfies a set of news reports Θ iff for each $\theta \in \Theta$,*

$$\begin{array}{l}\textit{If } \mathtt{T(X,D)} \in Formulate(\theta) \textit{ is an event time predicate,} \\ \textit{then for all } \phi \in Formulate(\theta), \pi(\lambda_i(\mathtt{D})) \models \phi\end{array}$$

Essentially, we are associating all the predicates obtained from a news report with the corresponding time point (as defined by the correspondence function λ_i) in the timeline $(\Omega_i, \leq_i)$. In this way, information that holds at a time point is true at the corresponding time point in all the models. Obviously, this construction only works if each report incorporates exactly one event time predicate. Also this construction places the log time predicate at the same point as the rest of the report.

Example 3.11 *Consider the following set of predicates obtained from a set of news reports, together with the correspondence function λ_{months}.*

```
date(r1, July2000),
buyer(r1, a),
acquisition(r1, b),
report(r1)
log(r1, 28/7/00)
```

Here, all models $(\Omega_{months}, \leq_{months}, \pi)$ *returned are such that*

$$\begin{array}{l}\pi(\lambda_{months}(\texttt{July2000})) \models \texttt{date}(\texttt{r1}, \texttt{July2000}) \\ \pi(\lambda_{months}(\texttt{July2000})) \models \texttt{buyer}(\texttt{r1}, \texttt{a}) \\ \pi(\lambda_{months}(\texttt{July2000})) \models \texttt{acquistion}(\texttt{r1}, \texttt{b}) \\ \pi(\lambda_{months}(\texttt{July2000})) \models \texttt{log}(\texttt{r1}, \texttt{28/7/00})\end{array}$$

The different granularities of temporal models directly reflect the different granularities of timestamp.

Example 3.12 *Consider the following predicates coming from three reports (*`r1`*,*`r2` *and* `r3`*), where for all models with model* $(\Omega_{days}, \leq_{days}, \pi)$ *and a correspondence function* λ_{days} *we have*

$$\begin{array}{l}\pi(\lambda_{days}(\texttt{1/7/00})) \models \texttt{acquisition}(\texttt{r1}, \texttt{b}) \\ \pi(\lambda_{days}(\texttt{7/7/00})) \models \texttt{acquisition}(\texttt{r2}, \texttt{c}) \\ \pi(\lambda_{days}(\texttt{9/7/00})) \models \texttt{acquisition}(\texttt{r3}, \texttt{d})\end{array}$$

Here, for all models $(\Omega_{months}, \leq_{months}, \pi)$*, and correspondence function* λ_{months}*, all three predicates hold at the same time point.*

$$\pi(\lambda_{months}(\texttt{July2000})) \models \texttt{acquisition}(\texttt{r1}, \texttt{b}) \wedge \texttt{acquisition}(\texttt{r1}, \texttt{c}) \wedge \texttt{acquisition}(\texttt{r1}, \texttt{d})$$

Whilst we have focused on predicates in news reports in this section, it is straightforward to generalize the definitions to handle the predicates obtained by the expansion and contraction axioms (as discussed in Section 4) and the merged predicates obtained by the merging axioms (as discussed 5.3). Furthermore, tense logic or US temporal logic can be directly harnessed to allow reasoning with formulae at each point in time and formulae over time. This includes extending the classical language with modal operators including Until and Since. For more details on temporal logic, see [GHR94].

One issue that may involve a more sophisticated temporal logic is the handling of the log time predicate. As an alternative to linear models, it may be appropriate to use a two-dimensional view on time [FG92]. So instead of time being based on $\mathbb{N}$ giving event time, we would use the matrix $\mathbb{N} \times \mathbb{N}$ where the first dimension would be for event time and the second dimension would be for log time.

4 Expansion and contraction inferences

There are many ways that we may describe a proposition in time. Consider the following descriptions taken from Shoham [Sho88].

Downward-hereditary P is downward-hereditary if whenever P holds over an interval, it holds over all of its subintervals. For example, `the robot travelled less than two miles`.

Upward-hereditary P is upward-hereditary if whenever P holds for all proper subintervals of some non-point interval, it also holds over the non-point interval itself. For example, `the robot travelled at a speed of two miles per hour`.

Liquid P is liquid if whenever P is both upward-hereditary and downward-hereditary. For example, `the robot arm was in the grasping state`.

Gestalt P is gestalt if P never holds over two intervals, one of which contains the other. For example, `exactly six minutes have passed`.

Solid P is solid if P never holds over two properly overlapping intervals. For example, `the robot executed the procedure (from start to finish)`.

These kinds of descriptions can be important if we want to reason with temporal information. In our case, we have news reports with information that holds over certain time periods and we may need to know if they hold over finer grained or coarser grained time periods. For this, we consider restricted forms of downward-hereditary and upward-hereditary properties which we call contraction and expansion respectively.

4.1 Granularity of time

In the motivation for this paper, we identified a need to relate information of different granularities. As a precursor to this, we need to clarify this notion.

Definition 4.1 *The* `Granularity` *predicate is of the form* $\mathtt{Granularity(D,}i\mathtt{)}$ *where* `D` *is a point-based text entry and* i *is a granularity. This relationship means that* `D` *is of granularity* i*. In other words, we are using a correspondence function* λ_i *such that* $\lambda_i(\mathtt{D}) \in \Omega_i$ *to ascribe a meaning to this relationship.*

Example 4.1 *To illustrate, Definition 4.1 consider the following.*

$\mathtt{Granularity(27/3/01,}days\mathtt{)}$ $\qquad$ $\mathtt{Granularity(March\ 2000,}month\mathtt{)}$

So looking at the first of these examples, we can ascribe a meaning as follows: There is a timeline $(\Omega_{days}, \leq_{days})$ *and a correspondence function* λ_{days} *where* $27/3/2001 \in \Omega_{days}$ *and* $\lambda_{days}(\mathtt{27/3/01})$ $= 27/3/2001$*.*

In order to maintain a clear difference between the syntax and semantics of our proposal, we should use different symbols for the syntactic and semantic notions of granularity, and then associate them via an appropriate function. However, we have proposed a slight blurring of this distinction in order to maintain a more lucid framework.

From Section 3.2, we can consider different granularities of time point by shifting from one timeline to another. For example Ω_{days} is a finer granularity than Ω_{weeks} and Ω_{weeks} is a finer granularity than Ω_{months}. We can represent this by the `Subgrain` predicate defined below.

