Control and Cybernetics

vol. **41** (2012) No. 4

Temporal reasoning in trajectories using an ontological modelling approach[1]

by

Jamal Malki[1], Rouaa Wannous[1], Alain Bouju[1] and Cécile Vincent[2]

[1] La Rochelle University, L3i laboratory, EA 2118, France
[2] La Rochelle University, LIENSs laboratory, UMR 7266, France

Abstract: Nowadays, with growing use of location-aware, wirelessly connected, mobile devices, we can easily capture trajectories of mobile objects. To exploit these raw trajectories, we need to enhance them with semantic information. Several research fields are currently focusing on semantic trajectories to support inferences and queries to help users validate and discover more knowledge about mobile objects. The inference mechanism is needed for queries on semantic trajectories connected to other sources of information. Time and space knowledge are fundamental sources of information used by the inference operation on semantic trajectories. This article discusses new approach for inference mechanisms on semantic trajectories. The proposed solution is based on an ontological approach for modelling semantic trajectories integrating time concepts and rules. We present a case study with experiments, optimization and evaluation to show the complexity of inference and queries. Then, we introduce a refinement algorithm based on temporal neighbour to enhance temporal inference. The results show the positive impact of our proposal on reducing the complexity of the inference mechanism.

Keywords: trajectory, domain ontology modelling, time ontology, inference rules

1. Introduction

Over the last few years, there has been a huge collection of real-time data on mobile objects. These data are obtained by GNSS – Global Navigation Satellite System (GPS: Global Positioning System or ARGOS: Advanced Research and Global Observation Satellite), phone location or RFID (Radio Frequency Identification). This opens new perspectives for several applications like road traffic supervision and animal tracking. The raw data captured, commonly called trajectories, trace moving objects from a departure point to a destination point as sequences of pairs (sample points captured, time of the capture). In

[1]Submitted: October 2012; Accepted: November 2012

Spaccapietra et al. (2008), authors give a general definition of a trajectory: *A trajectory is the user defined record of the evolution of the position (perceived as a point) of an object that is moving in space during a given time interval in order to achieve a given goal.* Trajectories can be constrained to existing networks (Popa et al., 2011), or unconstrained, like in our study. Raw trajectories do not contain contextual information about moving objects like goals of travelling nor activities accomplished (Baglioni et al., 2008). Semantic trajectories are defined as a result of the annotation process on raw data with semantic annotations. This annotation process can be done automatically or manually. Semantic trajectories can be seen as a high-level data layer on raw trajectories (Yan et al., 2010). In Malki et al. (2009), to model semantic trajectories, a domain ontology is constructed to represent domain concepts and rules. Ontologies represent high level concepts, their properties and their interrelationships (Euzenat and Shvaiko, 2007). For that, it becomes necessary to provide mechanisms for storage, modelling, efficient analysis and knowledge extraction from these data.

In the continuation of our previous work, we discuss strategies for time integration with evaluation on generated and real data. We study seal trajectories and focus on semantic annotations for their activities such as foraging, travelling and resting. The inference mechanism on semantic trajectories is connected to time knowledge and has time and space storage complexity problems. This work addresses these two problems and gives some ideas for improving the complexity of the proposed approach.

This paper is organized as follows: Section 2 summarizes the state of the art on semantic trajectories and some recent related work. Section 3 describes our domain application and queries we aim to answer. Section 4 introduces the seal trajectory and time ontologies. Section 5 illustrates an implementation framework for the ontologies using a semantic data store. Section 6 presents the domain ontology rules and the temporal ontology rules. Section 7 defines the connection between seal trajectory and time ontologies. Section 9 evaluates the proposed approach while answering the real query and discusses the evaluation of the proposed approach. Finally, Section 10 concludes this paper and presents some future prospects.

2. Related work

Data management techniques including modelling, indexing, inferencing and querying large spatio-temporal data are actively investigated during the last decade (Yan et al., 2011). Most of these techniques are only interested in raw trajectories. In the state of the art, we notice two main views: conceptual modelling and moving objects. Both need spatio-temporal data modelling and reasoning.

Projects like GeoPKDD[1] and MODAP[2] emphasized the need to address and to use semantic data about moving objects for efficient trajectory analyses. Recently, new projects are born like MOVE[3] which aims at improving methods for knowledge extraction from massive amounts of moving objects data. For example, in Spaccapietra et al. (2008) bird migration trajectories are analysed for better understanding of bird behaviour. Scientists try to answer queries such as: where, why and how long birds stop on their travels, which activities they do during their stops, and what weather conditions the birds face during their flight. Considering these new requirements, new techniques appeared offering data models that can easily be expanded taking into account semantic data. The trajectory is seen as a user defined time-space function from a temporal interval to a space interval. To consider semantics of trajectories, a conceptual view is defined by three main concepts: stops, moves, and begin-end of a trajectory. Each part contains a set of semantic data. This model is implemented and evaluated on a relational database. Most domain and temporal operations are SQL based and use elementary data comparators. Based on this conceptual model of trajectories, several approaches have been proposed such as Baglioni et al. (2008), Bogorny et al. (2010).

