CHRONOS: A Reasoning Engine for Qualitative Temporal Information in OWL

Eleftherios Anagnostopoulos, Sotiris Batsakis, and Euripides G.M. Petrakis

Department of Electronic and Computer Engineering
Technical University of Crete (TUC)
Chania, Greece
e-mail:eanagnostopoulos@hotmail.com,{batsakis,petrakis}@intelligence.tuc.gr

Abstract. We propose CHRONOS, a system for reasoning over temporal information in OWL ontologies. Representing both qualitative temporal (i.e., information whose temporal extents are unknown such as "before", "after" for temporal relations) in addition to quantitative information (i.e., where temporal information is defined precisely e.g., using dates) is a distinctive feature of the proposed approach. Qualitative representations are very common in natural language expressions such as in free text or speech and can be proven to be valuable in the Semantic Web. CHRONOS reasoner applies over temporal relations in order to infer implied relations and to detect inconsistencies while retaining soundness, completeness and tractability over the supported relations set. The experimental results demonstrated that CHRONOS runs up to several times faster compared to its SWRL counterpart and scales-up well for large relation sets.

1 Introduction

Ontologies are the means for representing high level concepts, their properties and their interrelationships. Dynamic ontologies will in addition enable representation of information evolvingin time. Representation of dynamic features calls for mechanisms allowing for the representation of the notions of time (and of properties varying in time) within an ontology. Methods for achieving this goal include (among others) N-ary relations [1] and the 4D-fluents (perdurantist) approach [3]. In our previous work [8], both, the N-ary relations and the 4D fluents approaches are enhanced with qualitative (in addition to quantitative) temporal expressions allowing for the representation of temporal intervals with unknown starting and ending points by means of their relation (e.g., "before", "after") to other time intervals. Reasoning is realized by means of SWRL rules implementing the allowable compositions over the supported (tractable) relation set combined with OWL 2.0 constructs (e.g., for declaring disjoint properties) ensuring path consistency while being sound and complete [12]. Reasoning is embedded within the ontology and can be executed by a general purpose reasoner such as Pellet[1]. Although fast and intuitive, this approach still suffers from the following

[1] http://clarkparsia.com/pellet/

disadvantages: (a) Relying in a general purpose reasoner, it is not possible to accommodate any optimizations within the implementation of path consistency (resorting entirely to the performance of the SWRL interpreter at hand) and, (b) Implementing path consistency in SWRL calls for additional temporal relations thus complicating the representation. The proposed approach tackles both these problems. CHRONOS is independent of temporal model representing the evolution of concepts in time (i.e., works with both the N-ary and the 4D-fluent representations).

Following the example of Pellet-Spatial [11] and CHOROS [4], CHRONOS is a dedicated temporal reasoning engine implemented in Java and allows for incorporating certain optimizations in implementing path consistency. We run several performance comparisons against SOWL [2], our SWRL implementation of the temporal reasoner, on a large synthetic (but realistic) relations set. The performance of the two competitor reasoner implementations are fully analysed and discussed. The experimental results demonstrate that CHRONOS runs several times faster than its SWRL counterpart and scales-up well for large relation sets.

Related work in the field of knowledge representation is discussed in Sec. 2. This includes issues related to representing and reasoning over information evolving in time. Temporal representation is discussed Sec. 3 and CHRONOS is presented in Sec. 4. Experimental results are presented in Sec. 5 followed by conclusions and issues for future work in Sec. 6.

2 Background and Related Work

The OWL-Time temporal ontology[2] describes the temporal content of Web pages and the temporal properties of Web services. Apart from language constructs for the representation of time in ontologies, there is still a need for mechanisms for the representation of the evolution of concepts (e.g., events) in time. Temporal relations are in fact ternary (i.e., properties of objects that change in time involve also a temporal value in addition to the object and the subject) and cannot be expressed directly in OWL. Representing temporal relations in OWL calls for specific methods such as N-ary relations [1] or 4D-fluents [3].

The N-ary relations approach [1] suggests representing an n-ary relation as two properties each related with a new object, which in turn is related with the temporal interval that this relation holds. This approach requires only one additional object for every temporal relation and maintains property semantics. Fig. 1 illustrates the representation of the temporal property "Employee WorksFor Company during TimeInterval" using the N-ary approach.

