SOWL: A Framework for Handling Spatio-Temporal Information in OWL

Batsakis Sotirios

Department of Electronic and Computer Engineering

Technical University of Crete (TUC)

A thesis submitted in partial fulfilment of the requirements for the degree of

Doctor of Philosophy

To my family

Acknowledgements

Writing a doctoral thesis needs assistance and supervision. Sincere thanks to my supervisor, Assoc. Prof. Euripides G.M. Petrakis for his unending support throughout the years of both my MSc and my PhD. His enthusiasm and desire for perfection was an inspiration for me.

I would also like to thank the members of my doctoral committee, Prof. Stavros Christodoulakis, Prof. Minos Garofalakis, Prof. Manolis Koubarakis, Prof. Dimitris Plexousakis, Prof. Grigoris Antoniou, and Prof. Timoleon Sellis for their comments and suggestions on improvements and extensions of this work.

In the last few years, I was a member of the Intelligent Systems Research Group. Many thanks to all my colleagues in the group for their continuous support and to the technical staff for providing a professional research infrastructure.

Throughout this thesis i received financial support from the Technical University of Crete (TUC) and from the Greek Ministry of Education. I would like to thank all the people in these institutions for these research grants.

Special thanks go to my family for their love and encouragement. Their support in difficulties and during long working hours was an extra motivation for this effort.

Abstract

The Semantic Web extends existing World Wide Web (WWW) with formal machine readable semantics that enable better understanding of Web content by both, people and machines. The OWL ontology language forms the basis of Semantic Web technologies by providing the means for defining formal semantics of concepts and their properties. Semantic Web standards also provide with the formalism and mechanisms for supporting reasoning and querying of information in OWL.

Representing dynamic information for concepts evolving in time and space is an important issue to deal with. This calls for mechanisms for representing dynamic properties of objects. In addition, these mechanisms must offer support for qualitative information (i.e., information represented using natural language expressions such as "before" for time or "north" for space) which are common in natural language expressions in many applications.

The current thesis addresses all these issues: representation of dynamic concepts is achieved using the "4D-fluents" or, alternatively, the "N-ary relations" mechanisms; Both mechanisms are thoroughly explored and are expanded for representing qualitative (in addition to quantitative) spatio-temporal information in OWL. Temporal representations based both, on intervals and on time instants (points) are supported and compared. Our approach models spatial content in description logics using the eight relations of the Region Connection Calculus (RCC8) complemented by directional relations encoded by cone-shaped or, alternatively, projection-based relations.

Qualitative spatial and temporal information is supported using sets of SWRL rules and OWL axioms offering a sound, complete and tractable reasoning procedure based on *path consistency* applying on the supported sets of relations. Polynomial time complexity of spatio-temporal reasoning is achieved by restricting the supported sets of relations to "tractable" sets. Furthermore, rules enforcing cardinality constrains on dynamic properties are defined. Reasoning support for qualitative information and restriction checking is also introduced in this work for increasing expressive power of existing approaches. Building-upon existing Semantic Web standards and on the idea of integrating reasoning support into the proposed representation are important advantages of the proposed approach.

Query support in SOWL is realized with two languages referred to as TOQL and SOWL Query Language. Both, are high-level query languages, independent from the underlying SOWL representation so that, the user need not be familiar with the peculiarities of the ontological spatio-temporal representation in SOWL (i.e., the 4D-fluents or the N-ary approach). TOQL handles ontologies almost like relational databases and relies on the idea of extending SQL with spatio-temporal operators. Respectively, SOWL builds-upon SPARQL, the current standard of the Semantic Web, which is also extended with a set of spatio-temporal operators similar to those introduced in TOQL.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The rapid growth of the World Wide Web (WWW) in recent years has generated the need for intelligent tools and mechanisms, which automatically handle tasks that are commonly handled manually by users. For example, planning a trip requires selecting and purchasing tickets and hotel reservations at specific dates and prices, according to user needs. Buying a product requires careful selection among different products that satisfy user needs, at the best available price. All these tasks are handled by end user by searching the Web (using a search engine). In recent years, there is an increasing need for Web services that accomplish these tasks automatically without requiring user intervention, besides task description. These services must be capable of understanding the meaning of the content of Web pages and reason over their content in a way similar to humans. Semantic Web is a solution to this need by introducing formal, machine readable semantics for representation of knowledge, combined with reasoning and querying support. These form the basis of the Semantic Web initiative.

Formal definitions of concepts and of their properties form ontologies, which are defined using the OWL language. OWL ontologies offer the means for representing high level concepts, their properties and their interrelationships. For example medical ontologies represent the anatomic features of the human body, diseases, their symptoms and corresponding medical exams and treatments. Ontologies contain definitions of concepts and their properties by means of binary relations between concepts or a concept and a numerical (concrete) domain. Query languages such as SPARQL, are typically applied for searching for information in

ontologies satisfying certain user criteria.

The syntactic restriction of OWL to binary relations complicates the representation of n-ary (e.g., ternary) relations. For example, an employment relation at a specific temporal interval that involves an employee, an employer and a temporal interval, is in fact a ternary relation. In general, properties of objects that change in time (dynamic properties) are not binary relations, since they involve a temporal interval in addition to the object and the subject. Representing information evolving in time is the problem this work is dealing with.

1.1 Problem Definition

Dynamic ontologies are not only suitable for describing static scenes with static objects (e.g., objects in photographs) but also to enable representation of events with objects and properties that change over time and space (e.g., moving objects in a video). Handling both static and dynamic information in the Semantic Web is an important problem to deal with. Representation of dynamic features calls for mechanisms that allow uniform representation of the notions of time (and of properties varying in time) within a single ontology. Existing methods for achieving this include, among others, temporal description logics [3], concrete domains [69], property labelling [44], versioning [58], named graphs [109], reification[1] and the 4D-fluents (perdurantist) approach [114].

Representing dynamic information in the Semantic Web is a complicated issue to deal with and is not handled in full by any of the aforementioned approaches. The syntactic restriction of OWL to binary relations complicates the representation of temporal properties since a property holding for a specific time instant or interval is a relation involving three objects (an object, a subject and a time instant or interval). Complying with existing Semantic Web standards and specifications (e.g., OWL 2.0, SWRL) is as basic design decision in SOWL.

Reasoning over qualitative spatial and temporal information (i.e., information defined using natural language expressions such as "before" or "left") is also a a requirement. For example, the description of a university campus using natural language involves expressions such as "north of", "into" instead of spatial

[1] http://www.w3.org/TR/swbp-n-aryRelations/

coordinates. Representing these expressions as well as inferring facts using their semantics are also supported by SOWL.

1.2 Proposed Solution

We introduce SOWL, an approach for handling spatio-temporal information in OWL. Representation of spatio-temporal information in SOWL extends those of previous approaches [114]. Welty and Fikes [114] showed how quantitative temporal information (i.e., in the form of temporal intervals whose start and end points are defined) and the evolution of concepts in time can be represented effectively in OWL using the so called "4D-fluents approach". Concepts varying in time are represented as 4D dimensional objects, with the 4th dimension being the time.

In this thesis this approach was extended in certain ways [14]: (a) The 4D-fluents mechanism was 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. SWRL and OWL 2.0 constructs (e.g., disjoint properties) are combined, offering a sound and complete reasoning procedure ensuring path consistency [111].

Relying on an interval-based representation, as in [114], didn't allow for reasoning over time instants similarly to intervals, thus an instant (or point) based representation is presented as well. The proposed approach also handles information, both quantitative and qualitative, using tractable sets of relations on which path consistency applies [14].

Quantitative and qualitative spatial relations are also represented in SOWL. Specifically, RCC8 topological relations, cone and projection based directional relations and quantitative defined distance relations are supported. The spatial representation is combined with the proposed temporal representation based on the extended 4D-fluents approach proposed in this work, offering a unified spatio-temporal representation and reasoning mechanism.

Apart from 4D-fluents, a representation of both forms of spatio-temporal information (i.e., quantitative, qualitative) based on N-ary relations [82] is also

proposed. The two representations (i.e., 4D-fluents, N-ary relations) are practically equivalent with 4D-fluents being more flexible for the representation of symmetric, inverse and transitive relations using fewer additional relations, while the N-ary approach requires fewer additional objects. Nevertheless, using either representation is solely a matter of user preference.

Reasoning in SOWL is implemented in SWRL and is capable of inferring spatial and temporal relations and detecting inconsistent assertions. The mechanism offers (besides soundness, completeness and tractability) compliance with existing W3C specifications, standards and tools. It is an integral part of the ontology and is handled by standard tools such as Pellet.

SOWL addresses issues concerning OWL property semantics (e.g., functional, inverse, transitive relations) and cardinality constrains and addresses issues concerning their applicability in conjunction with a temporal representation mechanism. Checking for restrictions holding on time dependent (fluent) properties requires particular attention. If a fluent property holds between two objects (classes), then, these objects are only indirectly associated through one or more artificial objects. A fluent property is declared between the artificial object and an actual object (N-ary) or between two artificial objects (4D-fluents) as presented in Chapter 2. Checking for property restrictions would require adjusting the domain and range of this property from the artificial to the actual objects. This in turn requires extra rules. To the best of our knowledge this is the first work addressing these issues.

Query support in SOWL is realized with two languages referred to as TOQL and SOWL Query Language. Both are high-level query languages, independent from the underlying SOWL representation so that, the user need not be familiar with the peculiarities of the ontological spatio-temporal representation in SOWL (i.e., the 4D-fluents or the N-ary approach). TOQL handles ontologies almost like relational databases and relies on the idea of extending SQL with spatio-temporal operators. Respectively, SOWL builds-upon SPARQL, the current standard of the Semantic Web and which is also extended with a set of spatio-temporal operators similar to those introduced in TOQL. Both query languages support a set of spatial and temporal operators that apply on the proposed representations.

1.3 Contributions of the Present Work

SOWL deals with the problems of representation, reasoning and querying of spatial and temporal information in the Semantic Web. Basic design decisions in this work are (a) full compliance with existed Semantic Web standards and specifications (b) support of both quantitative and qualitative information (c) integrated reasoning supporting inference of implied relations and inconsistency detection which is sound, complete and tractable (d) handling issues concerning applicability of OWL property semantics over properties evolving in time (fluent properties) and (e) querying support over both qualitative and quantitative spatio-temporal information.

Supporting both temporal and spatial relations, quantitative and qualitative defined in conjunction to the 4D-fluents mechanism (but not restricted to this since the N-ary relations mechanism is also proposed) is the first requirement of SOWL. A reasoning mechanism ensuring path consistency and preservation of property semantics while remaining full compliant with existing W3C standards and specifications are additional requirements that, to the best of our knowledge, are not handled by other approaches besides the proposed one.

Earlier approaches such as TOWL [38] rely on concrete domains and require extending OWL with temporal constructs. This approach is also valid and worthfull. It is fact, competitive to SOWL and requires further consideration. Notice though that, TOWL handles only temporal information, it is neither W3C standard nor it can be supported by existing ontology editors (e.g., Protege), programming environments (e.g., Jena [73], OWL-API [20]), reasoning tools (e.g., Pellet [101]) and query languages (e.g., SPARQL). In contrast to TOWL, SOWL extends the existing W3C ontology framework of handling ontologies to handle spatio-temporal information while being consistent with existing standards and tools such as those referred to above. The contributions of SOWL are summarized below:

- Supports representation of dynamic information using both the 4D-fluents and N-ary relations. Both approaches are evaluated.

- Support representation of both temporal and spatial relations. Temporal re-

lations between points and intervals, and representations based on both, are presented and compared. Spatial information is expressed using topological RCC8 relations or alternatively cone-shaped, or projection-based directional relations or combinations of the above (i.e., RCC8 and cone-shaped directional relations or RCC8 and projection-based directional relations).

- Supports qualitative expressions (i.e., information defined using natural language terms such as "before") of spatio-temporal relations. This is an issue not addressed by existing approaches for the Semantic Web. Many applications (such as scene descriptions in natural language) are based on this kind of information.

- Reasoning over spatio-temporal relations is a major issue. Supporting qualitative relations calls for a corresponding reasoning mechanism. Reasoning over quantitative relations is straightforward (i.e., a set of consistent relations can be computed from the provided numerical values). The case of qualitative information is much more involved. Qualitative reasoning deals with inference of implied facts and relations derived from asserted relations and detecting inconsistencies of these assertions. Reasoning is realized using path consistency on tractable sets of spatial and temporal relations.

- Existing approaches for spatio-temporal representation over the semantic web do not deal with preservation of property semantics and constrains, when dealing with properties and objects evolving in time. This work deals with these issues by means of additional rules.

- An extension to the SPARQL query language with temporal and spatial operators for effectively querying over spatio-temporal information in OWL ontologies is proposed. The query language is independent of the 4D-fluent and N-ary representations introduced into this work. A second query language, the TOQL query language, is a high-level SQL-like language that handles ontologies almost like relational databases. It maintains the basic structure of an SQL language (SELECT-FROM-WHERE) enhanced by a set of temporal and spatial operators. The SOWL ontology model is not part of the query language and it is not visible to the user, so the user need

not be familiar with peculiarities of the underlying mechanism for time and space representation. TOQL and SOWL Query Language support queries for qualitative spatio-temporal information rather than quantitative only, as it is common to existing work such as t-SPARQL [109] and T-SPARQL [39] which, nevertheless, developed in parallel to our work.

To the best of our knowledge, this is the first work dealing with both qualitative and quantitative temporal and spatial information in ontologies. At the same time, SOWL handles all issues concerning reasoning, preservation of property semantics and restrictions, and querying of data within an integrated mechanism, fully compliant with existing Semantic Web standards, specifications and tools.

1.4 Thesis Outline

Related work in the field of knowledge representation is discussed in Chapter 2. This includes issues related to Semantic Web such as representing information of time and space in ontologies, as well as reasoning using this information. Earlier work related to representation, reasoning and querying over temporal and spatial information are presented and discussed. The proposed SOWL ontology model for spatio-temporal information is presented in Chapter 3. The corresponding reasoning mechanism is presented in Chapter 4. Chapter 5 presents the SOWL Query Language followed by evaluation in Chapter 6 and conclusions and issues for future work in Chapter 7 respectively.

Chapter 2

Background and Related Work

2.1 Introduction

Semantic Web standards such as ontologies, ontology construction languages such as OWL, reasoning and rules (SWRL) are discussed in Sec. 2.2. Related work on the field of temporal and spatial representation is presented in Sec.2.3 followed by related work on reasoning presented in Sec.2.4. Previous work on temporal representation specialized for the Semantic Web is presented in Sec.2.5 while corresponding work concerning spatial and spatio-temporal information for the Semantic Web is discussed in Sec. 2.6. Besides representation and reasoning, querying support for the proposed model is provided by means of spatio-temporal query languages. Related work on spatio-temporal query languages and query languages for the Semantic Web is presented in Sec. 2.7.

2.2 Semantic Web

The rapid growth of available information in the World Wide Web (WWW) has complicated the task of information retrieval and processing over the Web. Search engines such as Google[1], Bing[2] and Ask[3] improve the task of information retrieval as much as possible, however in most cases, the users still have to browse through

[1] http://www.google.com
[2] http://www.bing.com
[3] http://www.ask.com

the returned Web pages in order to fulfil tasks such as on-line shopping, travel planning or ticket reservations. Advanced search mechanisms may also be implemented based-upon learning or focused crawlers [17; 27] but still are incapable of fully automating tasks such as those referred to above. Automating these tasks require that machine understandable semantics become available along with data readable by humans in HTML pages that are currently available. The requirement of machine interpreted Semantics is the core idea of the Semantic Web vision [21].

Introducing machine readable Semantics calls for a formal language for conceptualization of application domains, the related concepts, their properties and their relationships. Based-upon existing work on knowledge representation, logic and ontologies along with more recent approaches such as frames [76], Description Logics [4] form the basis for Semantic Web standards. Specifically the OWL representation language [75] and the SWRL rule language [51] are the description and rule languages respectively of the Semantic Web and are presented in following.

2.2.1 Description Logics

Description Logics (DLs) are a family of Knowledge Representation languages that form the basis for the Semantic Web standards. They are related with existing formalisms such as propositional and first order logic [35] and frames [76]. Compared to frames, they adopt formal semantics and they are more expressive than propositional logic but less expressive (in order to increase performance) compared to first order logic. A survey of Description Logics is presented in [5].

The basic components of a Description Logic formalism are the *Concepts* or *classes*, their *properties* or *roles* and the *individuals* or *objects*. The expressiveness of a description logic formalism is defined by the allowable constructs and expressions. The trade-off between expressive power and efficiency is the basic design decision in every DL. Every DL supports a different set of allowable expressions for defining concepts. These definitions are expressed as combinations of existing definitions and atomic (simple) concepts and of their properties. The formal, logic-based semantics of Description Logics allow for unary predicates

(concepts) and binary predicates (roles or properties).

Besides representation of concepts and of their properties, DL formal and unambiguous semantics allow for inference of implied facts from asserted knowledge. The expressive power of DLs is complemented by inference procedures dealing with *subsumption* (i.e., determining subclass-superclass relations), *consistency* (i.e., determining contradictions in concept definitions and individual assertions) and *instance* (i.e., determining the class(es) that an individual belongs to). Decidability of inference is a highly desirable characteristic so that, in practice, expressiveness is often sacrificed (i.e., restricted) in order to guarantee decidability. The OWL language is based on DL and it is the basic component of the Semantic Web initiative.

A Description Logic, or language, is fully characterized by the allowable constructs that are used for the definitions of concepts and properties, as expressions of basic (atomic) concepts and properties. The set of such definitions for an application domain forms the Terminological Box (TBOX) of an ontology. Assertions involving concepts and properties of individuals form the Assertional Box (ABox) of the ontology. Reasoning is applied on both, TBox definitions and ABox assertions.

The basic description language is the ALC language which allows concept negation, intersection and union. Specifically, concepts forming a set, abbreviated as N_c, and properties forming the set N_r can be defined. Atomic concepts and properties are part of N_c and N_r respectively. The negation of a concept C (atomic concept or description) abbreviated as $\neg C$, the intersection and union of concepts C and D (atomic concepts or descriptions), abbreviated as $C \sqcap D$ and $C \sqcup D$ respectively are also defined. Finally, if C is a concept then $\exists r.C$ (i.e., objects which relate with property r with at least one individual belonging to class C) and $\forall r.C$ (i.e., class of individuals that property r relates them only with individuals of class C), are also allowable ALC concept definitions. They are called *existential* and *value restrictions* respectively.

Formally given a set Δ^I then *interpretation* I $(\Delta^I, .^I)$ is a pair Δ^I and function $.^I$ that assigns to every concept A^I the set $A^I \subseteq \Delta^I$ and to every property r the set $r^I \subseteq \Delta^I \times \Delta^I$. In ALC the following also hold:

- $(C \sqcap D)^I = C^I \cap D^I$

- $(C \sqcup D)^I = C^I \cup D^I$
- $(\neg C)^I = \Delta^I \diagdown C^I$
- $(\exists r.C)^I = \{x \epsilon \Delta^I \text{ and exists } a, y \epsilon \Delta^I \text{ and } (x,y) \epsilon r^I, y \epsilon C^I\}$
- $(\forall r.C)^I = \{x \epsilon \Delta^I \text{ and } \forall y \epsilon \Delta^I, (x,y) \epsilon r^I \Rightarrow y \epsilon C^I\}$

Besides concept *conjunction, disjunction* and *negation* ALC description language includes the *top concept* $\top$ which is an abbreviation of the whole domain and $\bot$ *bottom concept* which represents the empty set. ABox assertions can be the following:

- $a \epsilon\ C$ where a is an individual and C a concept
- $(a,b) \epsilon\ r$ where a, b are individuals and r a property

An important part of a Description Logic expressiveness are *General Inclusion Axioms* that specify that a concept C is subsumed by another concept D (i.e., all individuals of C also belong to D. Formally:

- $C \sqsubseteq D \Longleftrightarrow C^I \subseteq D^I$

Equivalence axioms $C \equiv D$ correspond to axioms: $C \sqsubseteq D$ and $D \subseteq C$. Additional constructs not in ALC but parts of more expressive description logics are *qualified number restrictions*: they can be either $(\geq nr.C)$ (*at least* restriction) or $(\leq nr.C)$ (*at most* restriction) implying that individuals of a concept can be related with property r with at least (or at most) n individuals of concept C with property r:

- $(\geq nr.C)^I = \{d \epsilon \Delta^I \text{ and } cardinality(\{e|(d,e) \epsilon r^I \wedge e \epsilon C^I\}) \geq n\}$
- $(\leq nr.C)^I = \{d \epsilon \Delta^I \text{ and } cardinality(\{e|(d,e) \epsilon r^I \wedge e \epsilon C^I\}) \leq n\}$

The *unqualified number restrictions* are defined as the qualified ones but concept C in the corresponding definition is replaced by the top concept $\top$. The *exact cardinality restrictions* are defined as the result of a combination of an *at least* n and an *at most* n restriction. Exact value restrictions force a specific value for a property of individuals of a concept and *nominal* concepts whose individuals can

be one of a restricted list of specific individuals. *Inverse properties* r, r^- are also defined so that when r holds between two individuals r^- also holds but with the object and subject of the property interchanged:

- $(r^-)^I = \{(b,a)|(a,r)\epsilon r^I\}$

If a property r is the inverse of itself then r is *symmetric*. A $functional$ property represents *at most one* restriction over the property and $inversefunctional$ a property that it's inverse is functional. A property r is *transitive* when:

- $(r(a,b) \wedge r(b,c) \Rightarrow r(a,c)$

General Role Inclusion Axioms are an important part of the expressiveness of a description logic, given a set of relations $r_1, r_2, ..., r_n, r$ then:

- $r_1 \circ r_2 ... \circ r_n \sqsubseteq r \Leftrightarrow r_1(a_1,a_2) \wedge r_2(a_2,a_3) ... r_n(a_n,a_{n+1}) \Rightarrow r(a_1,a_{n+1})$

where $\circ$ denotes composition. Limited forms of General Role Inclusion Axioms are *subproperty axioms* (i.e., property r_1 is subproperty of property r_2 when $r_1(a,b) \Rightarrow r_2(a,b)$). Also *reflexive* properties (i.e., r is reflexive when $r(a,b) \Rightarrow r(a,a)$), $irreflexive$ properties (i.e., r is irreflexive when $\forall a : r(a,a) \sqsubseteq \neg r^I$) and $disjoint$ properties (i.e., properties r_1, r_2 are disjoint when $r_1^I \sqsubset \neg r_2^I$ and $r_2^I \sqsubset \neg r_i^I$) are also specific forms of role inclusion axioms. A description logic may also offer support of specific *datatypes* such as integer numbers and strings. The notation characterizing the expressiveness of a description logic can be summarized as follows:

- F: Functional properties
- ε:qualified existential restrictions
- U:Concept union
- C:Negation (including non atomic concepts)
- S: ALC with transitive properties
- H: Subproperty axioms

- R: reflexive, irreflexive and disjoint properties
- O: nominals
- I:inverse properties
- N: Unqualified cardinality restrictions
- Q: Qualified cardinality restrictions
- (D):datatypes
- AL: concept negation, intersection, value restrictions and limited (unqualified) existential restrictions.
- EL:intersection and full existential restrictions
- FL_o:FL^- without existential qualification

The expressiveness of a description language complicates reasoning tasks, so that Description Logic expressiveness and efficiency of reasoning tasks are always traded-off. Also, the following reductions apply to reasoning tasks, and have polynomial time complexity [5] :

- Subsumption is reduced to equivalence
- Equivalence can be reduced to subsumption
- Subsumption can be reduced to satisfiability
- Satisfiability can be reduced to subsumption
- Satisfiability can be reduced to consistency
- Instance problem can be reduced to consistency
- Consistency can be reduced to the instance problem

For terminological (TBox) reasoning implementing an algorithm for satisfiability is sufficient for all reasoning tasks (i.e., satisfiability, equivalence and subsumption). Accordingly, implementing a consistency algorithm is sufficient for implementing assertional (ABox) reasoning (i.e., consistency and instance). Description Logics are a fragment of First Order Logic and resolution based approaches (i.e., a reasoning method for first order logic) where initially employed for the required reasoning tasks. Recently, a shift towards the so called "tableaux" based reasoning is observed [5]. Popular reasoners such as FaCT++[1], Pellet[2], Hermit[3] and RACER[4] are examples of tableaux based reasoners.

Description Logics do not adopt the *Unique Name Assumption* and the *Closed World Assumption* as the Entity Relationship Model [30] does. Two individuals with different names are not considered distinct unless this is asserted or proved by the asserted axioms. DLs adopt the so called *Open World Assumption* implying that asserted knowledge is not considered to be complete, thus failing to prove a fact doesn't mean that the fact doesn't hold true. Only proven facts can be asserted into the knowledge base and lack of a proof doesn't imply proof of the negation of the fact.

2.2.2 OWL

The objective of Semantic Web standards is to offer means for formal machine understandable semantics of application domains. This formal conceptualization, or ontologies is based on the OWL language introduced initially in [52]. Building-upon DAML [2] and OIL [36] OWL was compatible with the existing RDF [67] specification for describing concepts and properties of objects, while offering increased expressiveness over the RDFS [74] vocabulary description language for RDF.

RDF and RDFS represent properties or relations between entities in the form of triplets of the form *object-predicate-subject* (e.g., IBM employs John). Specific individuals can belong to classes (e.g., John is-a Person, where *John* is an indi-

[1] http://owl.man.ac.uk/factplusplus/
[2] http://clarkparsia.com/pellet/
[3] http://www.hermit-reasoner.com/
[4] http://www.sts.tu-harburg.de/ r.f.moeller/racer/

vidual and *Person* is a class). Properties such as *employs* can relate individuals of specific classes, for example we can specify that the object of the property *employs* belongs to class *Company* and the subject belongs to the class *Employee*. Classes of the object and the subject of a property are abbreviated as *domain* ans *range* respectively. Basic taxonomic relations between classes and properties can be also specified, for example is can be stated that *Employee* is a *subclass* of *Person*, (i.e., every employee is also a person). OWL extends RDF/RDFS expressiveness and it will be described in detail in the following. Three variants of OWL where introduced in the OWL specification [19]:

- OWL-Full which is fully compliant with RDF,
- OWL-DL which is based on Description Logics and,
- OWL-Lite is subset of OWL-DL, it is less expressive than OWL-DL allowing for the definition of class hierarchies and simple constraint features.

OWL-full is the most expressive variant but doesn't retain decidability and it is not supported by existing OWL reasoners. OWL-DL is the most routinely used version of OWL and it is based in the $SHOIN(D)$ description logic [52]. Thus, OWL DLs available constructs are those of $SHOIN(D)$: Concept negation, union, intersection, value and existential restrictions and transitive properties (S), subproperties (H), nominals (O), inverse properties (I), unqualified cardinality restrictions (N) and datatypes. Reasoning over $SHOIN(D)$ is non deterministic exponential in time (NExpTIME) although, in practice, optimised tableaux based reasoners [5] offer tractable average case running times. OWL-Lite is based on $SHIF(D)$ description logic [75]. OWL-Lite ($SHIF(D)$) supports concept negation, union, intersection, value and existential restrictions and transitive properties (S), subproperties (H), and inverse properties (I), it doesn't support nominals and unqualified cardinality restrictions as OWL-DL does, but it supports functional properties (F), a limited form of cardinality restrictions. Reasoning over OWL-Lite is deterministic exponential in time (EXPTIME) although, average case complexity, as in the case of OWL-DL, is much lower in practice [5].

The evolution of the OWL specification was based on the observation that additional constructs can be added in OWL-DL without compromising decidability, while increasing expressiveness. Extending OWL-DL with the additional constructs leaded to the adoption of OWL 2 [40] as the current Semantic Web standard [80]. OWL 2 is based on $SROIQ(D)$ description logic [50] (i.e., $SROIQ(D)$ offers all constructs of $SHOIN(D)$ with the addition of qualified number restrictions (Q) and complex role inclusion axioms (R)). The computational properties of *SHROIQ(D)-OWL 2.0* are analysed in [50], along with a description of the corresponding tableaux algorithm for reasoning over OWL 2. In addition to constructs offered by OWL-DL, OWL 2 offers support for qualified number restrictions (Q) in addition to the unqualified ones and complex role inclusion axioms (R) along with disjoint, symmetric, asymmetric, reflexive, irreflexive properties and property negation in addition to subproperties offered by OWL-DL. $SHROIQ(D)$ is decidable thus offering additional expressiveness, while retaining the computational properties of OWL-DL. SOWL aims at expressing dynamic concepts using current Semantic Web standards, such as OWL 2 and SWRL.

