Temporal Interval Reasoning with CLP($\mathcal{Q}$)

Igor Mozetič
Jožef Stefan Institute
Jamova 39, 1000 Ljubljana, Slovenia
igor.mozetic@ijs.si

Abstract

Temporal reasoning is an important aspect of common-sense reasoning. The SUMO upper ontology incorporates Allen's influential axiomatization of temporal relations, but without reasoning capabilities. We propose a refinement of the SUMO temporal ontology, implemented in CLP($\mathcal{Q}$), a Constraint Logic Programming system over the domain of rational numbers.

1 Introduction

An ontology is a conceptualization of a domain of interest. In the Semantic Web context, it should be formalized and provide enough detail and structure to enable computers to process its content. An upper ontology is limited to concepts that are meta, generic and abstract, i.e., general enough to address a broad range of domain areas. There is an attempt by IEEE to define the Standard Upper Ontology (IEEE, 2003), currently embodied in the SUMO (Suggested Upper Merged Ontology). The SUMO (Niles and Pease, 2001) provides a foundation for middle-level and domain ontologies, and its purpose is to promote data interoperability, information retrieval, automated inference, and natural language processing. Some of the general topics covered in the SUMO include:

- Structural concepts such as instance and subclass
- General types of objects and processes
- Abstractions including set theory, attributes, and relations
- Numbers and measures
- Temporal concepts, such as duration
- Parts and wholes
- Basic semiotic relations
- Agency and intentionality.

SUMO is represented in a simplified form of the KIF language (Knowledge Interchange Format) which is based on the first-order logic and has formally defined semantics (Hayes and Menzel, 2001). However, to the best of our knowledge, there is no theorem prover or interpreter of the KIF language publicly available. As a consequence, the SUMO cannot be directly used for reasoning.

The motivation for this paper is the lack of reasoning capabilities in the existing SUMO ontology. We attempt to extend the SUMO, by defining a part of the ontology in compatible terms and implementing it in an executable language. Here we focus on temporal reasoning, as an important aspect of common-sense reasoning. To represent axioms about temporal concepts, SUMO adopted the influential Allen's temporal interval algebra (Allen, 1983), but without any reasoning capabilities.

We have actually implemented the temporal interval algebra in CLP($\mathcal{Q}$) . This simple implementation allows for the reasoning about temporal events, such as:

- "If X precedes Y, and Y overlaps with Z, what are the possible temporal relations between X and Z ?"
- "If X takes longer then Y, can X occur during Y ?"
- "Given a set of temporally related events, what are the possible consistent scenarios on the time line ?"

In Section 2, we give an overview of the Allen's interval algebra and some of its extensions. Section 3 presents Constraint Logic Programming (CLP), and CLP($\mathcal{Q}$) in particular. We show the CLP($\mathcal{Q}$) implementation of the temporal axioms in Section 4, and give some examples of reasoning.

2 Allen's interval algebra

Allen (1983) proposed an interval algebra framework to represent hierarchical and possibly indefinite and incomplete temporal information. This differs from the representation based on timestamps, since it allows for *relative* relations and at different levels of granularity. Events are represented by time intervals (in contrast to time points). There are thirteen basic relations between time intervals (Figure 1). The basic relations are disjoint and exhaustive.

Relation	**Symbol**	**Inverse**	**Meaning**
X before Y	b	bi	
X meets Y	m	mi	
X overlaps Y	o	oi	
X starts Y	s	si	
X during Y	d	di	
X finishes Y	f	fi	
X equals Y	eq		

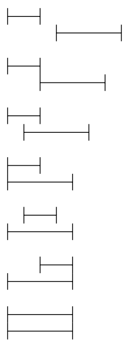

Figure 1: The basic relations between time intervals.

In order to represent indefinite information, Allen allows for any subset (disjunction) of the basic relations to hold between two time intervals. A set of temporally related events forms a network, with edges corresponding to (possibly disjunctive) relations between events. van Beek (1991) refers to such networks as IA (Interval Algebra) networks. There are two fundamental queries one can ask of IA networks:

1. Finding feasible relations between all pairs of events, and
2. Determining the consistency of temporal relations.

For IA networks, answering such queries was shown to be an NP-complete problem (Vilain et al., 1989). Therefore, van Beek (1991) proposed a restricted class of IA networks, called SIA (Simple Interval Algebra) networks. This class restricts the disjunctive relations between the time intervals just to those that can be expressed as *conjunctions* of equalities and inequalities between the interval endpoints. van Beek (1991) argues that the restricted class still covers most of the practical cases, while at the same time he gives a tractable polynomial time algorithms for answering the fundamental queries of the SIA networks.