Definition 4.2 *The Subgrain predicate is of the from* $\mathtt{Subgrain(}i_1\mathtt{,}i_2\mathtt{)}$ *where* i_1 *and* i_2 *are granularities. The meaning for this is either* $i_1 = i_2$ *or* i_1 *is finer grained than* i_2*.*

Example 4.2 *To illustrate Definition 4.2, consider the following.*

$\mathtt{Subgrain(}day, week\mathtt{)}$ $\qquad$ $\mathtt{Subgrain(}week, year\mathtt{)}$ $\qquad$ $\mathtt{Subgrain(}month, year\mathtt{)}$

Definition 4.3 *The* `In` *predicate is a binary predicate over pairs of pointbased text entries. For this,* `In(X,Y)` *means that the following conditions hold: (1)* $\mathtt{Granularity(X,}i\mathtt{)}$*; (2)* $\mathtt{Granularity(Y,}j\mathtt{)}$*; and (3)* $h^i_j(\lambda_i(\mathtt{X})) = \lambda_j(\mathtt{Y})$ *where* h^i_j *is a harmonization function, and* λ_i *and* λ_j *are correspondence functions.*

Equivalently, $\texttt{In(X,Y)}$ means that $\texttt{Granularity(X,}i\texttt{)}$ and $\texttt{Granularity(Y,}j\texttt{)}$ hold and there are correspondence functions λ_i and λ_j such that $\lambda_j(\texttt{X}) \in e^i_j(\lambda_i(\texttt{Y}))$.

Example 4.3 *To illustrate, Definition 4.3 consider the following.*

$$\texttt{In}(29/5/00, 2000) \qquad \texttt{In}(29/5/00, 29/5/00) \qquad \texttt{In}(29/5/00, \texttt{March}/00)$$

Looking at the first of these, we can ascribe the meaning to the predicate as follows: We assume a correspondence function λ_{days} such that $\lambda_{days}(29/5/00) = 29/5/2000$, a correspondence function λ_{year} such that $\lambda_{year}(2000) = 2000$, and a harmonization function h^{day}_{year} such that $h^{day}_{year}(\lambda_{days}(29/5/00)) = \lambda_{year}(2000)$.

In the rest of this paper, we will use the `Granularity`, `Subgrain`, and `In` predicates as metalanguage predicates. We explain how and why in the next section.

4.2 Metalanguage for expansion and contraction

We assume a metalanguage for formalizing expansion and contraction. We will use classical logic for this metalanguage, though a more specialized language such as Horn clauses may be adequate. We use a metalanguage so as to be able to identify which parts of a structured news report is expandable or contractable.

Metalanguages have been the subject of much development in logic generally and in logic programming in particular [BFG+96]. Here we only require the ability to treat formulae in the object language as terms in the metalanguage. So in particular, predicate symbols in the object language are treated as function symbols in the metalanguage. Apart from object language formulae, and object language terms, appearing as terms in the metalanguage, the object language and metalanguage are disjoint.

The main predicates that we require for the metalanguage are `Granularity`, `Subgrain`, `In`, `Expandable`, `Contractable`, `Holds`, `Eventtime` and `NewLabel`. We considered the first three of these in the previous section. We consider the remaining definitions for these in the following.

Definition 4.4 *The* `Expandable` *predicate is a binary relation that limits expansion. Using this,* $\texttt{Expandable}(\phi, \texttt{i})$ *means the semantic label ϕ is regarded as expandable, up to a maximum granularity of i. So for any complex feature $\langle \phi : \sigma_1, .., \sigma_n : \phi \rangle$, any offspring of this complex feature is regarded as expandable.*

The upper limit on the granularity means that expansion is not taken too far. For example, if we have the information that the `middaytemperature` is `10C` for a particular day, then it is reasonable to say that the `middaytemperature` is `10C` for the week that includes that day, or even for the month that includes that day. But to expand the information to a year is not appropriate.

Definition 4.5 *The* `Contractable` *predicate is a binary relation that limits contraction. Using this,* $\texttt{Contractable}(\phi, \texttt{i})$ *means the semantic label ϕ is regarded as contractable, down to a minimum granularity of i. So for any complex feature $\langle \phi : \sigma_1, .., \sigma_n : \phi \rangle$, any offspring of this complex feature is regarded as contractable.*

Definition 4.6 *The* `Holds` *predicate is a monadic predicate over object language formulae, where* $\texttt{Holds}(\alpha)$ *means the formula α is assumed to be a formula that can be inferred from the structured news report.*

So if $\alpha \in Formulate(\theta)$ for some structured report θ, then we have $\texttt{Holds}(\alpha)$.

Definition 4.7 *The* `event time` *predicate is a monadic predicate over object language formulae. For this,* $\texttt{Eventtime}(\alpha)$ *holds if the formula* α *is an event time predicate.*

Since we have assumed there is only one timestamp for the event time time in each news report, there is only one event time predicate in the logical formalization of each news report.

Definition 4.8 *The* `NewLabel` *predicate is a monadic predicate over object language terms. Informally,* `NewLabel(X)` *means the reference number* `X` *is a reference label that can be used as a new reference label (i.e. the reference label has not been used elsewhere in the structured text).*

In the rest of this paper, we will assume classical logic proof theory and semantics for reasoning with the classical formulae formed from these metalanguage predicates.

4.3 The expansion axiom

As an illustration of when we would want to use the date expansion axiom, consider a weather report about an eclipse occurring in Europe on 6 August 1999. Here we would want to apply the date expansion axiom and obtain that the eclipse occurs in Europe in August 1999.

As an illustration of when we would not want to use the date expansion axiom, consider a weather report about the rainfall in London being 10cm during May 2000. Here we would not want to apply the date expansion axiom and obtain that the rainfall for 2000 is 10cm.

In this way, the type of information, and the type of semantic label, dictates whether or not the date expansion axiom applies. Note also that the expansion predicate can be defined for parts of reports not just for whole reports. So suppose we have a weather report on Paris in August 1999 that includes a complex feature on the eclipse and a complex feature on rainfall, then we could assume that the former complex feature is expandable whilst the later is not. In this way, the expansion axiom provides a flexible way of inferencing with temporal information.

Definition 4.9 *Let* θ *be structured news report where the following holds:*

$$\{\texttt{P1(X,Z1), ... ,Pn(X,Zn),T(X,D)}\} \subseteq Formulate(\theta)$$

The **Expansion axiom** *is defined as follows, where* i *is the required granularity for the inferences, and there is implicit universal quantification outermost:*

$$
\begin{array}{l}
\texttt{Expandable(P1,}i_1\texttt{)} \wedge ... \wedge \texttt{Expandable(Pn,}i_n\texttt{)} \\
\wedge\texttt{Holds(P1(X,Z1))} \wedge ... \wedge \texttt{Holds(Pn(X,Zn))} \\
\wedge\texttt{Eventtime(T(X,D))} \wedge \texttt{NewLabel(Y)} \\
\wedge\texttt{In(D,D}'\texttt{)} \wedge \texttt{Granularity (D}'\texttt{,}i\texttt{)} \\
\wedge\texttt{Subgrain(}i,i_1\texttt{)} \wedge ... \wedge \texttt{Subgrain(}i,i_n\texttt{)} \\
\rightarrow \\
\texttt{Holds(T(Y,D}'\texttt{))} \wedge \texttt{Holds(P1(Y,Z1))} \wedge ... \wedge \texttt{Holds(Pn(Y,Zn))}
\end{array}
$$

So by application of the expansion axiom, we can assume `T(Y,D'), P1(Y,Z1), ... , Pn(Y,Zn)` *as additions to the literals obtained from the news report* θ.

