Abstract—We investigate on potential improvements to reasoning approaches designed for spatial information in OWL. First, we introduce CHOROS, a qualitative spatial reasoning engine for ontologies in OWL. Building upon PelletSpatial \cite{pelletspatial}, CHOROS supports consistency checking and query answering for spatial information using Region-Connection Calculus (RCC), but also using the Cone-Shaped Directional logic formalism. It works with all RCC and CSD relations in combination with standard RDF/OWL semantic relations in an OWL ontology and can answer SPARQL queries with spatial and non-spatial relations. We also present SOWL, a spatial reasoner for both relation calculi implemented in SWRL and runs under Pellet\cite{sowl}. We discuss and evaluate possible optimizations of CHOROS and compare its performance with that of SOWL. The experimental results demonstrate that CHOROS runs significantly faster than its respective SWRL implementation in most cases.

I. Introduction

Formal spatial representations have been studied extensively in the Database \cite{rouza} and recently, the Semantic Web literature \cite{halevy}. Spatial entities (e.g., objects, regions) in classic database systems are represented using points, lines (polygonal lines) or Minimum Bounding Rectangles (MBRs) enclosing objects or regions and their relationships. Relations among spatial entities can be topological, orientation or distance-based 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.). The motivation for using a qualitative approach is that it is considered to be closer to the way humans represent and reason about commonsense knowledge. Another motivation is that it is possible to deal with incomplete knowledge.

A spatial ontology can also be defined without a coordinate reference system, such as in the case of topological relations \cite{rouza}. Nevertheless, it is not always possible to directly encode the semantics of these relations using the expressivity of OWL and Description Logics (DL) that OWL is based on \cite{owl}. There might be inconsistencies within a set of spatial relations that will not be detected by an OWL reasoner or, an OWL reasoner might not compute all spatial inferences. To deal with this problem, reasoning rules for various relation sets have been proposed \cite{chon}, \cite{halevy}.

The most popular reasoning methods used in qualitative spatial reasoning are constraint based techniques adopted from previous work on temporal reasoning \cite{rouza}, \cite{halevy}. Reasoning applies on sets of qualitative spatial relations which are jointly exhaustive and pairwise disjoint (i.e., between any two spatial entities exactly one of the basic relations holds). The set of all possible relations is then the set of all possible unions of the basic relations. Reasoning is realized by exploiting composition of relations. For instance, if the binary relation $R_1$ holds between entities $A$ and $B$ and the binary relation $R_2$ holds between $B$ and $C$, then the composition of $R_1$ and $R_2$ restricts the possible relationships between $A$ and $C$. Compositions of relations are usually pre-computed and stored in a composition table.

In this work, we present CHOROS reasoner which supports consistency checking and query answering over spatial information in OWL expressed using the Region Connection Calculus (RCC) or the Cone-Shaped Directional logic formalism (CSD). Choosing either representation is a design decision that depends mainly on the application. However, both RCC and CSD expressions in OWL may co-exist within the same ontology together with standard OWL semantic relations. In that respect, CHOROS extends PelletSpatial \cite{pelletspatial} to support CSD-9 relations in addition to RCC-8 relations. As such, it can answer mixed SPARQL queries over all spatial and non-spatial relation types.

Reasoning relies on the path consistency algorithm of PelletSpatial \cite{pelletspatial} extended to take into account the CSD relations in addition to the RCC ones of the original implementation. Plausible optimizations of CHOROS are also discussed and evaluated including, a multi-threading (faster) implementation enabling the parallel execution of CSD and RCC reasoning. We also suggest reducing the 9 CSD relations to 8 by encoding the identical to relation using the sameAs OWL axiom. As shown in the experimental results, this optimization speed-up the reasoner by at least 10%.

Finally, we introduce SOWL \cite{sowl}, an implementation of the spatial reasoner for both the CSD-9 and RCC-8 relation models using SWRL under Pellet \cite{pelletspatial}. The performance of CHOROS is also compared to that of the SOWL in both the average and the worst case. The experimental results demonstrate that CHOROS outperforms SOWL in most cases, except for very small ontologies (i.e., with a few number of regions) in the average case. A summary of the SOWL framework for handling spatio-temporal information in OWL.
Background knowledge and related research are discussed in Sec. II. CHOROS architecture, its components, the reasoner and query engine as well as optimizations are discussed in Sec. IV. SOWL design is presented in Sec. III. Evaluation results are presented in Sec. V followed by conclusions and issues for further research in Sec. VI.