OWL 2 specification includes $profiles$[1], namely *OWL 2 EL,QL* and *RL*. *OWL 2 EL* supports class conjunction and existential restrictions while disallowing negation, disjunction, cardinality and value restrictions, in order to optimize classification tasks. *OWL 2 EL* offers tractable (i.e., polynomial time) reasoning performance. *OWL 2 QL* is offering optimized performance for conjunctive query answering by allowing class disjointness, domain, range of properties, existential restrictions, disallows disjunctions and value restrictions. *OWL 2 RL* is optimized for rule-based reasoning over individuals explicitly asserted into the ontology, thus disallowing constructs such as existential restrictions that introduce anonymous individuals.

OWL specification also includes the *abstract syntax* which is equivalent to the *Description Logic based syntax* presented in section 2.2.1. Specifically, class names (usually with a capital initial letter) represent concepts and keywords. For example, $owl : Thing$ and $owl : Nothing$ represent the top $\top$ and bottom $\bot$ concepts respectively. $intersectionOf(C_1, C_2...C_n)$ and $unionOf(C_1, C_2...C_n)$ represent the intersection and disjunction of classes $C_1, C_2...C_n$. The complement

[1] http://www.w3.org/TR/owl2-profiles/

of a class C is defined using keyword $complementOf(C)$ while an enumerated class defined as a set of individuals $o_1, o_2...o_n$ is defined using the declaration $oneOf(o_1...o_n)$.

Restrictions (expressed using the *restriction* keyword followed by the property P over which the restriction applies and the restriction keyword) can be one of the following: *someValuesFrom(C), allValuesFrom(C), hasValue(o), minCardinality(n)* and *maxCardinality(n)* representing qualified existential restrictions, value restrictions, exact value restriction, min and max cardinality restrictions respectively, where C is a class name, o an individual (or datatype value), and n an integer. An enumeration using the $oneOf$ keyword can be used instead of a class name C in the above definitions. TBox class definitions can be A *partial* $C_1..C_n$ (i.e., class A is a subclass of the conjunction of $C_1...C_n$), and A *complete*$(C_1...C_n)$ (i.e., A equals the intersection of $C_1...C_n$). Enumerated classes are defined using the *EnumeratedClass(A* $o_1...o_n$*)* keyword (where $o_1...o_n$ are individuals). *SubClassOf(*C_1 C_2*)* asserts that C_1 is a subclass of C_2, while the $EquivalentClasses(C_1...C_2)$ and $DisjointClasses(C_1...C_n)$ define class equivalence and disjointness for the list of class names.

Keyword $SubPropertyOf(p_1\ p_2)$ defines the subproperty relation between properties p_1 and p_2, while property equivalence is defined using *EquivalentProperties* keyword. Domains and ranges are defined using *domain* and *range* keywords with class definitions as arguments, while keywords *inverseOf, Symmetric, Asymmetric, Functional, InverseFunctional, Transitive, DisjointProperties, Reflexive* and *Irreflexive* apply on properties having the obvious interpretations. Finally, *SameIndividual* and *DifferentIndividuals* apply on lists of individuals specifying their equivalence or difference respectively. Both, abstract syntax and Description Logic based syntax will be used in this thesis.

2.2.3 SWRL

SWRL[1] is the language for specifying rules applying on Semantic Web ontologies. *Horn Clauses* (i.e., a disjunction of classes with at most one positive literal), can be expressed using SWRL, since Horn clauses can be written as implications (i.e.,

[1] http://www.w3.org/Submission/SWRL/

$\neg A \vee \neg B ... \vee C$ can be written as $A \wedge B \wedge ... \Rightarrow C$). The efficiency of reasoning over Horn clauses using forward chaining algorithms is a reason for choosing this form of rules. The antecedent (body) of the rule is a conjunction of clauses. Notice that, neither disjunction nor negation of clauses is supported in the body of rules. Also, the consequence (head) of a rule is one positive clause. Again, neither negation nor disjunction of clauses can appear as a consequence of a rule. Conjunction of clauses into the consequence part of the rule can be expressed indirectly by a set of rules with identical antecedents.

The clauses in the rule can be class names (e.g., C) with variables as arguments (e.g., $C(?x)$), property names p (in the form $p(?x, ?y)$ where x, y are variables) or the OWL $sameAs$, $differentThan$ keywords. Specific build-ins can be also be supported for numerical operators supporting specific datatypes. Whenever the antecedent of the rule holds for a given set of variable instantiations, the consequence is asserted into the knowledge base. To guarantee decidability, the rules are restricted to be *DL-safe rules* [79] that apply only on named individuals in the ontology ABox and not on implied anonymous individuals. Even with the syntactic restrictions imposed on SWRL rules, such as the lack of negation and disjunction and their applicability only to ABox reasoning, SWRL still extends OWL expressiveness while retaining decidability. For example, intersection of properties over named individuals can be expressed using SWRL (See Eq. 4.3) although, this is not part of OWL specification. Therefore, SWRL is an important tool for embedding rules into an ontology, while retaining decidability and OWL semantics. In this thesis, SWRL rules are presented using a first order notation in place of its equivalent SWRL notation.

2.3 Representation of Time and Space

Space and time are fundamentals aspects of the conceptualization of the physical world. The notions of time and space as well as the evolution of concepts and individuals into space and time are important issues in almost all application domains.

Time can be regarded as discrete or continuous, linear or cyclical, absolute or relative, qualitative or quantitative. Also, time can be presented using time

instances or intervals. Temporal concepts as used by humans in every day life are represented in the Semantic Web using (in many cases) the OWL-Time ontology [49]. OWL-Time provides definitions of time instants (or points), intervals, definitions for their relations and definitions of concepts such as days, weeks, months, years, dates, time zones, durations and measuring units. OWL-Time is an ontology of the concepts of time, but OWL-Time doesn't specify how these concepts can be used to represent evolving properties of objects (i.e., properties that change in time) and it doesn't specify how to reason over qualitative relations of temporal intervals and instants. This is a problem this work is dealing with. Nevertheless, since OWL-Time is a W3C recommendation (although other ontologies such as the SWRL Temporal Ontology[1] also exist) it is adopted by the current work for providing definitions of the concepts of time. In addition, this work will show how these concepts can be related to dynamic concepts evolving in time.

There is a fundamental philosophical controversy concerning the evolution of concepts in time, namely the perdurantist and the endurantist approaches [100] and this controversy also applies to the generalized spatio-temporal representation [42]. A discussion on these issues from the philosophical standpoint can be found in [48]. This controversy is related to issues such as the identity of objects as they evolve in time and whether objects endure in time although their properties may change, implying a fundamental distinction between objects and events or they perdure in time by relating properties in time and space in a form of a generalized event (4D perdurantist approach).

According to the perdurantist approach, every object is in fact a generalized event having specific spatial and temporal extensions. Even objects such as the sun and the solar system can be regarded as temporal entities having a specific duration, not different in their fundamental aspects from everyday events. In this work, both approaches are taken into account and modelling approaches based on 4D objects and 3D objects participating into events are presented. In this work, time is described using quantitative or qualitative terms using temporal instances and intervals. The employed representations are neutral to the discrete or continuous nature of time and they are based on absolute time specifying an

[1] http://swrl.stanford.edu/ontologies/built-ins/3.3/temporal.owl

ordering of time points.

Choosing between a point or an interval-based representation is an important issue [110]. Point-based representations assume linear ordering of time points with three possible relations the "$<$", "$>$", "$=$" often referred to as *before, after* and *equals* respectively. Based on this ordering relations, intervals can also be defined as ordered pairs of points s, e with $s < e$, often referred to as *start* and *end* of an interval respectively. Given two such ordered pairs of points (i.e., intervals) the following relations can be defined between the two intervals i_1, i_2 in terms of their endpoints s_1, e_1 and s_2, e_2 respectively:

$$
\begin{aligned}
&i_1\ before\ i_2 \equiv e_1 < s_2 \\
&i_1\ equals\ i_2 \equiv s_1 = s_2 \wedge e_1 = e_2 \\
&i_1\ overlaps\ i_2 \equiv s_1 < s_2 \wedge e_1 < e_2 \wedge e_1 > s_2 \\
&i_1\ meets\ i_2 \equiv e_1 = s_2 \\
&i_1\ during\ i_2 \equiv s_1 > s_2 \wedge e_1 < e_2 \\
&i_1\ starts\ i_2 \equiv s_1 = s_2 \wedge e_1 < e_2 \\
&i_1\ finishes\ i_2 \equiv s_1 > s_2 \wedge e_1 = e_2
\end{aligned}
$$

The relations *after, overlappedby, metby, contains, startedby and finishedby* are the inverse of *before, overlaps, meets, during, starts and finishes* and are defined accordingly (by interchanging s_1, s_2 and e_1, e_2 in their respective definitions). A temporal relation can be one of the 13 pairwise disjoint Allen's relations [1] of Fig. 2.1.

Using either a point or an interval-based representation, qualitative relations can be asserted, even when the specific time points or the temporal extends of intervals are unknown but their relative position is known; thus, the expressiveness of the representation increases. Quantitative representations involve specific datatypes such as $xsd : date$ supported by OWL, and these datatypes support comparison of dates, thus yielding the required ordering relation when specific dates of points are known.

Space is an important aspect of knowledge representation. Space can be regarded as two dimensional (2D) or three dimensional (3D) with most applications adopting a 2D space, which is the approach followed in this thesis. Cartesian co-

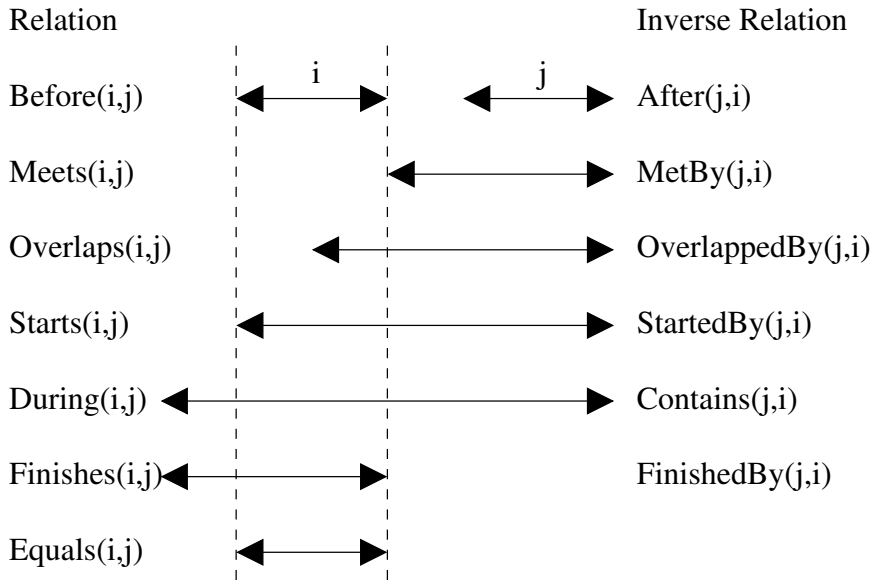

Figure 2.1: Allen's Temporal Relations

ordinates either 2D (x, y) or 3D (with z being the third axis) are employed; alternatively spherical coordinate systems can be employed.

Our approach is suitable for most applications, although there are limitations when a 2D projection is not adequate, for example in applications that handle earth in a global scale and not as a set of planes representing local areas. Each point is represented using a pair (or a triple) of coordinates of a specific datatype related to the coordinate system and the scale employed. Regions are represented using a set of points representing their contour or using minimum bounding rectangles (i.e., the minimal rectangle with sides parallel to the axis that contain the space of the region in question). Spatial information can be *topological, directional* and *distance* [92].

Topological relations represent the relative position of regions in the plane. The most widespread formalism for representing such relations is the so called Region Connection Calculus (RCC) Formalism introduced in [89]. A form of this calculus specifying 8 possible mutually exclusive relations between two regions, called RCC8 calculus is the most commonly used. The topological relations shown in Fig. 2.2, *(DC, EC, EQ, NTPP, NTPPi, TTP, TPPi, PO)*, referred to as RCC8 relations are supported in the SOWL model.

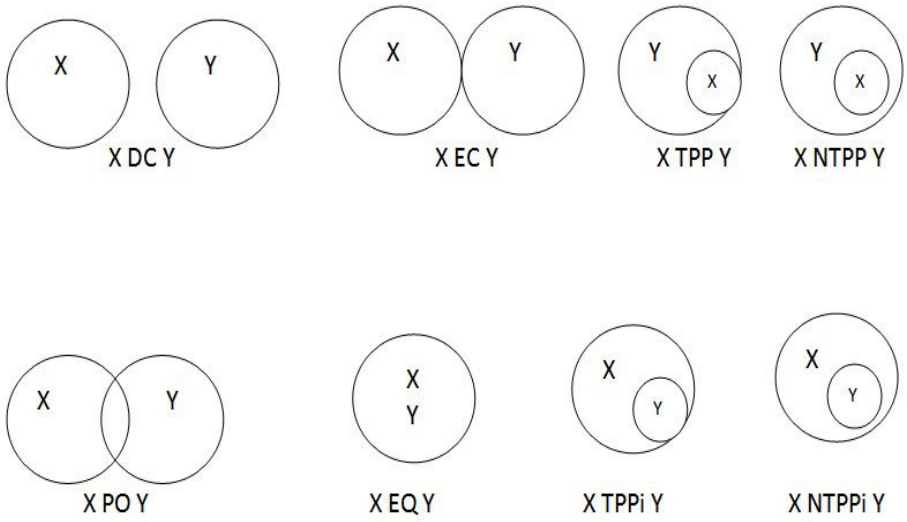

Figure 2.2: RCC8 topologic relations.

Directional relations are also defined based on cone-shaped areas [77]. As shown in Fig. 2.3, eight directional relations can be identified namely North (N), North East (NE), East (E), South East (SE), South (S), South West (SW), West (W) and North West (NW) following the cone-shaped regions approach of [77].

Alternative approaches based on 2D projections are discussed in [92; 102] and are also supported as well. The projection-based approach handles the projections of points (or regions) over the coordinate system axis, as shown in Fig. 2.4. In case of points, these projections are single point projections on two axes while, in case of regions these projections form a two-point projection (or interval) on each axis. These projections are in turn handled the same way as point and interval algebra handles time instants in the case of a temporal representation.

Finally, distance relations are defined and can be used in SOWL in conjunction with the above relation types. Notice that, qualitative distance relations (e.g., "far" and "near") may be ambiguous especially in applications where a common scale for measuring distances is not provided [92]. This problem is resolved when distance relations are expressed quantitatively (e.g., 3Km away from city A) and stored in the ontology as N-ary relations (i.e., by defining an object with attributes the two related locations and a numerical attribute representing their distance). In SOWL, we opt for the later (quantitative) approach for representing distance

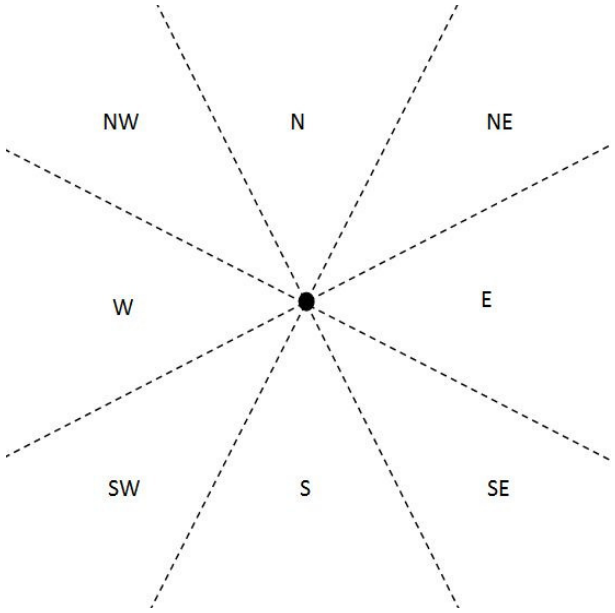

Figure 2.3: Cone-based direction relations.

information.

2.4 Temporal and Spatial Reasoning

Inferring implied relations and detecting inconsistencies are handled by a reasoning mechanism. In the case of a quantitative representation such a mechanism isn't required because spatial and temporal relations are extracted from the numerical representations in polynomial time (e.g., using datatype comparisons in the case of temporal relations and computational geometry algorithms in the case of spatial relations).

In cases where relations are qualitative, assertions of relations holding between spatial and temporal entities (e.g., intervals, points) restrict the possible assertions holding between other temporal and spatial entities in the knowledge base. For example, the assertion "point A is *north of* point B" impose a restriction on the arrangement of points is space. Also, it imposes a restriction on future assertions (e.g., a new assertion such as: "A is *south of* B" contradicts the existing one). Then, reasoning on qualitative spatial or temporal qualitative relations can

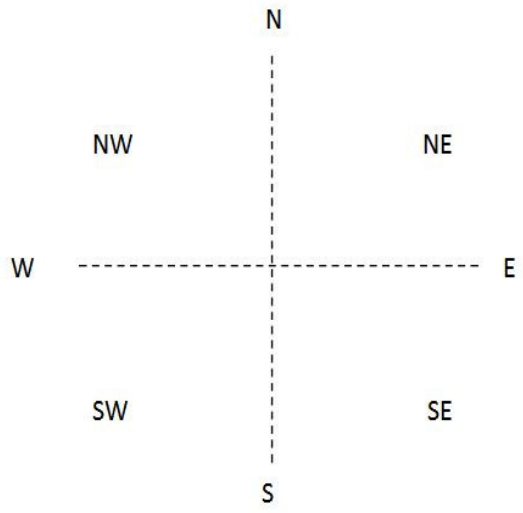

Figure 2.4: Projection-based direction relations.

be transformed into a *constrain satisfaction problem*, which is known to be an NP problem in the general case [92]. The worst case complexity appears in certain instances that are neither over-constrained nor under-constrained [92]. Notice that, a large number of constrains (i.e., assertion of relations) in comparison to the number of entities involved, usually leads to an inconsistency and a small number of constrains usually imply a small number of implied relations; both cases are are resolved in polynomial time complexity in practice [92].

Reasoning over qualitative spatial and temporal relations is achieved using [92]:

- An exponential worst case algorithm that has better performance on the average case.
- Approximation algorithms that are neither complete nor sound but they have polynomial worst case complexity.
- Polynomial time algorithms that are sound and complete by restricting the allowable relations to specific tractable sets.

Inferring implied relations depends on existing relations in the knowledge base and on their semantics. For example, directional relations may have different semantics if they are interpreted using the cone-shaped approach [77] or

the projection-based approach [7; 102]. Although, the relations in both cases are the same, their interpretations and semantics differ. Inferring implied relations is achieved by specifying the result of *compositions* of existing relations. Specifically, when a relation (or a set of possible relations) R_1 holds between entities A and B and a relation (or a set of relations) R_2 holds between entities B and C then, the *composition* of relations R_1, R_2 (denoted as $R_1 \circ R_2$) is the set (which may contain only one relation) R_3 of relations holding between A and C. Typically, compositions of pairs of relations are stored in *composition tables* [92].

Qualitative relations under the intended semantics may not apply simultaneously between a pair of individuals. For example, given time instants p_1 and p_2, p_1 can't be simultaneously $before$ and $after$ p_2. Typically, in spatio-temporal representations (e.g., using Allen and RCC8 relations) all basic relations (i.e., simple relations and not disjunctions of relations) are pairwise disjoint. When disjunctions of basic relations hold simultaneously then, their set intersection holds. For example, if p_1 is *before or equals* p_2 and simultaneously p_1 is *after or equals* p_2 then p_1 *equals* p_2. In case the intersection of two relations is empty these relations are disjoint. Checking for consistency means checking if asserted and implied relations are disjoint.

Reasoning over spatio-temporal relations is known to be an NP problem and identifying tractable cases of this problem has been in the center of many research efforts over the last few years [92]. The notion of $k-consistency$ is very important in this research. Given a set of n entities with relations asserted between them imposing certain restrictions, $k-consistency$ means that every subset of the n entities containing at most k entities doesn't contain an inconsistency. Obviously, when $n-cosistency$ holds then, there is not an inconsistency, but checking for all subsets of n entities for consistency is exponential on the number n. Simpler forms of consistency are $2-consistency$, (i.e., checking for asserted relations between all pairs of individuals for disjoint relations) and *3-way consistency* (i.e., checking all triples of individuals for inconsistencies caused by asserted relations and compositions of pairs of relations holding between the 3 entities).

There are cases where, $k-consistency$ for a specific value of k implies $n-consistency$ so that, a polynomial algorithm that enforces $k-consistency$ also solves the $n-consistency$ problem [92]. There also also cases where, although

$k-consistency$ doesn't imply $n-consistency$, there are specific sets of relations R_t (which are subsets of the set of all possible disjunctions of basic relations R), with the following property: if asserted relations are restricting to this set then $k-consistency$ implies $n-consistency$ and R_t is a *tractable set* of relations or a *tractable subset of* R [92].

Tractable subsets for point algebra have been identified in [110; 112; 113]. However, these results can be applied when point algebra is used for reasoning over projections of points in the 2D (or 3D) space in addition to the temporal case which involves only one dimension. In fact, point algebra is only applicable on point projections on each axis. Tractable sets of Allen interval algebra have been identified in [81] and [66]. These results apply in cases where Allen relations are used for spatial reasoning, since projections of 2D objects define 1D intervals on each axis, as in the case of temporal Allen relations [6].

Tractability of RCC8 subsets is analysed in [90] while, cone-shaped directional relations are examined in [91]. Combining points and intervals for temporal reasoning is analysed in [55] while, combined reasoning over intervals and their durations is discussed in [88]. Recent results for topological and temporal relations are presented in [22]. A survey is presented in [92].

2.5 Temporal Representation and Reasoning in the Semantic Web

The OWL-Time temporal ontology[1] 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. This is related to the problem of the representation of time in temporal (relational and object oriented) databases. Existing methods are relying mostly on temporal Entity Relation (ER) models [41] taking into account valid time (i.e., time interval during which a relation holds), transaction time (i.e., time at which a database entry is updated) or both. Also time is represented by time instants, intervals or finite sets

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

of intervals. However, representation of time in OWL differs because (a) OWL semantics are not equivalent to ER model semantics in Relational Databases (e.g., OWL adopts the *Open World Assumption* while DBs typically adopt the *Closed World Assumption*) and (b) relations in OWL syntax are restricted to binary ones in contrast to DBs. Representation of time in the Semantic Web can be achieved using *Temporal Description logics (TDLs)* [3], *Concrete domains* [69], *Reification*, *labeling of properties* [26; 44], *Versioning* [58], *named graphs* [109] and *4D-fluents* [114].

Temporal Description Logics (TDLs) [3; 70] extend standard description logics (DLs) that form the basis for semantic Web standards with additional constructs such as "always in the past", "sometime in the future". TDLs offer additional expressive capabilities over non temporal DLs and retain decidability (with an appropriate selection of allowable constructs) but they require extending OWL syntax and semantics with the additional temporal constructs (the same as property labelling introduced in [44]). Representing information concerning specific time points requires support for concrete domains, resulting to the proliferation of objects [3].

Concrete Domains [69] introduce datatypes and operators based on an underlying domain (such as decimal numbers). The concrete domains approach requires introducing additional datatypes and operators to OWL, while our work relies on existing OWL constructs. This is a basic design decision in our work. TOWL [38] is an approach combining 4D-fluents with concrete domains but didn't support qualitative relations, path consistency checking (as this work does) and is not compatible with existing OWL editing, querying and reasoning tools (e.g., Protege, Pellet, SPARQL).

Temporal RDF [44] proposes extending RDF by labelling properties with the time interval they hold. This approach also requires extending the syntax and semantics of the standard RDF. Note that Temporal-RDF cannot express incomplete information by means of qualitative relations. Although interval endpoints may be unspecified, since reasoning support over qualitative relations is not provided, Temporal-RDF does not provide the full expressiveness of the proposed approach. Temporal-RDF is combined with fuzzy logic in [106].

Temporal annotation of properties as proposed in [44] has been proposed

for OWL representation in [78], enhanced with support for undefined intervals. Querying support for annotated properties is provided as well [68].

Versioning [58] suggests that the ontology has different versions (one per instance of time). When a change takes place, a new version is created. Versioning suffers from several disadvantages: (a) changes even on single attributes require that a new version of the ontology be created leading to information redundancy (b) searching for events occurred at time instances or during time intervals requires exhaustive searches in multiple versions of the ontology, (c) it is not clear how the relation between evolving classes is represented. Furthermore, ontology languages such as OWL are based on binary relations (relations connecting two instances) with no time dimension regarding ontology versions.

Named Graphs [109] represent the temporal context of a property by inclusion of a triple representing the property in a named graph (i.e., a subgraph into the RDF graph of the ontology specified by a distinct name). The default (i.e., main) RDF graph contains definitions of interval start and end points for each named graph, so that a temporal property is represented by the start and end points corresponding to the temporal interval that the property holds. Named graphs are not part of the OWL specification[1] (i.e., there are not OWL constructs translated into named graphs) and they are not supported by OWL reasoners. In [109] a SPARQL based temporal query language is also introduced applying only on quantitative defined temporal intervals.

Reification is a general purpose technique for representing n-ary relations using a language such as OWL that permits only binary relations. Specifically, an n-ary relation is represented as a new object that has all the arguments of the n-ary relation as objects of properties. For example if the relation R holds between objects A and B at time t, this is expressed as *R(A,B,t)*. Furthermore, in OWL, using reification this is expressed as a new object with R, A, B and t being objects of properties. Fig. 2.5 illustrates the relation *WorksFor(Employee, Company, TimeInterval)* representing the fact that an employee works for a company during a time interval. Using reification, the extra class "ReifiedRelation" is created having all the attributes of the relation as objects of properties. Reification suffers mainly from two disadvantages: (a) a new object is created whenever a

[1] http://www.w3.org/TR/owl2-syntax/

temporal relation has to be represented (this problem is common to all approaches based on OWL) and (b) offers limited OWL reasoning capabilities [114] since relation R is represented as the object of a property, thus OWL semantics over properties (e.g., inverse properties) are no longer applicable (i.e., the properties of a relation are no longer associated directly with the relation itself). Examples of temporal representation based on reification (the reified temporal relations are named *Events* or *Actions*) are presented at [28; 98]. In [108] temporal representation is combined with application specific SWRL rules for representing clinical narratives.

Using an improved form of reification, the N-ary relations approach suggests representing an n-ary relation as two properties each related with a new object (rather than as the object of a property, as reification does). This approach requires only one additional object for every temporal relation, maintains property semantics but (compared to the 4D-fluents approach below) suffers from data redundancy in the case of inverse and symmetric properties (e.g., the inverse of a relation is added explicitly twice instead of once as in 4D-fluents). This is illustrated in Fig.2.7. In the case of transitive properties additional triples are introduced as well. Furthermore, domains and ranges of properties have to be adjusted taking into account the class of intermediate objects representing the relation (for example the $worksfor$ relation in no longer a relation having as object an individual of class *Company* and subject of class *Employee* as they are now related to the new object "TemporalEmployment").

Similarly to our proposed approach (see Chapter 4), property restrictions (e.g., cardinality constraints) cannot be expressed directly on properties and, subsequently, can't be identified by a reasoner as it is common in OWL ontologies. Instead, restriction checking on properties have to be implemented separately (on top of the ontology) with extra rules . Software in Java, instead of SWRL as in this work, handling a subset of restrictions over only quantitative intervals has been also developed in our laboratory [97].

A plug-in for the Protege editor supporting editing of N-ary based temporal ontologies is presented at [93]. A similar tool has been developed in our laboratory for the proposed representations, both for 4D-fluents [71] and N-ary relations [85].

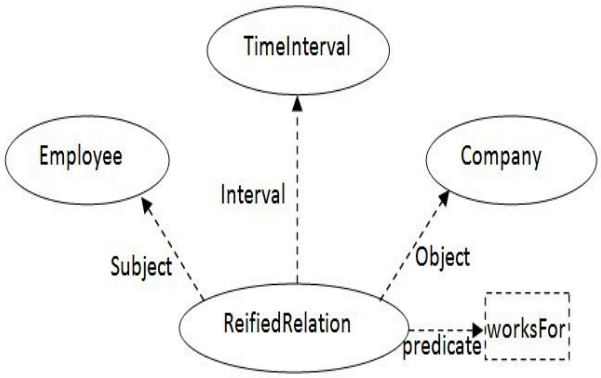

Figure 2.5: Example of Reification

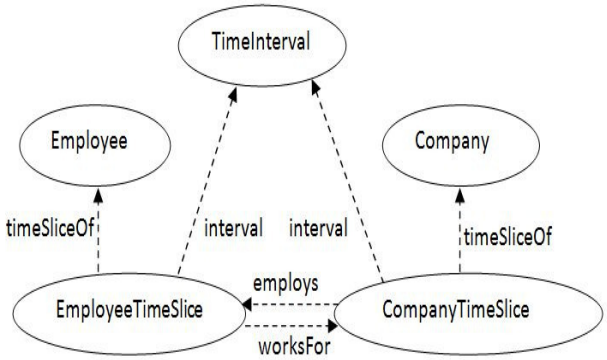

Figure 2.6: Example of 4D fluents

The *4D-fluent* (perdurantist) approach [114] shows how temporal information and the evolution of temporal concepts can be represented in OWL. Concepts in time are represented as 4-dimensional objects with the 4th dimension being the time (*timeslices*). Time instances and time intervals are represented as instances of a *TimeInterval* class, which in turn is related with concepts varying in time as shown in Fig.2.6. Changes occur on the properties of the temporal part of the ontology keeping the entities of the static part unchanged. The 4D-fluent approach still suffers from proliferation of objects since it introduces two additional objects for each temporal relation (instead of one in the case of reification and N-ary relations). The N-ary relations approach referred to above is considered to be an alternative to the 4D-fluents approach considered into this work. Examples of representations based on 4D-fluents are presented at [10; 15; 115].