Example 4.4 *Consider the the one day weather report* `r1`.

```
report(r1)
rainfall(r1, 10cm)
middaytemp(r1, 10C)
date(r1, 3/3/00)
log(r1, 4/3/00)
```

Assuming

Expandable(middaytemp,*month*)
Expandable(log,*month*)
NewLabel(r2)
Eventtime(date(r1,3/3/00))
Granularity(March2000,*month*)
Subgrain(*month*,*month*)
In(3/3/00,March2000)

and a required granularity of month, we get inferences `date(r2,March2000)`*,* `log(r2,4/3/00)`*, and* `middaytemp(r2,10C)`*.*

Example 4.5 *Consider the business reports* `r1`*,* `r2` *and* `r3`*,*

report(r1)	report(r2)	report(r3)
buyer(r1, a)	buyer(r2, a)	buyer(r3, a)
date(r1, 1/7/00)	date(r2, 7/7/00)	date(r3, 9/7/00)
acquisition(r1, b)	acquisition(r2, c)	acquisition(r3, d)
log(r1, 10/7/00)	log(r2, 10/7/00)	log(r3, 10/7/00)

Assuming a required granularity of month together with the following,

Expandable(buyer,*year*)	In(1/7/00, July2000)
Expandable(acquisition,*year*)	In(7/7/00, July2000)
Expandable(log,*year*)	In(9/7/00, July2000)
Granularity(July2000,*month*)	Subgrain(*month*,*year*)
NewLabel(r4)	Eventtime(date(r1,1/7/00))
NewLabel(r5)	Eventtime(date(r2,7/7/00))
NewLabel(r6)	Eventtime(date(r3,22/7/00))

we obtain

buyer(r4, a)	buyer(r5, a)	buyer(r6, a)
acquisition(r4, b)	acquisition(r5, b)	acquisition(r6, c)
date(r4, July2000)	date(r5, July2000)	date(r6, July2000)
log(r4, 10/7/00)	log(r5, 10/7/00)	log(r6, 10/7/00)

In practice, assumptions about predicates such as `Expandable` need to depend on the context. So for example, in the metalevel knowledge, there will be formulae of the form,

$$\alpha \rightarrow \texttt{Expandable(middaytemp,}\mathit{month}\texttt{)}$$

where α would be some condition that may take into account other information in the news report. For example, α could be a `Holds` literal.

4.4 Axioms for contraction

As an illustration of when we would want to use the date contraction axiom, consider a weather report about the weather in London being unsettled during May 2000. Here if we have no other weather reports about this period in London, then we would want to apply the date contraction axiom and obtain that the weather is unsettled on each day in May 2000.

As an illustration of when we would not want to use the date contraction axiom. Consider a weather report about an eclipse occurring in Europe in August 1999. Here we would not want to apply the date contraction axiom and obtain that the eclipse occurs on each day in August 1999.

Definition 4.10 *Let θ be structured news report where the following holds:*

$$\{\mathtt{P1(X,Z1)}, ..., \mathtt{Pn(X,Zn)}, \mathtt{T(X,D)}\} \subseteq \mathtt{Formulate}(\theta)$$

The **Contraction axiom** *is defined as follows, where i is the required granularity for the inferences, and there is implicit universal quantification outermost:*

$$\begin{array}{l}
\mathtt{Contractable(P1,}i_1\mathtt{)} \wedge ... \wedge \mathtt{Contractable(Pn,}i_n\mathtt{)} \\
\wedge \mathtt{Holds(P1(X,Z1))} \wedge ... \wedge \mathtt{Holds(Pn(X,Zn))} \\
\wedge \mathtt{Eventtime(T(X,D))} \wedge \mathtt{NewLabel(Y)} \\
\wedge \mathtt{In(D',D)} \wedge \mathtt{Granularity\ (D',}i\mathtt{)} \\
\wedge \mathtt{Subgrain(}i_1, i\mathtt{)} \wedge ... \wedge \mathtt{Subgrain(}i_n, i\mathtt{)} \\
\rightarrow \\
\mathtt{Holds(T(Y,D'))} \wedge \mathtt{Holds(P1(Y,Z1))} \wedge ... \wedge \mathtt{Holds(Pn(Y,Zn))}
\end{array}$$

So by application of the contraction axiom, we can assume `T(Y,D'), P1(Y,Z1), ... , Pn(Y,Zn)` *as additions to the literals obtained from the news report θ.*

Example 4.6 *Consider the the one month weather report* `r1`*.*

```
dailyreport(r1)
rainfall(r1, 10cm)
middaytemp(r1, 10C)
date(r1, March2000)
log(r1, 4/4/00)
```

Assuming

$$\begin{array}{l}
\mathtt{Contractable(middaytemp,}day\mathtt{)} \\
\mathtt{Contractable(log,}day\mathtt{)} \\
\mathtt{NewLabel(r2)}, ...\mathtt{NewLabel(r32)} \\
\mathtt{Eventtime(date(r1,March2000))} \\
\mathtt{Granularity(1/3/00,}day\mathtt{)} \wedge .. \wedge \mathtt{Granularity(31/3/00,}day\mathtt{)} \\
\mathtt{Subgrain(}day,day\mathtt{)} \\
\mathtt{In(1/3/00,March2000)} \wedge .. \wedge \mathtt{In(31/3/00,March2000)}
\end{array}$$

and a required granularity of day, we get the inferences:

$$\begin{array}{c}
\mathtt{date(r2,1/3/00)}, ..., \mathtt{date(r32,31/3/00)} \\
\mathtt{middaytemp(r2,10C)}, ..., \mathtt{middaytemp(r32,10C)} \\
\mathtt{log(r2,4/4/00)}, ..., \mathtt{log(r32,4/4/00)}
\end{array}$$

We use the output from the above example as part of an example of merging in Example 5.4.

Example 4.7 *Consider the one week weather report* `r2`*.*

```
report(r2)
weather(r2, sun)
date(r2,1/7/00-7/7/00)
log(r2, 1/7/00)
```

Using the assumptions,

$$\begin{array}{l}
\texttt{Contractable(weather,}\mathit{day}\texttt{)} \\
\texttt{Contractable(log,}\mathit{day}\texttt{)} \\
\texttt{NewLabel(r3)}, ...\texttt{NewLabel(r9)} \\
\texttt{Eventtime(date(r2,1/7/00-7/7/00))} \\
\texttt{Granularity(1/7/00,}\mathit{day}\texttt{)} \wedge .. \wedge \texttt{Granularity(7/7/00,}\mathit{day}\texttt{)} \\
\texttt{Subgrain(}\mathit{day,day}\texttt{)} \\
\texttt{In(1/7/00,1/7/00-7/7/00)} \wedge .. \wedge \texttt{In(31/7/00,1/7/00-7/7/00)}
\end{array}$$

and a required granularity of day, together with the contraction axiom, we obtain

$$\begin{array}{c}
\texttt{weather(r3, sun), weather(r4, sun)}, ..., \texttt{weather(r9, sun)} \\
\texttt{log(r3, 1/7/00), log(r4, 1/7/00)}, ..., \texttt{log(r9, 1/7/00)} \\
\texttt{date(r3, 1/7/00), date(r4, 2/7/00)}, ..., \texttt{date(r9, 7/7/00)}
\end{array}$$

As with the `Expandable` predicates discussed above, we assume that the `Contractable` predicates will in practice need to be made contingent on other information holding in a given news report.