II. BACKGROUND

Spatial information is typically encoded by means of a (binary) relation model of spatial entities. Most spatial calculi focus on a single aspect of space (e.g. topology, direction, distance or position [3]). Spatial relations between regions can be easily extracted from their surrounding minimum bounding rectangles (or their surrounding contours) by comparing their coordinates [11]. For computing directional relations in particular, it is convenient to approximate spatial entities such as regions, by points (e.g., by their centers of gravity). This is not sufficient for representing topological information, which encodes adjacency, overlap and containment relations between regions. Nevertheless, both representations may co-exist in a spatial representation (using one of them or both, is a design decision).

CHOROS supports qualitative topological and directional relations. An additional aspect of space information is distance. The projection-based approach [9] computes spatial relations between regions by means of their projections over the $X$ and $Y$ axes. In the case of points (e.g., city locations or regions represented by their centers of gravity) these projections are single point projections on the two axes. In the case of 2D regions, their projections form an interval on each axis. Furthermore, their relations on each axis are encoded using the 13 Allen relations.

Distance relations can also be defined and used in conjunction with the above relation models. However, qualitative distance relations (e.g., “far” and “near”) although common in natural scene descriptions, may be ambiguous, unless a reference coordinate system or scale for measuring distances is provided. This problem is resolved when distance relations are expressed quantitatively (e.g., “3Km away from city A”). Nonetheless, it is likely that only topological and directional relations will be used in practice.

A. RCC-8 Topologic Relations

Topologic 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 [3]. The most commonly used form of this calculus is referred to as RCC-8 calculus and specifies the 8 mutually exclusive relations between region pairs (DC, EC, EQ, NTPP, NTPPi, TTP, TTPi, PO) which are shown in Fig. 1.

B. CSD-9 Direction Relations

Directional relations are defined based on cone-shaped areas [12], [13]. As shown in Fig. 2, 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). Each relation covers a part of the 360 degrees range and all relations taken together cover the entire 360 degrees range.

Fig. 1: The set of RCC-8 Topologic Relations.

C. PelletSpatial

PelletSpatial [1] extends the Pellet OWL reasoner with qualitative spatial reasoning capabilities. It supports checking the consistency of spatial relations expressed using RCC-8 calculi and computes new spatial inferences from asserted relations. The spatial relations are expressed in RDF/OWL and can be combined with arbitrary domain ontologies.

PelletSpatial can answer SPARQL queries that mix spatial relations with RDF/OWL relations. It implements two reasoners: (a) A reasoner based on the semantics preserving translation of RCC-8 relations to OWL-DL class axioms and, (b) A reasoner that uses the RCC-8 composition table and implements a variant of the path-consistency algorithm by Renz and Nebel [14]. Without further optimizations, the performance of the former reasoner does not scale-up well with the size of the dataset. In addition to translating RCC-8 relations to OWL class axioms, one axiom is defined for each region to satisfy the regularity condition of region (i.e., to be a non-empty concept and to contain all of the regions interior points). This significantly affects non-determinism as well as the number of qualified existential quantifiers in the ontology. Qualified existential and universal quantifiers, is one of the sources of complexity (AND-branching) in DL reasoning. The later reasoner design using path-consistency, exhibited most promising performance and is the one we relied on for supporting reasoning with CSD-9 calculus.

Fig. 2: The set of Cone-Shaped Directional Relations.
III. SOWL

SOWL \(^1\) is 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. Both RCC-8 topological and CSD-9 relational calculi are supported. Handling both qualitative and quantitative information (i.e., information of 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. The reasoner is capable of inferring new relations and checking for their consistency, while retaining soundness, completeness, and tractability over the supported sets of relations. In contrast to PelletSpatial, reasoning is part of the ontology (rather than a separate system), so that maintenance of the ontology requires that changes are applied only to the ontology and not to the system.

Similarly to PelletSpatial, extracting spatial relations from the raw spatial data depends on the application and is not part of the reasoning mechanism. For more details on the implementation of the spatial reasoner the interested reader is referred to \([4]\).