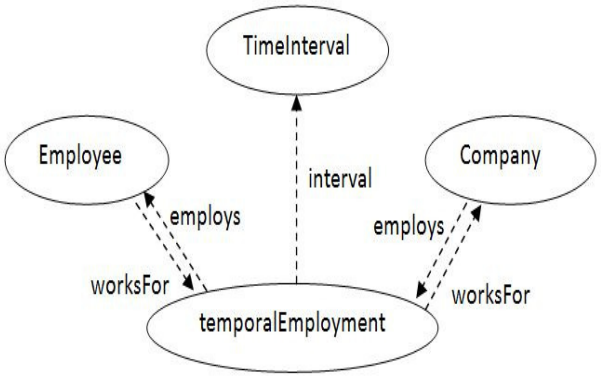

Figure 2.7: Example of N-ary Relations

The MUSING system where both a 4D-fluents based approach and an alternative approach based on extending RDF with temporal annotation [63; 64; 65] used in conjunction with OWL-Time, but without the qualitative reasoning support proposed in this thesis, is a related work.

2.6 Spatial Representation and Reasoning in the Semantic Web

Formal spatial, and spatio-temporal representations have been studied extensively in the Database [45] and recently, in the Semantic Web literature [25]. Spatial entities (e.g., objects, regions) in classic database systems are typically represented using points, lines (polygonal lines) or Minimum Bounding Rectangles *(MBRs)* enclosing objects or regions and their relationships [84]. Relations among spatial entities can be topological, orientation or distance relations. Furthermore, spatial relations are distinguished into qualitative (i.e., relations described using lexical terms such as "Into", "South" etc.) and quantitative (i.e., relations described using numerical values such as "10Km away", "45 degrees North" etc.). Accordingly, spatial ontologies are defined based upon a reference coordinate system in conjunction with a set of qualitative topological and direction relations (e.g., RCC8 relations). Reasoning rules for various relation sets have been proposed as well [34; 92].

Representing spatio-temporal knowledge has also motivated research within

the Semantic Web community. Katz et.al. [57] proposed representing RCC8 relations as OWL-DL class axioms (instead of object properties as in [43]) but this approach has limited scalability as shown in [105]. Chen et.al. [29] and Sotnykova et.al. [104] proposed an integrated spatio-temporal representation which includes qualitative relations but without specialized spatio-temporal reasoning support. Perry et.al. [99] proposed a representation based on quantitative spatio-temporal data and Christodoulakis et.al. [31] proposed a quantitative representation for data from GPS devices. A representation for quantitative spatio-temporal information based on linear constrains is presented in [61; 62].

Pellet Spatial [105] offers reasoning support for RCC8 topological relations. In SOWL [11; 13; 14; 16] support for topological and directional (quantitative and qualitative defined) spatial relations is provided in conjunction with the temporal representation mechanism. Applications of the SOWL model for dynamic medical information[1], video content and spatial descriptions using qualitative terms are presented at [72], [47] and [32] respectively.

2.7 Querying Spatio-Temporal Information in the Semantic Web

Examples of temporal query languages for temporal databases include TQuel [103], TSQL2 [59] and ATSQL [23]. Query languages for RDF and OWL ontological representations are of particular interest as they form the basis for developing the new type of temporal ontology query languages. SeRQL [24] and SPARQL [86] are good representatives of this category of query languages. SPARQL [86] is a W3C recommendation query language. SeRQL is a RDF/RDFS query language combining features of other (query) languages (e.g., RQL [56], RDQL [96], N-Triples, N3). Important features of SPARQL (and SeRQL) are: Graph transformation, RDF and XML Schema data type support, expressive path expression syntax and optional path matching. SPARQL (and SeRQL) supports comparison between date times.

Query languages for RDF and OWL ontological representations such as SPARQL

[1] Available at: http://www.intelligence.tuc.gr/HPV-4dcase/

[86] and SeRQL [24] form the basis for developing languages for querying spatio-temporal information in ontologies and the semantic Web. Querying spatio-temporal information over the semantic Web using languages such as SPARQL is a tedious task. Recent work on query languages for temporal ontologies include TOQL [9], t-SPARQL [109] and T-SPARQL [39] using 4D-fluents, named graphs and versioning respectively for the representation of temporal information.

A temporal query language supporting temporal annotation of ontologies is presented at [78]. In this work we extend TOQL [9] to handle spatial (in addition to temporal) and also qualitative spatial and temporal information. We also introduce the SOWL Query Language that extends SPARQL (notice that TOQL adopts an SQL-like syntax) with spatio-temporal operators supporting the SOWL model.

Chapter 3

The SOWL Model

3.1 Introduction

We propose SOWL, an ontology for representing and reasoning over spatio-temporal information in OWL. Building upon well established standards of the semantic web (OWL 2.0, SWRL) SOWL enables representation of static as well as of dynamic information based on the 4D-fluents [114] (or, equivalently, on the N-ary [82]) approach. Both RCC8 topological and cone-shaped directional relations are integrated in SOWL. Representing both qualitative temporal and spatial information (i.e., information whose temporal or spatial extents are unknown such as "left- of" for spatial and "before" for temporal relations) in addition to quantitative information (i.e., where temporal and spatial information is defined precisely) is a distinctive feature of SOWL. Both, the 4D-fuents and the N-ary relations approaches are expanded to accommodate this information. The SOWL reasoner implements path consistency [92], and is capable of inferring new relations and checking their consistency, while retaining soundness, completeness, and tractability over the supported sets of relations.

3.2 SOWL Model

Temporal representation in SOWL is based on the 4D-fluents approach enhanced with Allen relations which are defined as object properties between intervals.

Topological and directional spatial relations are represented as object properties defining the relation between the spatial extends of objects (which can be static or moving). In each case, the spatial representation is combined with the temporal representation with the location of objects being a static or fluent property respectively. An alternative implementation based on the N-ary approach has been implemented as well.

3.2.1 Temporal Representation using 4D-Fluents

Following the approach by Welty and Fikes [114], to add time dimension to an ontology, classes *TimeSlice* and *TimeInterval* with properties *tsTimeSliceOf* and *tsTimeInterval* are introduced. Class *TimeSlice* is the domain class for entities representing temporal parts (i.e., "time slices") and class *TimeInterval* is the domain class of time intervals. A time interval holds the temporal information of a time slice. Property *tsTimeSliceOf* connects an instance of class *TimeSlice* with an entity, and property *tsTimeInterval* connects an instance of class *TimeSlice* with an instance of class *TimeInterval*. Properties having a time dimension are called fluent properties and connect instances of class *TimeSlice*.

Fig. 3.1 illustrates a temporal ontology with classes *Company* (with datatype property *companyName*), *Product* (with datatype properties *price* and *productName*), and *Location* which represents spatial information (see Fig. 3.2 and Fig. 3.3). In this example, *CompanyName* is static property (it's value does not change in time), while properties *produces*, *productName*, *locatedAt* and *price* are dynamic (fluent) properties whose values may change in time. Because they are fluent properties, their domain (and range) is of class *TimeSlice*. *CompanyTimeSlice*, *LocationTimeslice* and *ProductTimeSlice* are instances of class *TimeSlice* and are provided to denote that the domain of properties *produces*, *locatedAt*, *productName* and *price* are time slices restricted to be slices of a specific class. For example, the domain of property *productName* is not class *TimeSlice* but it is restricted to instances that are time slices of class *Product*. All fluent properties are defined as *subproperties* of the property $fluent$.

In SOWL, the 4D-fluent representation is enhanced with qualitative temporal relations holding between time intervals whose starting and ending points are not

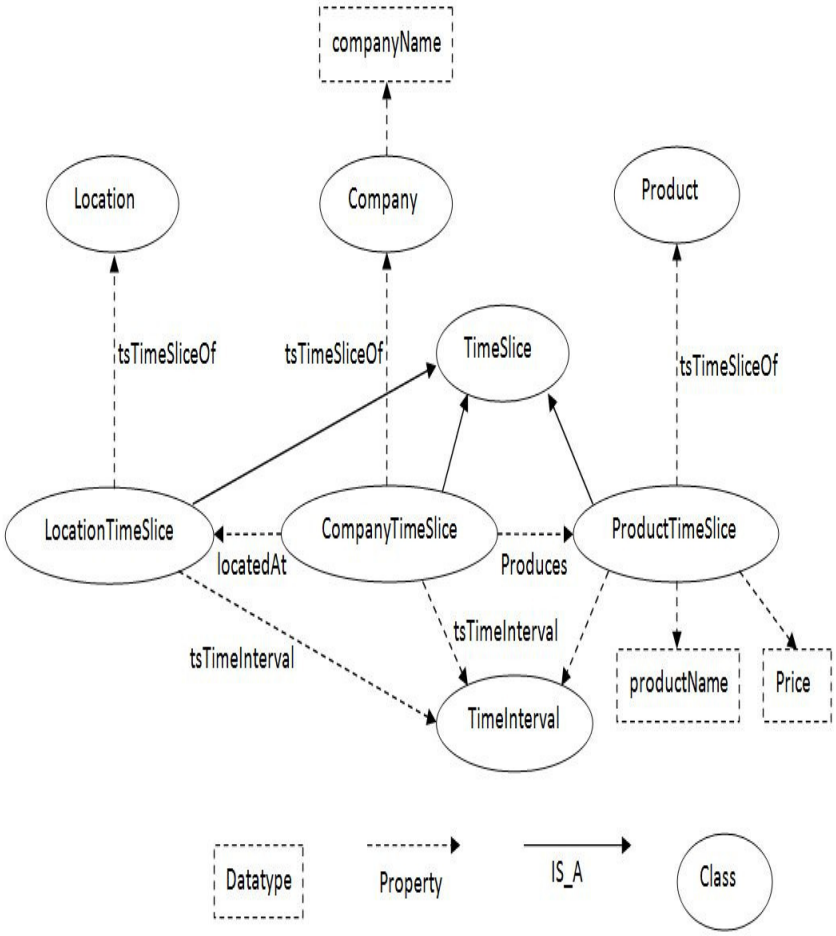

Figure 3.1: Dynamic Enterprise Ontology

specified. This is implemented by introducing temporal relationships as object relations between time intervals. This can be one of the 13 pairwise disjoint Allen's relations [1] of Fig. 2.1.

By allowing for qualitative relations the expressive power of the representation increases. Typically, the 4D-fluents model (similarly to other approaches such as Temporal RDF [44]), assume closed temporal intervals for the representation of temporal information, while semi-closed and open intervals can't be represented effectively in a formal way. In SOWL, this is handled by Allen relations: for example if interval t_1 is known and t_2 is unknown but we know that t_2 starts when t_1 ends, then we can assert that t_2 is *met by* t_1. Likewise, if t_3 is an interval

with unknown endpoints and t_3 is $before$ t_1 then, using compositions of Allen relations [1], we infer that t_3 is $before$ t_2 although both interval's endpoints are unknown and their relation is not represented explicitly in the ontology. Semi-closed intervals can be handled in a similar way. For example, if t_1 starts at time point 1, still holds at time point 2, but it's endpoint is unknown, we assert that t_1 has *started by* interval t_2:[1,2].

SOWL demonstrates enhanced expressiveness compared to previous approaches [25; 29; 38; 44; 54; 99; 104; 109] by combining 4D-fluents with Allen's temporal relations, their formal semantics and composition rules as defined in [1]. Notice that, temporal instants still cannot be expressed; subsequently, relations between time instants or between instants and intervals cannot be expressed explicitly.

In this work, an instant-based (or point-based) approach is adopted. Definitions for temporal entities (e.g., instants and intervals) are provided by incorporating OWL-Time into the same ontology. Each interval (which is an individual of the $ProperInterval$ class) is related with two temporal instants (individuals of the $Instant$ class) that specify it's starting and ending points using the $hasBegining$ and $hasEnd$ object properties respectively. In turn, each $Instant$ can be related with a specific date using the concrete $dateTime$ datatype.

One of the $before$, $after$ or $equals$ relations may hold between any two temporal instants with the obvious interpretation. In fact, only the relation $before$ is needed since relation $after$ is defined as the inverse of $before$ and relation $equals$ can be represented using the $sameAs$ OWL keyword applied on temporal instants. In this work, for readability we use all three relations. Notice also that, property $before$ may be also qualitative when holding between time instants or intervals whose values or end points are not specified. This way, we can assert and infer facts beyond the ones allowed when only instants or intervals with known values (e.g., dates) or end-points are allowed. Quantitative defined instants are specified using the $dateTime$ datatype and the supported operators can be applied between instants.

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" [111].

Let $i_1 = [s_1, e_1]$ and $i_2 = [s_2, e_2]$ be two intervals with starting and ending points s_1, s_2 *and* e_1,e_2 respectively; then, the 13 Allen relations of Fig. 2.1 are rewritten as follows:

$$
\begin{aligned}
&i_1\ before\ i_2 \equiv e_1 < s_2 \\
&i_1\ equals\ i_2 \equiv s_1 = s_2 \wedge e_1 = e_2 \\
&i_1\ overlaps\ i_2 \equiv s_1 < s_2 \wedge e_1 < e_2 \wedge e_1 > s_2 \\
&i_1\ meets\ i_2 \equiv e_1 = s_2 \\
&i_1\ during\ i_2 \equiv s_1 > s_2 \wedge e_1 < e_2 \\
&i_1\ starts\ i_2 \equiv s_1 = s_2 \wedge e_1 < e_2 \\
&i_1\ finishes\ i_2 \equiv s_1 > s_2 \wedge e_1 = e_2
\end{aligned}
$$

The relations *after, overlappedby, metby, contains, startedby and finishedby* are the inverse of *before, overlaps, meets, during, starts and finishes* and are defined accordingly (by interchanging s_1, s_2 and e_1, e_2 in their respective definitions). Notice that, in the case of Allen relations additional relations (representing disjunctions of basic relations) are introduced in order to implement path consistency, totalling a set of 29 supported relations (although, such relations are not required by a point algebra). Example of such relations is the disjunction of relations *during, overlaps* and *starts*. The full set of supported relations is presented in Appendix A. These temporal relations and the corresponding reasoning mechanism are integrated within the SOWL ontology.

In the original work by Welty and Fikes [114], the following restriction is imposed on timeslices: whenever two timeslices are related by means of a fluent property, their corresponding temporal intervals must be equal. However, no mechanism for enforcing this restriction is provided. In this work, the following SWRL rule in conjunction with the reasoning mechanism of Chapter 4 imposes the required restriction:

$$fluent(x, y) \wedge tsTimeInterval(y, z) \wedge tsTimeInterval(x, w) \rightarrow equals(w, z)$$

3.2.2 Temporal Representation using N-ary Relations

The N-ary version of the SOWL ontology introduces one additional object for representing a temporal property. This object is an individual of class *Event* and this name convention is also adopted by other approaches such as the LODE ontology [98]. In SOWL, the temporal property remains a property relating the additional object with both the objects (e.g., an *Employee* and a *Company*) involved in a temporal relation. This is illustrated in Fig. 2.7. The representation of qualitative relations between temporal intervals or points (and the corresponding reasoning mechanisms) remain identical to the 4D-fluents based version of the model.

The advantage of the N-ary approach over reification [98] is that property semantics are retained. For example, when a property is the inverse of another, the inverse property declaration is retained. As shown in Fig. 2.7, if *worksFor* is the inverse of *hasEmployee* and *Employee*1 is related with *Company*1 using the *worksFor* relation and an intermediate *EmploymentEvent*1, OWL semantics indicate that the relation *hasEmployee* holding between *Company*1 and *Employee*1 through the *EmploymentEvent*1 object can be inferred. This is not the case with reification because, as shown in Fig. 2.5, the *worksFor* relation is the object of a property, and objects of properties don't have inverses as the properties themselves. The same hold for symmetric and reflexive properties. Transitive properties are more involved since the equality of the related intervals must also hold when a transitive property applies. This can be achieved using an SWRL rule such as in the case of 4D-fluents.

N-ary relations (similarly to 4D-fluents) require modification of domains and ranges of fluent properties. Specifically, when a property is temporal, if the domain of property is *ClassA* and the range is *ClassB* (where domains and ranges can be composite class definitions or atomic concepts), then using the N-ary representation the domain becomes *ClassA OR Event* and the range *ClassB OR Event*. Compared to 4D-fluents, the disjunction of concepts appearing both in domain and ranges of properties limits specificity of references of the N-ary representation.

3.2.3 Spatial Representation

The 4D-fluent mechanism is also enhanced with several types of qualitative spatial relations. These can be either topological or directional [92]. Fig. 3.2 illustrates a general ontology representation model for spatial information. Class *Location* has attribute *name* (of type string). Also a *Location* object can be optionally connected with a *footprint* class with subclasses: *Point, Line, Polyline and MBR*. Class *Point* has two (or three in a three-dimensional representation) numerical attributes, namely X, Y (also Z in a three-dimensional representation). For example, *Point* will be the *footprint* of entities such as cities in a large scale map. Class *Line* has *point1* and *point2* as attributes representing the ending points of a line segment. Class *PolyLine* represents the surrounding contour of an object (or region) as a set of consecutive line segments.

An object (or region) may also be represented by its *Minimum Bounding Rectangle* (MBR) specified by the four numerical attributes *Xmax, Ymax, Xmin and Ymin*. Both representations may co-exist in SOWL (using one of them or both is a design decision).

The spatial relations between regions can be easily extracted from their surrounding *MBRs* (or contours) by comparing their coordinates. In the case of MBR or point-based representations, extraction of qualitative relations from the underlying quantitative representations has been implemented with SWRL rules and embedded into the ontology as part of the reasoning mechanism. In the case of polygons, a separate software component is used for extracting qualitative relations [47]. In an ontology, each *spatialRelation* connects two locations and has two subproperties namely: *topologicRelation* and *directionalRelation*. Fig. 3.3 summarizes all types of spatial relations within a common ontology schema. Omitting one or more types of spatial relations is also a design decision.

The topologic relations shown in Fig. 2.2, *(DC, EC, EQ, NTTP, NTTPi, TTP, TPPi, PO)*, referred to as RCC8 relations [89], are also defined in SOWL. In order to implement a sound and complete reasoning mechanism additional relations are are introduced totalling a minimal set of 49 relations (See Chapter 4). Direction relations are defined based on cone-shaped areas [37]. Other alternative approaches based on 2D-projections are presented at [92; 102]. As shown in

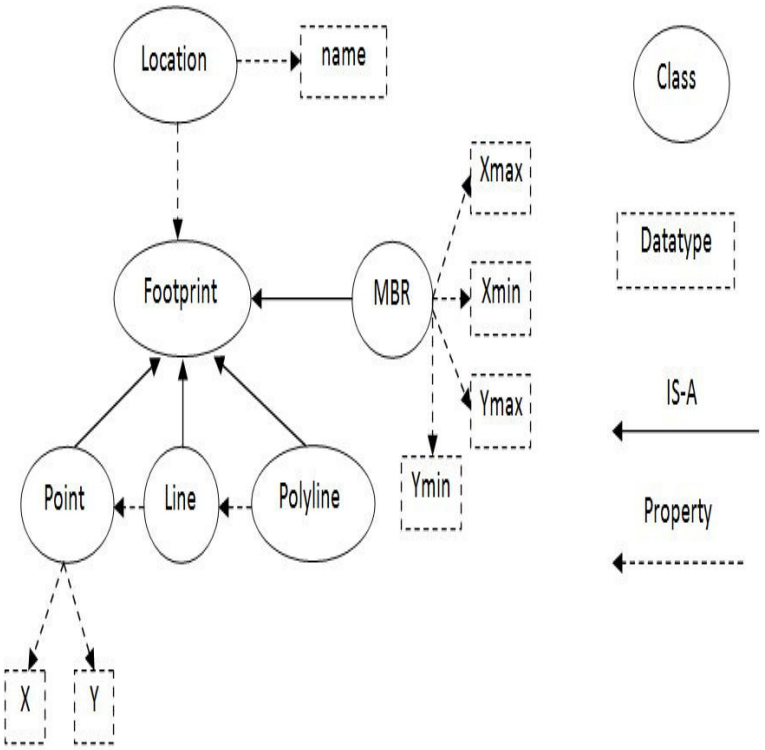

Figure 3.2: Ontology representation of spatial objects.

Fig. 2.3, eight direction relations can be identified namely *North (N), North East (NE), East (E), South East (SE), South (S), South West (SW), West (W) and North West (NW)* following the cone-shaped areas approach of [37].

The cone shaped approach is suitable for objects represented by points (e.g., by their centroid). It complements topological relations that apply on regions. Nevertheless, for completeness, a projection-based approach has been implemented as well. When applied to points, the projection based approach is equivalent to applying point algebra on a pair of orthogonal axes (instead of one in the temporal case), while representing regions corresponds to applying Allen's interval algebra on two axes (one to each dimension) instead of one as in the temporal case. The projections over the horizontal axis define relations $East$ and $West$ (equivalently $left$ and $right$ relations) corresponding to the $Before$ and $After$ relations respectively of the temporal representation. The projections over the vertical axis define the relations $North$ and $South$, (equivalently $front$ and

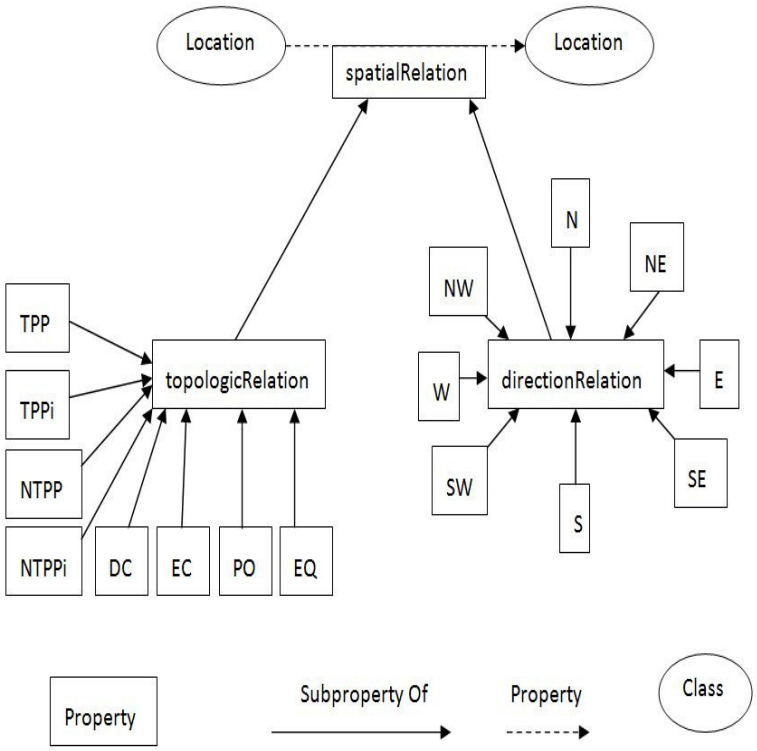

Figure 3.3: Ontology schema with spatial relations.

behind relations).

In contrast to the cone-shaped approach, each axis is independent and different relations (thus and their conjunction) may hold as long as they are defined over a different axis (e.g., *North* and *South* are disjoint properties). Thus, relations *Noth*, *West* and consequently the relation *North-West* may hold simultaneously. This is not the case in the cone-shaped approach where all basic properties are pairwise disjoint. Notice also that, although the cone-shaped and the projection based directional relations result in the same relations set, they convey different semantics and consequently call for different reasoning mechanism. Selecting one of the two approaches referred to above (i.e, cone-shaped or project-based) is subject to user preference or may depend on the application at hand. For example, one may opt for the projection-based approach in the case of large objects or the cone-shaped approach in the case of small objects or objects specified by their coordinates. Both approaches are implemented in SOWL. If the cone-shaped

approach is selected, a set of additional disjunctive relations (totalling a minimal set of 33 relations presented in the Appendix C) are required and they are part of the representation as well. In case of the projection based approach the additional relations are not required (See Chapter 4).

Finally, distance relations are defined and can be used in conjunction with the above relation types. Notice that, qualitative distance relations (e.g., *far* and *near*) may be ambiguous especially in applications where a common scale for measuring distances is not provided. This is resolved when distance relations are expressed quantitatively (e.g., 3Km away from city A) and stored in the ontology as N-ary relations [82] (i.e., by defining an object with attributes the two related locations and a numerical attribute representing their distance). In SOWL, we opt for the later (quantitative) approach for representing distance information.

3.2.4 Combining Spatial and Temporal Representations

In the case of a moving object, its location is a property of a timeslice holding for a specific time interval (Fig. 3.5) while, in the case of a static object, its location is a property of the object and not a property of a timeslice.

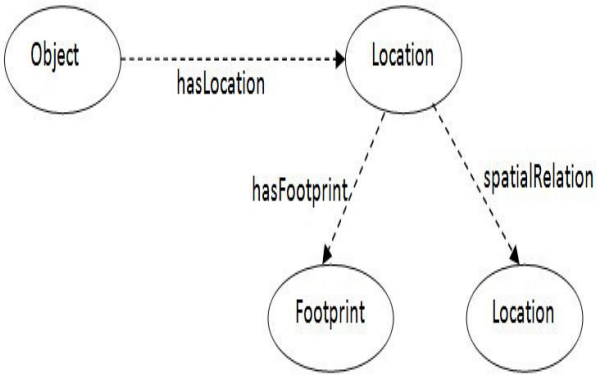

Figure 3.4: Ontology representation of static objects.

Notice that, even if the location of the object is static, some of its properties may change in time so that, there can be timeslices associated with it (e.g., timeslices of a building for different owners in time). In the N-ary based approach, the location of a moving object is a property of the *Event* object that has specific temporal extends. In the case of a static object, its location remains a property of

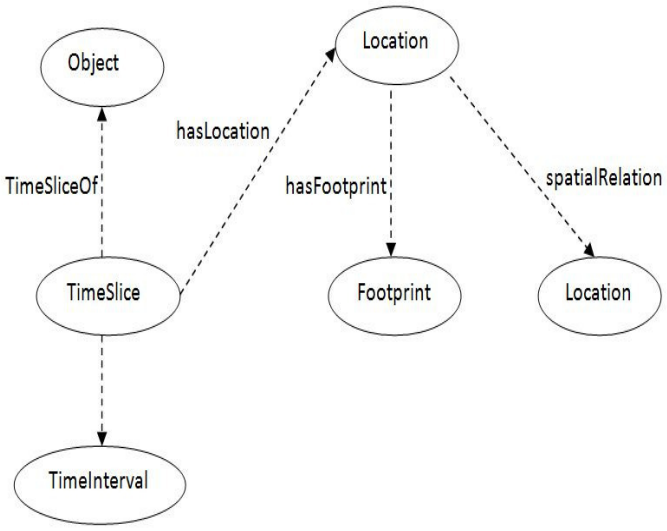

Figure 3.5: Ontology representation of moving objects.

the object. As in 4D-fluents, the object can have temporal properties represented by *Event* objects while it's location is a static property.

Fig. 3.6 illustrates the dynamic ontology schema representing the scenario "T1 Radio was produced in Patras, a city west of Athens from May 2006 to May 2010, since then it is produced at Athens". In this example, we don't know whether the product is still produced in Athens. Only the first temporal interval is defined. The second interval and both locations are unknown and only qualitative relations about them appear into the ontology.

The example of Fig. 3.6 illustrates the applicability of the model in the case of missing or inaccurate information (as it is usually the case with natural language descriptions). In these cases, models based on quantitative information only, are insufficient.