Using ontologies for semantic spatio-temporal data modelling is a new research field. In Matthew (2008), authors work on a military application domain with complex queries that require sophisticated inferences methods. For this application, they present an upper-level ontology defining a general hierarchy of thematic and spatial entity classes and associating relationships to connect these entity classes. They intend for application-specific domain ontologies in the thematic dimension to be integrated into the upper-level ontology through subclassing of appropriate classes and relationships. Temporal information is integrated into the ontology by labelling relationship instances with their valid times. The author used the temporal and spatial dimensions which are included in the global ontology. Moreover, the ontology is formalised by the RDFS vocabulary and implemented on a relational database. Consequently, the inference mechanism is based on several domain specific table functions. The inference mechanism defined uses only the RDFS rules indexes. In Yan et al. (2010), authors design a conceptual model of trajectories from low-level real-life GNSS data to different semantically abstracted levels. Their application concerns daily trips of employees from home to work office and coming back. In Malki et al. (2012), authors define an ontological approach to modelling and reasoning on trajectories. This approach takes into account thematic, temporal and spatial rules. The ontologies constructed are formalised using both RDFS and OWL vocabulary. The inference mechanism is based on rules defined as entailments.

[1]GeoPKDD: Geographic Privacy-aware Knowledge Discovery and Delivery - European Project - http://www.geopkdd.eu

[2]MODAP: Mobility, Data Mining and Privacy - Coordination Action type project funded by EU, FET OPEN, 2009-2012 - http://www.modap.org

[3] MOVE: is an Action of the COST Programme (European Cooperation in Science and Technology) funded in the period of 11/2009 to 10/2013 by the European Science Foundation - http://move-cost.info/

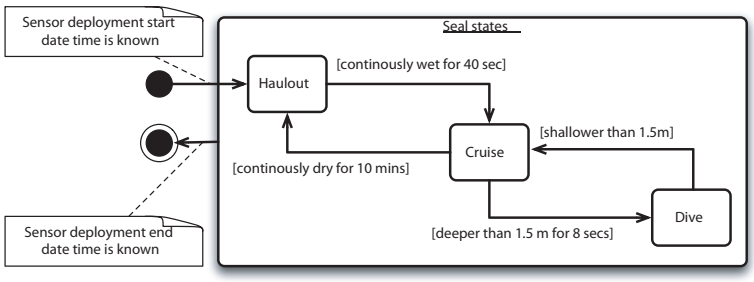

Figure 1. The three states of seal trajectory

Finally, in Malki et al. (2009), authors present time knowledge integration using inference mechanism on semantic trajectories. Nevertheless, this work did not mention evaluation of the proposed approach. The present work addresses limitations and gives experiments and evaluations of the performance problems of ontological time integration on trajectories.

3. Application domain

3.1. Seal trajectory data model

As in Malki et al. (2012), this paper considers trajectories of seals. The data comes from the LIENSS[4] (CNRS/University of La Rochelle) in collaboration with SMRU[5]. These laboratories work on marine mammal ecology. Trajectories of seals between their haulout sites along the coasts of the English Channel or in the Celtic and Irish seas are captured using GNSS systems provided by SMRU Instrumentation. We use trajectories data coming from GPS/GSM tags. The captured spatio-temporal data of seal trajectories can be classified into three main states: haulout, cruise and dive. Fig. 1 shows the three states, the transitions and their guard conditions (Malki et al., 2009).

3.2. Semantic seal trajectory

We focus on studying seals' activities to identify, for example their foraging areas. The main activities of seal, like foraging, travelling and resting, occur in parts of trajectory related to seal states. We aim at answering queries, such as:

1. Foraging activities.
2. Foraging activities during a given time interval.
3. Foraging activities performed after travelling during a given time interval.

For all these queries, we have to define a seal trajectory domain rule called `Foraging`. However, for the last two queries, time rules must be defined be-

[4]http://lienss.univ-larochelle.fr
[5]SMRU: Sea Mammal Research Unitm, http://www.smru.st-and.ac.uk

Table 1. Domain, time concepts and rules needed for answering the query 3

Concepts and rules			Description
Concepts	Domain	Dive	specific part of the seal trajectory
	Time	Temporal interval	the given temporal interval
Rules	Domain	Travelling	seal activity
		Foraging	seal activity
	Time	After	temporal relationship between the two activities
		During	temporal relationship between activity and time interval

tween trajectory parts. For example, query 3 needs `Foraging` and `Travelling` domain rules and `During` time rule as illustrated by Table 1.

4. Ontological modelling approach

4.1. Seal trajectory ontology

Seal trajectory ontology, called `owlSealTrajectory`, is a result of a model transformation of the semantic seal trajectory. An extract of this ontology concepts and properties is shown in Fig. 2, where:

- `Seal`: is a mobile object. It represents the animal equipped with a tag.
- `Sequence`: captures in the form of temporal intervals a spatial part called `GeoSequence` and can be `Haulout`, `Cruise` or `Dive`. Metadata parts are called `Summary` and `CTD` (Conductivity-Temperature-Depth).
- `Trajectory`: is a logical form to represent a set of sequences.
- `Activity`: is the seal activity for a sequence or for a trajectory.