5 Merging news reports

In this section, we enhance our framework for merging news reports (given in [Hun00a]) by incorporating temporal domain knowledge.

5.1 Representing merged predicates

Given a set of news reports represented by a set of predicates, we want to infer a set of predicates that contain information merged from the original set of predicates. These inferred predicates are called merged predicates. To clarify, we need some subsidiary definitions.

Definition 5.1 *In the language for the domain knowledge, the* **label conjunction** *function, denoted* $+$*, is a function symbol denoting conjunction of reference labels. So if* `X` *and* `Y` *are reference labels, then* $\texttt{X} + \texttt{Y}$ *is a reference label. The domain knowledge includes axioms for associativity, commutativity and idempotence for this function symbol.*

Definition 5.2 *The* **labels** *function, denoted Labels, is defined as follows: For a structured report* θ*,* $Labels(\theta)$ *is the set of reference labels in* $Formulate(\theta)$*. For a set of structured reports* $\Theta = \{\theta_1, .., \theta_n\}$*,*

$$Labels(\Theta) = \{r_1 + ... + r_n \mid r_1 \in Labels(\theta_1) \text{ and } .... \text{ and } r_n \in Labels(\theta_n)\}$$

Essentially, predicates that represent information merged from a set of reports Θ will have reference labels from $Labels(\Theta)$. This is formalized in Definition 5.4.

Definition 5.3 *A* **closure** *function, denoted* $Closure_\mu$*, is defined as follows. Let* Δ *be the domain knowledge, let* Γ *be a set of merging axioms, let* $\mu = (\Delta, \Gamma)$*, let* $\Theta = \{\theta_1, .., \theta_n\}$ *be a set of structured reports, and let* $\vdash$ *be the classical logic consequence relation.*

$$\begin{aligned} Closure_\mu(\Theta) = \{\alpha(\beta, \gamma) \mid \Delta \cup \Gamma \cup Formulate(\Theta) \vdash \alpha(\beta, \gamma)\} \\ \cup \{\alpha(\beta) \mid \Delta \cup \Gamma \cup Formulate(\Theta) \vdash \alpha(\beta)\} \end{aligned}$$

Each formula in $Closure_\mu(\Theta)$ *is an inferred predicate.*

In [Hun00a], we provide details on the kinds of axioms that could be included in the domain knowledge and merging axioms. In the following sections, we review some of the details and extend the framework with temporal knowledge.

Definition 5.4 *The set of merged predicates in* $Closure_\mu(\Theta)$ *is* $Merged_\mu(\Theta)$ *defined as*

$$\begin{aligned} Merged_\mu(\Theta) = \{\alpha(\beta,\gamma) \in Closure_\mu(\Theta) \mid \beta \in Labels(\Theta)\} \\ \cup \{\alpha(\beta) \in Closure_\mu(\Theta) \mid \beta \in Labels(\Theta)\} \end{aligned}$$

Any predicate in $Merged_\mu(\Theta)$ *is a merged predicate.*

To illustrate, the above definition consider the following example.

Example 5.1 *Suppose* Θ *is based on two reports, and from these reports we have predicates* `log(r1,29/3/00)` *and* `log(r2,29/3/00)`*, and we have the following axiom.*

$$\texttt{log(X,D)} \land \texttt{log(Y,D)} \rightarrow \texttt{log(X} + \texttt{Y,D)}$$

Then `log(r1+r2,29/3/00)` *would be a merged predicate.*

When merging a set of news reports, the merged predicates are the formulae that contain the merged information. An item of structured text can be constructed from the merged predicates.

5.2 Domain knowledge

To reason with structured reports, we use the logical representation obtained via Definition 2.9. Since this definition only gives a set of positive literals for each structured report, we can only identify interesting inferences and inconsistencies in structured reports, by reasoning with domain knowledge.

Definition 5.5 *For an application,* **domain knowledge***, denoted* Δ*, is a set of classical first-order formulae.*

The choice of axioms in the domain knowledge depends on the application. We explain in the following definitions and examples the kinds of formulae that are included in such a set.

Definition 5.6 *In the language for merging systems (i.e. the language for domain knowledge and merging axioms),* $\oplus$ *is a function symbol denoting a conjunction of text entries, and* $\otimes$ *is a function symbol denoting a disjunction of text entries. For this, we assume the following schema for every predicate symbol* α*, and every term* X*,* A *and* B *in the language:*

$$\alpha(X, A \oplus B) \leftrightarrow (\alpha(X,A) \land \alpha(X,B))$$

$$\alpha(X, A \otimes B) \leftrightarrow (\alpha(X,A) \lor \alpha(X,B))$$

Inferencing and consistency checking with a set of structured reports and associated domain knowledge Δ is therefore viewed as logical reasoning with $\Delta \cup Formulate(\Theta)$. We base the logical reasoning on the usual definition of the first-order classical consequence relation.

Key types of temporal knowledge that are required include timestamp equivalence axioms (see 2.4) and pointwise axioms (see 2.5). Further types of temporal domain knowledge that could be used include temporal integrity constraints and axioms to capture relationships between events and time.

In order to determine whether a pair of structured reports represented as formulae are on the same subject, we need axioms to reason with formulae, such as in Example 2.12. These axioms are needed to identify equivalences between text entries including synonymous text entries. In practice, we would need axioms for text entries to semantic labels such as `name`, `date`, `location`, `source`, etc.

We also require domain knowledge for defining when certain combinations of text entry are inconsistent. For example, for a semantic label that refers to the weather in London at midday today, suppose the text entry in one report is `rain` and in another report is `sun`, then `sun` and `rain` are "inconsistent". We represent this by the predicate `incoherent(rain,sun)`. More generally, `incoherent(X,Y)` means `X` and `Y` are incoherent. We discuss this set up in more detail in [Hun00a].

Depending on the application, further specialized knowledge may also be required including:

Context-sensitive lexical knowledge Synonymous words; More general terms; More specialized terms, Preferred terms; Related terms; Meronyms; Antonyms; etc.

Context-sensitive world knowledge Concept definitions; Relationships between concepts; Geographic knowledge; Knowledge about calendars and clocks; Knowledge about people and organizations; Common-sense knowledge; etc.

Some options for representing and reasoning with context-sensitive knowledge in text handling are discussed in [Hun96, Hun97, HM99, HM01, Hun01a, Hun01b]. However, we do not consider context-dependent knowledge further in this paper.

5.3 Merging axioms

We now show how we can exploit the framework for temporal domain knowledge given in Section 5.2. For this, we use merging axioms (as presented in [Hun00a]) which are represented in first-order classical logic. These merging axioms are used with the domain knowledge to derive merged predicates. This is done by application of the classical consequence relation. In the following examples, we illustrate how domain knowledge and merging axioms can be used to obtain merged predicates.

In Examples 5.2 to 5.6, merging is done on information that is of the same granularity of time. This therefore involves the use of temporal granularity axioms and timestamp equivalence axioms. Once the information in the report is in a common form, the merging axioms form merged predicates that select information in case of conflict. For further details on merging axioms, including how we can recover structured text that represents the merged information, see [Hun00a].