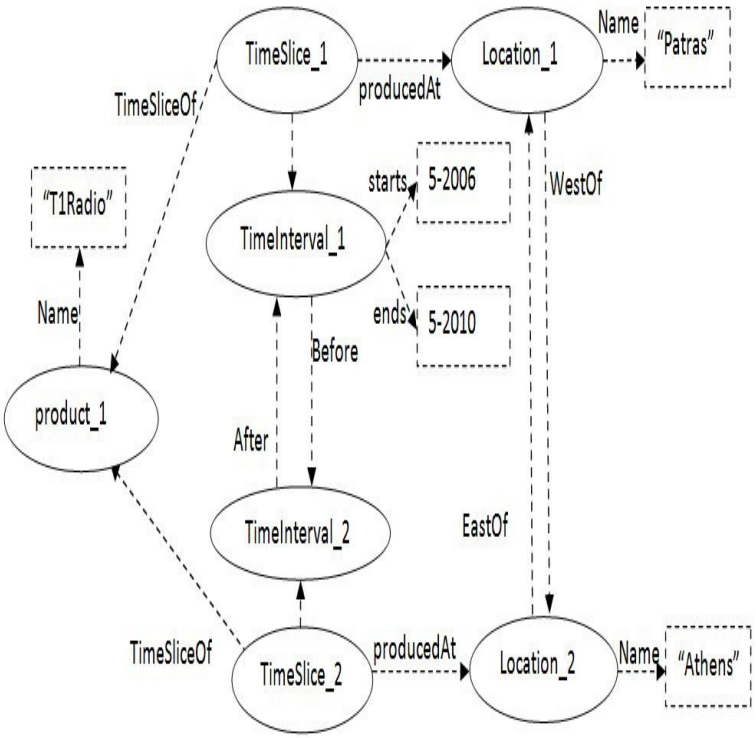

Figure 3.6: Instantiation example.

Chapter 4

Reasoning in SOWL

4.1 Introduction

Temporal and spatial reasoning in SOWL is realized by introducing a set of SWRL[1] rules operating on spatial (topological or directional) relations as well as, by a set SWRL rules for asserting inferred temporal Allen relations. Reasoners that support DL-safe rules (i.e., rules that apply only on named individuals in the knowledge base) such as Pellet [101] can be used for inference and consistency checking over spatio-temporal relations. Alternatively, OWL axioms on temporal properties can be used instead of SWRL. However, this approach cannot guarantee decidability and is therefore not compatible with W3C specifications. In addition to reasoning applying on temporal and spatial relations, the Pellet reasoner applies to the ontology schema for inferring additional facts using OWL semantics (e.g., facts due to symmetric relationships and class-subclass relationships). Checking for property restrictions on temporal properties (*fluent* properties) is also implemented and discussed.

4.2 Temporal Reasoning

Reasoning is applied either on temporal intervals directly [16] or by applying point-based reasoning [18] operating on representations of intervals involving their

[1] http://www.w3.org/Submission/SWRL/

starting and ending points. Both approaches have been implemented and are discussed in the following.

4.2.1 Temporal Reasoning over Interval-Based Representations

Reasoning is realized by introducing a set of SWRL rules operating on temporal intervals. The temporal reasoning rules are based on the composition of pairs of the basic Allen's relations of Fig. 2.1 as defined in [1]. The composition table of basic Allen's relations is presented in Table 4.1. Relations BEFORE, AFTER, MEETS, METBY, OVERLAPS, OVERLAPPEDBY, DURING, CONTAINS, STARTS, STARTEDBY, ENDS, ENDEDBY and EQUALS are represented using symbols B, A, M, Mi, O, Oi, D, Di, S, Si, F, Fi and = respectively. Compositions with EQUALS are not presented since these compositions keep the initial relations unchanged.

The composition table represents the result of the composition of two Allen relations. For example, if relation R_1 holds between $interval_1$ and $interval_2$ and relation R_2 holds between $interval_2$ and $interval_3$ then, the entry of Table 4.1 corresponding to row R_1 and column R_2 denotes the possible relation(s) holding between $interval_1$ and $interval_3$. Not all compositions yield a unique relation as a result. For example, the composition of relations *During* and $Meets$ yields the relation $Before$ as a result while, the composition of relations *Overlaps* and *During* yields three possible relations namely *Starts, Overlaps and During*. Rules corresponding to compositions of relations R_1, R_2 yielding a unique relation R_3 as a result can be represented using SWRL as follows:

$$R_1(x, y) \wedge R_2(y, z) \rightarrow R_3(x, z) \tag{4.1}$$

An example of temporal inference rule is the following:

$$DURING(x, y) \wedge MEETS(y, z) \rightarrow BEFORE(x, z)$$

Rules yielding a set of possible relations can't be represented in SWRL since, disjunctions of atomic formulas are not permitted as a rule head. Instead, disjunc-

	B	A	D	Di	O	Oi	M	Mi	S	Si	F	Fi
B	B	B,A,D,Di,O,Oi M,MiS,Si,F,Fi, Eq	B,O,M, D,S	B	B	B,O,M,D,S	B	B,O,M,D, S	B	B	B,O,M, D,S	B
A	B,A,D,Di, O,Oi,M, Mi,S,Si,F, Fi,Eq	A	A,Oi,Mi D,F	A	A,Oi, Mi,D,F	A	A,Oi,Mi, D,F	A	A,Oi,Mi D,F	A	A	A
D	B	A	D	B,A,D,Di,O,Oi M,MiS,Si,F,Fi, Eq	B,O,M, D,S	A,Oi,Mi,D,F	B	A	D	A,Oi,Mi D,F	D	B,O,M, D,S
Di	B,O,M,Di, Fi	A,Oi,Di,Mi,Si	O,Oi,D, Di,S,Si, F,Fi,Eq	Di	O,Di,Fi	Oi,Di,Si	O,Di,Fi	Oi,Di,Si	O,Di,Fi	Di	Oi,Di,Si	Di
O	B	A,Oi,Di,Mi,Si	O,D,S	B,O,M,Di,Fi	B,O,M	O,Oi,D,Di,S,Si, F,Fi,Eq	B	Oi,Di,Si	O	O,Di,Fi	O,D,S	B,O,M
Oi	B,O,M,Di, Fi	A	Oi,D,F	A,Oi,Di,Mi,Si	O,Oi,D, Di,S,Si, F,Fi,Eq	A,Oi,Mi	O,Di,Fi	A	Oi,D,F	Oi,A,Mi	Oi	Oi,Di,Si
M	B	A,Oi,Di,Mi,Si	O,D,S	B	B	O,D,S	B	F,Fi,Eq	M	M	O,D,S	B
Mi	B,O,M,Di, Fi	A	Oi,D,F	A	Oi,D,F	A	S,Si,Eq	A	Oi,D,F	A	Mi	Mi
S	B	A	D	B,O,M,Di,Fi	B,O,M	Oi,D,F	B	Mi	S	S,Si,Eq	D	B,O,M
Si	B,O,M,Di, Fi	A	Oi,D,F	Di	O,Di,Fi	Oi	O,Di,Fi	Mi	S,Si,Eq	Si	Oi	Di
F	B	A	D	A,Oi,Di,Mi,Si	O,D,S	A,Oi,Mi	M	A	D	A,Oi,Mi	F	F,Fi,Eq
Fi	B	A,Oi,Di,Mi,Si	O,D,S	Di	O	Oi,Di,Si	M	Oi,Di,Si	O	Di	F,Fi,Eq	Fi

Table 4.1: Composition table for Allen's temporal relations.

tions of relations are represented using new relations whose compositions must also be defined and asserted into the knowledge base. For example, the composition of relations *Overlaps* and *During* yields the disjunction of three possible relations (*During, Overlaps* and *Starts*) as a result:

$$OVERLAPS(x, y) \wedge DURING(y, z) \rightarrow$$

$$During \vee Starts \vee Overlaps$$

If the relation DOS represents the disjunction of relations *During, Overlaps* and *Starts*, then the composition of *Overlaps* and *During* can be represented using SWRL as follows:

$$OVERLAPS(x, y) \wedge DURING(y, z) \rightarrow DOS(x, z)$$

The set of possible disjunctions over all basic Allen's relations contains 2^{13} relations, and reasoning over all temporal Allen relations has exponential time complexity. However, subsets of this set that are closed under composition (i.e., compositions of relation pairs from this subset yield also a relation in this subset) are also known to exist [81; 112]. In addition, inverse axioms (relations AFTER, METBY, OVERLAPPEDBY, STARTEDBY, CONTAINS and FINISHEDBY are the inverse of BEFORE, MEETS, OVERLAPS, STARTS, DURING and FINISHES respectively) and rules defining the relation holding between two intervals with known starting and ending points (e.g., if the ending point of $interval_1$ is before the starting point of $interval_2$ then, $interval_1$ is *before* $interval_2$) are also asserted into the knowledge base.

The starting and ending points of intervals are represented using concrete datatypes such as *xsd:date* that support ordering relations. Axioms involving disjunctions of basic relations are denoted using the corresponding axioms for these basic relations. Specifically, compositions of disjunctions of basic relations are defined as the disjunction of the compositions of these basic relations. For example, the composition of relation DOS (representing the disjunction of *During, Overlaps and Starts*), and the relation *During* yields the relation DOS as a result as follows:

$$DOS \circ During \rightarrow (During \vee Overlaps \vee Starts) \circ During \rightarrow$$

$$(During \circ During) \vee (Overlaps \circ During) \vee (Starts \circ During)$$

$$\rightarrow (During) \vee (During \vee Overlaps \vee Starts) \vee (During)$$

$$\rightarrow During \vee Starts \vee Overlaps \rightarrow DOS$$

The symbol $\circ$ denotes composition of relations. Compositions of basic (non-disjunctive) relations are defined using Table 4.1. Similarly, the inverse of a disjunction of basic relations is the disjunction of the inverses of these basic relations illustrated in Fig. 2.1. For example, the inverse of the disjunction of relations $Before$ and $Meets$ is the disjunction of their inverse relations, $After$ and $MetBy$ respectively.

By applying compositions of relations, the implied relations may be inconsistent. Consistency checking is achieved by applying path consistency [81; 92; 112]. Path consistency is implemented by consecutive application of the formula:

$$\forall x, y, k \; R_s(x, y) \leftarrow R_i(x, y) \cap (R_j(x, k) \circ R_k(k, y)) \tag{4.2}$$

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

An additional set of rules defining the result of intersection of relations holding between two intervals is also introduced. These rules are of the form:

$$R_1(x, y) \wedge R_2(x, y) \rightarrow R_3(x, y), \tag{4.3}$$

where R_3 can be the empty relation. For example, the intersection of relation DOS (represents the disjunction of *During, Overlaps and Starts*) with relation $During$, yields relation $During$ as a result:

$$DOS(x, y) \wedge During(x, y) \rightarrow During(x, y).$$

The intersection of relations $During$ and $Starts$ yields the empty relation, and an inconsistency is detected:

$$Starts(x, y) \wedge During(x, y) \rightarrow \bot.$$

The maximal tractable subset of Allen relations containing all basic relations

when applying path consistency comprises of 868 relations [81]. Tractable subsets of Allen relations containing 83 or 188 relations [112] can be used instead, offering reduced expressiveness but increased efficiency over the maximal subset of [81]. Furthermore, since the proposed temporal reasoning mechanism affects only relations of temporal intervals, it can be also applied to other temporal representation methods (besides 4D-fluents) such as N-ary relations. Reasoning operating on temporal instants rather on intervals is also feasible [112]. Specifically, qualitative relations involving instants form a tractable set if relation $\neq$ (i.e., a temporal instant is before *or* after another instant) is excluded. Reasoning involving relations between interval and instants is achieved by translating relations between intervals to relations between their endpoints [1].

Path consistency requires composition of properties, intersection of properties and role complement. Notice that, disjointness of properties can be represented in terms of complement of properties (i.e., two properties are disjoint when one of them is *subproperty* of the *complement* of the second property). However, the combination of property composition, intersection and complement has been proven to be undecidable [94]. Instead of property complement, the disjointness of two properties can be represented as an *at most 0* cardinality constraint over their intersection. However, the intersection and the composition of two properties is a composite (i.e., not simple) property and applying cardinality constrains over composite properties has been proven to be undecidable [53]. Therefore, reasoning using SWRL, as proposed in this thesis, is the only solution complying with current OWL specifications while retaining decidability.

Implementing path consistency over Allen relations (or topological and directional spatial relations) requires minimizing the required additional relations and rules for implementing the mechanism. Existing work (e.g., [90]) emphasizes on determining maximal tractable subsets of relations while, practical implementations calls for minimizing of such relation sets (i.e., finding the minimal tractable set that contain the required relations). For example, implementing path consistency over the maximal tractable set of Allen relations [90], containing 868 relations is impractical, since defining all intersections and compositions of pairs of relations by means of SWRL rules requires millions of such rules.

In this work, minimal relation sets containing a tractable set of basic relations

are detected by applying the *closure method* of Table 4.2 (i.e., starting with a set of relations, intersections and compositions of relations are applied iteratively until no new relations are produced). Applying the closure method over the set of basic Allen relations yields a tractable set containing 29 relations, illustrated in Appendix A.

```
Input: Set S of tractable relations
 Table C of compositions
 WHILE S size changes
      BEGIN
           Compute C:Set of compositions of relations in S
           S=S ∪ C
           Compute I:set of intersections of relations in S
           S= S ∪ I
      END
 RETURN S
```

Table 4.2: Closure method

Notice that, implementing path consistency using rules of the form of Eq. 4.2 over n relations requires $O(n^3)$ rules (i.e., rules for every possible selection of three relations must be defined), while implementing path consistency using rules according to Eq. 4.1 and Eq. 4.3 (as implemented in this work) requires $O(n^2)$ rules, since rules for every pair of relations must be defined. Further improvements and reductions can be achieved by observing that the disjunction of all basic Allen relations when composed with other relations yields the same relation, while intersections yield the other relation. Specifically, given that *All* represents the disjunction of all basic relations and, R_x is a relation in the supported set then the following hold for every R_x:

$$All(x, y) \wedge R_x(x, y) \rightarrow R_x(x, y)$$

$$All(x, y) \wedge R_x(y, z) \rightarrow All(x, z)$$

$$R_x(x, y) \wedge All(y, z) \rightarrow All(x, z)$$

Since relation *All* always holds between two individuals, because it is the disjunction of all possible relations, all rules involving this relation, both com-

positions and intersections, do not add new relations into the ontology and they can be safely removed. Also, all rules yielding the relation *All* as a result of the composition of two supported relations R_{x1}, R_{x2} :

$$R_{x1}(x, y) \wedge R_{x2}(y, z) \rightarrow All(x, z)$$

can be removed too. Thus, since intersections yield existing relations and the fact that the disjunction over all basic relations must hold between two intervals, all rules involving the disjunction of all basic relations and consequently all rules yielding this relation can be safely removed from the knowledge base. After applying this optimization the required number of axioms for implementing path consistency over the minimal tractable set of Allen relations is reduced to 983.

4.2.2 Reasoning over Point-Based Representations

The possible relations between temporal instants are $before$, $after$ and $equals$, denoted as "<",">","=" respectively. Table 4.3 illustrates the set of reasoning rules defined on the composition of existing relation pairs.

Relations	$<$	$=$	$>$
$<$	$<$	$<$	$<, =, >$
$=$	$<$	$=$	$>$
$>$	$<, =, >$	$>$	$>$

Table 4.3: Composition Table for point-based temporal relations.

The composition table represents the result of the composition of two temporal relations. For example, if relation R_1 holds between $instant_1$ and $instant_2$ and relation R_2 holds between $instant_2$ and $instant_3$ then, the entry of Table 4.3 corresponding to row R_1 and column R_2 denotes the possible relation(s) holding between $instant_1$ and $instant_3$. Also, the three temporal relations are declared as pairwise disjoint, since they can't simultaneously hold between two instants. Not all compositions yield a unique relation as a result. For example, the composition of relations $before$ and $after$ yields all possible relations as a result. Because such compositions don't yield new information these rules are discarded. Rules corresponding to compositions of relations R_1 *and* R_2 yielding a unique relation

R_3 as a result are retained (7 out of the 9 entries of Table 4.3 are retained) and are expressed in SWRL using rules of the form (Eq. 4.1):

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

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

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

Therefore, 7 out of the 9 entries in Table 4.1 can be expressed using SWRL rules while, the two remaining entries don't convey new information. A series of compositions of relations may imply relations which are inconsistent with existing ones. Consistency checking is achieved by imposing path consistency [112]. Path consistency is implemented by iteratively applying formula of Eq. 4.2:

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

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. In addition to rules implementing compositions of temporal relations, a set of rules defining the result of intersecting relations holding between two instances must also be defined in order to implement path consistency. These rules are of the form of Eq.4.3:

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

where R_3 can be the empty relation. For example, the intersection of the relation representing the disjunction of *before, after and equals* (abbreviated as ALL), and the relation $before$ yields the relation $before$ as result:

$$ALL(x, y) \wedge before(x, y) \rightarrow before(x, y)$$

The intersection of relations $before$ and $after$ yields the empty relation, and

an inconsistency is detected:

$$before(x, y) \wedge after(x, y) \rightarrow \bot$$

As shown in Table 4.3, compositions of relations may yield one of the following four relations: *before, after, equals* and the disjunction of these three relations. Intersecting the disjunction of all three relations with any of these leaves existing relations unchanged. Intersecting any one of the tree basic (non disjunctive) relations with itself also leaves existing relations unaffected. Only compositions of pairs of different basic relations affect the ontology by yielding the empty relation as a result, thus detecting an inconsistency. By declaring the three basic relations *before, after, equals* as pairwise disjoint, all intersections that can affect the ontology are defined. Path consistency is implemented by defining compositions of relations using SWRL rules and by declaring the three basic relations as disjoint. Notice that, path consistency is sound and complete when applied on the three basic relations [111].

Alternatively, we can define the composition of $before$ with itself as a transitivity axiom rather than by an SWRL rule. In this case, there would be no need for SWRL rules applying only on named individuals into the ontology ABox. The resulting representation will apply on the TBox as well. However, this is not compatible with OWL 2.0 thus imposing the use of SWRL rules: relation $before$ must be declared as transitive in order to infer implied relations and disjoint with $after$, it's inverse relation, (also $before$ is asymmetric and irreflexive) in order to detect inconsistencies. However, OWL specifications[1] disallow the combination of transitivity and disjointness (or asymmetry) axioms on a property since they can lead to undecidability [50]. This restriction is necessary in order to guarantee decidability of the basic reasoning problems for OWL 2 DL.

In cases where temporal information is provided as dates, the qualitative relations are specified using SWRL rules that apply on the quantitative representation. An example of such a rule is the following:

$$Instant(x) \wedge Instant(z) \wedge inXSDDateTime(x, y)$$

[1] http://www.w3.org/TR/2009/REC-owl2-syntax-20091027/#The_Restrictions_on_the_Axiom_Closure

$$\wedge inXSDDateTime(z, w) \wedge lessThan(y, w) \rightarrow before(z, x)$$

Replacing the *lessThan* operator in the rule with *greaterThan* and *equal* yields the corresponding rules for relations $after$ and *equals* respectively. These qualitative relations can be combined with asserted and inferred qualitative relations using path consistency.

All interval relations can be represented by means of point relations between their end-points. Rules implementing transformation of Allen relations to endpoint relations and rules yielding Allen relations from endpoint relations have been implemented as well. For example, the rule yielding the *During* Allen relation from endpoint relations is the following:

$$ProperInterval(a) \wedge ProperInterval(x) \wedge before(b, y)$$

$$\wedge before(z, c) \wedge hasBeginning(a, b)$$

$$\wedge hasBeginning(x, y) \wedge hasEnd(a, c) \wedge hasEnd(x, z) \rightarrow intervalDuring(x, a)$$

Rules similar to the above, yielding all basic Allen relations are implemented. Notice that, the inverse transformation can't be expressed by a single SWRL rule: one Allen relation corresponds to four end-point relations and conjunctions at the rule head are not supported in SWRL. Conjunctions can be expressed as rules with identical antecedent part and different head. For example, the following rules represent the transformation of relation $IntervalOverlaps$:

$$hasBeginning(a, b) \wedge hasBeginning(x, y) \wedge hasEnd(a, c) \wedge hasEnd(x, z)$$

$$\wedge intervalOverlaps(x, a) \rightarrow before(z, c)$$

$$hasBeginning(a, b) \wedge hasBeginning(x, y) \wedge hasEnd(a, c) \wedge hasEnd(x, z)$$

$$\wedge intervalOverlaps(x, a) \rightarrow before(b, z)$$

$$hasBeginning(a, b) \wedge hasBeginning(x, y) \wedge hasEnd(a, c) \wedge hasEnd(x, z)$$

$$\wedge intervalOverlaps(x, a) \rightarrow before(y, b)$$

$$hasBeginning(a, b) \wedge hasBeginning(x, y) \wedge hasEnd(a, c) \wedge hasEnd(x, z)$$

$$\wedge intervalOverlaps(x, a) \rightarrow before(y, c)$$

In fact, the last rule in the above example is implied by the previous rules and the rules that specify that the start of an interval is before the end and it can be omitted.

Notice that, if data consistency can be assured, then reasoning can be significantly speeded-up. In cases where all relations are specified quantitatively (i.e., by numerical values) reasoning with path consistency can be dropped. For example, for intervals with known end-points, all possible relations between them can be computed in quadratic time from their end-point dates. The computed set of relations is guaranteed to be consistent and reasoning is not needed.

If consistency checking is not needed (in case instance assertions doesn't contain conflicts -implied or direct) then, temporal properties need not be declared disjoint. For example if sequences of events are recorded using sensors, then there is a valid arrangement of the events on the axis of time (i.e., the sequence of their recording), thus their temporal relations are consistent by definition. In this case, reasoning can be achieved using OWL role inclusion axioms instead of SWRL rules that apply on the ontology TBOX as well. Such axioms are of the form:

$$before \circ equals \sqsubset before$$

All relation compositions can be defined similarly. Intersections of relations are not required in case of basic point algebra relations and if the consistency checking requirement is dropped, only OWL axioms are sufficient for implementing the reasoning mechanism. In this case, a great speed up is achieved, since in our experiments for over 20,000 random instances, reasoning is achieved in about 18 seconds (which is comparable to the time required for reasoning over 80 instances using SWRL when consistency checking is required) as presented in Fig. 6.11. This speed-up can be achieved only in special cases where consistency of data is guaranteed (thus consistency checking can be dropped), which is not the case for

example in natural language text.

4.3 Spatial Reasoning

In the following, reasoning over both, topological and directional relations is discussed. Choosing either representation is a design decision that depends mainly on the application. However, both representations may co-exist in the SOWL model (both are common in natural language expressions and may co-exist e.g., in text descriptions over the Web). A third case, is reasoning over interval-based (or their equivalent point-based representations) obtained as the projections of spatial entities (i.e., points or regions in a two-dimensional or three-dimension space). Reasoning over interval-based (or point-based) spatial projections is equivalent to reasoning over temporal intervals (or points) discussed in Sec. 4.2.

Table 4.5 illustrates a composition table for RCC8 topological relations [16]. The corresponding composition table for directional relations is illustrated in Table 4.4. Spatial reasoning is then achieved by applying rules implementing the inferred relations of the composition table at hand.

As shown in Table 4.5, only a limited set of table entries leads to an unambiguous result. For example, the composition of the *NTPP* and *DC* topologic relations (i.e., object A is into B and object B outside of C) yields the *DC* relation as a result meaning that A is outside of C. However, the composition of *NTPP* and *PO* relations doesn't yield a unique relation as a result. Only 27 out of the 64 (RCC-8) entries of Table 4.5, and only 8 out of the 64 compositions of basic cone-shaped directional relations [37] can be used to infer unique relations.

The SOWL spatial representation implements reasoning rules for RCC8 relations and cone-shaped direction relations using SWRL and OWL 2.0 property axioms. All basic relations are pairwise disjoint. Their inverse relations (e.g., North is the inverse of South) are defined as well. Furthermore, the point identity relation (O) is handled using the OWL *SameAs* keyword applied on points instead of explicitly asserting the relation. Path consistency is implemented by introducing rules defining compositions and intersections of supported relations until a fixed point is reached or until an inconsistency is detected [33; 37; 91]. The supported directional relations are the 9 basic relations and their disjunctions

	N	NE	E	SE	S	SW	W	NW	O
N	N	N,NE	N,NE,E	N,NE,E,SE	N,NE,E,SE S, SW,W NW,O	W,NW, SW,N	NW,N,W	NW,N	N
NE	NE,N	NE	NE,E	E,NE,SE	E,NE, SE,S	N,NE,E,SE S, SW, W,NW,O	N,NE, NW,W	N,NE,NW	NE
E	NE,E,N	NE,E	E	SE,E	SE,E,S	S,SW,SE E,	N,NE,E, SE,S, SW W,NW,O	N,NW, NE,E	E
SE	E,SE, NE,N	E,SE,NE	SE,E	SE	SE,S	S,SE,SW	S,SE,SW	N,NE,E,SE,S, SW,W,NW,O	SE
S	N,NE,E,SE,S SW,W NW,O	E,S,NE,SE	SE,E,S	SE,S	S	S,SW	S,W,SW	W,S,NW, SW	S
SW	W,SW N,NW	N,NE,E,SE,S SW,W NW,O	S,SW SE,E	S,SW,SE	SW,S	SW	SW,W	W,NW,SW	SW
W	N,W,NW	N,NW,NE W	N,NE,E,SE, S ,SW,W NW,O	S,SE,SW W	W,S,SW	W,SW	W	W,NW	W
NW	N,NW	N,NW,NE	N,NW,NE,E	N,NE,E,SE S,SW,W, NW,O	W,NW,SW, S	W,NW, SW	NW,W	NW	NW
O	N	NE	E	SE	S	SW	W	NW	O

Table 4.4: Composition table for cone-shaped directional relations.

appearing in Table. 4.4. Compositions and intersections of disjunctive relations are defined using the compositions and intersections of basic relations as in the case of temporal reasoning.

The directional relations in SOWL (under the assumption that the line separating two 2D cone-shaped areas e.g., North from North-West, is part of only one of these areas, preserving the disjointness of basic relations) are a special case of the revised Star Calculus [91] and is decided by path consistency when applied to basic relations. Furthermore, given a tractable set of relations, by applying com-

positions, intersections and inverse operations until a set of relations that is closed under these operations is yielded, the resulting relations set is also tractable [92]. By applying this *closure method* to the basic relations of Fig. 2.3 a tractable set of relations containing the basic directional relations and all relations appearing in Table 4.4 is yielded. This set of directional relations is used in this work for directional spatial reasoning.

	DC	EC	PO	TPP	NTPP	TPPi	NTPPi	EQ
DC	DC,EC,PO, TPP,NTPP, TPPi,NTPPi, EQ	DC,EC,PO, TPP,NTPP	DC,EC,PO, TPP,NTPP	DC,EC,PO, TPP,NTPP	DC,EC,PO, TPP,NTPP	DC	DC	DC
EC	DC,EC,PO, TPPi,NTPPi	DC,EC,PO,TPP TPPi,EQ	DC,EC,PO, TPP,NTPP	EC,PO, TPP,NTPP	PO,TPP,NTPP	DC,EC	DC	EC
PO	DC,EC,PO, TPPi,NTPPi	DC,EC,PO, TPPi,NTPPi	DC,EC,PO,TPP, NTPP,TPPi, NTPPi,EQ	PO,TPP,NTPP	PO,TPP,NTPP	DC,EC,PO, TPPi,NTPPi	DC,EC,PO, TPPi,NTPPi	PO
TPP	DC	DC,EC	DC,EC,PO, TPP,NTPP	TPP,NTPP	NTPP	DC,EC,PO, TPP,NTPP	DC,EC,PO, TPPi,NTPPi	TPP
NTPP	DC	DC	DC,EC,PO, TPP,NTPP	NTPP	NTPP	DC,EC,PO, TPP,NTPP	DC,EC,PO TPP,NTPP, TPPi,NTPPi EQ	NTPP
TPPi	DC,EC,PO, TPPi,NTPPi	EC,PO, TPPi,NTPPi	PO,TPPi, NTPPi	EQ,PO,TPPi, TPP	PO,TPP,NTPP	TPPi,NTPPi	NTPPi	TPPi
NTPPi	DC,EC,PO, TPPi,NTPPi	PO,TPPi, NTPPi	PO,TPPi, NTPPi	PO,TPPi, NTPPi	PO,TPP,NTPP EQ,TPPi,NTPPi	NTPPi	NTPPi	NTPPi
EQ	DC	EC	PO	TPP	NTPP	TPPi	NTPPi	EQ

Table 4.5: Composition table for RCC8 topological relations.