A. Spatial Reasoning in SOWL

Reasoning in SOWL is realized by introducing a set of SWRL\(^2\) rules operating on spatial (topological or directional) and temporal relations. Notice that, OWL does not support role intersection and that (in order to retain decidability), transitivity of properties cannot be combined with property disjointness. In SOWL, path consistency is implemented using SWRL. Reasoners that support DL-safe rules such as Pellet\(^3\) can be used for inference and consistency checking over spatio-temporal relations. In addition to reasoning applying on temporal and spatial relations, the Pellet reasoner applies to the ontology schema to infer additional facts using OWL semantics (e.g., facts due to symmetric relationships and class-subclass relationships).

The SOWL spatial reasoner in particular, implements rules for RCC-8 and CSD-9 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. The point identity relation \((O)\) of CSD-9 formalism is handled using the OWL SameAs keyword applied on points instead of explicitly asserting the relation. Specifically, the nine directional relations have been declared as transitive OWL relations (i.e., a relation such as South is transitive meaning that if the relation holds between locations \(A\) and \(B\), and between locations \(B\) and \(C\), it also holds between locations \(A\) and \(C\)). Their inverse relations (e.g., North is the inverse of South) are defined as well. Furthermore, the identity relation \((O)\) is symmetric. All basic relations are pairwise disjoint.

By defining the transitive and inverse relations, all inferences of basic cone-shaped directional relations can be obtained. Path consistency is implemented by introducing rules defining compositions and intersections of supported relations until a fixpoint is reached or until an inconsistency is detected \([13]\).

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 one of these areas, preserving the disjointness of basic relations) are a special case of the revised Star Calculus \([13]\) and therefore, are decided by path consistency when applied on basic relations. Furthermore, given a tractable set of relations and by applying compositions, intersections and inverse operations until a set of relations that is closed under these operations is yielded, the resulting set of relations is also tractable \([3]\). By applying this closure method on the basic directional relations, a tractable set of relations containing the basic directional relations (and all relations appearing as the result of their composition or intersection) is yielded. This set of directional relations is used for directional spatial reasoning.

Reasoning on RCC-8 relations combines OWL property axioms with a set of composition rules (i.e., rules defining compositions of RCC-8 relations) and intersection rules. Relations DC, EC and PO are symmetric and relations NTPPi and TPPi are the inverse of NTPP and TPP respectively. In SOWL, the spatial reasoner implements the RCC-8 composition rules defined in \([13]\). For example the rule defining the composition of relations TPPi and NTPPi for locations \(x, y, z\) is:

\[
T P P i(x, y) \land N T P P i(y, z) \rightarrow N T P P i(x, z)
\]

Notice that, using the full set of relations \((2^8 - 1\) relations in the case of RCC-8 model) leads to intractability since this set cannot be decided by path consistency. However, tractable subsets of the full set are known to exist \([1, 13]\) and have been used in this work. In SOWL, after applying the closure method over the RCC-8 and CSD-9 relations sets, the minimal tractable sets containing the basic relations for the two models consist of 49 and 33 relations. For these sets, the number of OWL axioms (and SWRL rules) are 1,439 and 964 for RCC-8 and CSD-9 respectively \([13]\).

B. Complexity of SOWL

Following the analysis by Renz and Nebel \([3]\), by restricting the supported relations set to tractable subsets of RCC-8 and CSD-9 relations, path consistency has \(O(n^3)\) worst time complexity (with \(n\) being the number of locations) and is sound and complete. Nevertheless, this is a theoretical upper limit which is obtained when at most \(O(n^2)\) relations are asserted into the Knowledge Base (KB) using \(n\) locations and a relations set with \(k\) rules. At each inference step, the reasoner selects \(3\) variables among \(n\) points or regions which corresponds to \(O(n^3)\) possible different choices.

This upper bound is a rather pessimistic one, since the overall number of steps may be lower than \(O(n^2)\) because an

\(^1\)http://www.w3.org/Submission/SWRL

\(^2\)http://www.w3.org/TR/2009/REC-owl2-syntax-20091027

\(^3\)http://clarkparsia.com/pellet
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 may employ several optimizations (such as the Rete algorithm used within the SWRL implementation of Pellet [7]), so that the selection of appropriate variables usually involves fewer than \(O(n^3)\) trials. Because an 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, by retaining control over the order of variable selection and application of rules yields \(O(n^3)\) as a more realistic upper bound for path consistency.