In our first merging example (Example 5.2), we consider a pair of reports that conflict on the weather on a particular day, but one of the reports has a more recent log time, and so contains information that is selected in preference to the other report.

Example 5.2 *Consider the one day weather reports* r1 *and* r2

dayreport(r1)	dayreport(r2)
weather(r1, rain)	weather(r2, sun)
date(r1, 28/7/00)	date(r2, 28/7/00)
log(r1, 1/7/00)	log(r2, 5/7/00)

Using the following merging axiom, where T1 *and* T2 *are pointbased text entries,* T1-T2 *is an intervalbased text entry,* $<$ *is an ordering relation over timepoints, and* cograined(T1,T2) *holds when* T1 *and* T2 *are of the same granularity.*

$$\begin{array}{l}\forall \mathtt{X}, \mathtt{Y}, \mathtt{A}, \mathtt{B}, \mathtt{D}, \mathtt{T1}, \mathtt{T2} \\ \mathtt{weather}(\mathtt{X}, \mathtt{A}) \wedge \mathtt{weather}(\mathtt{Y}, \mathtt{B}) \wedge \mathtt{incoherent}(\mathtt{A}, \mathtt{B}) \\ \wedge \mathtt{log}(\mathtt{X}, \mathtt{T1}) \wedge \mathtt{log}(\mathtt{Y}, \mathtt{T2}) \wedge \mathtt{T1} < \mathtt{T2} \wedge \mathtt{cograined}(\mathtt{T1}, \mathtt{T2}) \\ \wedge \mathtt{date}(\mathtt{X}, \mathtt{D}) \wedge \mathtt{date}(\mathtt{Y}, \mathtt{D}) \\ \rightarrow \mathtt{weather}(\mathtt{X} + \mathtt{Y}, \mathtt{B}) \wedge \mathtt{date}(\mathtt{X} + \mathtt{Y}, \mathtt{D}) \wedge \mathtt{log}(\mathtt{X} + \mathtt{Y}, \mathtt{T1\text{-}T2})\end{array}$$

we obtain,

$$\mathtt{weather}(\mathtt{r1} + \mathtt{r2}, \mathtt{sun}), \mathtt{date}(\mathtt{r1} + \mathtt{r2}, 28/7/00), \mathtt{log}(\mathtt{r1} + \mathtt{r2}, 1/7/00\text{-}5/7/00)$$

Here this merging axiom has selected the information about weather from r2 *in preference to* r1 *because the log time for* r2 *is more recent.*

In our second merging example (Example 5.3), we consider a pair of reports where the event time of the second report is one day after the first. We can merge these to form a multiday report.

Example 5.3 *Consider the one day weather reports* r1 *and* r2

dayreport(r1)	dayreport(r2)
weather(r1, rain)	weather(r2, sun)
date(r1, 28/7/00)	date(r2, 29/7/00)
log(r1, 1/7/00)	log(r2, 5/7/00)

Using the merging axiom where nextday(D1,D2) *holds when* D2 *is the next day after* D1.

$$\begin{array}{l}\forall \mathtt{X}, \mathtt{Y}, \mathtt{D1}, \mathtt{D2} \\ \mathtt{dayreport}(\mathtt{X}) \wedge \mathtt{dayreport}(\mathtt{Y}) \wedge \mathtt{date}(\mathtt{X}, \mathtt{D1}) \wedge \mathtt{date}(\mathtt{Y}, \mathtt{D2}) \wedge \mathtt{nextday}(\mathtt{D1}, \mathtt{D2}) \\ \rightarrow \mathtt{multidayreport}(\mathtt{X} + \mathtt{Y}) \wedge \mathtt{multidayreport}(\mathtt{X} + \mathtt{Y}, \mathtt{X}) \wedge \mathtt{multidayreport}(\mathtt{X} + \mathtt{Y}, \mathtt{Y})\end{array}$$

we obtain

$$\mathtt{multidayreport}(\mathtt{r1} + \mathtt{r2}), \mathtt{multidayreport}(\mathtt{r1} + \mathtt{r2}, \mathtt{r1}), \mathtt{multidayreport}(\mathtt{r1} + \mathtt{r2}, \mathtt{r2})$$

Here this merging axiom has merged r1 *and* r2 *by forming a joint report* r1+r2 *where* r1 *and* r2 *are both nested reports. So* r1+r2 *is the unique reference label.*

In our third merging example (Example 5.4), we consider a pair of reports where the event times are of different granularities, but the first occurs for a day within the week of the second report. To address this example, we draw on the contraction axiom, as discussed in Example 4.7.

Example 5.4 *Consider the one day weather report* r1 *and the one week weather report* r2.

report(r1)	report(r2)
weather(r1, rain)	weather(r2, sun)
date(r1, 3/7/00)	date(r2,1/7/00-7/7/00)
log(r1, 1/7/00)	log(r2, 1/7/00)

As discussed in Example 4.7, we obtain `date(r3,1/7/00), date(r4,2/7/00), ..., date(r9,7/7/00)` *using the contraction axiom. Then using the merging axiom,*

$$\begin{array}{l}\forall \mathtt{X}, \mathtt{Y}, \mathtt{A}, \mathtt{B}, \mathtt{D}, \mathtt{E} \\ \mathtt{date}(\mathtt{X}, \mathtt{D}) \wedge \mathtt{date}(\mathtt{Y}, \mathtt{D}) \wedge \mathtt{weather}(\mathtt{X}, \mathtt{A}) \wedge \mathtt{weather}(\mathtt{Y}, \mathtt{B}) \\ \wedge \mathtt{incoherent}(\mathtt{A}, \mathtt{B}) \wedge \mathtt{log}(\mathtt{X}, \mathtt{E}) \wedge \mathtt{log}(\mathtt{Y}, \mathtt{E}) \\ \rightarrow \mathtt{date}(\mathtt{X}+\mathtt{Y}, \mathtt{D}) \wedge \mathtt{weather}(\mathtt{X}+\mathtt{Y}, \mathtt{A} \otimes \mathtt{B}) \wedge \mathtt{log}(\mathtt{X}+\mathtt{Y}, \mathtt{E})\end{array}$$

we obtain,

$$\mathtt{date}(\mathtt{r1}+\mathtt{r5}, 3/7/00), \mathtt{weather}(\mathtt{r1}+\mathtt{r5}, \mathtt{sun} \otimes \mathtt{rain}), \mathtt{log}(\mathtt{r1}+\mathtt{r5}, 1/7/00)$$

Here, we get the disjunction $\mathtt{sun} \otimes \mathtt{rain}$ *as the text entry for* `weather` *in the merged predicate.*

In our fourth example (Example 5.5), we consider three reports that have event times on different days of the same month. Using the Expansion axiom, the information in these reports can be merged as information occurring in the same month. To illustrate this, we draw on Example 4.5 that covers the application of the Expansion axiom to these reports.