Reasoning on RCC8 relations also combines OWL property axioms along with a set of composition rules (i.e., rules defining compositions of RCC8 relations)

and intersection rules. Specifically, relations *DC, EC* and *PO* are symmetric, and relations *NTPPi* and *TPPi* are inverse of *NTPP* and *TPP* respectively. *EQ* corresponds to the equality relation for *Location* objects. In SOWL, the spatial reasoner implements the RCC8 composition rules of Table 4.5. For example the rule defining the composition of relations $TPPi$ and $NTPPi$ for locations x, y, z is the following:

$$TPPi(x, y) \wedge NTPPi(y, z) \rightarrow NTPPi(x, z)$$

Extracting spatial relations from the raw spatial data depends on the application and is not part of the reasoning mechanism, besides the specific case of MBRs where rules for extracting both projection based directional relations and RCC8 relations given the MBRs coordinates have been implemented. Extracting directional and topologic applications from random polygons has been implemented in our laboratory as an external application [47].

Notice that, using the full set of relations (totalling $2^8 - 1$ relations in case of RCC8) leads to intractability since this set is not decided by path consistency. However, tractable subsets of the full set are known to exist [90; 92]. Such subsets are used in this work offering increased expressive power while retaining tractability. Specifically applying the closure method of Table 4.2 over RCC8 topological and cone-shaped directional relations yields two sets with 49 and 33 relations respectively. These relation sets are presented at the Appendixes B and C respectively. Implementing path consistency over these sets as described in Sec. 4.2.1 requires a total of 1439 and 964 axioms respectively [16].

4.3.1 Reasoning over Point-Based Spatial Representations using Point Algebra

The point based representation and reasoning method presented in Sec. 4.2.2 applies also to spatial data in 2 (or 3) dimensions with points represented by their x, y (and z) coordinates. Then, relations *East* and *West* between regions or locations are defined using their point projections on X axis, relations *South* and *North* are defined using projections on Y axis while, relations *Bellow* and

Above are defined using projections on Z axis. The reasoning mechanism for point based temporal information applies in the case of spatial information as it is, by replacing the temporal relations with the corresponding spatial relations on each axis (i.e., $North$ and $South$, $East$ and $West$, $Bellow$ and $Above$ replace relation $before$ and $after$ on the X, Y and Z axis respectively). Rules defining relations representing conjunctions of basic relations (e.g., $NorthWest$ is the conjunction of $North$ and $West$)are also defined:

$$North(x, y) \wedge West(x, y) \rightarrow NorthWest(x, y)$$

and the inverse:

$$NorthWest(x, y) \rightarrow North(x, y)$$

$$NorthWest(x, y) \rightarrow West(x, y)$$

Adding equivalent rules for relations $NorthEast$, $SouthWest$ and $SouthEast$ along with rules extracting qualitative relations by comparing point coordinates, are sufficient for realizing a projection based spatial reasoning mechanism for directional relations. Notice that, the reasoning mechanisms referred to above is based on projections and is not compatible with reasoning over cone-shaped relations. In the later case relations such as $NorthWest$ are not the conjunction of relations such as $North$ and $West$ as in the case of projections. Reasoning for cone-shaped relations is discussed in Section Sec. 4.3. Whether a projection based or a cone-shaped approach is adopted is a design decision.

In case of regions represented by their Minimum Bounding Rectangles (MBRs), the point based approach still applies. Two SWRL rules (one for each axis) are introduced for computing the coordinates of the centroid (i.e., as the average of the maximum and minimum values over each axis). Then, the directional relations between regions are defined as the directional relations between their centroids. Finally, topological relations can be also extracted by comparing the coordinates of an MBR representation.

4.4 Restriction Checking over Temporal Properties

Checking for restrictions holding on time dependent (fluent) properties requires particular attention. If a fluent property holds between two objects (classes), then, these objects are only indirectly associated through one or more artificial objects (e.g., TimeSlice object in 4D-fluents). Notice that both, reification and the 4D-fluents approach introduce additional objects for expressing fluent properties. A fluent property is declared between the artificial object and an actual object (as in Fig. 2.7 and 2.5) or between two artificial objects (as in Fig. 2.6). Checking for property restrictions would require adjusting the domain and range of this property from the artificial to the actual objects (e.g., to *Company* and *Employee* objects in Fig 2.7). This, in turn, calls for extra rules or software, which is a disadvantage pertaining to all methods considered in this work (i.e., 4D-fluents and N-ary relations). For example, for the *worksfor* property in Fig. 2.6, the domain of the property is no longer class *Employee* but *timeslice of Employee*. Accordingly, it's range is *timeslice of Company*.

Similar adjustments must be made in the case of N-ary relations but, in this case, combining transitivity of properties while retaining domain and range restrictions becomes problematic: for example, the $worksfor$ relation in Fig. 2.7 must be provided with two alternative domains and ranges. Other restrictions on properties such as symmetric, asymmetric, reflexive, irreflexive and transitive can be applied directly on the temporal property retaining the intended semantics.

Universal restrictions (e.g., "all Employees work for a company") also require adjusting domains and ranges (i.e., all timeslices of employees $workfor$ timeslices of companies). Existential restrictions are adjusted as well (if for example each employee must work for some company then, timeslices of employees must work for some timeslices of companies). Notice that, an existential restriction corresponds to an *at least one* qualified cardinality restriction in OWL and the way it is handled is discussed in the rest of this section.

Adjusting cardinality restrictions, functional and inverse functional properties is somewhat more complicated. Functional properties are a special case of cardinality restrictions (i.e., if a property is functional, then each object must

be connected to *at most* one subject which is different for each object). Cardinality restrictions are amenable to two different interpretations depending on the specific application and the intended semantics: Cardinality restrictions may be interpreted, either as restricting the total number of individuals of a class (e.g., *Company*) related with each individual of another class (e.g., *Employee*) through a fluent property at all times or, as restricting the number of individuals for each specific temporal interval that the fluent property holds true. The first interpretation is handled simply by counting on the number of individuals of a class related with this property and is implemented in SWRL (because OWL cardinality restrictions cannot handle fluent properties connecting objects through intermediate objects).

The following rule expresses the restriction that each employee can work for at most n companies. If $n+1$ company individuals are found to connect with an employee individual, then the restriction is violated (the $Alldifferent$ keyword is an abbreviation for a series of axioms imposing that the n+1 individuals $z_1, z_2 ... z_{n+1}$ are all different). By imposing a *max* cardinality restriction of 0 over property *error* at the definition of class *Employee* the violation of the cardinality restriction is detected by standard reasoners such as Pellet using the rule:

$$(At-most-rule1)Employee(x) \wedge (tsTimesliceOf(x_1, x)$$

$$\wedge ... \wedge tsTimesliceOf(x_{n+1}, x)$$

$$\wedge worksfor(x_1, y_1) \wedge worksfor(x_{n+1}, y_{n+1}$$

$$\wedge tsTimesliceOf(y_1, z_1) ... \wedge tsTimesliceOf(y_{n+1}, z_{n+1})$$

$$\wedge Alldifferent(z_1, z_2, ..., z_{n+1})$$

$$\wedge Company(z_1) ... \rightarrow error(x, z_1)$$

An $at-least$ restriction is expressed similarly as follows: an *at most* $n-1$ rule is applied (changing the asserted property to $satisfies(x, n)$) followed by an *at least one* cardinality restriction on the $satisfies$ property for class *Employee*. An *exact* cardinality restriction can be expressed by combining an *at least* n with an *at most* n restriction. All rules impose also a restriction on the type of objects

involved (e.g., they require that only company objects are involved by checking only for objects connected with timeslices of companies). Dropping such a check leads to an unqualified numeric restriction on the property. Notice that, the *Open World Assumption* of OWL will cause a reasoner (e.g., Pellet) not to detect an inconsistency of an $at-least$ restriction as future assertions might cause invalidity of this inconsistency detection. Instead, the the user can retrieve individuals that are not yet proven to satisfy the restriction using the following SPARQL query:

```
select distinct ?x
where {
?x rdf:type ex1:Employee.
OPTIONAL{
?x ex1:satisfies ?y.}
FILTER(!bound(?y))}
```

The second interpretation imposes restrictions on the number of individuals associated with an individual of a specific class through a fluent property for every temporal interval that the property holds true. Checking for such restrictions requires applying reasoning rules over the relations between the temporal intervals associated with the fluent property as described in Sec. 4.2. The next step is to detect overlapping and non-overlapping intervals. After the Allen relations holding between intervals have been inferred, Allen properties *during, contains, starts, startedby, finishes, finishedby, overlaps, overlapedby, equals* are defined as subproperties of property *overlapping*, thus detecting overlapping and non-overlapping intervals. Also, properties *before, after, meets, metby* are defined as subproperties of property *non-overlapping*.

Expressing an *at most* restriction for every time interval is based on the following observation: the restriction is violated iff $n + 1$ distinct individuals are connected with a given individual (through their timeslices) with the relation at hand, and their corresponding intervals are all pairwise overlapping. Iff $n + 1$ intervals are pairwise overlapping then, there exist an interval where $n + 1$ intervals share a common sub-interval, and this can be proven by induction on n. The existence of such an interval implies that for this interval the *at least* restriction is violated. The corresponding rule (used in combination with a cardinality restriction on property error for inconsistency detection by reasoners) is expressed

as (the *pairwiseoverlapping* is an abbreviation for a set of *overlapping* relations between all pairs of intervals at hand) :

$$(At-most-rule2)Employee(x) \wedge (tsTimesliceOf(x_1, x)$$

$$\wedge ... \wedge tsTimesliceOf(x_{n+1}, x) \wedge hasinterval(x_{n+1}, w_{n+1})$$

$$\wedge worksfor(x_1, y_1) \wedge worksfor(x_{n+1}, y_{n+1}$$

$$\wedge tsTimesliceOf(y_1, z_1) ... \wedge tsTimesliceOf(y_{n+1}, z_{n+1}) \wedge$$

$$Alldifferent(z_1, ..., z_{n+1}) \wedge pairwiseoverlapping(w_1, ...w_{n+1})$$

$$\wedge Company(z_1) ... \rightarrow error(x, z_1)$$

The case of an *at least* restriction applying for every interval that a fluent property holds is handled as follows: every time instant related with an interval that the fluent property in question holds, is also related with the object of the corresponding class. For example if a *worksFor* property holds for each *Employee* then all temporal instants that the property holds are detected by asserting an *OccuresAt* relation between the *Employee* and the time instants. There are three sub-properties of *OccuresAt* namely, *duringAt, endsAt* and *startsAt* indicating that the time instant is *during*, at the *start* or at the *end* of an interval that the property holds. For each such sub-property assertion, an SWRL rule is applied.

The second step is to check if for each time instant that the property *occuresAt* for an individual, the restriction at hand is satisfied. For example, in the case of an *at least 2* restriction applied on individuals of a class, a time instant that the fluent holds satisfies the restriction iff (a) the fluent *startsAt* this point and also another point that *equals* the point in question *startsAt* this fluent (b) the fluent *endsAt* the point and another point that *equals* the point in question *endsAt* this fluent (c) the fluent holds *duringAt* the point in question and also this point is *into* a second interval that the property holds or is *equals* both the end of such an interval and the beginning of another. Each case is implemented as an SWRL rule.

Each instant indicating the end or the start of an interval is a distinct individual from other points even if they represent the same time point (e.g., if a point

is the end of an interval and the beginning of another then two points declared as *equal* must be asserted). Finally, as in the case of the *at least* restriction in the first interpretation, since an inconsistency can't be detected by OWL reasoners, a SPARQL query is issued in order to detect individuals for which the restriction is not satisfied yet.

All rules under both interpretations involve a time consuming selection of all possible subsets of individuals and intervals. Therefore, expressing restrictions using SWRL may become time consuming. Specifically, rules for imposing a cardinality of *at-least* or *at most* n involves selection of all combinations of n among k timeslices (or reified relations) (where k is the number of temporal individuals in the ontology) it is not scalable for large values of n. In the case of reification or N-ary relations, cardinality constraints are expressed accordingly, using appropriate adjustments on classes and on properties of involved objects. To the best of our knowledge this is the only known solution to the problem of cardinality restriction checking on temporal representations.

Besides representation of cardinality and value restrictions and adjustments of domains and ranges the following object property characteristics are redefined using 4D-fluents as follows:

- *Functional*: It is handled as an at most 1 unqualified cardinality restriction.
- *Inverse Functional*: The inverse property is handled as an at most one unqualified cardinality restriction.
- *Symmetric*: The fluent property is *symmetric* too, thus the symmetry axioms applies on the interval that the involved timeslices exist.
- *Asymmetric*: This is handled as a form of cardinality restriction where the same property can't hold for interchanged subjects and objects for timeslices that share an overlapping interval.
- *Equivalent*: The fluent properties are equivalent too.
- *Reflexive*: The fluent property is reflexive too, thus when a timeslice has the property for an interval it is also the subject of the property for this interval.

- *Irreflexive*: This is handled as an cardinality restriction; two timeslices of an object can not be related with the property in question if their intervals overlap.
- *Subproperty*: subproperty axioms apply for the fluent properties with the intended semantics.
- *Transitive*: Fluent properties can be safely declared transitive since related timeslices must have equal intervals (by the definition of the 4D-fluent model) and for these intervals the transitivity is applied.

Datatype properties have fewer characteristics (i.e., *subproperty, equivalence disjointness, functional*) and they are handled as is the case of object properties. In case of the N-ary relations the above adjustments must take into account the different objects involved (i.e., *Events* instead of *timeslices*).

Chapter 5

Querying in SOWL

5.1 Introduction

Representing spatial and temporal information in OWL using mechanisms such as 4D-fluents or N-ary relations requires introducing additional objects that complicate the ontology as well as querying over the represented information. Querying information using query languages such as SPARQL leads to complicated queries and requires that the users be familiar with the underlying representation mechanism. Adding spatial and temporal operators that hide the underlying representation from the end user is an important issue to deal with. Two query languages based on SQL syntax and SPARQL syntax respectively are developed and they are presented in the following.

The main goal of spatio-temporal query languages is to maintain simplicity of expression while the time and space dimension is added. In particular, desirable features of temporal query languages include, temporal upward compatibility (i.e., conventional queries and modifications on temporal relations act on the current state), point and interval-based views of data, expressive power and ease of implementation.

5.2 TOQL

TOQL (Temporal Ontology Query Language) is an SQL-like language for OWL, supporting the basic structure of SQL (SELECT - FROM - WHERE) and treats classes and properties of an ontology almost like tables and columns of a database. TOQL supports queries over quantitative and qualitative spatio-temporal information in 4D-fluent ontologies. Nevertheless, TOQL syntax is independent of the underlying ontology representation mechanism. In ontologies the basic terms are classes (also named concepts) and properties (object or datatype). Classes represent concepts of the world. Properties represent relations between two concepts or between a concept and a value. Properties relating two classes (concepts) are referred to as object properties, while properties relating a class with a value are referred to as datatype properties. As an example of object property consider the relation between the *Company* and the *Employee*. These two classes are connected with the object property *hasEmployee*. As an example of datatype property consider the name of an *Employee*. Class *Employee* is connected with a name (string value) with datatype property *employeeName*.

TOQL not only uses SQL-like clauses and a similar syntax, but also treats ontologies almost like relational databases. Tables representing concepts correspond to classes and tables representing relations correspond to object properties. Attributes correspond to datatype properties. In addition, 1:1 and 1:N relations correspond to object properties. Table 5.1 summarizes the mapping between database relations and ontology concepts used by TOQL.

Relational Database	**Ontology**
Table representing concept	Class
Table representing N:N relation	Object Property
1:N or 1:1 relation	Object Property
Attribute	Datatype Property

Table 5.1: Mapping between database relations and ontology concepts.

In TOQL, classes are declared in FROM clauses just like SQL handles tables. To access a datatype property of a class, the name of the class is followed by a

dot (".") and the name of the datatype property, just like SQL handles tables and attributes:

ClassName.DatatypePropertyName

To access object properties (properties connecting two classes), the name of the domain class is followed by a dot (".), the name of the object property, double dot (":") and finally the name of range class:

DomainClassName.objectPropertyName:RangeClassName

The following query can be used to access the names of companies producing products called "x" in the ontology of Figure 3.1:

SELECT Company.companyName
FROM Company, Product
WHERE Company.produces:Product
AND Product.productName LIKE "x"

5.2.1 TOQL: Syntax

TOQL takes into account differences in the type of relations in the two representations and also supports temporal operators. The following temporal operators are supported: BEFORE, AFTER, EQUALS, MEETS, METBY, OVERLAPS, OVERLAPPEDBY, DURING, CONTAINS, STARTS, STARTEDBY, ENDS, ENDEDBY and operators AT(time point) and AT(time point, time point). Allen operators [1] compare datatype properties e.g., *A.B like "x" before C.D like "y"* (all keywords are *case insensitive*). The language also supports additional functionalities such as LIMIT, OFFSET that limit the number of answers to be returned, and nested queries. A detailed description of the language's syntax can be found in [9] and it's implementation is presented in [8]. The spatial extensions and the support for qualitative relations are presented in [12; 14]. TOQL supports most of an SQL language syntax and clauses, the most important of them being:

- **SELECT**: specifies the object property values or class of objects to be returned.

- **FROM**: declares the class or classes to query from. Always follows SELECT.
- **WHERE**: includes logic operations and comparisons between object property values that restrict the number of answers returned by the query. Always follows FROM.

TOQL supports the following operators:

- **AS**: renames a class (in a FROM clause) or a property (in a SELECT clause). Renaming of a class allows using more than one instances of a class in a query (e.g., FROM Company AS C1, Company AS C2). Renaming of a property allows changing its name in the results (e.g., SELECT Company.companyName AS Name).
- **AND**: connects two expressions involving properties (datatype or object properties) in WHERE and returns objects satisfying both expressions.
- **OR**: connects two expressions involving properties (datatype or object properties) in WHERE and returns objects satisfying at least one of them.
- **LIKE**: checks whether a datatype property value matches a specified string in WHERE. Comparison is case sensitive.
- **LIKE "string" IGNORE CASE**: checks whether a datatype property value matches a specified string ignoring case.

Table 5.2 summarizes TOQL syntax:

Syntax
SELECT *Property(ies) OR Class(es)* AS *Literal(s)*
FROM *Class(es)* AS *Literals*
WHERE *Condition(s)*

Table 5.2: Generic TOQL syntax.

There are operation clauses connecting two (or more) queries in a nested query:

- **MINUS**: returns query results retrieved by the first operand, excluding results retrieved by the second operand.
- **UNION**: returns the union of results returned by both operands. Duplicate answers are filtered out.
- **UNION ALL**: returns the union of results returned by both operands. Duplicate answers are not filtered out.
- **INTERSECT**: returns the intersection of results retrieved by both operands.
- **EXISTS**: this is a unary operator that has a nested SELECT-query as its operand. The operator is an existential quantifier that succeeds when the nested query has at least one result.
- **ALL**: this is an operator that has a nested SELECT-query as one of its operands. It always follows a comparison operator (i.e., "=", "!=", "<", ">", "<=", ">="). It indicates that for every value of the nested query the comparison must hold.
- **ANY**: has a nested SELECT-query as one of its operands. It always follows a comparison operator (i.e., "=", "!=", "<", ">", "<=", ">="). It indicates for at least one value of the nested query the comparison must hold.
- **IN**: has a nested SELECT-query as one of its operands. Allows set membership checking. The set is defined by the nested SELECT-query.

Table 5.3 summarizes TOQL syntax with operator clauses:

Case 1	Case 2	Case 3	Case 4
Query MINUS Query	Query UNION Query	Query UNION ALL Query	Query INTERSECT Query
Case 5	**Case 6**	**Case 7**	**Case 8**
SELECT ... FROM ... WHERE EXISTS (QUERY)	SELECT ... FROM ... WHERE ... CO[1] ALL (Query)	SELECT ... FROM ... WHERE ... CO[1] ANY (Query)	SELECT ... FROM ... WHERE ... IN (Query)

Table 5.3: TOQL syntax with operator clauses.

5.2.2 Dealing with Time

TOQL is a high level language, hiding from the users the implementation of time at the ontology level. A temporal ontology consists of (a) the static part where application classes, properties and their instances are defined and (b) the dynamic part where the additional temporal classes (i.e., classes *TimeSlice*, *TimeInterval*), properties and instances of the above temporal classes and fluent properties are defined (i.e., *tsTimeSliceOf*, *tsTimeInterval*). TOQL automatically determines references to time related information.

To do this, TOQL:

- Retrieves the time slices associated with a class of the static ontology.
- Determines whether a property (object or datatype) in the query is a fluent property (i.e., a property that connects time slices or a time slice with a datatype) or not (i.e., a property that connects "static" classes or a "static" class with a datatype).
- Uses the ontology's dynamic part to answer the query, if a property specified by the query is a fluent one.

[1]CO: comparison operator can be any of "=", "!=", "<", ">", "<=", ">="

- Uses the ontology's static part to answer the query, if a property specified by the query is not a fluent one.

TOQL does not require that the users be familiar with the representation of time in ontologies. As an example consider the *DEn* Ontology of Figure 3.1. Typically to retrieve companies that hired employees, one should be familiar with the 4D-fluent mechanism and ask for all time slices (instances) of class *Company* and all time slices of class *Employee* and then query on the object property *hasEmployee* that connects those instances. In TOQL (without implementing the high level functionality described above), this is expressed as:

SELECT Company.companyName
FROM Company, Employee, TimeSlice AS T1 ,
TimeSlice AS T2
WHERE T1.tsTimeSliceOf:Company AND
T2.tsTimeSliceOf:Employee AND T1.hasEmployee:T2 AND
Employee.employeeName LIKE "x"

This is a rather complicated expression and requires the user to be familiar with the implementation of time at the level of the ontology (the 4D-fluent method in this work). However, this is not necessary in TOQL and the same query can be expressed as:

SELECT Company.companyName
FROM Company, Employee
WHERE Company.hasEmployee:Employee
AND Employee.employeeName LIKE "x"

The second query is much more easy to write than the first one. Notice that the object property *hasEmployee* is treated like its domain class *Company* and its range class *Employee*. Query execution of TOQL queries relies on the intermediate translation of these queries into an equivalent SeRQL query[8; 9; 14]. For example the above TOQL query is translated into the following SeRQL query :

SELECT companyName_Company
FROM {Company} ex1:companyName {companyName_Company},
{Company} rdf:type {ex1:Company},
{Employee} rdf:type {ex1:Employee},
{CompanySlice_1} rdf:type {ex1:TimeSlice},
{CompanySlice_1} ex1:hasEmployee {EmployeeSlice_1},
{Employee} ex1:employeeName {employeeName_Employee},
{CompanySlice_1} ex1:tsTimeSliceOf {Company},
{EmployeeSlice_1} ex1:tsTimeSliceOf {Employee}
WHERE employeeName_Employee Like "x"
USING NAMESPACE
ex1=http://www.intelligence.tuc.gr/ontologies/2008/4/Den.owl

5.2.3 Allen Operators

In TOQL, the implementation of ALLEN operators correspond to comparisons between fluent properties. Fluent properties connect time slices and time slices are associated with time intervals. Consequently, the implementation of Allen operators correspond to comparisons between time intervals. The following operators are supported in TOQL: BEFORE, AFTER, MEETS, METBY, OVERLAPS, OVERLAPPEDBY, DURING, CONTAINS, STARTS, STARTEDBY, ENDS, ENDEDBY and EQUALS, representing the corresponding relations holding between two time intervals.

The following TOQL query retrieves the name of the company that hired employee "x" and *then* employee "y":

SELECT Company.companyName
FROM Company, Employee AS E1, Employee AS E2
WHERE Company.hasEmployee:E1 BEFORE Company.hasEmployee:E2
AND E1.employeeName like "x" AND E1.employeeName LIKE "y"

5.2.4 AT, TIME Operators

TOQL also introduces clause "AT" which compares a fluent property (i.e., the time interval in which the property is true) with a time period (time interval) or time point:

- **AT(time point)** operation returns true if the time interval holds true at the time specified.
- **AT(start time point, end time point)** operation returns true if the time interval holds true for *all* the time interval.

The following TOQL query retrieves the name of the company employee "x" was working for, from time=3 to time=5:

SELECT Company.companyName
FROM Company, Employee
WHERE Company.hasEmployee:Employee AT(3,5)
AND Employee.employeeName LIKE "x"

Because TOQL is independent of the mechanism implementing time, there is no way to directly access class *TimeInterval* (i.e., the class holding values of time). In order for TOQL to return values of time, the keyword TIME is introduced. It follows datatype or object properties and can be used only in SELECT. It returns the start and end time point (if any) in which the property holds true (the time interval in which the property is true). If no end point exists, it returns only its start point. As an example, the following TOQL query retrieves the time for which a company had employee "x":

SELECT Company.hasEmployee.TIME
FROM Company, Employee
WHERE Company.hasEmployee:Employee AND
Employee.employeeName LIKE "x"

5.2.5 Special Cases

This section describes TOQL special features. These are related to the way TOQL deals with Class keys, wildcards (*) and Scope.

Dealing with keys: In relational databases each tuple is uniquely characterized by a key. A key can refer to more than one attributes (compound key). Consider a relational database that has the table Company and that this table uses the attribute ID as key. To access this key, in SQL, a user should write:

SELECT Company.ID

In OWL, each class instance and each property have a unique name. This unique name is considered to be equivalent to the unique key of relational databases. The difference is that this unique name is not an ordinary datatype property, and so it can not be accessed by writing the name of the class followed by a dot "." and the datatype property. In TOQL, the (unique) name of a class instance is accessed using the name of the class itself (without reference to a property). For example, to access the unique name of a company we write:

SELECT Company

Dealing with wild cards (*): In TOQL, wild cards can be used only in SELECT. In SQL the presence of wildcard in SELECT implies that all the columns of all the tables declared in clause FROM will be returned. If the wild card follows a table (*tableName*.*), all the columns of the specific table will be returned. In TOQL the presence of wild card in SELECT implies that all the datatype properties of *all* the classes declared in FROM will be returned. If the wild card follows a class, the datatype properties of the specific class will be returned. Notice that the class unique name is not returned (only its datatype properties are returned). The following query retrieves companies producing product with unique name "x", as well as the product's name.

SELECT *
FROM Company, Product
WHERE Company.hasProduct:Product
AND Product LIKE “x”

Dealing with scope: TOQL supports set combination operations in queries as well as nested queries. Both set operations and nested queries imply that a TOQL query may be composed of more than one sub-queries. Each sub-query has its own class declarations, class and property usage and this introduces the need for the handling of scopes.

Queries combined by set operators have different scopes. Classes declared in any of them are local to this query and are not visible to the others. The following query retrieves names of *“Company_1”* and also names of *“Company_2”* from the *DEn* Ontology:

SELECT C1.companyName
FROM Company As C1
WHERE C1 like “Company_1”
UNION
SELECT C1.companyName
FROM Company As C1
WHERE C1 like “Company_2”

This TOQL expression specifies two separate queries combined by the set operator UNION. Each sub-query has a different scope: classes declared in the first sub-query are not visible to the second one. Even if the same class is used by the second sub-query, it must be redeclared.

In TOQL, a nested query inherits all the classes declared in the query it is nested into. A nested query can use these classes, but cannot (re)declare any of them. The following nested TOQL query (a second query follows clause ANY) retrieves products whose price is at least 10 and not smaller than than the price of any other product. Both sub-queries use class *Product* but with different names (*P1* and *P2* respectively) otherwise a semantic error will be reported.

SELECT P1
FROM Product As P1
WHERE P1.price >= 10 AND NOT
P1.price < ANY
(**SELECT** P2.value **FROM** Product As P2)

5.2.6 Spatial Operators

In [14] our previous work on TOQL [9] for querying temporal information in OWL was extended to handle spatial and spatio-temporal information. In addition to the existing set of temporal operators (i.e. the AT and Allen operators) the language is enhanced with spatial operators for handling both, spatial and temporal relations, thus the IN_RANGE and all RCC8 and directional relations are supported by corresponding operators.

Spatial operators refer to topological or directional spatial relations represented in the underlying ontology. The result of applying spatial operators are locations qualifying the expressions specified by the query. Locations are assessed using their names (although the user can issue queries addressing the underlying quantitative representation using coordinates, but formulating such queries requires that the user be familiar with the underlying spatio-temporal representation). The following spatial operators are supported:

- NORTH_OF
- NORTHEAST_OF
- EAST_OF
- SOUTH_OF
- WEST_OF
- SOUTHWEST_OF
- SOUTHEAST_OF
- INTO

- OUTSIDE_OF
- SAME_LOCATION_AS
- BORDERING
- OVERLAPPING
- CONTAINS
- INTERNALLY_BORDERING
- CONTAINS_AND_BORDERING
- IN_RANGE.