IV. CHOROS REASONING

CHOROS defines an RDF/OWL vocabulary for expressing qualitative spatial relations, with both the RCC-8 topological and CSD-9 models. Specifically, a region is represented as an OWL individual (e.g., \(\text{Individual} : \text{Town1, Individual} : \text{Town2 of type Class : Town}\)) and a spatial relation between two regions is represented as an OWL object property assertion (e.g., \(\text{Individual} : \text{Town1 ObjectProperty : westOf Individual : Town2}\)). As illustrated in Fig. 3, RCC-8 and CSD-9 terms are defined as simple object properties with no extra characteristics (e.g., inverse, transitive). One can either use the vocabulary provided, or use his own by defining sub-property axioms. Non-spatial relations, such as region type, size etc., are represented as ordinary OWL assertions (e.g., \(\text{Individual} : \text{Town1 DatatypeProperty : hasName Individual : Chania}\)). Similarly to PelletSpatial, CHOROS strictly separates spatial reasoning from semantic OWL-DL reasoning.

![Fig. 3: RCC-8 and CSD-9 as object properties, in Protégé ontology editor.](http://jena.apache.org/)

The architecture of CHOROS reasoner is illustrated in Fig. 4. It consists of several modules, the most important of them being the following:

A. Parser

Implements an RDF and an ARQ parser for parsing ontologies and queries respectively. The RDF parser extracts RCC-8 and CSD-9 spatial triples from the RDF graph for creating their respective RCC-8 and CSD-9 constraint networks (i.e., two constraint networks, one for each spatial formalism) for checking the consistency of spatial relations defined in the ontology and for query answering. The spatial triples in the RDF graph are loaded in their respective constraint (i.e., RCC-8 or CSD-9) network. The rest of the graph (non-spatial relations) is handled by Pellet’s KB, a component that contains assertions about individuals (ABox) and axioms about classes (TBox). RDF parsing is implemented using Jena API and internal spatial vocabularies denoting the relations of the RCC-8 and CSD-9 formalisms. In query (ARQ) parsing, query atoms are characterized as RCC-8, CSD-9 or non-spatial.

B. Reasoner

CHOROS separates spatial from semantic DL reasoning and uses an exclusive reasoner for each spatial calculus. RCC-8 and CSD-9 relations are handled separately by their respective constraint networks. They form sets of jointly exclusive and pairwise disjoint basic relations (all basic relations are pairwise disjoint). Spatial reasoning for each spatial calculus is then achieved by applying the path consistency algorithm of [1] (notice though, that PelletSpatial implements path consistency for RCC-8 relations alone) separately for each calculus. Path consistency, computes all inferred relations using compositions of existing relations defined until a fixpoint is reached or until an inconsistency is detected. The possible compositions for basic RCC-8 and CSD-9 relations are defined by Cohn et.al [15] and by Renz and Mitra [13] respectively. Similarly to PelletSpatial, CHOROS implements paths consistency for basic relations only [3].

![Fig. 4: Main Components of the CHOROS reasoner.](http://jena.apache.org/)

As an example, consider the case with four objects on a plane. The question is whether a relation between \(\text{Object}_1\) and \(\text{Object}_3\) can be implied. The following list defines their CSD-9 constraint network:

- \(\text{Object}_1\ N \text{Object}_2\)
- \(\text{Object}_2\ NW \text{Object}_3\)
- \(\text{Object}_1\ NE \text{Object}_4\)
- \(\text{Object}_4\ N \text{Object}_3\)

[^4]: http://jena.apache.org/
By applying path consistency, the following relations are implied:

\[Object_1 \{N, NW\} Object_3\]
\[Object_1 \{N, NE\} Object_3\]

As a conclusion, \(Object_4\) is North (N) of \(Object_3\) (i.e., the intersection of the above two relations). In the same example, if \(Object_4\) is East (E) of \(Object_3\), path consistency results in the following relations set:

\[Object_1 \{N, NW\} Object_3\]
\[Object_1 \{E, NE\} Object_3\]

In the later case, the network is inconsistent (and as a result, no new relation is implied) because the intersection of the above two relations is the empty relation.