Example 5.5 *Consider the business reports* `r1`*,* `r2` *and* `r3`*,*

$$\begin{array}{lll}\mathtt{report}(\mathtt{r1}) & \mathtt{report}(\mathtt{r2}) & \mathtt{report}(\mathtt{r3}) \\ \mathtt{buyer}(\mathtt{r1}, \mathtt{a}) & \mathtt{buyer}(\mathtt{r2}, \mathtt{a}) & \mathtt{buyer}(\mathtt{r3}, \mathtt{a}) \\ \mathtt{date}(\mathtt{r1}, 1/7/00) & \mathtt{date}(\mathtt{r2}, 7/7/00) & \mathtt{date}(\mathtt{r3}, 9/7/00) \\ \mathtt{acquisition}(\mathtt{r1}, \mathtt{b}) & \mathtt{acquisition}(\mathtt{r2}, \mathtt{c}) & \mathtt{acquisition}(\mathtt{r3}, \mathtt{d}) \\ \mathtt{log}(\mathtt{r1}, 10/7/00) & \mathtt{log}(\mathtt{r2}, 10/7/00) & \mathtt{log}(\mathtt{r3}, 10/7/00)\end{array}$$

Using the expansion axiom, as discussed in Example 4.5, we obtain the following predicates:

$$\begin{array}{c}\mathtt{date}(\mathtt{r4}, \mathtt{July2000}), \mathtt{date}(\mathtt{r5}, \mathtt{July2000}), \mathtt{date}(\mathtt{r6}, \mathtt{July2000}) \\ \mathtt{acquisition}(\mathtt{r4}, \mathtt{b}), \mathtt{acquisition}(\mathtt{r5}, \mathtt{c}), \mathtt{acquisition}(\mathtt{r6}, \mathtt{d}) \\ \mathtt{buyer}(\mathtt{r4}, \mathtt{a}), \mathtt{buyer}(\mathtt{r5}, \mathtt{a}), \mathtt{buyer}(\mathtt{r6}, \mathtt{a}) \\ \mathtt{log}(\mathtt{r4}, 10/7/00), \mathtt{log}(\mathtt{r5}, 10/7/00), \mathtt{log}(\mathtt{r6}, 10/7/00)\end{array}$$

Now using the following merging axiom,

$$\begin{array}{l}\forall \mathtt{X}, \mathtt{Y}, \mathtt{A}, \mathtt{B}, \mathtt{C}, \mathtt{T} \\ \mathtt{date}(\mathtt{X}, \mathtt{D}) \wedge \mathtt{date}(\mathtt{Y}, \mathtt{D}) \\ \wedge \mathtt{buyer}(\mathtt{X}, \mathtt{A}) \wedge \mathtt{buyer}(\mathtt{Y}, \mathtt{A}) \\ \wedge \mathtt{acquisition}(\mathtt{X}, \mathtt{B}) \wedge \mathtt{acquisition}(\mathtt{Y}, \mathtt{C}) \\ \wedge \mathtt{log}(\mathtt{X}, \mathtt{T}) \wedge \mathtt{log}(\mathtt{Y}, \mathtt{T}) \\ \rightarrow \mathtt{date}(\mathtt{X}+\mathtt{Y}, \mathtt{D}) \wedge \mathtt{buyer}(\mathtt{X}+\mathtt{Y}, \mathtt{A}) \wedge \mathtt{acquisition}(\mathtt{X}+\mathtt{Y}, \mathtt{B} \oplus \mathtt{C}) \wedge \mathtt{log}(\mathtt{X}+\mathtt{Y}, \mathtt{T})\end{array}$$

we obtain,

$$\begin{array}{l}\mathtt{date}(\mathtt{r4}+\mathtt{r5}+\mathtt{r6}, \mathtt{July2000}) \\ \mathtt{buyer}(\mathtt{r4}+\mathtt{r5}+\mathtt{r6}, \mathtt{a}) \\ \mathtt{acquisition}(\mathtt{r4}+\mathtt{r5}+\mathtt{r6}, \mathtt{b} \oplus \mathtt{c} \oplus \mathtt{d}) \\ \mathtt{log}(\mathtt{r4}+\mathtt{r5}+\mathtt{r6}, 10/7/00)\end{array}$$

So by expansion, all the reports refer to the same timepoint, namely `July2000`*, and the information pertaining to* `buyer`*,* `acquisition` *and* `log` *can be conjoined in the merged predicate.*

In our fifth example (Example 5.6), we consider a pair of news reports of different granularities where some, but not all, of the information is expandable. To address, this example, we draw on the expansion axiom as discussed in Example 4.4.

Example 5.6 *Consider the one day weather report* `r1` *and the one month weather report* `s1`.

`report(r1)`	`report(s1)`
`rainfall(r1, 10cm)`	`rainfall(s1, 10cm)`
`middaytemp(r1, 10C)`	`middaytemp(s1, 10C)`
`date(r1, 3/3/00)`	`date(s1, March)`
`log(r1, 4/3/00)`	`log(s1, 1/4/00)`

Applying the expansion axiom to report `r1`, *we get inferences* `date(r2,March 2000)`, `log(r2,4/3/00)`, *and* `middaytemp(r2,10C)`. *Now using the following merging axiom,*

$$\begin{array}{l}\forall \mathtt{X,Y,A,B,D,E}\\ \mathtt{date(X,D)} \wedge \mathtt{date(Y,D)} \wedge \mathtt{middaytemp(X,A)} \wedge \mathtt{middaytemp(Y,B)}\\ \wedge\neg\mathtt{incoherent(A,B)} \wedge \mathtt{log(X,E)} \wedge \mathtt{log(Y,F)}\\ \rightarrow \mathtt{date(X+Y,D)} \wedge \mathtt{middaytemp(X+Y,A \oplus B)} \wedge \mathtt{log(X+Y,E \otimes F)}\end{array}$$

we get `date(r2+s1,March)`, $\mathtt{log(r2+s1,4/3/00}\otimes\mathtt{1/4/00)}$ *and* `middaytemp(r2+s1,10C)` *as merged predicates.*

This framework for merging is intended to be used as a specification for a merging system, rather than as a proposal to use logic directly in the implementation. Nonetheless it is useful to consider the computational viability of the framework. Each structured new report is represented by a set of ground predicates. It is also reasonable to restrict the universe of terms used in the domain knowledge to a finite set by bounding the set of time points under consideration and restricting the reference labels and other constant symbols to those appearing in the structured news reports. In this way, we can consider replacing the domain knowledge and merging axioms by a set of ground instances (ie. by a set of propositional formulae). In this way $Closure_{\mu}(Formulate(\theta))$ can be reduced to reasoning with propositional classical logic which is in the co-NP class.

6 Conclusions

In this paper, we have presented a framework for representing and reasoning with temporal domain knowledge. We have shown how this can be used with structured text, and that the inferences obtained from the domain knowledge and structured text can be useful for reasoning with potentially inconsistent structured reports and for merging potentially inconsistent structured reports.

In our framework, we assume each structured report incorporates a log time and a time for the story of the report. Furthermore we assume the structured reports may be heterogeneous in structure, and heterogeneous in source. Using the inferences from temporal domain knowledge, we can see how reports relate to each other according to log time and according to story time. These inferences allow us to reason about the reports over time, and so allow us to see how a subject of the reports evolves over time.