The 4D fluents approach is discussed in our previous work [2, 8]. Using 4D fluents, for each object participating in a temporal property a new intermediate object is asserted, in contrast to a single object for the entire relation of the N-ary approach. Overall, the N-ary approach requires fewer objects and is the

[2] http://www.w3.org/TR/owl-time/

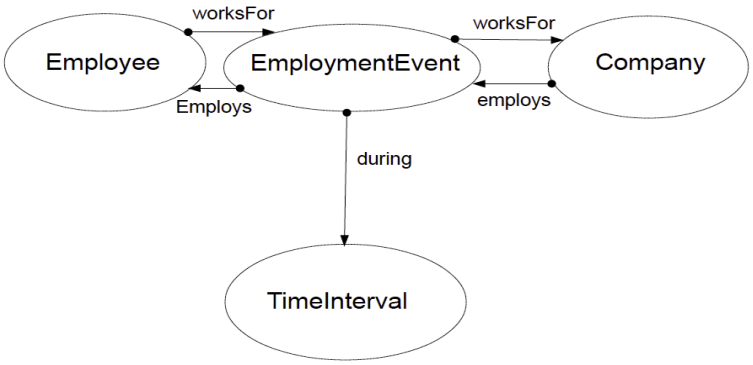

Fig. 1: N-ary relations Example

preferred approach. In [8], temporal intervals or instants can be defined either quantitatively (i.e., by specifying exact starting and ending points or dates) or, qualitatively, using natural language expressions such as "after", "before". In this later case, temporal intervals or instants are defined by means of their relations to other temporal intervals or time instants in the ontology.

Reasoning over temporal relations in [8] is achieved by means of SWRL rules embedded into the ontology. Limitations of SWRL (e.g., disjunctions of clauses as a rule result are not permitted) lead to additional temporal relations and more involved reasoning.

3 Temporal ontology

Following the N-ary relations approach, to add the time dimension to an ontology, classes *Event* and *TimeInterval* are introduced. Class *Event* represents temporal relations (which can be specialized subclasses of *Event* such as *EmploymentEvent* and class *TimeInterval* is the domain class of time intervals. Properties having a temporal dimension are called *fluent properties* and connect initial objects with instances of class *Event* (as in Fig.1). In [2] the temporal representation was enhanced with qualitative temporal relations (i.e., relations holding between time intervals whose starting and ending points are not specified) by introducing temporal relationships as object relations between time intervals. A temporal relation can be one of the 13 pairwise disjoint Allen's relations [7] of Fig. 2. Definitions for temporal entities (e.g., intervals) are provided by incorporating OWL-Time into the same ontology.

Relations between intervals are expressed as relations between their starting and ending points which in turn are expressed as a function of the three possible relations between points (time instants) namely $equals$, $before$ and $after$ denoted by "=", "<" and ">" respectively, forming the so called "point algebra" [12]. Let $i1 = [s1, e1]$ and $i2 = [s2, e2]$ be two intervals with starting and ending

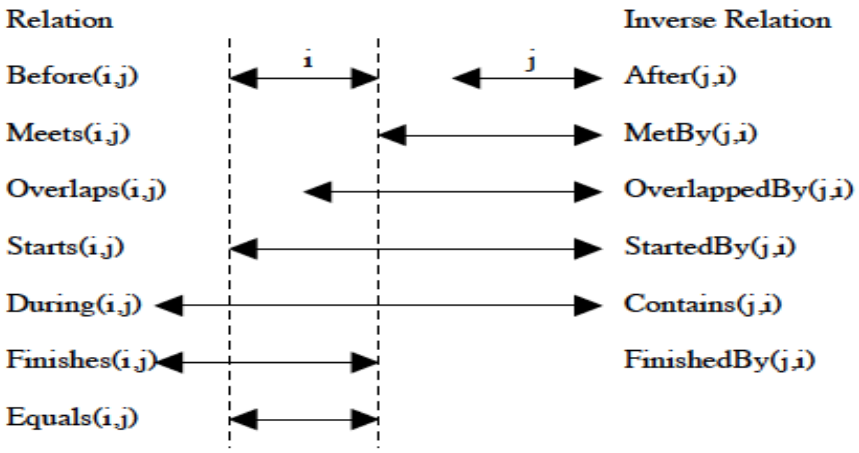

Fig. 2: Allen's Temporal Relations

points *s1, s2 and e1,e2* respectively; then, the 13 Allen relations of Fig. 2 are rewritten as follows:

$i1\ before\ i2 \equiv e1 < s2$
$i1\ equals\ i2 \equiv s1 = s2 \wedge e1 = e2$
$i1\ overlaps\ i2 \equiv s1 < s2 \wedge e1 < e2 \wedge e1 > s2$
$i1\ meets\ i2 \equiv e1 = s2$
$i1\ during\ i2 \equiv s1 > s2 \wedge e1 < e2$
$i1\ starts\ i2 \equiv s1 = s2 \wedge e1 < e2$
$i1\ finishes\ i2 \equiv s1 > s2 \wedge e1 = e2$

The relations *after, overlappedby, metby, contains, startedby and finishedby* are the inverse of *before, overlaps, meets, during, starts and finishes*, relation $equals$ is symmetric and relations $before$ and $after$ are transitive. These temporal relations and the corresponding reasoning mechanism are integrated both in the 4D-fluents and the N-ary based ontologies.

4 CHRONOS Reasoning

The architecture of CHRONOS reasoner is illustrated in Fig. 3. It consists of several modules, the most important of them being the following:

Parser: Implements an RDF[3] and an ARQ[4] parser for parsing ontologies and queries respectively. CHRONOS parses the ontology and temporal relations between intervals are detected and extracted, forming a Constraint Network. A

[3] http://www.w3.org/RDF/
[4] http://jena.apache.org/documentation/query/

Constraint Network (CN) is a set of variables together with a set of constraints defined on these variables. The constraint network of CHRONOS comprises of temporal triples in the ontology graph (extracted by the Ontology Parser), as well as, of non-temporal OWL assertions, which are stored in the Knowledge Base of Pellet reasoner, as it is typical in standard (i.e., non temporal) reasoning.

The user can search for relations (by specifying the temporal interval of interest), count the number of temporal and non-temporal relations, check the temporal network for consistency using the composition table of the temporal reasoner, and also check consistency of the non-temporal knowledge base by using Pellet [6].

Reasoning: CHRONOS separates temporal from semantic DL reasoning and uses an exclusive reasoner for temporal calculus. Allen relations are handled separately by their respective constraint network.

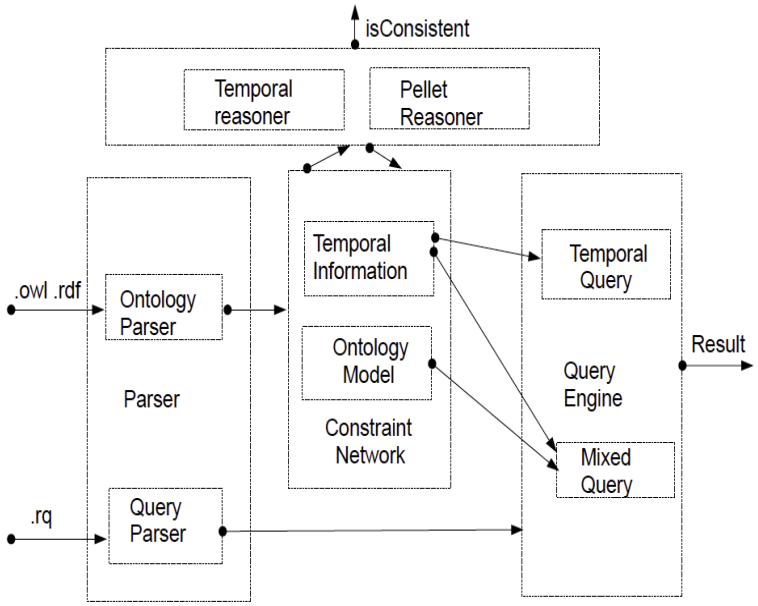

Fig. 3: CHRONOS Architecture

CHRONOS separates temporal from semantic DL reasoning, which is managed by Pellet. Temporal relations are expressed by a set of jointly exclusive and pairwise disjoint basic relations. Temporal reasoning (i.e., inferring implied relations and detecting inconsistencies) can be viewed as a form of a constraint satisfaction problem which is NP in the general case. CHRONOS handles tractable cases of such problems by limiting asserted temporal relations to the basic Allen relations of Fig. 2.

Temporal reasoning is then achieved by applying the path consistency algorithm of [4]. Path consistency computes all inferred relations using compositions of existing relations defined, until a fixed point is reached (i.e., algorithm does not yield new inferences) or until an inconsistency is detected (i.e., yield the empty relation as a result). Path consistency when applied on a set of assertions containing only basic relations retains tractability and guarantees soundness and completeness of reasoning [9].