C. Query Engine

CHOROS supports similar query functionality to PelletSpatial with the addition of CSD-9 operators. As such it answers conjunctive queries specifying spatial and non-spatial patterns. More specifically, this module can process queries written in SPARQL satisfying the following conditions: (a) No variable is used in the predicate position, (b) Each property used in the predicate position is either an (object or datatype) property defined in the ontology or one of the following built-in properties: \(rdf\): type, \(owl\): sameIndividualAs, \(owl\): differentFrom and, (c) At least one of the triples must contain a spatial object property in the predicate position (otherwise it is a non-spatial query).

Similarly to PelletSpatial, the query engine can process queries for regions specifying their properties described in an RDF graph along with their (CSD-9 or RCC-8) spatial relations with other regions. The query engine implements a dual stage query answering technique. The set of query solutions returned by the first stage in response to a given a spatial query are fed to the second stage whose purpose is to guarantee that the non-spatial part of the query is satisfied as well.

D. Optimizations

In this section we present and discuss different versions of CHOROS along with possible optimizations.

a) CSD Calculus Approaches: The first version of CHOROS, referred to as henceforth CHOROS 0.1, applies consistency checking over the full set of 9 basic CSD-9 relations. This implementation requires pre-computing the compositions of 9 relations (i.e., \(2^9\)) and storing them in an \(512 \times 512\) table. Version 0.2 applies consistency checking over 8 out of 9 basic relations by omitting spatial relation identical-to which is replaced by the OWL axiom sameAs. CHOROS 0.2 pre-computes and stores less data (i.e., by computing the compositions of 8 relations instead of 9).

b) Multi-threading: Multi-threading allows two parts of the same program to run concurrently. A simple scheduler ensures that the number of threads running at any instance of time equals the number of CPU cores (2 in our case). In CHOROS, multi-threading enables parallel execution of CSD-9 and RCC-8 reasoning.

c) Complexity: In CHOROS, similarly to PelletSpatial, path consistency (and hence reasoning over CSD-9 or RCC-8 relation sets) has \(O(n^3)\) complexity in the worst case.

V. Evaluation

The purpose of the following set of experiments is to compare the performance of CHOROS with that of the SWRL spatial reasoner of SOWL [9] that runs under Pellet. In particular, we carried out two different experiments, corresponding to measurements of performance in the average and in the worst case. The average case performance of a spatial reasoner is encountered when less than \(n^2\) individuals are asserted into the KB from an input set of \(n\) locations or regions. In the first experiment, exactly \(kn\) relations \((k = 8\) in the case of RCC-8 and \(9\) in the case of CSD-9) are asserted. Accordingly, the worst case performance is encountered when the number of asserted relations are in the order of \(n^2\). In both experiments, we compare the running time of the two implementations of the reasoner as a function of the number of instances (representing points or regions) in the ontology.

Table I summarizes the differences between the two implementations of spatial reasoner considered in this work.

<table>
<thead>
<tr>
<th>Spatial Representation</th>
<th>Reasoning Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOWL: RCC-8 and CSD-9</td>
<td>CHOROS:</td>
</tr>
<tr>
<td>relations are defined as object properties with extra characteristics (e.g., (DC,BC,PO) relations are declared symmetric).</td>
<td>RCC-8 and CSD-9 relations are defined as simple object properties with no extra characteristics (e.g., inverse, symmetric relations are not declared).</td>
</tr>
<tr>
<td>CHOROS:</td>
<td></td>
</tr>
<tr>
<td>Path consistency is implemented by introducing SWRL rules defining compositions and intersections of supported relations until a fixpoint is reached or until an inconsistency is detected. Pellet is used for inference and consistency checking over spatial and non spatial relations.</td>
<td>Consistency checking separates spatial from semantic OWL-DL reasoning by using one exclusive reasoner for each calculus. Pellet is used only for inference and consistency checking over non spatial relations.</td>
</tr>
</tbody>
</table>

TABLE I: Differences between the CHOROS and SOWL reasoners.