Besides these concepts, `owlSealTrajectory` defines relationships like:

- `seqHasActivity`: is the object property between an activity and a sequence.
- `isSeqOf`: is the object property between a trajectory and a sequence.
- `s_date` and `e_date`: are data properties for beginning and ending time, respectively.
- `dive_dur, sur_dur` and `max_depth`: are dive duration, surface duration and maximum depth of the dive, respectively.
- `TAD`: is Time Allocation at Depth which defines the shape of a seal's dive, as mentioned in Fedak et al. (2001).

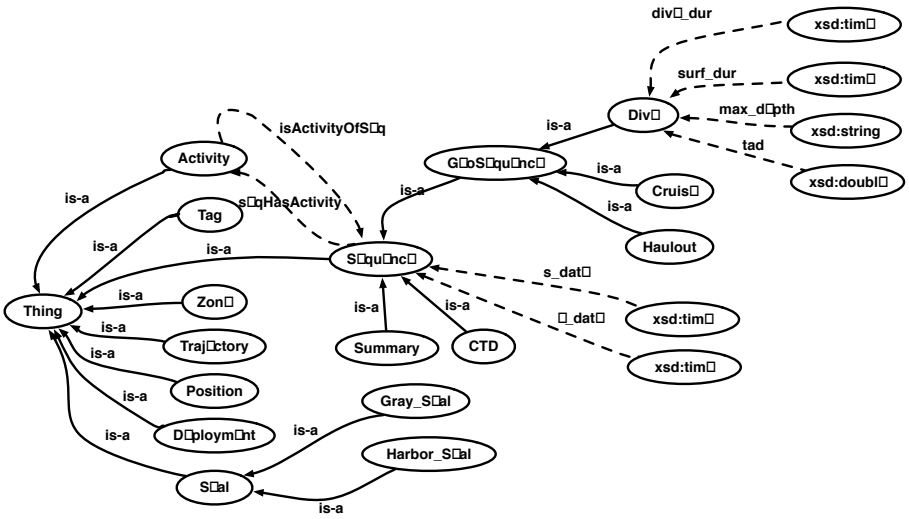

Figure 2. An extract of `owlSealTrajectory` ontology

4.2. Time ontology

Table 1 clearly highlights the need of temporal concepts as well as temporal relationships between these concepts. In our approach, we chose `owlTime` [6] ontology (**?**) developed by the World Wide Web Consortium (W3C). An extract of the declarative part of this ontology is shown in figure 3 described in detail in Jerry and Feng (2004). We are mainly interested in the `ProperInterval` concept and its two properties `hasBeginning` and `hasEnd`.

5. Semantic data store

The built ontologies `owlSealTrajectory` and `owlTime` are based on the Ontology Web Language (OWL) witch is a vocabulary extension of the Resource Description Framework (RDF). Therefore, our ontological data can be seen as a set of RDF triples, also known as `triplestore`. Many semantic stores (including Jena, 2000; Oracle, 2012; Vituoso RDF Triple Store, Virtuoso, 2006; Open RDF.org, 2007, and C-Store, Abadi et al., 2007) use database to store and manage RDF data. In this work, we use a semantic store implemented in the Oracle Spatial Database 11g.

5.1. Semantic data storage in Oracle

RDF data store in Oracle 11g is built on the top of Oracle Spatial Network Data Model (NDM). NDM is an Oracle solution for storing, managing and analysing

[6] `http://www.w3.org/2006/time`

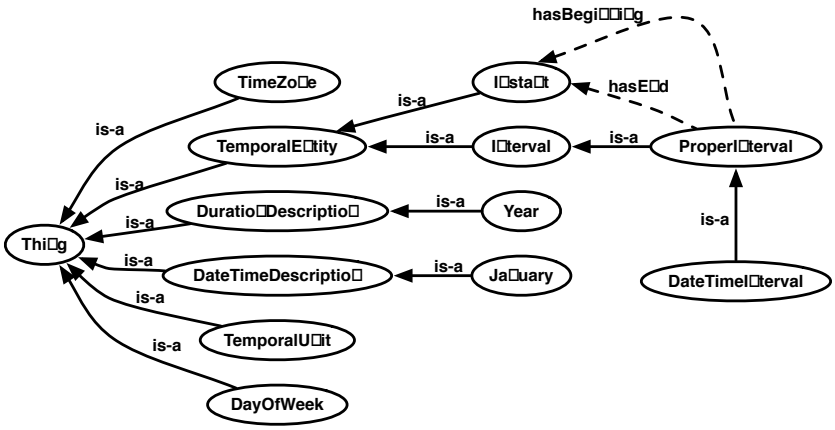

Figure 3. A view of owlTime ontology

networks or graphs in the database. RDF graphs are modelled as a directed logical network in NDM. A set of triples is known as an RDF graph or a model. Each RDF statement is represented using a triple where:

- Subject: is represented by a URI or a blank node.
- Predicate or Property: is represented by a URI.
- Object: is represented by a URI, a blank node, or a literal.