The possible compositions of basic Allen temporal relations are defined in a composition table [7]. The composition table represents the result of the composition of two temporal relations. For example, if relation R_1 holds between $interval1$ and $interval2$, and relation R_2 holds between $interval2$ and $interval3$, then the entry of the composition table corresponding to line R_1 and column R_2 denotes the possible relation(s) holding between $interval1$ and $interval3$.

$$R_1(x, y) \wedge R_2(y, z) \rightarrow R_3(x, z)$$

The following is an example of such a temporal inference rule:

$$before(x, y) \wedge equals(y, z) \rightarrow before(x, z)$$

A series of compositions of relations may imply relations which are inconsistent with existing ones (for example the rule referred to above will yield a contradiction if *after(x,z)* has been asserted into the ontology for specific values of *x,y,z*). Consistency checking is achieved by ensuring path consistency [12]. Path consistency is implemented by consecutively applying the following formula:

$$\forall x, y, k \; R_s(x, y) \leftarrow R_i(x, y) \cap (R_j(x, k) \circ R_k(k, y))$$

representing intersection of compositions of relations with existing relations (symbol $\cap$ denotes intersection, symbol $\circ$ denotes composition and R_i, R_j, R_k, R_s denote temporal relations). The formula is applied until a fixed point is reached (i.e., the consecutive application of the rules above doesn't yield new inferences) or until the empty set is reached, implying that the ontology is inconsistent.

The composition of base relations may infer disjunctions of such relations because disjunctive entries exist in the composition table of Allen relations. For example, the composition of $Meets$ and $During$ yields the disjunction of relations *During, Overlaps* and *Starts* as a result. Composing disjunctions of relations can be achieved in two different ways: (a) By pre-computing the composition of disjunctions and storing the result in a table and (b) By computing the composition of disjunctions "on the fly". In our earlier work for CHOROS [4] spatial reasoner, we adopted the first approach, which is feasible for the set of 8 basic RCC-8 topological or for the set of 9 Directional realtions. For topologic relations the composition table has $2^8 \times 2^8$ entries (i.e., up to 2^8 disjunctions can appear). Similarly, for directional relations, the composition table has $2^9 \times 2^9$ entries. However, in CHRONOS and for the set of 13 basic Allen relations, the table must have $2^{13} \times 2^{13}$ entries which takes more memory to store and more time to compute. Following the second approach, calculating the compositions

on the fly, in practice results in less than $2^{13} \times 2^{13}$ combinations of disjunctions of basic relations and each one involves a simple look-up operation in the 13×13 Allen composition table [7].

Query Engine: CHRONOS supports similar query functionality to PelletSpatial and CHOROS, but for temporal operators. As such, it answers conjunctive queries specifying temporal and non-temporal patterns. More specifically, this module can process queries written in SPARQL satisfying the following conditions: (a) No variable is used in the predicate position, (b) Each property used in the predicate position is either an (object or datatype) property defined in the ontology or one of the following built-in properties: rdf : type, owl : sameIndividualAs, owl : differentFrom and, (c) At least one of the triples must contain a temporal object property in the predicate position (otherwise it is a non-temporal query). Similarly to PelletSpatial and CHOROS, the query engine can process temporal queries specifying temporal properties in an RDF graph or Allen relations between temporal intervals. The query engine implements a dual stage query answering technique: The set of query solutions returned by the first stage in response to a given a temporal query are fed to the second stage whose purpose is to guarantee that the non-temporal part of the query is satisfied as well.

5 Evaluation

In the following, the efficiency of our reasoning approach is assessed both, theoretically and experimentally. Reasoning calls for rules applying over temporal relations whose purpose is to infer implied relations, detect inconsistencies and ensure path consistency.

If only the basic Allen relations are supported (as it is the case in our present work), reasoning has polynomial time complexity [9, 12]. Because, within n intervals, any time interval can be related with every other interval with one basic Allen relation (basic Allen relations are mutually exclusive), at most $O(n^2)$ relations can be inferred. Furthermore, path consistency has $O(n^3)$ worst case complexity [5] and is sound and complete for the supported sets of relations. This upper bound is pessimistic, because an inconsistency detection may terminate the reasoning process early, or the asserted relations may yield a small number of inferences.