They correspond to the eight directional relations in Fig. 2.3, the RCC8 relations in Fig. 2.2 and one operator (range) involving distance information. Spatial operators are issued in WHERE, followed by a string denoting the location name according to the pattern <SPATIAL OPERATOR> <STRING>. For example the following query retrieves the name of the company located north of a given location :

SELECT Company.companyName
FROM Company
WHERE Company NORTH_OF "Attica"

Queries involving the IN_RANGE operator have the following syntax: IN_RANGE <Comparison operator> <Number> OFF <String> where the string denotes location name. The following query retrieves the names of employees working for companies located in distance greater than 100Km away from "Athens":

SELECT Employee.employeeName
FROM Company, Employee
WHERE Company IN_RANGE >100 OFF "Athens" AND
Company.hasEmployee:Employee

5.2.7 Spatio-Temporal Queries

All spatial and temporal operators can be combined in TOQL. For example the following query retrieves the name of the company located *north* of a given location *at* a specific instant of time:

SELECT Company.companyName
FROM Company
WHERE Company NORTH_OF "Attica" AT(5)

The following query retrieves the names of employees working for companies located in distance greater than 100Km away from "Athens" during the time interval [5,10]:

SELECT Employee.employeeName
FROM Company, Employee
WHERE Company IN_RANGE >100 OFF "Athens" AND
Company.hasEmployee:Employee AT(5,10)

5.3 SOWL Query Language

The SOWL query language relies on the idea of extending SPARQL (rather than SQL as TOQL does) with spatio-temporal operators following the examples of [109]. Compared to the work referred to above, SOWL has the following two advantages (a) supports both spatial and temporal operators and (b) is capable of querying over both quantitative and qualitative spatial and temporal information. Similarly to TOQL, SOWL syntax is independent on the underlying ontological implementation. The working version of SOWL is implemented on top an N-ary relations representation, although a representation based on 4D-fluents has been implemented as well.

5.3.1 SOWL Query Language Syntax

The set of spatial and temporal operators of SOWL are discussed in the following. Similarly to TOQL, SOWL supports Allen, AT, topological and directional operators for both spatial and temporal information (i.e., time instants or points, temporal and spatial intervals).

Query execution relies on the intermediate translation of these operators into an equivalent SPARQL query. This translation depends on the underlying ontology representation [18; 107]. The motivation of extending SPARQL with new operators (rather than using SPARQL with its existing operators for querying temporal information) is that this approach offers additional flexibility in expressing temporal queries concisely, while ensuring independence of the temporal expressions from the peculiarities of the underlying ontological representation (i.e., query syntax is the same regardless of temporal representation).

The language is enhanced with spatial operators for handling both, spatial and temporal relations, thus all RCC8 and directional relations are supported by corresponding operators. The query language inherits SPARQL syntax and semantics (e.g., queries over non-temporal information are expressed in SPARQL) with the addition of the temporal operators. The query language also inherits the computational complexity of SPARQL [83] since it can be considered as a compact form of SPARQL for spatio-temporal queries. Table 5.4 summarizes SOWL Query Language syntax:

Syntax
SELECT *Variable(s)*
WHERE { *Condition(s)* Spatial or Temporal Operator(s)
AND *Condition(s)* }

Table 5.4: Generic Query Language syntax.

5.3.2 Temporal Operators

SOWL introduces clauses “AT”, “STARTS_AT”, “ENDS_AT”, “SOMETIME_AT”, “ALWAYS_AT” for comparing a fluent property (i.e., the time interval during which the property holds true) with a time period (time interval) or time point.

Such queries return fluents holding true at the specified time interval or point. Queries involving static properties (properties not changing in time) are issued as normal queries applied on the static part of the ontology. The following query applies on a fluent property:

```
select distinct ?x ?y
where{ ?x ex1:hasEmployee ?y.
?x ex1:companyName "C1"
}
```

SOWL queries are translated into equivalent SPARQL queries. For example when a query involving a fluent property (such as *hasEmployee*) is issued, the query is translated into a SPARQL query (applied on the 4D-fluents or N-ary representation). The above query in translated into the following equivalent SPARQL query (4D-fluents representation):

```
select distinct ?x ?y
where {
?_timeSlice_0 ex1:tsTimeSliceOf ?x.
?_timeSlice_0 ex1:tsTimeInterval ?_interval_0.
?_timeSlice_0 ex1:hasEmployee ?_timeSlice_1 .
?_timeSlice_1 ex1:tsTimeSliceOf ?y.
?_timeSlice_1 ex1:tsTimeInterval ?_interval_0.
?x ex1:companyName "C1" }
```

Operator "SOMETIME_AT" returns fluents holding for intervals that share common time points with the interval in question. The following example illustrates a query involving the "SOMETIME_AT" operator:

```
select ?x ?y
where{
?x ex1:hasEmployee ?y SOMETIME_AT
"2007 − 02 − 01T00 : 00 : 00Z", "2007 − 02 − 05T00 : 00 : 00"}
```

Operator "ALWAYS_AT" returns fluents holding for intervals which contain all points of the interval in question. For example the following query retrieves the name of the company "x" that employee "y" was always working for, at the specified interval:

select ?x ?y
where{
?x ex1:hasEmployee ?y ALWAYS_AT
"$2007-02-01T00:00:00Z$", "$2007-02-05T00:00:00$"}

The "AT" operator returns fluents holding at intervals that contain the time point in question. Notice that, integrating interval and instance representations allows for inferring relations between points and intervals in addition to relations between intervals. For example, the following query retrieves the name of the company "x" that employee "y" was working for, at the specified date:

select ?x ?y
where{
?x ex1:hasEmployee ?y AT"$2007-02-05T00:00:00$"}

The reasoning mechanism allows for retrieving results that implicitly satisfy the conditions imposed by the query. For example, if company C_1 has Employee E_1 for the interval $interval_1$, that spans throughout year 2007, and Employee E_2 for an unknown $interval_2$ that *contains* $interval_1$, then by using the point based reasoning mechanism it can be inferred that both $interval_1$ and $interval_2$ contain the time point in the previous query. Thus the results: C_1,E_1 and C_1,E_2 will be returned.

The following Allen operators are also supported: BEFORE, AFTER, MEETS, METBY, OVERLAPS, OVERLAPPEDBY, DURING, CONTAINS, STARTS, STARTEDBY, ENDS, ENDEDBY and EQUALS, representing the relations holding between two time intervals specified (operators BEFORE and AFTER support temporal points as well). In this work, relations can be quantitative (i.e., involving specific temporal instants or intervals) or qualitative (i.e., the exact values of temporal instants or intervals are unknown or not specified). For example

the following query retrieves the name of the company "x" that employee "y" was working for, before the specified date:

select ?x ?y
where{
?x ex1:hasEmployee ?y before"$2007-02-08T00:00:00$"}

the following query retrieves the name of the company "x" that employee "y" was working for, before *Employee2* worked for *Company1*:

select ?x ?y
where{
?x ex1:hasEmployee ?y before
ex1:Company1 ex1:hasEmployee ex1:Employee2}

The following query retrieves the name of the company "x" that employee "y" was waking for, during the specified interval:

select ?x ?y
where{
?x ex1:hasEmployee ?y
during"$2007-02-01T00:00:00Z$", "$2007-02-05T00:00:00$"

Finally the following query retrieves the names of employees of company "C1" and the intervals they where employed by this company (note that the interval is a variable and not a specified date in this example):

select distinct ?x ?y ?z
where{
?x ex1:hasEmployee ?y AT(?z).
?x ex1:companyName "C1"
}

The SOWL query language has been implemented on top of the N-ary relations ontological representation. However, similarly to TOQL, SOWL syntax

is independent of ontological representation. Notice that, an earlier version of SOWL was implemented using the 4D-fluents model. The language itself, it's syntax and the set of supported operators, remain identical (only the translation changes to take into account the peculiarities of the underlying representation). The following query applied on the N-ary representation involves a fluent property:

```
select distinct ?x ?y
where{
?x ex1:hasEmployee ?y.
}
```

which is translated to:

```
select distinct ?x ?y
where {
?x ex1:hasEmployee ?_event_0.
?_event_0 ex1:hasEmployee ?y.
?_event_0 nary:atTime ?_interval_0.
}
```

Note that the translation introduces objects of class *Event* representing relations with temporal extends while the 4D-fluents based representation involves *Timeslice* objects.

5.3.3 Spatial Operators

Similarly to temporal, spatial operators can retrieve both qualitative and quantitative spatial information. Spatial operators are basically dynamic predicates that are identified by the system and treated separately. In order to use a quantitative parameter in conjunction with a spatial operator in a triplet, we must always use a spatial operator as predicate and use the quantitative parameter as the object of the triplet, to declare a value and connect it to the subject.

Specifically, the basic spatial query patterns are two :

- subject spatial-Operator object : The spatial operator is treated like a "special" dynamic predicate.
- subject spatial-Operator Quantitative-Spatial-Parameter : The Quantitative Spatial Parameter here declares a spatial location by using points to describe it. The only Quantitative Spatial Parameter implemented right now is *POINT(x,y)* (Although in a projection based representations using MBRs two points specifying the MBR can be also used).

We use two kinds of spatial operators : Directional (those that declare the directional relation of two objects in space) and Topological (those that declare the relative location of the objects surface).

- Directional : (Assume that we have two spatial objects A and B that are connected with the following spatial operators)
  - Nof : A is North of B
  - Sof : A is South of B
  - Eof : A is East of B
  - Wof : A is West of B
  - NWof : A is North and West of B
  - NEof : A is North and East of B
  - SEof : A is South East of B
  - SWof : A is South West of B
  - SameX : A and B have the same 'x' coordinate (e.g A(2,3) , B(2,5))
  - SameY : A and B have the same 'y' coordinate (e.g A(2,3) , B(5,3))
  - SameXY : A and B have the same center (e.g A(2,3) , B(2,3))
- Topological : (Assume that we have two spatial objects A and B that are connected with the following spatial operators)
  - Contains (NTTPi): Object A contains object B (not touching).
  - ContainsTouches (TTPi): Object A contains and touches object B.

- Disjoint (DC) : Objects A and B are not one inside another and not touching.
- Touches (EC) : Objects A and B touch each other from outside.
- Equals(EQ) : A and B are the same size and occupy the same location on space.
- IntoTouches(TPP) : Object A is contained and touched by object B.
- Whithin(NTTP): Object A is contained by object B
- Overlaps(PO): Objects overlap.

Example Query:

```
select ?x ?y
where{
?x spatial:Nof ?y
}
```

SPARQL Translation :

```
select ?x ?y where {
?x spatial:locatedAt ?_location_0.
?_location_0 spatial:hasGeometry ?_geometry_0.
?y spatial:locatedAt ?_location_1.
?_location_1 spatial:hasGeometry ?_geometry_1.
?_geometry_0 spatial:Nof ?_geometry_1
}
```

Apart from using operators between triplets the language is capable of answering queries containing quantitative values of type “point” (e.g., spatial coordinates). The query pattern for a triplet of this kind is : *Subject Spatial-Operator POINT(x,y)* where x,y are floats declaring points on space.

Example Query :
select ?x
where{
?x spatial:Nof POINT([10.8,9.2])
}

5.3.4 Spatio-Temporal Queries

A combination of spatial and temporal operators is possible in SOWL. Both a spatial and a temporal operator are used within the same expression (i.e., a triplet).

There are 3 query triplet patterns for using both spatial and temporal operators:

- Subject spatial-Operator Object Temporal-Operator: This triplet declares that a spatial event takes place at a time instant/interval
- Subject spatial-Operator Quantitative-Spatial-Parameter Temporal-Operator: This is to declare that an event at a quantitative defined spatial point happens at a time instant/interval (E.g., Car North-Of Point(3,4) AT(timepoint))
- Subject spatial-Operator Object Quantitative-Spatial-Parameter Temporal-Operator Subject2 spatial-Operator Object2 Quantitative-Spatial-Parameter: This declares a time-relation between two spatial events (e.g car_A is North-Of car_B before Car_c is North-Of Car_b)

Below we can see an example of combining the "At" temporal operator with the "Nof" spatial operator:

select ?x ?y
where{
?x spatial:Nof ?y AT("2007-02-05T00:00:00").
}

Below is an example of an Allen temporal operator connecting two triplets with spatial operators:

```
select ?x ?y
where{
?x spatial:Nof ?y before ?k spatial:EC ?z
}
```

Variables can also be used as arguments when combining spatio-temporal operators, as in the following example query:

```
select ?x ?y ?z
where{
?x spatial:Nof ?y STARTS(?z).
}
```

Chapter 6

Evaluation

6.1 Introduction

In the following, the efficiency of SOWL reasoning is assessed both, theoretically and experimentally. Reasoning calls for rules applying over spatial and temporal relations whose purpose is to infer implied relations, detect inconsistencies and ensure path consistency. The purpose of the theoretical analysis is to show that SOWL reasoning retains soundness, completeness and tractability over the supported sets of relations. We focus on the performance of SOWL reasoning rather than on the performance of SOWL query engine for the following reasons: (a) queries are translated to equivalent SPARQL queries (or SeRQL queries in the case of TOQL); Therefore, query translation is not optimized and (b) There is no indexing of the spatio-temporal data. Both (a) and (b) above are important issues for future research.

To evaluate the performance of SOWL experimentally we run several groups of experiments using synthetic (but realistic) data sets. The purpose of this set of experiments is, to demonstrate the run-time efficiency of the reasoner as a function of the size of the data set, investigate dependencies of the run-time efficiency of reasoning on the type and peculiarities of the underlying representations (i.e., 4D-fluents versus N-ary representation, point and interval-based representation, reasoning using SWRL versus reasoning applying on OWL axioms).

6.2 Theoretical Evaluation

The resulting OWL ontology is characterized by *SHRIF(D)* DL expressiveness (as presented in Section 2.2.1) and is decidable. Specifically, the required *SHRIF(D)* expressiveness is analysed as follows:

- S:ALC basic logic with transitive roles (e.g., SOUTH, BEFORE)
- H: Role hierarchy axioms (e.g., $Meets$ is a sub-property of interval relation)
- R: Role axioms such as disjoint relations (e.g., $Before$ and $After$ are disjoint properties)
- I: Inverse properties (e.g., $Meets$ and $MetBy$)
- F: Functional Properties (e.g., the *interval* of a *timeslice*).
- (D): Datatypes (e.g., *xsd:date*)

The required expressiveness of the proposed representation is within the limits of OWL 2 expressiveness. Notice that, an application might require additional expressiveness which can be evaluated once the application and its respective ontological representation has been analysed. This type of evaluation is outside the scope of the analysis discussed below.

Reasoning is achieved by employing DL-safe rules expressed in SWRL that apply on named individuals in the ontology A-box, thus retaining decidability while offering a sound and complete inference procedure for asserted temporal intervals. Furthermore, computing the rules has polynomial time complexity since tractable sets of relations are supported [81; 111].

Because any time interval can be related with every other interval with one basic Allen relation (basic Allen relations are mutually exclusive) between n intervals, at most $(n-1)^2$ relations can be asserted and this also holds in the case of instants. Furthermore, path consistency has $O(n^5)$ time worst case complexity (with n being the number of intervals or instants) and is sound and complete [92]. In the most general case where disjunctive relations are supported in addition to the basic ones, any interval (or instant) can be related with every other interval (or instant) 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. In the case of temporal instants (point algebra), qualitative relations on time instants form a tractable set [111] (i.e., a set of relations applying path consistency on this is a sound and complete method) if the relation $\neq$ (i.e., a temporal instant is before *or* after another instant) is excluded. Thus, the proposed reasoning method can be extended with disjunctive relations such as $\geq$ denoting that an instant is *after or equals* to another. Applying the *closure method* over Allen, RCC8 and directional relations the minimal tractable sets containing the basic relations consist of 29,49 and 33 relations respectively [16]. For these sets the required number of OWL axioms and SWRL rules are 983, 1439 and 964 respectively [16]. Reasoning over basic point algebra relations does not require additional relations and a total of 20 axioms are adequate for implementing path consistency [18].

The $O(n^5)$ upper limit referred to above is obtained as follows: At most $O(n^2)$ relations can be added in the knowledge base. At each such addition step, the reasoner selects 3 variables among n intervals (or points or regions) which corresponds to $O(n^3)$ possible different choices. Clearly, this upper bound is pessimistic, since the overall number of steps may be lower than $O(n^2)$ because an inconsistency detection may terminate the reasoning process early, or the asserted relations may yield a small number of inferences. Also, forward chaining rule execution engines employ several optimizations (e.g., the Rete algorithm employed at the SWRL implementation of Pellet as presented at [60]), thus the selection of appropriate variables usually involves fewer than $O(n^3)$ trials. Nevertheless, since the end user may use any reasoner supporting SWRL, a worst case selection of variables can be assumed in order to obtain an upper bound for complexity. Nevertheless retaining control over the order of variable selection and application of rules yields an $O(n^3)$ upper bound for path consistency [105].

6.3 Experimental Evaluation

To evaluate the performance of SOWL reasoning, we run several sets of experiments whose purpose is to demonstrate the run-time efficiency of the proposed reasoning approach as a function of the size of the data sets and its dependence

on the type and peculiarities of the underlying ontological representation. The efficiency tests of the SOWL reasoner require a spatio-temporal ontology. Because such an ontology is not available to us for these experiments, as a test-bed, we used data-set of 10 to 100 individuals generated randomly. In fact, these randomly generated data are used to populate 10 ontologies with random instances. Reasoning response times of both, a spatial and a temporal reasoner, are measured as the average over 10 runs. Pellet 2.1.2 running as a plug-in of Protege 4.1 beta was the reasoner used in the experiments. All experiments run on a home PC, with Intel Core 2 CPU at 2.13 GHz, 1 GB RAM, and Windows Vista.

6.3.1 Spatio-Temporal Reasoning

In this work, a point-based representation is adopted for handling both temporal (and spatial in the projection-based approach) instants and intervals. Relations between intervals are expressed as a function of relations between their end-points. A representation relying on intervals is also implemented. However, since the number of basic relations is 13 (Fig. 2.1) and because all possible disjunctions appearing in the supported tractable set must also be supported, the representation may become particularly involved. Notice also that, temporal instants, the same as semi-closed temporal intervals (i.e., intervals whose one of their end-points is undefined) cannot be represented efficiently in an interval-based representation. On the other hand, the interval based representation directly handles temporal intervals without the need of an intermediate translation to instants as in the point-based approach. Also, the definition of n intervals requires $2n$ points (although this number can be reduced in the case of temporal intervals with end-points in common). A point-based representation has the advantage that, if intervals share end-points then, a reduction in the number of instances is achieved, since using n points (depending on the number of shared endpoints) the number of derived intervals ranges from $n*(n-1)/2$ to $n/2$ in the worst case.

In the following experiment, we measure the performance of reasoning in the cases of both, an interval-based and a point-based representation and their performance is discussed. In both cases, n random instances with n random Allen or point relations defined between them were asserted, and reasoning times using

Pellet are measured. This measurement doesn't provide a direct comparison between interval and point-based reasoning since, up to $2n$ points may be required in order to define n intervals. Measurements of the time required by each approach for producing all inferred relations from a data set of random intervals or points are reported in Fig. 6.2 and Fig. 6.1. Fig. 6.2 and Fig. 6.1 illustrate the dependence of average reasoner response time as a function of the data-set size. Each point in the plot is the average over 10 measurements. Interval and instance assertions were selected to form a consistent set of axioms, thus the reasoning procedure did not terminate early due to inconsistencies. If an inconsistency is detected by the reasoner, the inconsistent random instance is removed. Reasoning using Pellet is terminated directly when an inconsistency is detected, thus measuring reasoning times calls for consistent assertions. The inconsistent assertions where replaced by other random assertions until a consistent set of 10 randomly generated ontologies for every instance set size was created.

The evaluation indicated that the point-based approach is faster, although the number of temporal instants involved is typically larger than the number if intervals from which they are derived. The point-based representation requires fewer rules applied on a largest set of points (in the worst case) compared to the interval-based representation which requires 29 relations (the minimal tractable subset containing basic relations) and 983 OWL axioms and SWRL rules as opposed to just 20 axioms in the case of point algebra. Overall, point-based representations are more flexible (facilitate representation of time instants and semi-closed intervals and they are space efficient involving only 3 relations) and are preferred. Also, if consistency checking is not needed (in case instance assertions doesn't contain conflicts -implied or direct) then, temporal properties need not be declared disjoint, and reasoning can be achieved using OWL role inclusion axioms that apply to the ontology TBOX instead of SWRL rules.

The point-based representation and reasoning method presented here applies also for spatial data in the 2 (or 3) dimensions with points represented by their x, y (and z) coordinates and by using relations $East$ and $West$ (applying on the x coordinates), *South, North* (applying on the y coordinates) and $Bellow$, *Above* applying on the z coordinates in place of the $before$ and $after$ temporal relations respectively.

Fig.6.1 represents the average time (for 10 random generated ontologies for every instance set size) required for reasoning using point algebra.

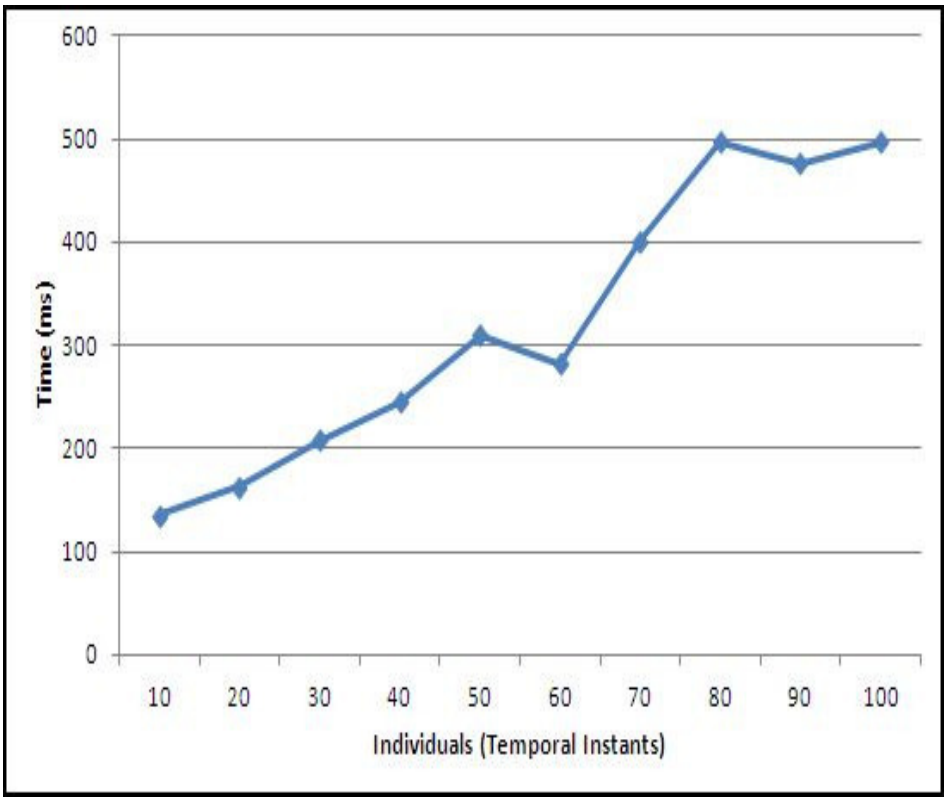

Figure 6.1: Average response time (over 10 runs) of temporal reasoning operating on an instant-based representation as a function of the number of temporal instants.

Fig.6.2 represents the average reasoning time required for reasoning using interval algebra.

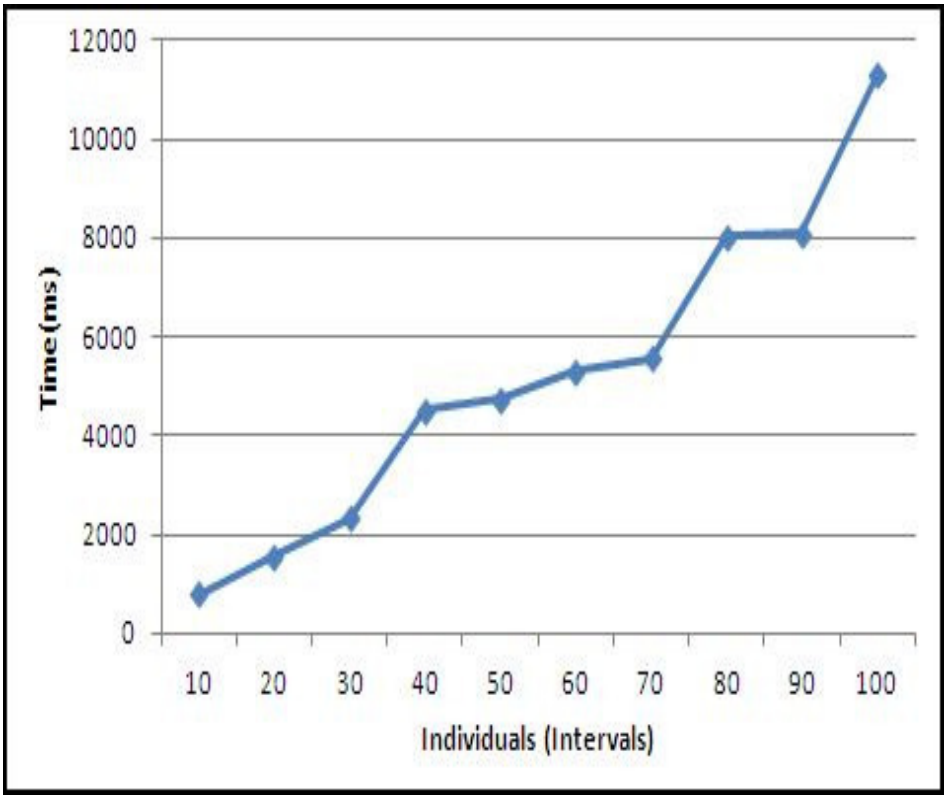

Figure 6.2: Average response time (over 10 runs) of temporal reasoning operating on an interval-based representation as a function of the number of temporal intervals.

Reasoning over instance-based representations is faster due to the small number of axioms involved and the smaller number of inferences over a set of random relations. This can be attributed to the fact that a larger percentage of entries in the composition table of Allen relations yield new relations (i.e., they do not yield the disjunction of all basic relations that does not provide information) as a result, in contrast to the composition table of point algebra.

In the following experiments we measure the performance of a spatial reasoner working over RCC8 and cone-shaped relations respectively. Fig. 6.4 and Fig. 6.3 illustrate the dependence of the reasoning response time as a function of the number of instances. Fig. 6.3 represents the average reasoning time (for 10 random generated ontologies for every instance set size) required for reasoning over cone-shaped directional relations, asserted between individuals representing

points. Fig. 6.4 represents the average time (for 10 random generated ontologies for every instance set size) required for reasoning over RCC8 relations. Individuals represent regions with topological relations between them.

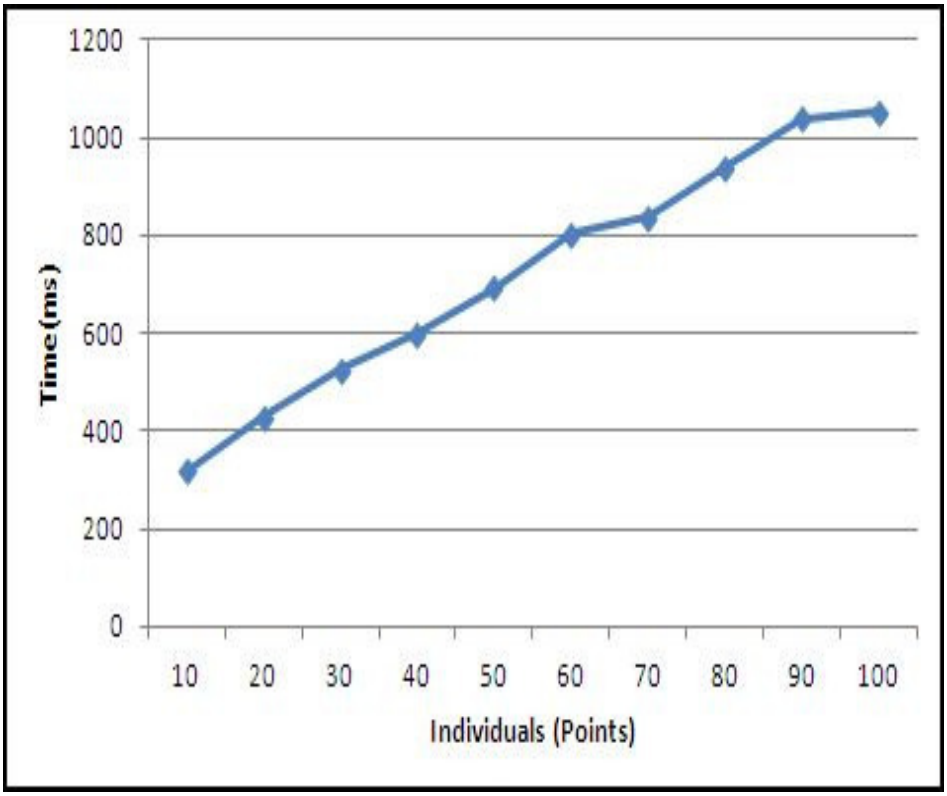

Figure 6.3: Average response time (over 10 runs) of spatial reasoning operating on cone-shaped directional relations as a function of the number of points.