3 Constraint Logic Programming

Constraint Logic Programming (CLP, Jaffar and Lassez (1987); Cohen (1990)) is a generalization of logic programming. Unification, the basic operation in logic programs, is replaced by a more general mechanism of constraint satisfaction over a specific computation domain. An instance of the general CLP scheme is obtained by selecting a computation domain, a set of allowed constraints and designing a solver for the constraints. CLP combines the advantages of logic programming (declarative semantics, nondeterminism, partial answers) with the efficiency of specialized constraint satisfaction algorithms. CLP($\mathcal{Q}$) is an instance of the CLP scheme which extends logic programs with interpreted arithmetic functions and a solver for systems of linear equations and inequalities over the domain of $\mathcal{Q}$ (rational numbers). In our experiments we use an implementation of CLP($\mathcal{Q}$) (Holzbaur, 1995) which is incorporated in the SICStus and Yap Prolog.

A CLP($\mathcal{Q}$) program is a set of clauses of the form:

$H \ \leftarrow \ C_1, \ldots, C_n.$

and a CLP($\mathcal{Q}$) query is a clause without head:

$\leftarrow \ C_1, \ldots, C_n.$

where H is an atom and C_i are negated or non-negated atoms or arithmetic constraints. Arithmetic constraints are bracketed by { and }, and consist of equations or inequalities, built up from rational constants, variables, $+, -, *, /$ and $=, \geq, \leq, >, <$. All of these symbols have the usual meaning and parentheses may be used. An atom is a predicate symbol applied to a number of terms. A term is a constant, a variable, an uninterpreted functor applied to a number of terms, or an arithmetic term. Variables start with capitals and are implicitly universally quantified in front of a clause, and constants start with lower-case letters.

CLP($\mathcal{Q}$) is restricted to systems of *linear* equations and inequalities. Non-linear constraints are accepted but not resolved — they are just delayed until (if) they eventually become linear. In the case of the Allen's temporal algebra, all constraints are linear. In general, however, a reply to a query is not just an answer substitution (as is the case with logic programs), but also potentially unresolved residual constraints between the variables involved.

4 CLP($\mathcal{Q}$) implementation and examples

In CLP($\mathcal{Q}$) we represent a temporal interval by a term *i(X1,X2)*, where *X1* and *X2* are rationals, representing the start and end points of the interval:

```
interval( i(X1,X2) )  ←  {X1 < X2}.
```

Duration of an interval is trivially defined by the following clause:

```
duration( i(X1,X2), Dur )  ←
     {X1 < X2, Dur = X2−X1}.
```

Basic temporal relations are defined in terms of equalities and inequalities between the endpoints. The following are the six basic relations and equality, their inverses are obvious:

```
temp( b, i(X1,X2), i(Y1,Y2) )  ←  {X2 < Y1}.
temp( m, i(X1,X2), i(Y1,Y2) )  ←  {X2 = Y1}.
temp( o, i(X1,X2), i(Y1,Y2) )  ←
     {X1 < Y1, X2 > Y1, X2 < Y2}.
temp( s, i(X1,X2), i(Y1,Y2) )  ←
     {X1 = Y1, X2 < Y2}.
temp( d, i(X1,X2), i(Y1,Y2) )  ←
     {X1 > Y1, X2 < Y2}.
temp( f, i(X1,X2), i(Y1,Y2) )  ←
     {X1 > Y1, X2 = Y2}.
temp( eq, i(X1,X2), i(Y1,Y2) )  ←
     {X1 = Y1, X2 = Y2}.
```

The SIA relations are all powersets of the basic relations which are consistent and can be expressed as conjunctions of (in)equalities between the endpoints. There are altogether 82 SIA relations which can be derived from the above basic definitions (and their inverses). For example, a disjunctive interval relation (X meets or overlaps or starts Y, in CLP($\mathcal{Q}$) represented by a list *[m,o,s]*) is defined as:

```
sia( [m,o,s], i(X1,X2), i(Y1,Y2) )  ←
     {X1 =< Y1, X2 >= Y1, X2 < Y2}.
```

Simple query. What are the constraints on the interval Y which overlaps with *i(1,4)*:

```
 ←  temp(o, i(1,4), i(Y1,Y2)).
Y1 > 1, Y1 < 4, Y2 > 4
```

Composition of two relations. Given "X starts Y and Y overlaps with Z", what is the relation between X and Z:

```
 ←  temp(s, i(X1,X2), i(Y1,Y2)),
    temp(o, i(Y1,Y2), i(Z1,Z2)),
    temp(Rel, i(X1,X2), i(Z1,Z2)).
Rel = b ? ;
Rel = m ? ;
Rel = o ?
```

Backtracking yields three answer substitutions which can be simplified into a SIA *[b,m,o]* with the only two relevant constrains remaining: *X1* < *Y1* and *X2* < *Y2*.