A. Average Case Performance

As a case study and to objectively assess the performance of CHOROS in the average case we developed the “TUC spatial ontology” providing a qualitative spatial description of regions that comprise the campus of the Technical University of Crete (TUC) which is illustrated in Fig. 3. We used the satellite image from Google’s map for discriminating the
relative position of regions depicted on the map. RCC-8 and CSD-9 relations co-exist within the same ontology. For example, a region $A$ can be North West of region $B$ (i.e., $A NW B$ using CSD-9) and, at the same time, region $A$ can be “Externally Connected” to region $B$ (i.e., $A EC B$ using RCC-8). A detailed description of TUC ontology can be found in [13].

Additional instances are generated using a random number generator forming a realistic spatial ontology with 100 regions total. In particular, 50 more instances (representing points or regions) each one having one CSD-9 and one RCC-8 relation (i.e., also selected at random among the 9 CSD and 8 CSD relations) with existing instances are added into the ontology. Generating random instances and reasoning on the resulting ontology is repeated 10 times so that the reasoner response time reported below is the average over 10 runs.

Additional instances are generated using a random number generator forming a realistic spatial ontology with 100 regions total. In particular, 50 more instances (representing points or regions) each one having one CSD-9 and one RCC-8 relation (i.e., also selected at random among the 9 CSD and 8 CSD relations) with existing instances are added into the ontology. Generating random instances and reasoning on the resulting ontology is repeated 10 times so that the reasoner response time reported below is the average over 10 runs.

**VI. Conclusions and Future Work**

CHOROS is a qualitative spatial reasoning engine implemented in Java that extends PelletSpatial’s hybrid architecture to support the Cone-Shaped directional logic formalism (CSD) in addition to the Region Connection Calculus (RCC) of the native implementation. CHOROS implements path consistency separately for each one of the CSD-9 and RCC-8 relation models. Evaluation results on a realistic ontology with 100 regions demonstrated that spatial reasoning performs almost linearly with the number of regions in the input ontology in the average case. For an ontology specifically designed for the worst case, reasoning resembles close to cubic performance (as expected theoretically).

We also introduce SOWL, an implementation of the spatial reasoner under Protégé for both the RCC-8 and the CSD-9 relations can be inferred. This is achieved by asserting relations that (a) are not inconsistent and (b) reasoning over these relations yields relations between all pairs of individuals into the ontology. The $n$ relations where selected in such a way, so as $n^2$ consistent relations can be inferred from them. For this, $n$ regions are selected in a sequence so that, each one is included inside the next (i.e., $R_i NTPP R_{i+1}$) in the case of RCC-8 and, at the same time, each one is in the same direction (e.g., North) with respect to its next (i.e., $R_i N R_{i+1}$). Although the worst case is less likely to occur in practice, it is being applied for yielding an upper limit (worst-case) measurement of the response time of reasoning.

**VI. Conclusions and Future Work**

CHOROS is a qualitative spatial reasoning engine implemented in Java that extends PelletSpatial’s hybrid architecture to support the Cone-Shaped directional logic formalism (CSD) in addition to the Region Connection Calculus (RCC) of the native implementation. CHOROS implements path consistency separately for each one of the CSD-9 and RCC-8 relation models. Evaluation results on a realistic ontology with 100 regions demonstrated that spatial reasoning performs almost linearly with the number of regions in the input ontology in the average case. For an ontology specifically designed for the worst case, reasoning resembles close to cubic performance (as expected theoretically).

We also introduce SOWL, an implementation of the spatial reasoner under Protégé for both the RCC-8 and the CSD-9
relation calculi using SWRL and compared its performance with that of CHOROS. SOWL exhibited very good performance, close to that of the best implementation variant of CHOROS, in the average case (which is the expected for most practical applications). Nevertheless, CHOROS is much faster than SOWL in the worst case where SOWL actually failed to run, as the large number of inferred relations caused the computer to run out of memory.

Extending CHOROS to support qualitative temporal in addition to qualitative spatial relations is an interesting direction for future work. Notice that, SOWL supports arbitrary temporal and spatial relation models including projection-based and distance models already [9]. We are also planning to support reasoning beyond the base relations of each calculus. Dealing with scalability issues for large scale applications (i.e., ontologies) are also important issues for future research. Finally, we are planning to support OWL 2 restrictions on spatial relations (e.g., “a country A borders with exactly 3 other countries”).

Acknowledgment

This research leading to these results has received funding from the European Community’s Seventh Framework Program (FP7/2007–2013) under grant agreement No 296170 (Project Port.Dial).

REFERENCES