Subjects and objects are mapped to nodes, and predicates are mapped to links that have subject and object as start-nodes and end-nodes, respectively.

In Oracle, two new semantic datatypes are defined for RDF-modeled data:

- SDO_RDF_TRIPLE: defined to serve as triple representation of RDF data (subject, predicate, object).
- SDO_RDF_TRIPLE_S: defined to store persistent data in the database (the _S for storage). This type has references to the data, because the semantic data is stored only in the central RDF schema. It has methods to retrieve the entire triple or part of the triple.

The SDO_RDF_TRIPLE type provides a triple view of the data and the type SDO_RDF_TRIPLE_S stores the IDs of the triple.

5.2. Loading semantic data in Oracle

In the Oracle semantic data store, both the declarative part of an ontology and its individuals can be inserted by the same way and even in the same place. The following prerequisites are needed to load RDF data in the database:

1. Creating a tablespace: recommended for all RDF data tables, since RDF data store tends to be very large.
2. Create an RDF network: enables RDF store in Oracle database.
3. Create a table to store references to RDF data: must contain a column of type SDO_RDF_TRIPLE_S, which will contain references to all data

associated with a single RDF model. It is recommended that this table include a column named `TRIPLE` of the type `SDO_RDF_TRIPLE_S`.

4. Create an RDF model: created by specifying a model name, the table name to hold references to RDF data for the model and the column of the type `SDO_RDF_TRIPLE_S` in that table.

Loading RDF data in Oracle database semantic store supports three forms of loading (Das and Srinivasan, 2009):

- Bulk loading: a highly optimized method for loading medium to large number (e.g., billions) of triples.
- Batch loading: an optimized method to handle loading a medium number (e.g., a few million) of triples. Its advantage is that, unlike bulk loading, does not require object values to stay within 4000 bytes.
- Loading via SQL INSERT into the application table: a recommended method for small number (e.g., up to a few thousands) of triples.

For bulk loading and batch loading, only N-Triple format file-based input is supported. SQL INSERT requires use of an object type constructor, `SDO_RDF_TRIPLE_S`, with target RDF model name and lexical values for subject, predicate and object components of the triple used as arguments.

5.3. Semantic data and relational database mapping

The majority of data underpinning the Web are stored in Relational DataBases (RDB) with their proven track record of scalability, efficient storage, optimized query execution and reliability. As compared to the relational data model, RDF is a more expressive data model and data expressed in RDF can be interpreted, processed and reasoned over by software agents (Sahoo et al., 2009). This is why the strategies for mapping relational data to RDF abound. The direct mapping defines an RDF graph representation of the data in a relational database. It can be seen as a transformation witch takes as input a relational database (data and schema) and generates an RDF graph. In our work, we use the D2RQ Mapping Language. D2RQ is a declarative language for mapping relational database schemas to RDF vocabularies and OWL ontologies. The language is implemented in the D2RQ Platform. Fig. 4 illustrates the RDB-RDF mapping process in our work.

5.4. The inference

The inference can be simply characterized as a process of discovering new relationships. Otherwise, inference means that automatic procedures can generate new relationships based on the data and based on some additional information in the form of a vocabulary, e.g., a set of rules or rulebase. Inferencing, or computing entailment, is a major attribute of semantic technologies that differentiates it from other relevant technologies. We can distinguish two entailment regimes (Souripriya and Jagannathan, 2009):

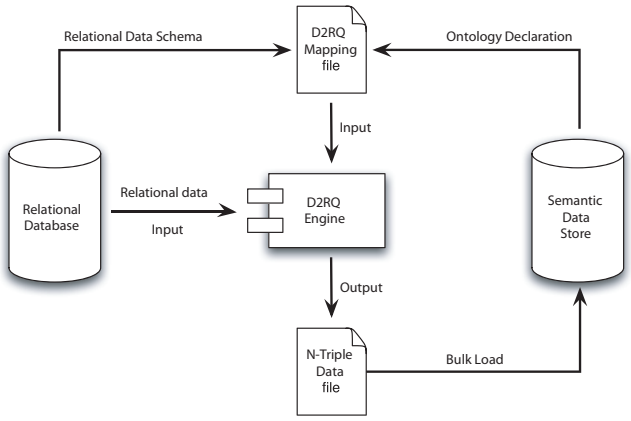

Figure 4. RDB to RDF mapping process using D2RQ

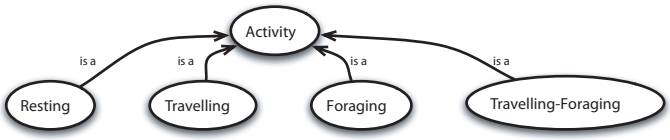

Figure 5. Declarative part of seal activities

- Standard entailment regimes: there are several standard entailment regimes: semantics of RDF, RDFS and OWL. Support for RDF and RDFS is simplified by the availability of axioms and rules that represent their semantics. Support for major subsets of OWL-Lite and OWL-DL vocabularies is provided in current semantic stores. In Oracle Semantic Data Store, this subset is called OWLPRIME.
- Custom entailment regimes: since the standard vocabularies cannot handle the full repertoire of custom entailment regime a semantic application may require, it becomes important to provide support for entailment based on arbitrary user-defined rules. In our work, we define two user-rulebases: domain seal trajectory rules and temporal rules.