The technique for translating structured text into logic used in this paper is just one of a number techniques we could use. Advantages of this technique are that it provides a clear expression of the original tree structure used in the structured text and it is directly implementable in logic-based technology such as Prolog. A disadvantage of the technique is that it introduces numerous extra constant symbols which decrease the clarity of the information in the logical representation.

An alternative technique for translating structured text into logic is to use literals with more arguments and arguments with function symbols. For example, function symbols can be used to capture the tree structure of the original structure text where each semantic label is represented by a function symbols. In this way, an item of structured text can be captured by one literal. As another example, function symbols can be used to provide a decomposed representation of text entries (e.g. $\langle$`date` : `31October2001`$\rangle$ might be represented by the ground term `date(31,October,2001)`. An additional advantage of this richer representation is that we may decrease the number of axioms required including timestamp equivalence axioms and pointwise axioms. We leave the details to future papers.

As another technqiue for translating structured text into logic, we could use a logic that incorporates specialized data structures that are isomorphic to the items of structured text. Possibilities include the logic of frames [Hay95] and the logic of typed feature structures [Car92]. The downside with using such logics is that incorporation of temporal reasoning is more difficult once we deviate from classical logic as the underlying formalism.

Our framework draws on some established results on temporal logics. Even though temporal domain knowledge is based on first-order classical logic, it is established that this language is as expressive as temporal logic with the Until and Since modal operators. For a more detailed discussion see [Gol87, GHR94]. The novelty of the work here is that we formalize different granularities of timeline and their inter-relationships. This gives a more appropriate language for reasoning with structured reports.

In some structured text, we may find that some timestamps are underspecified. In other words, there are timestamps that are relative (for example `yesterday`, `one year ago`, `next week`, etc) or ambiguous (for example `Thursday`, `October`, etc). For relative timestamps, we can make the timestamp explicit if we know what the timestamp is relative to. For example, we can replace the relative timestamp `yesterday` by `28 October 2001` if we know that `yesterday` is relative to `29 October 2001`. Similarly for ambiguous timestamps, we can disambiguate if we know the temporal context of the ambiguous timestamp. For example, if the context of the timestamp `Thursday` is the first week of October 2001, then we could replace this timestamp by `4 October 2001`. Now, suppose it is only the event timestamps that can be underspecified. Then we can axiomatize the use of log timestamp to address this underspecification. We leave the details to a future paper.

The types of news reports we have considered have been relatively simple. The events — in this case weather reports and business reports — have a relatively simple ontology and the temporal information is explicitly represented and so there are no co-reference problems. However, if we increase the complexity of the news reports allowing a wider ontology for semantic labels and text entries, then we need richer domain knowledge. This includes the need for commonsense reasoning about time, and for general knowledge about relationships between events and time.

As an illustration of the kinds of relationship between events and time that we may need, consider the event calculus [KS86]. Using Horn clause logic, together with negation-as-failure, event calculus is an approach to reasoning about time, and particularly about events. A key aim of event calculus is for updating inferential databases. The calculus is intended to cope with updates that do not occur in the same order as the real world. It is a non-monotonic system, as default assumptions are made as to the nature of the updates, and these updates can be amended in the light of new information. The non-monotonic capability is mechanized by use of negation-as-failure. In many conventional databases, when new knowledge indicates that some existing information is no longer true, the existing information is deleted from the database. In this approach, new information is used to delineate the time periods over which existing information is true. So in the event calculus, updates are additive — they do not delete information.

A fundamental component of the event calculus is that events are regarded as more primitive than time, and that events are represented explicitly in Horn clause logic, in order to reason about time. Time periods can be regarded as a function of an event and the relationship that started the event. So, for example, *after(R,E)* represents a time period, where R is a relationship and E is an event. Information about states can then be derived by a relationship *holds(R, after(R,E))*. This can be abbreviated to *holds(after(R,E))*. Similarly, we can state this as *holds(before(R,E))*. An initial state can be given explicitly, or separate initialising events can result in an initial state. (eg. *holds(after(R,E)) if initiates(R,E)*, or *holds(before(R,E)) if terminates(R,E)*. So using a general rule, as above, we can now state specific rules for different applications.

This approach may best be developed by harnessing formalisms for common-sense reasoning. This is a research topic initiated by calls for "naive physics" models of the real world [MH69, Hay79], with a number of sophisticated proposals made (see for example [San94, Sha97, MS98]).

References

[Abi97] S Abiteboul. Querying semi-structured data. In *International Conference on Database Theory*, pages 1–18, 1997.

[All84] J Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23:123–154, 1984.

[ARP98] ARPA. *Message Understanding Conference: Proceedings of the Seventh Conference*. Morgan Kaufmann, 1998.

[BCD+93] S Benferhat, C Cayrol, D Dubois, J Lang, and H Prade. Inconsistency management and prioritized syntax-based entailment. In *Proc. of the 13th Int. Joint Conf. on AI*, 1993.

[BDP95] S Benferhat, D Dubois, and H Prade. How to infer from inconsistent beliefs without revising. In *Proc. of the 14th Int. Joint Conf. on AI*, 1995.

[BFG+96] H Barringer, M Fisher, D Gabbay, G Gough, I Hodkinson, A Hunter, P McBrien, R Owens, and M Reynolds. Languages, meta-languages, and metatem. *Journal of the Interest Group for Pure and Applied Logic*, pages 255–272, 1996.

[BH01] Ph Besnard and A Hunter. A logic-based theory of deductive arguments. *Artificial Intelligence*, 128:203–235, 2001.

[BJW00] C Bettini, S Jajodia, and S Wang. *Time Granularities in Databases, Data Mining, and Temporal Reasoning*. Springer, 2000.

[BKMS92] C Baral, S Kraus, J Minker, and V Subrahmanian. Combining knowledgebases of consisting of first-order theories. *Computational Intelligence*, 8:45–71, 1992.

[Bre89] G Brewka. Preferred subtheories: An extended logical framework for default reasoning. In *Proceedings of the Eleventh International Conference on Artificial Intelligence (IJCAI'89)*, pages 1043–1048, 1989.

[Bun97] P Buneman. Semistructured data. In *Proceedings of the ACM Symposium on Principles of Database Systems*, 1997.

[Bur82] J Burgess. Axioms for tense logic 1. since and until. *Notre Dame Journal of Formal Logic*, 23, 1982.

[Car92] B Carpenter. *The Logic of Typed-feature Structures*. Cambridge University Press, 1992.

[CGL+98a] D Calvanese, G De Giacomo, M Lenzerini, D Nardi, and R Rosati. Description logic framework for information integration. In *Proceedings of the 6th Conference on the Principles of Knowledge Representation and Reasoning (KR'98)*, pages 2–13. Morgan Kaufmann, 1998.

[CGL+98b] D Calvanese, G De Giacomo, M Lenzerini, D Nardi, and R Rosati. Source integration in data warehousing. In *Proceedings of the 9th International Workshop on Database and Expert Systems (DEXA'98)*, pages 192–197. IEEE Computer Society Press, 1998.