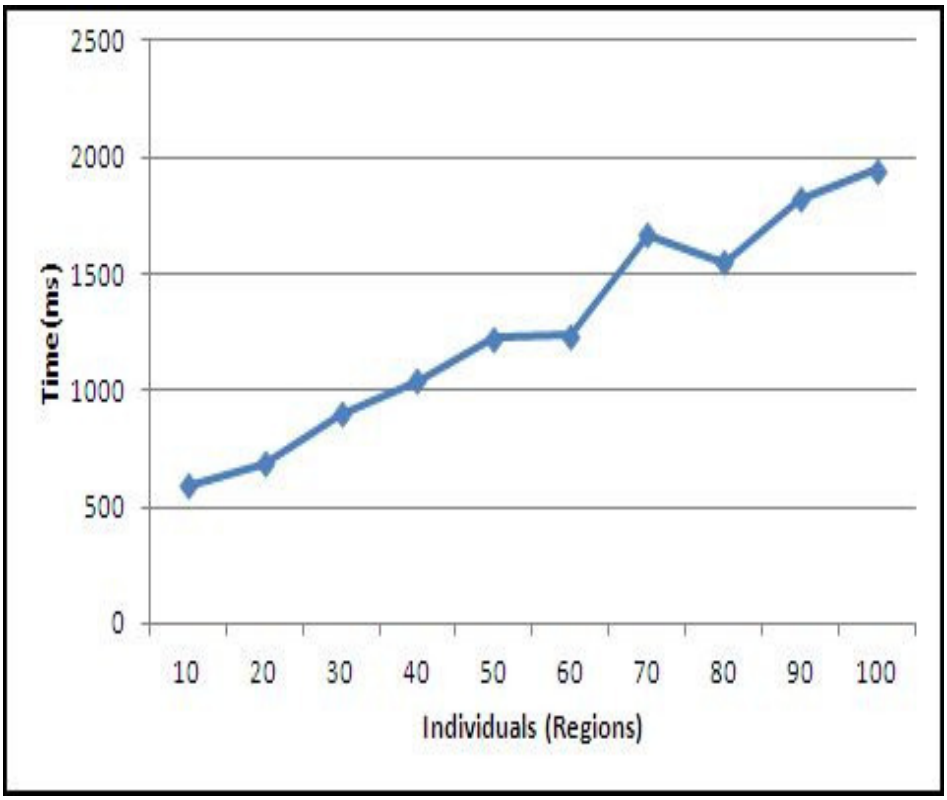

Figure 6.4: Average response time (over 10 runs) of spatial reasoning operating on RCC8 topological relations as a function of the number of regions.

In the following, measurements of time in the worst case are discussed. In this case from the n asserted relations n^2 relations can be inferred. This is achieved by asserting relations that (a) are not inconsistent and (b) applying reasoning over these relations yields relations between all pairs of individuals into the ontology. For example, if assertions of temporal instants define their ordering (e.g., by assigning points using ordered letters *a,b,...,z* and asserting a series of *after* relations for consecutive letters) then, from n assertions, n^2 consistent relations can be inferred. Although such a scenario is not as common in practice as the random selection of relations, as in the average case, it yields an upper limit (worst-case) for the reasoning time. Fig. 6.5 represents the worst case time required for reasoning over point algebra. Fig. 6.6 represents the worst case time required for reasoning over interval algebra.

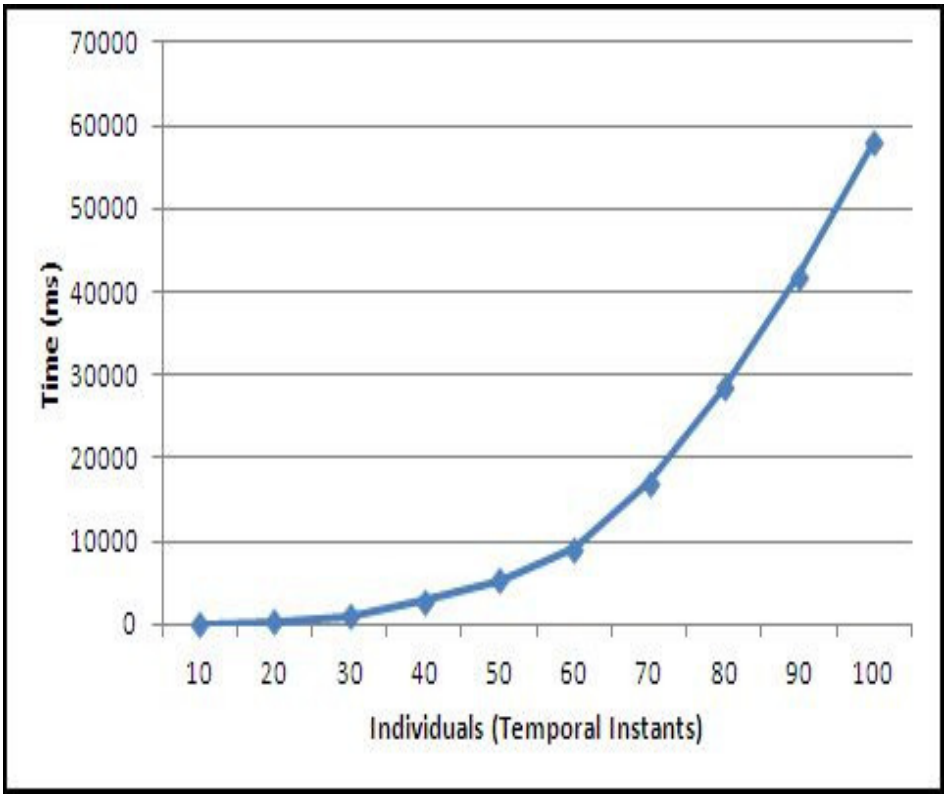

Figure 6.5: Worst case response time of temporal reasoning operating on an instant-based representation as a function of the number of temporal instants.

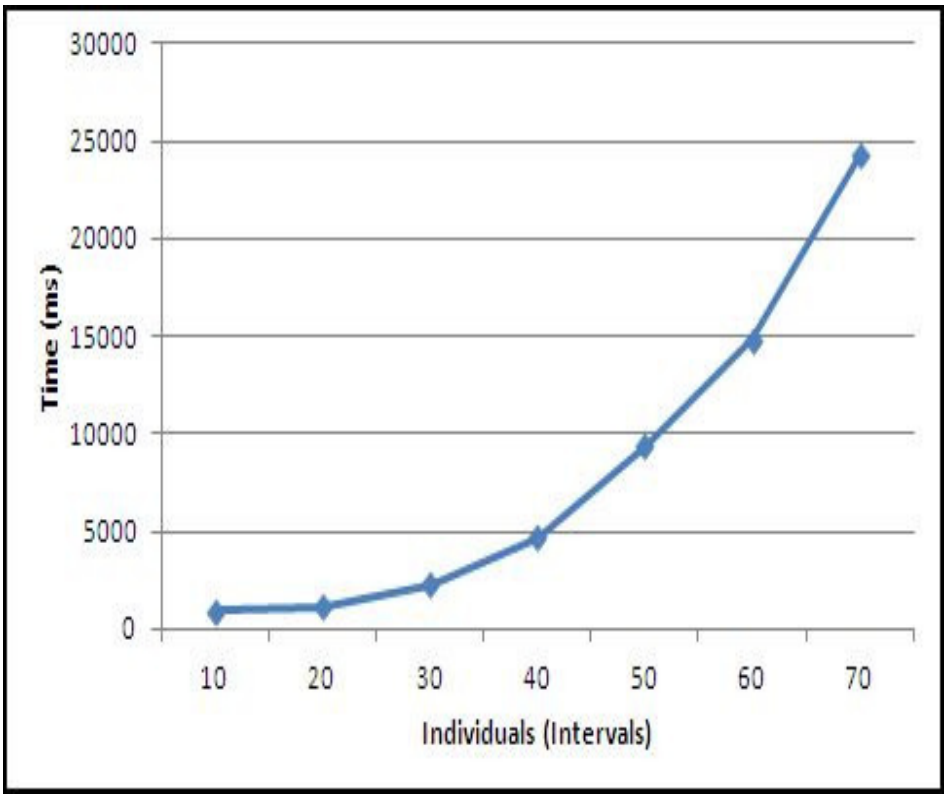

Figure 6.6: Worst-case response time of temporal reasoning operating on an interval-based representation as a function of the number of temporal intervals.

Fig. 6.7 represents the worst-case time required for reasoning over cone-shaped directional relations. Fig. 6.8 represents the worst case response time required for reasoning over RCC8 relations. As in the case of interval and point relations, the n asserted relations where selected in such a way as to produce a set of n^2 consistent relations. An ordering of regions such that each one is included into the previous one (i.e., $R_i\ NTPP\ R_{i-1}$) provides such a set for topologic relations. An ordered set of points, with all of them located towards the same direction in relation to the point preceding them (e.g., $P_i\ North-Of\ P_{i-1}$) is such a set for cone-shaped directional relations.

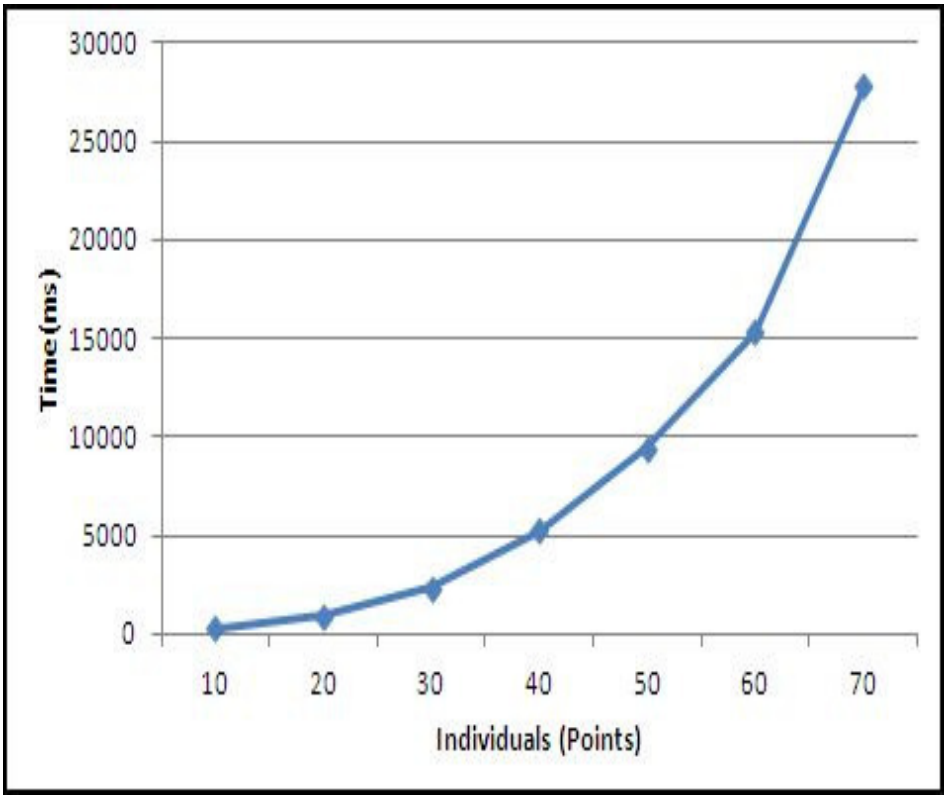

Figure 6.7: Worst-case response time of spatial reasoning operating on cone-shaped directional relations as a function of the number of points.

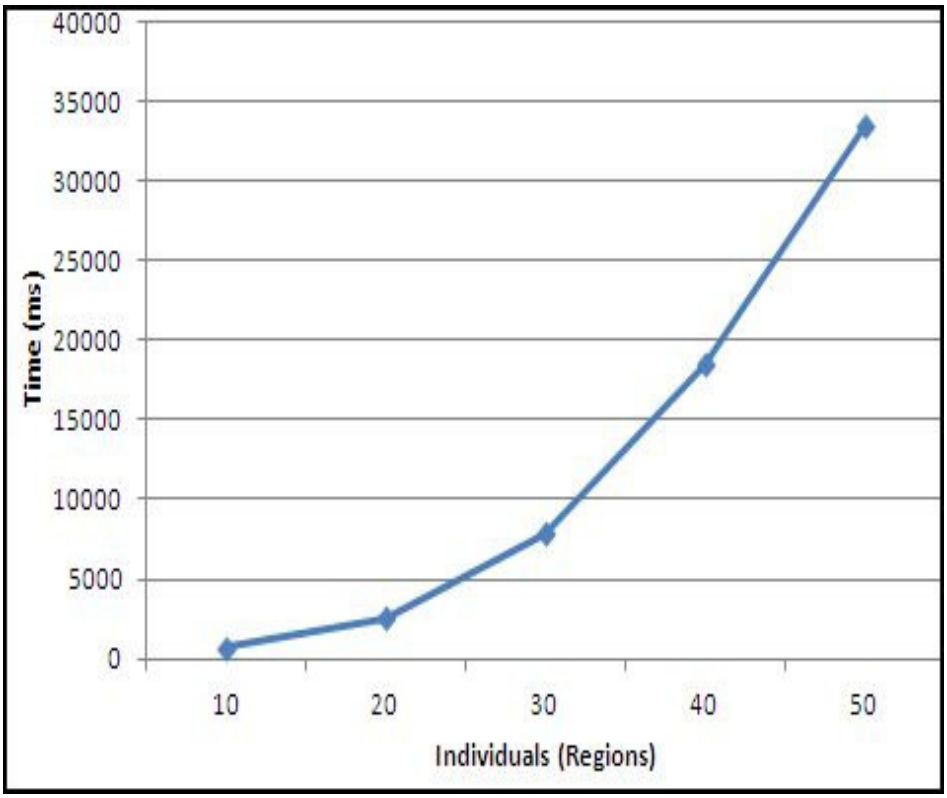

Figure 6.8: Worst-case response time of spatial reasoning operating on RCC8 topological relations as a function of the number of regions.

The number of inferences is the same for all sets of relations (in the worst-case). Therefore, reasoning response time depends solely on the number of axioms. Topological spatial relations involve more rules, thus require more time for reasoning. Notice that, besides the two cases presented here, spatio-temporal reasoning is a special case of constraint satisfaction problems and running times as well as the overall hardness of the reasoning task is highly dependent on input data as presented in detail at [92].

6.3.2 Comparative Evaluation of 4D-Fluent and N-ary representations.

Fig. 6.9 represents the comparison of the number of axioms required for expressing a number of temporal relations using the 4D-fluents and the N-ary relations

approach respectively.

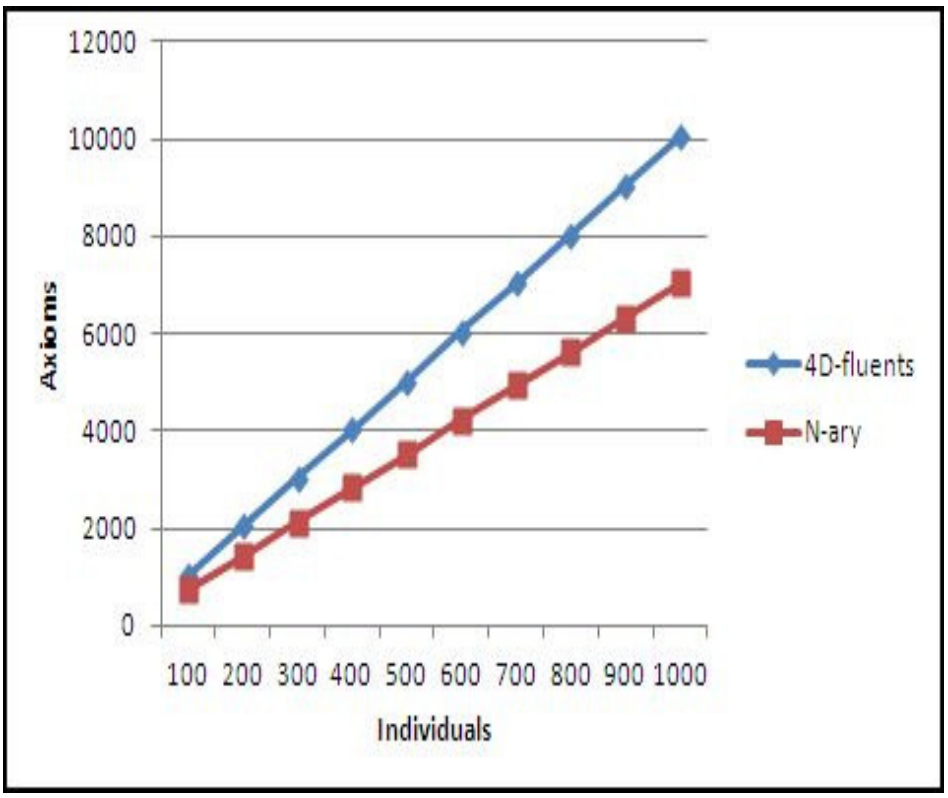

Figure 6.9: Axioms required for representing temporal properties using the 4D-fluents and the N-ary approach.

Fig. 6.10 represents the average time (for 5 random generated ontologies for every instance set size) required for reasoning over temporal properties using the N-ary and the 4D-fluents approach for representing fluent properties as in Fig. 2.6 and Fig. 2.7. The representation involve both simple properties and properties that are transitive, symmetric or subproperties of other properties in order to provide a realistic comparison.

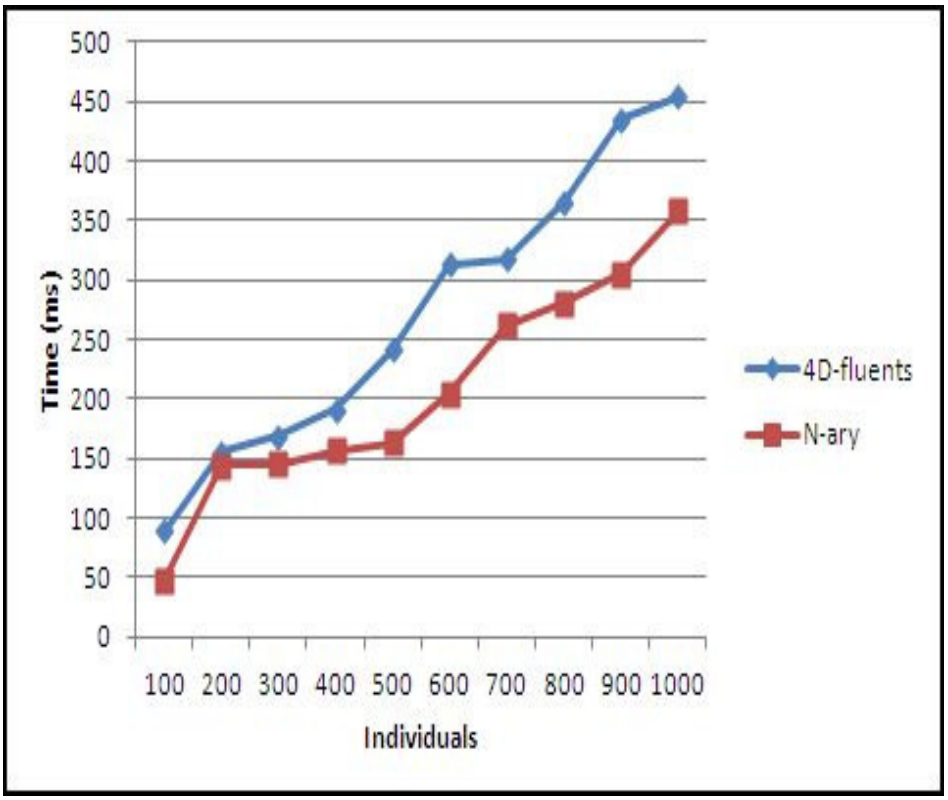

Figure 6.10: Time required for reasoning over temporal properties using the 4D-fluents and the N-ary approach

The quantitative comparison of the two approaches indicates that N-ary approach outperforms the 4D-fluents representation in terms of required axioms and consequently in reasoning time. This is attributed to the fact that fewer additional objects are required as illustrated in Fig. 2.6 and Fig. 2.7.

6.3.3 OWL Axioms-Based Implementation

As mentioned in Chapter 4, offering a sound, complete and decidable spatio-temporal reasoning mechanism using OWL axioms is not possible because of lack of the combination of adequate constructs (e.g., intersection of properties) which in turn is due to restrictions imposed by the requirement to retain decidability. Nevertheless, in the case of basic point relations, when the requirement of inconsistency detection is dropped, an implementation based on OWL axioms

is feasible. Bellow, this implementation is compared to the SWRL based implementation. Fig. 6.11 represents the average time (for 10 random generated ontologies for every instance set size) required for point relations using SWRL rules and OWL axioms (only for inference and not inconsistency detection in the later case).

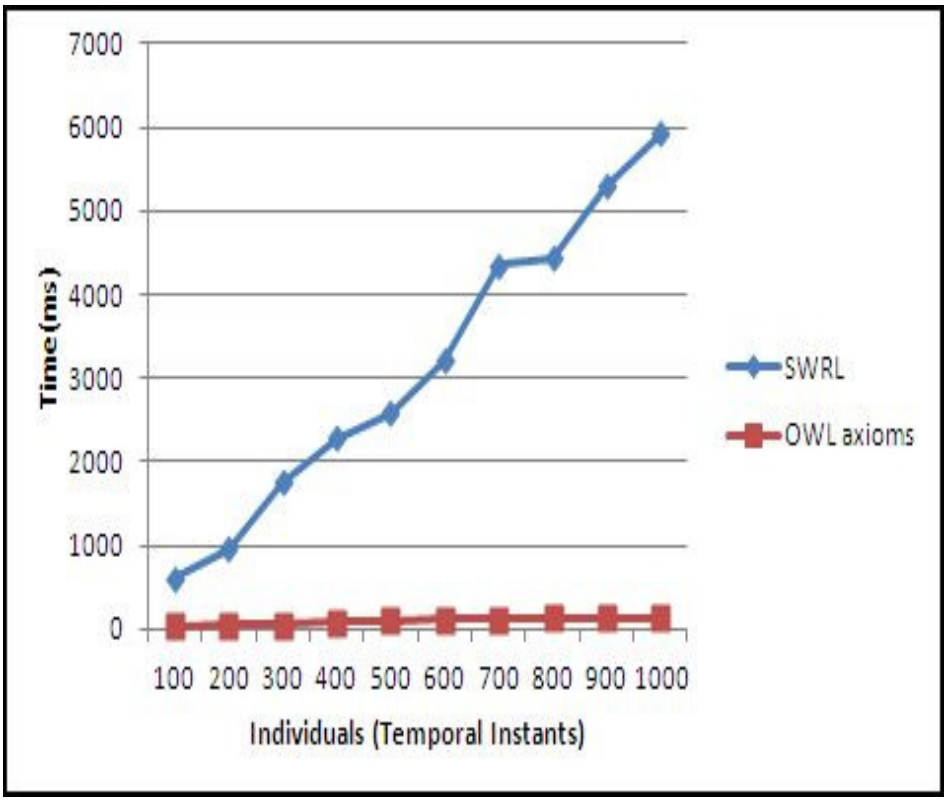

Figure 6.11: Average response time (over 10 runs) required for reasoning over temporal points using SWRL rules and OWL axioms as a function of the number of points.

The result indicated that when inconsistency detection is not a requirement of the reasoning mechanism (i.e., assertions are guaranteed to be correct, thus only inference and not inconsistency detection is required) the implementation based on OWL axioms is faster and can be preferred. Optimizations employed in current reasoning engines such as Pellet over OWL axioms provide much faster reasoning time than the corresponding reasoning time using SWRL when the

two approaches are directly comparable (i.e., when inconsistency detection is not required). Also, the OWL axioms-based approach applies on the TBOX of the ontology, thus on implied anonymous individuals and Concept definitions, and is not restricted to asserted named individuals as the SWRL-based reasoning mechanism.

Chapter 7

Conclusions and Future work

We propose SOWL an approach for handling temporal and spatial knowledge in ontologies. SOWL handles both, time instants and temporal intervals (and also semi-closed intervals) equally well using a sound and complete inference procedure based on path consistency and also handles property restrictions on fluent properties. SOWL applies on both, spatial and spatio-temporal information, by offering support for representing and reasoning over topological and directional spatial relations. Two alternative representations based on the 4D-fluents and the N-ary relations are presented and evaluated. Finally, two spatio-temporal query languages, first TOQL (based on the idea of extending SQL with spatio-temporal operators) and SOWL Query Language (based on SPARQL) are proposed and discussed. These query languages offer support for proposed spatio-temporal representation which in turn supports both qualitative and quantitative information. Other existing query languages such as t-SPARQL [109] and T-SPARQL [39] support representations based on named graphs and versioning respectively, which does not offer the expressive power of the SOWL framework. Neither support queries over qualitative information, nor they are supported by reasoning over a rich in spatio-temporal semantics ontological frameworks such as SOWL.

7.1 Conclusions

The proposed approach effectively represents and offers reasoning support for qualitative in addition to quantitative defined spatio-temporal information. It is

fully compliant with existing Semantic Web standards and specifications which increases it's applicability. Being compatible with W3C specifications SOWL is compatible and can be used in conjunction with existing editors, reasoners and querying tools such as Protege and Pellet without requiring specific additional software. Therefore, information in SOWL can be distributed and shared without specific software. Users can modify and redistribute this information using common tools such as Pellet and Protege. This is a highly desirable characteristic according to the overall Semantic Web vision. The contributions of SOWL are summarized below:

- Supports representation of dynamic information using both the 4D-fluents and N-ary relations. Both approaches are evaluated.
- Support representation of both temporal and spatial relations. Temporal relations between points and intervals, and representations based on both, are presented and compared. Spatial information is expressed using are topological RCC8 relations or alternatively cone-shaped, or projection-based directional relations or combinations of the above.
- Supports qualitative expressions (i.e., information defined using natural language terms such as "before") of spatio-temporal relations. This is an issue not addressed by existing approaches for the Semantic Web.
- Offers a reasoning mechanism dealing with inference of implied facts, and relations derived from asserted relations, and detecting inconsistencies of these assertions. Reasoning is realized using path consistency on tractable sets of spatial and temporal relations.
- Existing approaches for spatio-temporal representation over the semantic web do not deal with preservation of property semantics and constrains, when dealing with properties and objects evolving in time. This work deals with these issues by means of additional rules.
- Introduces query languages that offer support for the proposed spatio-temporal representation. The SOWL ontology model is not part of the query languages and it is not visible to the user, so the user need not be

familiar with peculiarities of the underlying mechanism for time and space representation. Other existing query languages such as t-SPARQL [109] and T-SPARQL [39] does not support queries over qualitative information, nor they are supported by reasoning over a rich in spatio-temporal semantics ontological frameworks such as SOWL.

7.2 Future Work

Directions for future work include:

- Addressing scalability issues by offering optimizations tailored for specific datasets in large scale applications. Optimizations can apply on both the reasoning and the querying process. For example, indexing mechanisms tailored for quantitative spatial datasets can be applied in certain applications. Examples of indexing support for temporal intervals for the Semantic Web are provided at [87; 109]. Examples for query optimization are provided at [46; 95].

- Extending the representation to handle information for qualitative distance relations such as *far, near* (which are application specific) is an additional direction for future research.

- Combining qualitative distance and directional information into the reasoning mechanism. Notice that, qualitative distance reasoning interferes with directional reasoning [92], thus qualitative distance reasoning must be integrated into the existing mechanism.

- Supporting duration information into the temporal reasoning mechanism. Although information about durations can, in some cases, be expressed in term of end-point relations, this is not always the case (for example when duration is known but both end-points of an interval are not). As with the case of qualitative distance relation this can be handled by extending the SOWL framework, since path consistency also applies in this case [88].

- Developing a tool that automates the process of converting class definitions and restrictions into a temporal representation. Although the issue is addressed in this work, converting an arbitrary expression is an tedious and error prone process that must be handled by specific software and not by the ontology developer.

- Proposing extensions on the OWL specification that will increase expressiveness and compactness of spatial and temporal representations. An example of this approach is TOWL [38] which handles only quantitative defined temporal information. Also developing tools that will offer support for such extensions is an important direction for future work.