In the most general case where disjunctive relations are supported in addition to the basic ones, any time interval can be related with every other interval by at most k relations, where k is the size of the set of supported relations. Therefore, for n intervals or instants, using $O(k^2)$ rules, at most $O(kn^2)$ relations can be asserted into the knowledge base.

5.1 Experimental Evaluation

The purpose of the following set of experiments is to demonstrate the run-time efficiency of CHRONOS over SOWL [2], a temporal reasoner implemented in

SWRL. SOWL is an ontology for representing and reasoning over spatiotemporal information in OWL. Temporal reasoning in SOWL is realized by introducing a set of SWRL rules for asserting inferred temporal Allen relations.

To evaluate the reasoning performance of CHRONOS, we run several experiments demonstrating the run-time efficiency of reasoning as a function of the size of input data set. In our experiments, we compare the running time of the two competiror implementations (CHRONOS and SWRL) as a function of the number of temporal assertions in the input ontology. To study the run-time efficiency of reasoning, we used a data set comprising between 10 and 100 assertions. These assertions are used to populate 10 ontologies with random instances. Reasoning times are computed as averages over 10 iterations.

We carried out two different experiments, corresponding to measurements of performance in the average and in the worst case [13]. The average case performance is encountered when less than n^2 individuals are inferred from a input set of n temporal intervals. In the following experiment, these are in the order of n. Accordingly, the worst case performance is encountered when the number of infrerred relations are in the order of n^2. All experiments were run on a PC with 4 GB RAM running Windows 7 at 2.40 GHz.

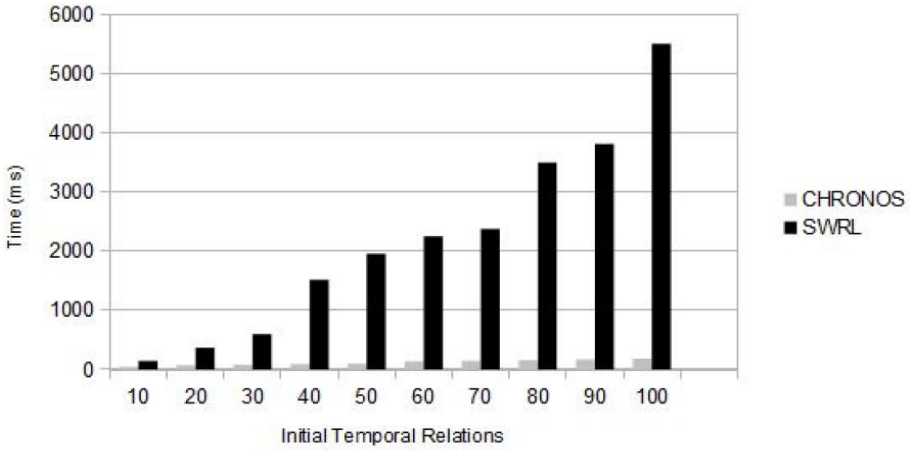

Fig. 4: Average response time (over 10 iterations) of reasoning as a function of the number of temporal assertions in an ontology

Fig. 4 illustrates the average response time of the CHRONOS and SOWL reasoners. Fig. 5 illustrates their run-time performance in the worst case. Obviously, CHRONOS clearly outperforms SOWL in all cases. In the average case, the number of inferred relations is in the order of n and the path-consistency algorithm of CHRONOS exhibits nearly linear running time performance.

Notice that, the run-time performance of SOWL declines drastically for large data sets as the large number of inferred relations caused the memory to over-

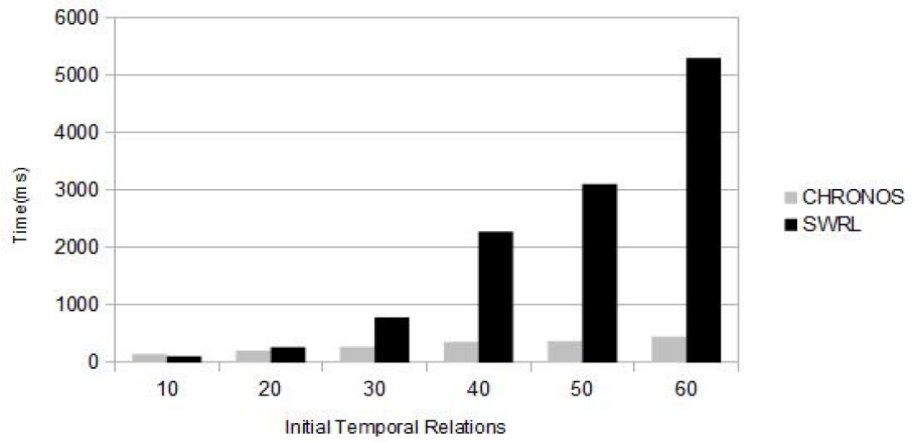

Fig. 5: Worst case response time of reasoning as a function of the number of temporal assertions in an ontology