[Cho98] L Cholvy. Reasoning with data provided by federated databases. *Journal of Intelligent Information Systems*, 10:49–80, 1998.

[CL96] J Cowie and W Lehnert. Information extraction. *Communications of the ACM*, 39:81–91, 1996.

[CM01] L Cholvy and S Moral. Merging databases: Problems and examples. *International Journal of Intelligent Systems*, 16:1193–1221, 2001.

[Coh98] W Cohen. A web-based information system that reasons with structured collections of text. In *Proceedings of Autonomous Agents'98*, 1998.

[CRS93] C Cayrol, V Royer, and C Saurel. Management of preferences in assumption based reasoning. In *Information Processing and the Management of Uncertainty in Knowledge based Systems (IPMU'92)*, volume 682 of *Lecture Notes in Computer Science*. Springer, 1993.

[DP98] D Dubois and H Prade, editors. *Handbook of Defeasible Resoning and Uncertainty Management Systems*, volume 3. Kluwer, 1998.

[FG92] M Finger and D Gabbay. Adding a temporal dimension to a logical system. *Journal of Logic, Language and Information*, 1:203–234, 1992.

[FS99] E Franconi and U Sattler. A data warehouse conceptual data model for multidimensional aggregation. In S Gatziu, M Jeusfeld, M Staudt, and Y Vassiliou, editors, *Proceedings of the Workshop in Design and Management of Data Warehouses*, 1999.

[Gar88] P Gardenfors. *Knowledge in Flux*. MIT Press, 1988.

[GHR94] D Gabbay, I Hodkinson, and M Reynolds. *Temporal Logic: Mathematical Foundations and Computational Aspects*. Oxford University Press, 1994.

[GLO+01] I Goralwalla, Y Leontiev, M Tamer Ozsu, D Szafron, and C Combi. Temporal granularity: Completing the puzzle. *Journal of Intelligent Information Systems*, 16:41–64, 2001.

[Gol87] R Goldblatt. *Logics of Time and Computation*. CLSI, 1987.

[Gri97] R Grishman. Information extraction techniques and challenges. In M Pazienza, editor, *Information Extraction*. Springer, 1997.

[Hay79] P Hayes. Naive physics manifesto. In D Michie, editor, *Expert Systems in the Microelectronic Age*. Edinburgh University Press, 1979.

[Hay95] P Hayes. Logic of frames. In R Brachman, editor, *Readings in Knowledge Representation*. Morgan Kaufmann, 1995.

[HG00] K Hui and P Gray. Developing finite domain constraints – a data model approach. In *Proceedings of Computation Logic 2000 Conference*, pages 448–462. Springer, 2000.

[HGNY97] J Hammer, H Garcia-Molina, S Nestorov, and R Yerneni. Template-based wrappers in the TSIMMIS system. In *Proceedings of ACM SIGMOD'97*. ACM, 1997.

[HM99] A Hunter and L Marten. Context-sensitive reasoning with lexical and world knowledge. In *SOAS Working Papers in Linguisitcs*, volume 9, pages 373–386, 1999.

[HM01] A Hunter and L Marten. Default reasoning with structured text using world knowledge. Technical report, Department of Computer Science, University College London, 2001.

[Hun96] A Hunter. Intelligent text handling using default logic. In *Proceedings of the IEEE Conference on Tools with Artificial Intelligence (TAI'96)*, pages 34–40. IEEE Computer Society Press, 1996.

[Hun97] A. Hunter. Using default logic for lexical knowledge. In *Qualitative and Quantitative Practical Reasoning*, volume 1244 of *Lecture Notes in Computer Science*, pages 86–95. Springer, 1997.

[Hun00a] A Hunter. Merging potentially inconsistent items of structured text. *Data and Knowledge Engineering*, 34:305–332, 2000.

[Hun00b] A Hunter. Ramification analysis using causal mapping. *Data and Knowledge Engineering*, 32:1–27, 2000.

[Hun00c] A Hunter. Reasoning with inconsistency in structured text. *Knowledge Engineering Review*, 15:317–337, 2000.

[Hun01a] A Hunter. A default logic-based framework for context-dependent reasoning with lexical knowledge. *Journal of Intelligent Information Systems*, 16:62–87, 2001.

[Hun01b] A Hunter. Reasoning with structured text using lexical knowledge. Technical report, Department of Computer Science, University College London, 2001.

[Hun02a] A Hunter. Logical fusion rules for merging structured news reports. *Data and Knowledge Engineering*, 2002. (in press).

[Hun02b] A Hunter. Hybrid argumentation systems for structured news reports. *Knowledge Engineering Review*, 2002. (in press).

[KM91] H Katsuno and A Mendelzon. On the difference between updating a knowledgebase and revising it. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference (KR'91)*, pages 387–394. Morgan Kaufmann, 1991.

[KP98] S Konieczny and R Pino Perez. On the logic of merging. In *Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98)*, pages 488–498. Morgan Kaufmann, 1998.

[KS86] R Kowalski and M Sergot. A logic-based calculus of events. *New Generation Computing*, 4:67–95, 1986.

[LS98] P Liberatore and M Schaerf. Arbitration (or how to merge knowledgebases). *IEEE Transactions on Knowledge and Data Engineering*, 10:76–90, 1998.

[McA76] R McArthur. *Tense Logic*. Reidel, 1976.

[MH69] J McCarthy and P Hayes. Some philosophical problems from the standpoint of artifical intelligence. In B Meltzer and D Michie, editors, *Machine Intelligence 4*. Edinburgh University Press, 1969.

[Mot96] A Motro. Cooperative database systems. *International Journal of Intelligent Systems*, 11:717–732, 1996.

[MR70] R Manor and N Rescher. On inferences from inconsistent information. *Theory and Decision*, 1:179–219, 1970.

[MS98] R Miller and M Shanahan. Working papers of commonsense'98: The fourth symposium on logical formalization of commonsense reasoning. Technical report, Queen Mary and Westfield College, London, 1998.

[PHG+99] A Preece, K Hui, A Gray, P Marti, T Bench-Capon, D Jeans, and Z Cui. The kraft architecture for knowledge fusion and transformation. In *Expert Systems*. Springer, 1999.

[PM98] A Poulovassilis and P McBrien. A general formal framework for schema transformation. *Data and Knowledge Engineering*, 28:47–71, 1998.

[SA99] A Sahuguet and F Azavant. Building light-weight wrappers for legacy web datasources using W4F. In *Proceedings of the International Conference on Very Large Databases (VLDB'99)*, 1999.

[San94] E Sandewall. *Features and Fluents: The Representation of Dynamical Systems (Volume 1)*. Oxford University Press, 1994.

[Sha97] M Shanahan. *Solving the Frame Problem*. MIT Press, 1997.

[Sho88] Y Shoham. *Reasoning about Change*. MIT Press, 1988.

[SL90] A Sheth and J Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Computing Surveys*, 22:183–236, 1990.

[SO99] K Smith and L Obrst. Unpacking the semantics of source and usage to perform semantic reconciliation in large-scale information systems. In *ACM SIGMOD RECORD*, volume 28, pages 26–31, 1999.