6. Domain and temporal ontological rules

6.1. Domain seal trajectory rules

We define four seal activities during their trajectory: resting, travelling, foraging and travelling-foraging. In our approach, each seal activity is defined in the ontology and has both a declarative and an imperative corresponding parts. Fig. 5 shows the declarative part.

The imperative parts are based on the decision table (Table 3). This decision

Table 3. Decision table associated with seal activities

Rules	**Max dive depth** (meter)	**Dive shape** or TAD	**Surface ratio** = surface dur/dive dur
Resting	< 10	all	> 0.5
Travelling	> 3	> 0 & < 0.7	< 0.5
Foraging	> 3	> 0.9 & < 1	< 0.5
Travelling_Foraging	> 3	> 0.7 & < 0.9	< 0.5

table shows the classification of seal activities based on parameters and considerations established by the domain expert. To implement the imperative parts in Oracle Semantic Data Store, we create the rule base `sealActivities_rb` to hold the implementation of each activity. As an example, Code 1 shows the implementation of `foraging_rule`. Referring to value conditions in Table 3, in Code 1, line 6 checks if the maximum dive depth is more than 3 meters, the TAD is 0.9 and the surface duration divided by the dive duration `v`, if smaller than 0.5.

```
1 EXECUTE SEM_APIS.CREATE_RULEBASE('sealActivities_rb');
2 INSERT INTO mdsys.semr_sealActivities_rb
3 VALUES( 'foraging_rule',
4 '(?diveObject rdf:type ost:Dive)
5 (?diveObject ost:tad ?tad)(?diveObject ost:max_depth ?maxDepth)(?diveObject
      ost:surf_dur ?surfaceDur)(?diveObject ost:dive_dur ?diveDur)',
6 '(maxDepth > 3) and (tad > 0.9) and (surfaceDur/diveDur < 0.5)',
7 '(?diveObject ost:seqHasActivity ?activityProberty) (?activityProberty rdf:type
       ost:Foraging)',
8 SEM_ALIASES(SEM_ALIAS('ost', 'http://l3i.univ-larochelle.fr/Sido/
      owlSealTrajectory#')));
```

Code 1. The imperative part of the seal activity `foraging`

6.2. Time ontology rules

The `owlTime` ontology declares 13 relationships based on Allen algebra (Allen, 1983). These are: `intervalEquals`, `intervalBefore`, `intervalDuring`, `intervalOverlaps`, `intervalStartedBy`, `intervalOverlappedBy`, `intervalFinishes`, `intervalFinishedBy`, `intervalContains`, `intervalMetBy`, `intervalStarts`, `intervalMeets`, `intervalAfter`. We implement the rule base `owlTime_rb` to hold interval temporal relationships. For example, Code 2 presents the implementation of the imperative part of the `intervalAfter_rule` based on operations defined in the table `TM_RelativePosition` of the ISO/TC 211 specification of temporal schema ISO–TC—-201, 2002. In Code 2, line 6 expresses the condition:

$$\texttt{sself.begin.position} > \texttt{other.end.position}$$
$$\text{where}$$
$$\begin{cases} \texttt{self} & = \texttt{timeObject2} \\ \texttt{other} & = \texttt{timeObject1} \\ \texttt{self.begin.position} & = \texttt{BeginTime2} \\ \texttt{other.end.position} & = \texttt{EndTime1} \end{cases}$$

```
1 EXECUTE SEM_APIS.CREATE_RULEBASE('owlTime_rb')
2 INSERT INTO mdsys.semr_owltime_rb
3 VALUES('intervalAfter_rule',
4 '(?timeObject1 rdf:type owltime:ProperInterval)(?timeObject1 owltime:hasEnd ?
      End1)(?End1 :inXSDDateTime ?EndTime1)
5 (?timeObject2 rdf:type owltime:ProperInterval)(?timeObject2 owltime:
      hasBeginning ?Begin2)(?Begin2 :inXSDDateTime ?BeginTime2)',
6  '(BeginTime2 > EndTime1)',
7 '(?timeObject2 owltime:intervalAfter ?timeObject1)',
8 SEM_ALIASES(SEM_ALIAS('owltime','http://www.w3.org/2006/time#')));
```

Code 2. The imperative part of `intervalAfter` temporal rule

7. Semantic integration by ontological mapping

The need of semantic integration is fundamental when considering different and independent sources of knowledge, like seal trajectory and time ontologies in our work. This ontological mapping is necessary for expressing semantic queries evolving different kind of information and in our domain application may lead to discovering more semantic trajectory patterns. In Choi,Song and Han (2006), we find an interesting survey on ontology mapping.