Appendix A

Implementing the reasoning mechanism over Allen relations described in Section 4.2.1 is based on a tractable set of relations. These relations are basic Allen relations or disjunctions of basic relations represented as a set of relations into brackets. The tractable set of Allen relations used in this work is:
{B},{A},{A, D, Di, O, Oi, Mi, S, Si, F, Fi, Eq}, {A, D, Oi, Mi, F}, {A, Di, Oi, Mi, Si}, {A, Oi, Mi}, {B, D, Di, O, Oi, M, S, Si, F, Fi, Eq}, {B, D, O, M, S}, {B, Di, O, M, Fi}, {B, O, M}, {D}, {D, Di, O, Oi, S, Si, F, Fi, Eq}, {D, Oi, F}, {D, O, S}, {Di}, {Di, Oi, Si}, {Di, O, Fi}, {Eq}, {F}, {F, Fi, Eq}, {Fi}, {M}, {Mi}, {O}, {Oi}, {S}, {S, Si, Eq}, {Si}.

Appendix B

Implementing the reasoning mechanism over RCC8 relations described in Section 4.3 is based on a tractable set of relations. These relations are basic RCC8 relations or disjunctions of basic relations represented as a set of relations into brackets. The tractable set of RCC-8 relations used in this work is:
{DC}, {DC, EC}, {DC, EC, PO}, {DC, EC, PO, TPP}, {DC, EC, PO, TPP, NTPP}, {DC, EC, PO, TPP, NTPP, TPPi}, {DC, EC, PO, TPP, NTPP, TPPi, EQ}, {DC, EC, PO, TPP, NTPP, TPPi, NTPPi}, {DC, EC, PO, TPP, TPPi}, {DC, EC, PO, TPP, TPPi, EQ}, {DC, EC, PO, TPP, TPPi, NTPPi}, {DC, EC, PO, TPP, TPPi, NTPPi, EQ}, {DC, EC, PO, TPPi}, {DC, EC, PO, TPPi, NTPPi}, {EC}, {EC, PO}, {EC, PO, TPP}, {EC, PO, TPP, NTPP}, {EC, PO, TPP, NTPP, TPPi}, {EC, PO, TPP, NTPP, TPPi, EQ}, {EC, PO, TPP, NTPP, TPPi, NTPPi}, {EC, PO, TPP, NTPP, TPPi, NTPPi, EQ}, {EC, PO, TPP, TPPi}, {EC, PO, TPP, TPPi, EQ}, {EC, PO, TPP, TPPi, NTPPi}, {EC, PO, TPP, TPPi, NTPPi, EQ}, {EC, PO, TPPi}, {EC, PO, TPPi, NTPPi}, {EQ}, {NTPP}, {NTPPi}, {PO}, {PO, TPP}, {PO, TPP, NTPP}, {PO, TPP, NTPP, TPPi}, {PO, TPP, NTPP, TPPi, EQ}, {PO, TPP, NTPP, TPPi, NTPPi}, {PO, TPP, NTPP, TPPi, NTPPi, EQ}, {PO, TPP, TPPi}, {PO, TPP, TPPi, EQ}, {PO, TPP, TPPi, NTPPi}, {PO, TPP, TPPi, NTPPi, EQ}, {PO, TPPi}, {PO, TPPi, NTPPi}, {TPP}, {TPP, NTPP}, {TPPi}, {TPPi, NTPPi}.

Appendix C

Implementing the reasoning mechanism over cone-shaped directional relations described in Section 4.3 is based on a tractable set of relations. These relations are basic directional relations or disjunctions of basic relations represented as a set of relations into brackets. The tractable set of cone-shaped directional relations used in this work is:
{E}, {E, SE}, {E, SE, S}, {E, SE, S, SW}, {N}, {N, NE}, {N, NE, E}, {N, NE, E, SE}, {NE}, {NE, E}, {NE, E, SE}, {NE, E, SE, S}, {SE}, {SE, S}, {SE, S, SW}, {SE, S, SW, W}, {S}, {S, SW}, {S, SW, W}, {S, SW, W, NW}, {SW}, {SW, W}, {SW, W, NW}, {SW, W, NW, N}, {W}, {W, NW}, {W, NW, N}, {W, NW, N, NE}, {NW}, {NW, N}, {NW, N, NE}, {NW, N, NE, E}.

References

[1] J.F. Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, **26**[11]:832–843, 1983. 21, 37, 38, 48, 52, 73

[2] A. Ankolekar, M. Burstein, et al. DAML-S: Web Service Description for the Semantic Web. In *First International Semantic Web Conference*, pages 348–363. Citeseer, 2002. 15

[3] A. Artale and E. Franconi. A Survey of Temporal Extensions of Description Logics. *Annals of Mathematics and Artificial Intelligence*, **30**[1]:171–210, 2000. 2, 28

[4] F. Baader. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge Univ. Pr., 2003. 10

[5] F. Baader. Description Logics. In *Reasoning Web: Semantic Technologies for Information Systems, 5th International Summer School 2009*, **5689** of *Lecture Notes in Computer Science*, pages 1–39. Springer–Verlag, 2009. 10, 14, 15, 16

[6] P. Balbiani, J.F. Condotta, and L.F. del Cerro. A Model for Reasoning about Bidimensional Temporal Relations. In *International Conference On Principles Of Knowledge Representation And Reasoning*, pages 124–130. Citeseer, 1998. 27

[7] P. Balbiani, J.F. Condotta, and L. Farinas del Cerro. A new Tractable Subclass of the Rectangle Algebra. In *International Joint Conference On Artificial Intelligence*, **16**, pages 442–447. Citeseer, 1999. 26

[8] E. Baratis, N. Maris, E. Petrakis, S. Batsakis, and N. Papadakis. The TOQL System. *11th International Symposium on Spatial and Temporal Databases (SSTD 2009), Demo Paper*, pages 450–454, 2009. 73, 77

[9] E. Baratis, E. Petrakis, S. Batsakis, N. Maris, and N. Papadakis. Toql: Temporal Ontology Querying Language. *11th International Symposium on Spatial and Temporal Databases (SSTD 2009)*, pages 338–354, 2009. 34, 73, 77, 82

[10] R. Batres, M. West, D. Leal, D. Price, K. Masaki, Y. Shimada, T. Fuchino, and Y. Naka. An Upper Ontology Based on ISO 15926. *Computers and Chemical Engineering*, **31**[5-6]:519 – 534, 2007. 31

[11] S. Batsakis and E.G.M. Petrakis. Representing and Reasoning over Spatio-Temporal Information in OWL 2.0. *6th Workshop on Semantic Web Applications and Perspectives (SWAP' 2010)*, 2010. 33

[12] S. Batsakis and E.G.M. Petrakis. Representing Temporal Knowledge in the Semantic Web: The Extended 4D-Fluents Approach. *2nd International Workshop on Combinations of Intelligent Methods and Applications (CIMA' 2010)*, pages 55–69, 2010. 73

[13] S. Batsakis and E.G.M. Petrakis. SOWL: Representing Spatio-Temporal Information in OWL. In *7th Extended Semantic Web Conference (ESWC'2010). Poster Paper*, 2010. 33

[14] S. Batsakis and E.G.M. Petrakis. SOWL: Spatio-Temporal Representation, Reasoning and Querying Over the Semantic Web. In *Proceedings of the 6th International Conference on Semantic Systems, (I-SEMANTICS' 2010)*, pages 15:1–15:9. ACM, 2010. 3, 33, 73, 77, 82

[15] S. Batsakis and E.G.M. Petrakis. Integrated Ontologies for Spatial Scene Descriptions. In *Qualitative Spatio-Temporal Representation and Reasoning: Trends and Future Directions, IGI Global (book chapter), S. M Hazarika (Eds)*. Citeseer, 2011. 31

[16] S. Batsakis and E.G.M. Petrakis. SOWL: A Framework for Handling Spatio-Temporal Information in OWL 2.0. *5th International Symposium on Rules: Research Based and Industry Focused (RuleML' 2011)*, pages 242–249, 2011. 33, 47, 59, 62, 97

[17] S. Batsakis, E.G.M. Petrakis, and E. Milios. Improving the Performance of Focused Web Crawlers. *Data & Knowledge Engineering*, **68**[10]:1001–1013, 2009. 10

[18] S. Batsakis, K. Stravoskoufos, and E.G.M. Petrakis. Temporal Reasoning for Supporting Temporal Queries in OWL 2.0. *15th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems (KES' 2011)*, **6881**:558–567, 2011. 47, 85, 97

[19] S. Bechhofer, F. Van Harmelen, J. Hendler, I. Horrocks, D.L. McGuinness, P.F. Patel-Schneider, L.A. Stein, et al. OWL Web Ontology Language Reference. *W3C recommendation*, **10**:2006–01, 2004. 16

[20] S. Bechhofer, R. Volz, and P. Lord. Cooking the Semantic Web with the OWL API. *Proceedings of the 2nd International Semantic Web Conference (ISWC 2003)*, pages 659–675, 2003. 5

[21] T. Berners-Lee, J. Hendler, O. Lassila, et al. The Semantic Web. *Scientific American*, **284**[5]:28–37, 2001. 10

[22] M. Bodirsky and H. Chen. Qualitative Temporal and Spatial Reasoning Revisited. *Journal of Logic and Computation*, **19**:1359–1383, December 2009. 27

[23] M.H. Bohlen and C.S. Jensen. Seamless Integration of Time Into SQL. In *Technical Report TR-962049, Aalborg University, Department of Computer Science.* Aalborg University, Institute for Electronic Systems, Dept. of Mathematics and Computer Science, 1996. 33

[24] J. Broekstra and A. Kampman. SeRQL: A Second Generation RDF Query Language. In *Proc. SWAD-Europe Workshop on Semantic Web Storage and Retrieval*, pages 13–14. Citeseer, 2003. 33, 34

[25] I. Budak Arpinar, A. Sheth, C. Ramakrishnan, E. Lynn Usery, M. Azami, and M.P. Kwan. Geospatial Ontology Development and Semantic Analytics. *Transactions in GIS*, **10**[4]:551–575, 2006. 32, 38

[26] P. Buneman and E. Kostylev. Annotation Algebras for RDFS. In *The Second International Workshop on the Role of Semantic Web in Provenance Management (SWPM-10), CEUR Workshop Proceedings*, 2010. 28

[27] S. Chakrabarti, M. Van den Berg, and B. Dom. Focused Crawling: A New Approach to Topic-specific Web Resource Discovery. *Computer Networks*, **31**[11-16]:1623–1640, 1999. 10

[28] P.A. Champin and A. Passant. SIOC in Action Representing the Dynamics of Online Communities. In *Proceedings of the 6th International Conference on Semantic Systems*, pages 1–7. ACM, 2010. 30

[29] H. Chen, F. Perich, T. Finin, and A. Joshi. SOUPA: Standard Ontology for Ubiquitous and Pervasive Applications. *Mobile and Ubiquitous Systems, Annual International Conference on*, pages 258–267, 2004. 33, 38

[30] P.P. Chen. The Entity-Relationship Model—Toward a Unified View of Data. *ACM Transactions on Database Systems*, **1**[1]:9–36, 1976. 15

[31] S. Christodoulakis, M. Foukarakis, L. Ragia, H. Uchiyama, and T. Imai. Semantic Maps and Mobile Context Capturing for Picture Content Visualization and Management of Picture Databases. pages 130–136, 2008. 33

[32] G. Christodoulou. A Reasoning and Query Engine over Qualitative Spatial Information. *Diploma Thesis, Department of Electronic and Computer Engineering, Technical University of Crete (to appear)*, 2012. 33

[33] A.G. Cohn, B. Bennett, J. Gooday, and N.M. Gotts. Qualitative Spatial Representation and Reasoning with the Region Connection Calculus. *GeoInformatica*, **1**[3]:275–316, 1997. 59

[34] A.G. Cohn and S.M. Hazarika. Qualitative Spatial Representation and Reasoning: An Overview. *Fundamenta Informaticae*, **46**[1-2]:1–29, 2001. 32

[35] H.B. Enderton. *A Mathematical Introduction to Logic*, **40**. Academic press New York, 1972. 10

[36] D. Fensel, F. Van Harmelen, I. Horrocks, D.L. McGuinness, and P.F. Patel-Schneider. OIL: An Ontology Infrastructure for the Semantic Web. *Intelligent Systems, IEEE*, **16**[2]:38–45, 2001. 15

[37] A.U. Frank. Qualitative Spatial Reasoning: Cardinal Directions as an Example. *International Journal of Geographical Information Science*, **10**[3]:269–290, 1996. 41, 42, 59

[38] F. Frasincar, V. Milea, and U. Kaymak. tOWL: Integrating Time in OWL. *Semantic Web Information Management: A Model-Based Perspective*, pages 225–246, 2010. 5, 28, 38, 116

[39] F. Grandi. T-SPARQL: a TSQL2-like Temporal Query Language for RDF. In *International Workshop on on Querying Graph Structured Data*, pages 21–30, 2010. 7, 34, 113, 115

[40] B.C. Grau, I. Horrocks, B. Motik, B. Parsia, P. Patel-Schneider, and U. Sattler. OWL 2: The Next Step for OWL. *Web Semantics: Science, Services and Agents on the World Wide Web*, **6**[4]:309–322, 2008. 17

[41] H. Gregersen and C.S. Jensen. Temporal Entity-Relationship Models-a Survey. *Knowledge and Data Engineering, IEEE Transactions on*, **11**[3]:464–497, 1999. 27

[42] P. Grenona and B. Smith. SNAP and SPAN: Towards Dynamic Spatial Ontology. *Event-Oriented Approaches in Geographic Information Science:*

A Special Issue of Spatial Cognition and Computation, **4**[1]:69–104, 2004. 20

[43] R. Grutter and B. Bauer-Messmer. Towards Spatial Reasoning in the Semantic Web: A Hybrid Knowledge Representation System Architecture. *The European Information Society*, pages 349–364, 2007. 33

[44] C. Gutierrez, C. Hurtado, and A. Vaisman. Temporal RDF. In *Second European Semantic Web Conference (ESWC 2005)*, pages 93–107, 2005. 2, 28, 37, 38

[45] R.H. Guting. An Introduction to Spatial Database Systems. *The VLDB Journal*, **3**[4]:357–399, 1994. 32

[46] O. Hartig and R. Heese. The SPARQL Query Graph Model for Query Optimization. **4519**:564–578, 2007. 115

[47] G. Hatzigeorgakidis. Management of Spatio-temporal Information in Semantic Web Applications. *Diploma Thesis, Department of Electronic and Computer Engineering, Technical University of Crete*, 2011. 33, 41, 62

[48] K. Hawley. Temporal Parts. *The Stanford Encyclopaedia of Philosophy, Edward N. Zalta (ed.)*, 2004. 20

[49] J.R. Hobbs and F. Pan. Time Ontology in OWL. *W3C Working Draft, September 2006*, 2006. 20

[50] I. Horrocks, O. Kutz, and U. Sattler. The Even More Irresistible SROIQ. In *Proc. of the 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2006)*, pages 57–67, 2006. 17, 56

[51] 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*, **21**, 2004. 10

[52] I. Horrocks, P.F. Patel-Schneider, and F. Van Harmelen. From SHIQ and RDF to OWL: The Making of a Web Ontology Language. *Web*

Semantics: Science, Services and Agents on the World Wide Web, **1**[1]:7–26, 2003. 15, 16

[53] I. Horrocks, U. Sattler, and S. Tobies. Practical Reasoning for Expressive Description Logics. In *Logic for Programming and Automated Reasoning*, pages 161–180. Springer, 1999. 52

[54] C.B. Jones, A.I. Abdelmoty, D. Finch, G. Fu, and S. Vaid. The Spirit Spatial Search Engine: Architecture Ontologies and Spatial Indexing. *Geographic Information Science*, pages 125–139, 2004. 38

[55] P. Jonsson and A. Krokhin. Complexity Classification in Qualitative Temporal Constraint Reasoning. *Artificial Intelligence*, **160**[1-2]:35–51, 2004. 27

[56] G. Karvounarakis, S. Alexaki, V. Christophides, D. Plexousakis, and M. Scholl. RQL: a Declarative Query Language for RDF. In *Proceedings of the 11th international conference on World Wide Web*, pages 592–603. ACM, 2002. 33

[57] Y. Katz and B.C. Grau. Representing Qualitative Spatial Information in OWL-DL. In *In Proceedings of OWL: Experiences and Directions. CEUR Workshop Proceedings vol. 188*. Citeseer, 2005. 33

[58] M. Klein and D. Fensel. Ontology Versioning on the Semantic Web. In *Proceedings of the International Semantic Web Working Symposium (SWWS)*, pages 75–91. Citeseer, 2001. 2, 28, 29

[59] N. Kline and R.T. Snodgrass. Computing Temporal Aggregates. In *Data Engineering, Proceedings of the Eleventh International Conference on*, page 222. Published by the IEEE Computer Society, 1995. 33

[60] V. Kolovski, B. Parsia, and E. Sirin. Extending the SHOIQ (D) Tableaux with DL-Safe Rules: First Results. In *Proceedings International Workshop on Description Logic, DL-2006*. Citeseer, 2006. 97

[61] K. Koubarakis, K. Kyzirakos, M. Karpathiotakis, C. Nikolaou, M. Sioutis, S. Vassos, D. Michail, T. Herekakis, C. Kontoes, and I. Papoutsis. Challenges for Qualitative Spatial Reasoning in Linked Geospatial Data. *Proceedings of the IJCAI 2011 Workshop on Benchmarks and Applications of Spatial Reasoning (BASR-11)*, 2011. 33

[62] M. Koubarakis and K. Kyzirakos. Modeling and Querying Metadata in the Semantic Sensor Web: the Model stRDF and the Query Language stSPARQL. *Proceedings of the 7th Extended Semantic Web Conference (ESWC2010), Part I, volume 6088 of Lecture Notes in Computer Science, Springer*, pages 425–439, 2010. 33

[63] H.U. Krieger. A General Methodology for Equipping Ontologies With Time. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC'10), ELRA*, pages 3165–3172. Citeseer, 2010. 32

[64] H.U. Krieger. A Temporal Extension of the Hayes and ter Horst Entailment Rules for RDFS and OWL. In *AAAI 2011 Spring Symposium Logical Formalizations of Commonsense Reasoning*, pages 143–146, 2011. 32

[65] H.U. Krieger, B. Kiefer, and T. Declerck. A Framework for Temporal Representation and Reasoning in Business Intelligence Applications. In *AAAI 2008 Spring Symposium on AI Meets Business Rules and Process Management*, pages 59–70, 2008. 32

[66] A. Krokhin, P. Jeavons, and P. Jonsson. Reasoning About Temporal Relations: The Tractable Subalgebras of Allen's Interval Algebra. *Journal of the ACM (JACM)*, **50**[5]:591–640, 2003. 27

[67] O. Lassila, R.R. Swick, et al. Resource Description Framework (RDF) Model and Syntax Specification. *W3C Recommendation, World Wide Web Consortium*, 1999. 15

[68] N. Lopes, A. Polleres, U. Straccia, and A. Zimmermann. AnQL: SPARQLing up Annotated RDFS. *The Semantic Web–ISWC 2010*, pages 518–533, 2010. 29

[69] C. Lutz. Description logics with concrete domains-a survey. *In Advances in Modal Logics, volume 4. King's College Publications*, 2003. 2, 28

[70] C. Lutz, F. Wolter, and M. Zakharyashev. Temporal Description Logics: A Survey. In *15th International Symposium on Temporal Representation and Reasoning, TIME'08*, pages 3–14. IEEE, 2008. 28

[71] X. Makri. 4D-Fluents Plug-In: A Tool for Handling Temporal Ontologies in Protg. *Diploma Thesis, Department of Electronic and Computer Engineering, Technical University of Crete*, 2011. 30

[72] E. Mavrogiannaki. The Time Dimension in Clinical Guidelines for the Management of Abnormal Gynaecological Exam Results. *Master's thesis, Department of Electronic and Computer Engineering, Technical University of Crete*, 2011. 33

[73] B. McBride. Jena: A Semantic Web Toolkit. *Internet Computing, IEEE*, **6**[6]:55–59, 2002. 5

[74] B. McBride. The Resource Description Framework (RDF) and its Vocabulary Description Language RDFS. *Handbook on Ontologies*, pages 51–66, 2004. 15

[75] D.L. McGuinness, F. Van Harmelen, et al. OWL Web Ontology Language Overview. *W3C Recommendation*, **10**:2004–03, 2004. 10, 16

[76] M. Minsky. Frame-system Theory. *Thinking: Readings in Cognitive Science*, pages 355–376, 1977. 10

[77] D.R. Montello and A.U. Frank. Modeling Directional Knowledge and Reasoning in Environmental Space: Testing Qualitative Metrics. *The Construction of Cognitive Maps GeoJournal Library*, **32**[3]:321–344, 1996. 23, 25

[78] B. Motik. Representing and Querying Validity Time in RDF and OWL: a Logic-Based Approach. *The Semantic Web–ISWC 2010*, pages 550–565, 2010. 29, 34

[79] B. Motik, I. Horrocks, R. Rosati, and U. Sattler. Can OWL and Logic Programming Live Together Happily Ever After? *5th International Semantic Web Conference, ISWC 2006*, pages 501–514, 2006. 19

[80] B. Motik, P.F. Patel-Schneider, B. Parsia, C. Bock, A. Fokoue, P. Haase, R. Hoekstra, I. Horrocks, A. Ruttenberg, U. Sattler, et al. OWL 2 Web Ontology Language: Structural Specification and Functional-Style Syntax. *W3C Recommendation*, **27**, 2009. 17

[81] B. Nebel and H.J. Burckert. Reasoning About Temporal Relations: A Maximal Tractable Subclass of Allen's Interval Algebra. *Journal of the ACM (JACM)*, **42**[1]:43–66, 1995. 27, 50, 51, 52, 96

[82] N. Noy, A. Rector, P. Hayes, and C. Welty. Defining N-ary Relations on the Semantic Web. *W3C Working Group Note*, **12**, 2006. 3, 35, 44

[83] J. Perez, M. Arenas, and C. Gutierrez. Semantics and Complexity of SPARQL. *ACM Trans. Database Syst.*, **34**:16:1–16:45, September 2009. 85

[84] E.G.M. Petrakis, C. Faloutsos, and K.I. Lin. Imagemap: An Image Indexing Method Based on Spatial Similarity. *Knowledge and Data Engineering, IEEE Transactions on*, **14**[5]:979–987, 2002. 32

[85] A. Preventis. Chronos: A Tool for Handling Temporal Ontologies in Protege. *Diploma Thesis, Department of Electronic and Computer Engineering, Technical University of Crete (to appear)*, 2012. 30

[86] E. PrudHommeaux and A. Seaborne. SPARQL Query Language for RDF. *W3C working draft*, **4**, 2006. 33, 34

[87] A. Pugliese, O. Udrea, and VS Subrahmanian. Scaling RDF with Time. In *Proceeding of the 17th international conference on World Wide Web*, pages 605–614. ACM, 2008. 115

[88] A.K. Pujari and A. Sattar. A New Framework for Reasoning About Points, Intervals and Durations. In *International Joint Conference On Artificial Intelligence*, **16**, pages 1259–1267. LAWRENCE ERLBAUM ASSOCIATES LTD, 1999. 27, 115

[89] D.A. Randell, Z. Cui, and A.G. Cohn. A Spatial Logic Based on Regions and Connection. *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference, (KR 92)*, **92**:165–176, 1992. 22, 41

[90] J. Renz. Maximal Tractable Fragments of the Region Connection Calculus: A Complete Analysis. In *International Joint Conference On Artificial Intelligence*, **16**, pages 448–455. Citeseer, 1999. 27, 52, 62

[91] J. Renz and D. Mitra. Qualitative Direction Calculi with Arbitrary Granularity. In *PRICAI 2004: Trends in Artificial Intelligence: 8th Pacific Rim International Conference on Artificial Intelligence, Proceedings*, pages 65–74. Springer-Verlag New York Inc, 2004. 27, 59, 60

[92] J. Renz and B. Nebel. Qualitative Spatial Reasoning using Constraint Calculi. *Handbook of Spatial Logics*, pages 161–215, 2007. 22, 23, 25, 26, 27, 32, 35, 41, 51, 61, 62, 96, 107, 115

[93] J. Santos, L. Braga, and A.G. Cohn. FONTE: A Protégé Plug-in for Engineering Complex Ontologies. *Enterprise Information Systems*, pages 222–236, 2011. 30

[94] K. Schild. *Towards a Theory of Frames and Rules*. PhD thesis, Technische Universitaet Berlin, Berlin, Germany, 1989. 52

[95] M. Schmidt, M. Meier, and G. Lausen. Foundations of SPARQL Query Optimization. pages 4–33, 2010. 115

[96] A. Seaborne. RDQL-a Query Language for RDF. *W3C Member Submission*, **9**:29–21, 2004. 33

[97] N. Sepetas. Restriction Checking on OWL:2 Temporal Ontologies. *Diploma Thesis, Department of Electronic and Computer Engineering, Technical University of Crete*, 2011. 30

[98] R. Shaw, R. Troncy, and L. Hardman. Lode: Linking Open Descriptions of Events. *The Semantic Web*, pages 153–167, 2009. 30, 40

[99] A. Sheth and M. Perry. Traveling the Semantic Web Through Space, Time, and Theme. *Internet Computing, IEEE*, **12**[2]:81–86, 2008. 33, 38

[100] T. Sider. *Four-Dimensionalism: An Ontology of Persistence and Time (Mind Association Occasional Series)*. Oxford University Press, 2003. 20

[101] E. Sirin, B. Parsia, B.C. Grau, A. Kalyanpur, and Y. Katz. Pellet: A Practical OWL-DL Reasoner. *Web Semantics: Science, Services and Agents on the World Wide Web*, **5**[2]:51–53, 2007. 5, 47

[102] S. Skiadopoulos and M. Koubarakis. On the Consistency of Cardinal Direction Constraints. *Artificial Intelligence*, **163**[1]:91–135, 2005. 23, 26, 41

[103] R. Snodgrass. The Temporal Query Language TQuel. *ACM Transactions on Database Systems (TODS)*, **12**[2]:247–298, 1987. 33

[104] A. Sotnykova, C. Vangenot, N. Cullot, N. Bennacer, and M.A. Aufaure. Semantic Mappings in Description Logics for Spatio-Temporal Database Schema Integration. *Journal on Data Semantics III*, pages 143–167, 2005. 33, 38

[105] M. Stocker and E. Sirin. Pelletspatial: A Hybrid RCC-8 and RDF/OWL Reasoning and Query Engine. In *CEUR Workshop Proceedings*, **529**. Citeseer, 2009. 33, 97

[106] U. Straccia, N. Lopes, G. Lukácsy, and A. Polleres. A General Framework for Representing and Reasoning with Annotated Semantic Web Data. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10), AAAI Press*, 2010. 28

[107] K. Stravoskoufos. A Query Language for Spatio-Temporal Ontologies in OWL 2.0. *Master Thesis, Department of Electronic and Computer Engineering, Technical University of Crete (to appear)*, 2012. 85

[108] C. Tao, W.Q. Wei, H.R. Solbrig, G. Savova, and C.G. Chute. CNTRO: A Semantic Web Ontology for Temporal Relation Inferencing in Clinical Narratives. In *AMIA Annual Symposium Proceedings*, **2010**, pages 787–91. American Medical Informatics Association, 2010. 30

[109] J. Tappolet and A. Bernstein. Applied Temporal RDF: Efficient Temporal Querying of RDF Data with SPARQL. In *Proceedings of the 6th European Semantic Web Conference on The Semantic Web: Research and Applications*, pages 308–322. Springer-Verlag, 2009. 2, 7, 28, 29, 34, 38, 84, 113, 115

[110] P. Van Beek. Approximation Algorithms for Temporal Reasoning. *Proceedings of the 11th International Joint Conference on Artificial Intelligence- Volume 2*, pages 1291–1296, 1989. 21, 27

[111] P. Van Beek. *Exact and Approximate Reasoning about Qualitative Temporal Relations*. PhD thesis, University of Waterloo, 1990. 3, 38, 56, 96, 97

[112] P. Van Beek and R. Cohen. Exact and Approximate Reasoning about Temporal Relations. *Computational Intelligence*, **6**[3]:132–144, 1990. 27, 50, 51, 52, 55

[113] M. Vilain and H. Kautz. Constraint Propagation Algorithms for Temporal Reasoning. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, pages 377–382, 1986. 27

[114] C. Welty and R. Fikes. A Reusable Ontology for Fluents in OWL. In *Formal Ontology in Information Systems: Proceedings of the Fourth International Conference (FOIS 2006)*, pages 226–336, 2006. 2, 3, 28, 30, 31, 35, 36, 39

[115] V. Zamborlini and G. Guizzardi. On the Representation of Temporally Changing Information in OWL. In *Enterprise Distributed Object Computing Conference Workshops (EDOCW), 2010 14th IEEE International*, pages 283–292. IEEE, 2010. 31