flow. Although the SOWL reasoner may perform better on computers with more memory, CHRONOS scales-up much better than SOWL with the size of the input (i.e., the performance gap between the two reasoner implementations increases with the size of the data set) and will run much faster than SOWL for large data sets even on average computers.

6 Conclusions and Future Work

We propose CHRONOS, a stand-alone temporal OWL reasoner following the example of Pellet-Spatial and CHOROS for reasoning over temporal knowledge in ontologies using OWL. CHRONOS implements certain optimizations and exhibited increased performance improvements over SOWL in both the average and the worst cases. Extending CHRONOS to handle both temporal instants in addition to temporal intervals (as SOWL does) is a direction of future work.

Acknowledgement

Research leading to these results has received funding from the European Community's Seventh Framework Program (FP7/2007-2013) under grant agreement No 604691 (Project Fi-Star).

References

1. N. Noy, and B. Rector. "Defining N-ary Relations on the Semantic Web". W3C Working Group Note 12, April 2006, http://www.w3.org/TR/swbp-n-aryRelations/

2. S. Batsakis, and E.G.M. Petrakis. "SOWL: Spatio-temporal Representation, Reasoning and Querying over the Semantic Web" 6^{th} International Conference on Semantic Systems (I-SEMANTICS 2010), Graz, Austria, 1–3 September 2010.
3. C. Welty, and R. Fikes. "A Reusable Ontology for Fluents in OWL". *Frontiers in Artificial Intelligence and Applications*, 150:226–236, 2006.
4. G. Christodoulou, E. Petrakis E., and S. Batsakis. "Qualitative Spatial Reasoning using Topological and Directional Information in OWL", 24^{th} International Conference on Tools with Artificial Intelligence (ICTAI 2012), Athens, Greece, November 7-9, 2012.
5. J. Renz, and B. Nebel. "Qualitative Spatial Reasoning using Constraint Calculi". In Handbook of Spatial Logics, Springer, Netherlands , pp. 161-215,
6. B. Parsia, and E. Sirin. "Pellet: An OWL DL reasoner." Third International Semantic Web Conference (ISWC 2004), 2004.
7. J. F. Allen. "Maintaining Knowledge About Temporal Intervals". *Communications of the ACM*. 26:832-843, 1983.
8. S. Batsakis, K. Stravoskoufos, and E.G.M. Petrakis. Temporal Reasoning for Supporting Temporal Queries in OWL 2.0. 15^{th} International Conference on Knowledge-Based and Intelligent Information and Engineering Systems (KES 2011), 6881:558–567, 2011.
9. B. Nebel, and H.J. Burckert "Reasoning about Temporal Relations: A Maximal Tractable Subclass of Allen's Interval Algebra" Journal of the ACM (JACM), Vol.42(1), pages:43–66, 1995.
10. I. Horrocks, P.F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean "SWRL: A Semantic Web Rule Language Combining OWL and RuleML." W3C Member submission, 2004. http://www.w3.org/Submission/SWRL/
11. M. Stocker and E. Sirin. Pelletspatial: A Hybrid RCC-8 and RDF/OWL Reasoning and Query Engine. In *CEUR Workshop Proceedings*, **529**. Citeseer, 2009.
12. P. van Beek, and R. Cohen "Exact and approximate reasoning about temporal relations." Computational intelligence, Vol 6(3), pp. 132–147, 1990.
13. S. Batsakis, "SOWL: A Framework for Handling Spatio-Temporal Information in OWL". Ph.D. dissertation, Technical Univ. of Crete (TUC), Dept. of Electronics and Comp. Engineering, Chania, Crete, Greece, Dec. 2011. http://www.intelligence.tuc.gr/publications.php?pub_author=All-&pub_type=10&pub_subject=All