Mapping between two ontologies can be done by considering the hierarchy concepts with the `rdfs:subClassOf` property. This built-in RFDS is not appropriate in separate and heterogeneous ontologies. The mapping can also be done by the built-in OWL property `owl:sameAs` that links an individual to an individual. However, it does not go further for their properties. Consequently, the properties `owl:equivalentClass` and `owl:equivalentProperty` are the most appropriate connection in our case. The Oracle Semantic Data Store reasoner takes into account the OWL property `owl:equivalentClass` which allows the inference that each `Sequence` is equivalent to a `ProperInterval`. Therefore, the interval temporal rules are also valid for trajectory sequences, which means valid for dives as well. The mapping process, shown in Fig. 6, follows these steps:

1. `owlSealTrajectory:Sequence` is mapped by the OWL construct `owl:equivalentClass` to `owltime:ProperInterval`.
2. `owlSealTrajectory:s_date` is mapped by OWL construct `owl:equivalentProperty` to `owlTime:hasBeginning`.
3. `owlSealTrajectory:e_date` is mapped by the OWL construct `owl:equivalentProperty` to `owlTime:hasEnd`.

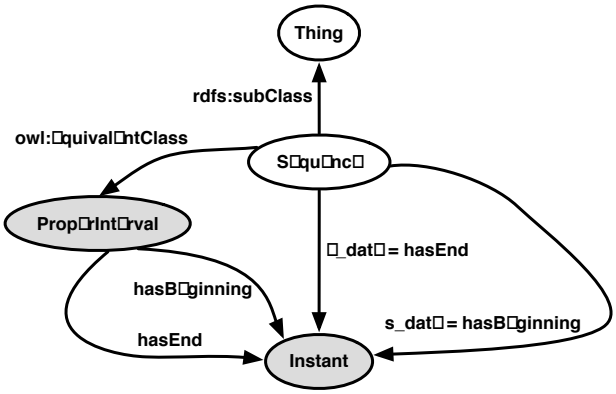

Figure 6. Connecting `owlSealTrajectory` to `owlTime`

8. Temporal rule refinement

The inference mechanism is needed for queries on the trajectory ontology mapped to the time ontology. Calculating the inference between all sequences of trajectories considering all temporal rules takes a lot of computation time and a large storage space (Figs. 7 and 8). To enhance the inference process, we define a refinement called *temporal neighbour refinement*. A temporal neighbour is a conceptual distance between two neighbouring sequences. The goal of this refinement, Algorithm 1, is to optimize the distance between two sequences in order to calculate the corresponding temporal relationships. In this solution, the optimization coefficient depends on the application domain and can be estimated after considering the statistical dispersion data.

Algorithm 1 Temporal neighbour refinement algorithm

Require: Two sequences: a referent S_r and an argument S_a
Require: A neighbour of S_r
Ensure: Temporal rule between S_r and S_a
  if $S_a \in$ to the neighbour of S_r **then**
    calculate the temporal rule between S_r and S_a
  end if
  go the next sequence S_{a+1}

9. Evaluation and analysis

We performed experiments with the aim of measuring the impact of the introduction of temporal refinement in the inference process calculation. For this, we consider two sets of data: generated data and GPS-GSM real data. In the case of real data, we consider trajectory data of one seal. For the experiments,

we consider query 3 given in Section 3.2 as:

```
Find foraging activities performed after travelling
activities during the time interval:
[2007-08-02T00:00:00, 2007-08-09T23:59:00]
```

Code 3 gives the SQL formulation of the query, where the Oracle table function SEM_MATCH (line 2 of Code 3) is used to extend SQL with SPARQL.

Figs. 7 and 8 show experiment results for the computation time in seconds and the storage space in triples needed by the inference calculation. The evolution curves are given by the number of dives. In all the following experiments, shown in Figs. 7 and 8, we consider the domain rules:

1. The experiment named *Temporal rules* analyses the inference on real data taking into account classic version of temporal rules.
2. The experiment named *Temporal rules refined - Real data* analyses the inference on real data considering the refinement of temporal rules given by Algorithm 1.
3. The experiment named *Temporal rules refined - Generated data* analyses the inference on generated data as in the previous experiment.

```
1  SELECT Dive1, Dive2
2  FROM TABLE (SEM_MATCH(
3  '(?Dive1 rdf:type ost:Dive) (?Dive1 ost:sequenceHasActivity ?activiteD1) (?
       activiteD1 rdf:type ost:Foraging)
4  (?Dive2 rdf:type ost:Dive) (?Dive2 ost:sequenceHasActivity ?activiteD2) (?
       activiteD2 rdf:type ost:Travelling)
5  (?Dive1 ot:intervalAfter ?Dive2)
6  (?timeObject rdf:type ot:ProperInterval) (?timeObject ot:hasBeginning ?
       beginTime) (?beginTime ot:inXSDDateTime "2007-08-02T00:00:00"^^xsd:
       datetime) (?timeObject ot:hasEnd ?endTime)(?endTime ot:inXSDDateTime
       "2007-08-09T23:59:00"^^xsd:datetime)
7  (?Dive1 ot:intervalDuring ?timeObject)(?Dive2 ot:intervalDuring ?timeObject)',
8  SEM_Models('owlSealTrajectory','owlTime'),
9  SEM_Rulebases('OWLPRIME','sealActivities_rb','owlTime_rb'),
10 SEM_ALIASES(SEM_ALIAS('ost', 'http://l3i.univ-larochelle.fr/Sido/
       owlSealTrajectory#'),
11 SEM_ALIAS('ot','http://www.w3.org/2006/time#')),
12 null));
```

Code 3. The SQL code of the query 3

It clearly appears that Experiment 1 gives the inference result with poor characteristics in terms of computation time and space storage. For example, for 500 dives, the inference takes around 67,000 seconds ($\simeq$ 18.5 hours) and generates 2,300,000 triples. This problem occurs because of time integration without applying any domain constraints on temporal rules. In this work, we propose a first solution to this problem by defining a domain constraint on temporal intervals based on the conceptual distance in the ontology hierarchy. This constraint limits the calculation of temporal rules into the neighbourhood of the current interval. From the seal trajectory domain and with our biological feedback, the candidate for the conceptual distance between two sequences is five minutes (300 seconds). So, we modify the implementation of the temporal rules considering this candidate. For instance, the implementation of the

intervalAfter_Refined rule, is given by Code 4.

In Experiment 2, we consider real GPS/GSM data and the inference uses the refined temporal rules. The computation time and space storage results show the improvement made on the inference calculation compared to Experiment 1. For example, for 500 dives, the inference takes less than 30,000 seconds ($\simeq$ 8 hours) and generates less than 1,100,000 triples. In Experiment 3, the inference is calculated on generated data and uses the refined temporal rules. Generated data contain temporal intervals with the same initial density for temporal relationships. The results show reasonable computation time and space storage taken by the inference mechanism. These experiments provide a view of the inference behaviour while considering independent or neutral data.

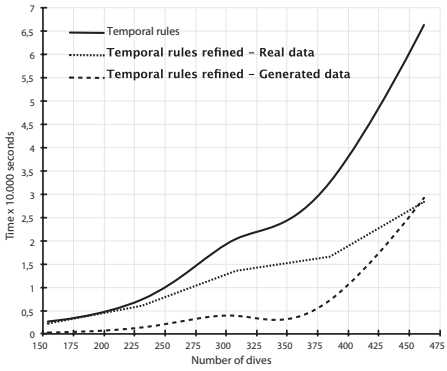

Figure 7. Inference computation time with(out) the temporal rules refinement

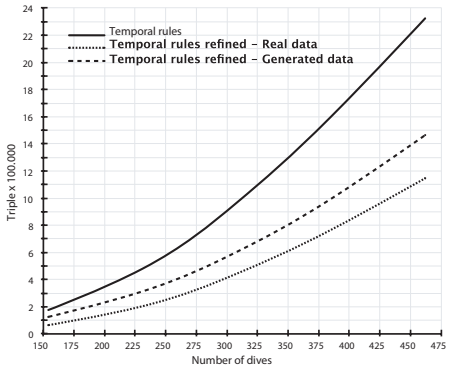

Figure 8. Inference storage space taken with(out) the temporal rules refinement

```
1 EXECUTE SEM_APIS.CREATE_RULEBASE('owlTime_rb')
2 INSERT INTO mdsys.semr_owltime_rb VALUES(
3 'intervalAfter_Refined_rule',
4 '(?timeObject1 rdf:type owltime:ProperInterval)(?timeObject1 owltime:hasEnd ?
      End1)(?End1 :inXSDDateTime ?EndTime1)
5 (?timeObject2 rdf:type owltime:ProperInterval)(?timeObject2 owltime:
      hasBeginning ?Begin2)(?Begin2 :inXSDDateTime ?BeginTime2)',
6 '(BeginTime2 > EndTime1)
7 ((timeIntervalLengthInSeconds(dateTime2TimeStamp(EndTime1),
      dateTime2TimeStamp(BeginTime2))<300)',
8 '(?timeObject2 owltime:intervalAfter_Refined ?timeObject1)',
9 SEM_ALIASES(SEM_ALIAS('owltime','http://www.w3.org/2006/time#')));
```

Code 4. Create the temporal intervalAfter_Refined rule

10. Conclusion and future work

Trajectories are usually available as raw data and lack semantics which is of fundamental importance for their efficient use. In this work, we present an ontological based approach for modelling of semantic trajectories. We describe our domain application, the underlying captured data and the trajectory data

model. We illustrate the need for time reasoning while studying the domain queries. From the principle of reusing existing ontologies, we show the time ontology. We give declarative and imperative parts of the introduced ontologies. We discuss the problem of connection between ontologies and show the implemented solution. We choose an environment to implement and test our proposals based on a database management system. Then, we provide the main technical steps for this implementation. The main contributions of this work are:

- How to use an ontological approach to modelling for semantic trajectories.
- How to connect separated and different needed ontologies.
- How to define inferences on semantic trajectories to answer user queries.
- What are the costs of these inferences.
- What can we do to face and reduce the inferences costs.