Scenario. Let's take an example description of events from (van Beek, 1991): "Fred was reading the paper while eating his breakfast. He put the paper down and drank the last of his coffee. After breakfast he went for a walk." Here we have four events: *Paper, Break, Coffee, Walk*. The (indefinite) temporal relations between them are described by the following four SIA relations:

```
sia( [d,di,eq,f,fi,o,oi,s,si], Paper, Break )
sia( [d,o,s], Paper, Coffee )
sia( [d], Coffee, Break )
sia( [b], Break, Walk )
```

Feasible relations. Assume that each event, e.g., *Coffee* is represented by an interval *i(C1,C2)*, and similarly *Walk* by *i(W1,W2)*. Given the above SIA network, we can compute all feasible temporal relations between any pair of events. E.g., between *Coffee* and *Walk*, the only feasible relation is *before*:

```
 ←  sia_net([i(P1,P2), i(B1,B2),
        i(C1,C2), i(W1,W2)]),
    temp(Feasible, i(C1,C2), i(W1,W2)).
Feasible = b,
```

... and some residual constraints

Consistent scenario. Another interesting question about a SIA network is finding a consistent scenario, i.e., a projection of the network to the time line. This can be simply realized by sorting the interval endpoints, while maintaining the consistency of constraints.

$\leftarrow$ *sia_net([i(P1,P2), i(B1,B2), i(C1,C2), i(W1,W2)]), sort([P1, P2, B1, B2, C1, C2, W1, W2], Scenario).*

We get five consistent scenarios:

Scenario = [P1,C1,P2,C2,B2,W1,W2], B1=P1
Scenario = [P1,B1,C1,P2,C2,B2,W1,W2]
Scenario = [B1,P1,P2,C2,B2,W1,W2], C1=P1
Scenario = [B1,P1,C1,P2,C2,B2,W1,W2]
Scenario = [B1,C1,P1,P2,C2,B2,W1,W2]

Note that the endpoints can be sorted without assigning actual numerical values to them!

Duration. Assume an interval X with duration longer then an interval Y:

$\leftarrow$ *duration(i(X1,X2), Xd), duration(i(Y1,Y2), Yd),* $\{Xd > Yd\}$.
Xd= -X1+X2, Y1-Y2 < 0, Yd= -Y1+Y2, X1-X2-Y1+Y2 < 0

Can X occur during Y? No, the corresponding query fails. Query for all the feasible relations between X and Y yields the following answers: *[b, m, o, bi, mi, oi, si, di, fi]*. Note that this is no longer a SIA relation since it cannot be represented by conjunctive constraints.

5 Conclusion

This modest contribution can be regarded as an attempt at making an ontology operational. In our view, it does not suffice to use or develop an expressive language to formalize an ontology. The language must also be executable in order to enable automated reasoning and derivation of explicit consequences from implicit knowledge in the ontology. When choosing between different competing representation languages, their operationality should be an important consideration.

Acknowledgements

This research was supported by the 6th FP Project IST-2003-506826 SEKT.

References

Allen, J.F. Maintaining knowledge about temporal intervals. *Communications of the ACM 26*, pp. 832-843, 1983.

Cohen, J. Constraint logic programming languages. *Communications of the ACM 33 (7)*, pp. 52-68, 1990.

Hayes, P., Menzel. C. A Semantics for the Knowledge Interchange Format. *http://reliant.teknowledge.com/IJCAI01/HayesMenzel-SKIF-IJCAI2001.pdf*, 2001.

Holzbaur, C. OFAI clp(q,r) Manual, Edition 1.3.3. Austrian Research Institute for Artificial Intelligence, Vienna, TR-95-09, 1995.

IEEE P1600.1. Standard Upper Ontology Working Group (SUO WG). *http://suo.ieee.org/*, 2003.

Jaffar, J., Lassez, J.-L. Constraint logic programming. *Proc. 14th ACM Symp. on Principles of Programming Languages*, pp. 111-119, Munich, 1987.

Niles, I., and Pease, A. 2001. Towards a Standard Upper Ontology. *Proc. 2nd Intl. Conf. on Formal Ontology in Information Systems (FOIS-2001)*, Chris Welty and Barry Smith (Eds), Ogunquit, Maine, October 17-19, 2001.

van Beek, P. Temporal query processing with indefinite information. *Artificial Intelligence in Medicine 3*, pp. 325-339, 1991.

Vilain, M., Kautz, H., van Beek, P. Constraint propagation algorithms for temporal reasoning: A revised report. *Readings in Qualitative Reasoning about Physical Systems*, pp. 325-339, 1991.