We evaluate our approach on generated and real GPS/GSM data. The experimental results verify the positive impact of the temporal neighbour refinement as a domain constraint to reduce the complexity of the inference calculation. In conclusion, we propose a first possible solution based on domain constraints to define driven inference process. In our future work, we will explore other approaches to define a class of driven inference processes to improve the feasibility of such solutions in business applications.

11. Reference

Abadi, D.J., Marcus, A., Madden, S.R. and Hollenbach, K. (2007) Scalable semantic web data management using vertical partitioning. In: *Proc. of the 33rd International conference on very large data bases*, VLDB, 411-422.

Allen, J.F. (1983) Maintaining knowledge about temporal intervals. *Commun. ACM* **26**(11), 832-843.

Baglion, M., Macedo, J., Renso, C. and Wachowicz, M. (2008) An ontology-based approach for the semantic modelling and reasoning on trajectories. In: *Advances in Conceptual Modeling - Challenges and Opportunities* **5232**, Springer Berlin/Heidelberg, 344-353.

Bizer, C. (2004) D2rq-treating non-RDF databases as virtual RDF graphs. In: *Proceedings of the 3rd International Semantic Web Conference* (ISWC 2004).

Bogorny, V., Heuser, C. and Alvares, L. (2010) A conceptual data model for trajectory data mining. In: *Geographic Information Sci.* **6292**, Springer Berlin/Heidelberg, 1-15.

Choi, N., Song, I. and Han, H. (2006) A survey on ontology mapping. *SIGMOD Rec.* **35**(3), 34-41.

Das, S. and Srinivasan, J. (2009) Database technologies for RDF. In: *Reasoning Web. Semantic Technologies for Information Systems,* Springer Berlin/Heidelbergr, 205-221.

Euzenat, J. and Shvaiko, P. (2007) *Ontology matching*. Springer-Verlag.

Fedak, M.A. Lovell, P. and Grant S.M. (2001) Two approaches to compressing and interpreting time-depth information as collected by time-depth recorders and satellite-linked data recorders. *Mar Mamm Sci* **17**(1), 94-11.

ISO-TC-211 (2002) Geographic information - temporal schema, ISO 19108: 2002.

Jena (2000) A semantic web framework for java, http://jena.sourceforge.net.

Jerry, R.H. and Feng, P. (2004) An ontology of time for the semantic web. In: *ACM Transactions on Asian Language Information Processing* **3**, 66-85, http://www.w3.org/2006/time.

Malki, J., Mefteh, W. and Bouju, A. (2009) Une approche ontologique pour la modélisation et le raisonnement sur les trajectoires. Prise en compte des règles métiers, spatiales et temporelles. In: *JFO 2009 3ème édition des journées Francofones sur les Ontologies*, 157-168.

Malki, J., Bouju, A. and Mefteh, W. (2012) An ontological approach modeling and reasoning on trajectories. Taking into account thematic, temporal and spatial rules. In: *TSI. Technique et Science Informatiques* **31, 71-96.**

Matthew, P. (2008) A framework to support spatial, temporal and thematic analytics over semantic web data. PhD thesis, Wright State University.

OpenRDF.org. (2007) Sesame, http://www.openrdf.org/.

Oracle (2012) Oracle Database Semantic Technologies Developer's guide 11g release 2, http://www.oracle.com/technology/tech/semantic-technologies.

Popa, I.S., Zeitouni, K., Oria, V., Barth, D. and Vial, S. (2011) Indexing in network trajectory flows. In: *The VLDB Journal* **20**, 643-669. Springer-Verlag.

Sahoo, S.S., Halb, W., Hellmann, S., Idehen, K., Thibodeau, T. and Auer, S. (2009) A survey of current approaches for mapping of relational databases to RDF, W3C RDB2RDF Incubator Group.

Sequeda, J., Tirmizi, S.H., Corcho, Ó. and Miranker, P. (2011) Survey of directly mapping sql databases to the semantic web. In: *Knowledge Eng.* **26**, 445-486.

Souripriya, D. and Jagannathan, S. (2009) Database technologies for RDF. In: *Reasoning Web. Semantic Technologies for Information Systems* **5689**, 205-221, Springer.

Spaccapietra, S., Parent, C., Damiani, M., Demacedo, J., Porto, F. Vangenot, C. (2008) A conceptual view on trajectories. In: *Including Special Section, Privacy Aspects of Data Mining Workshop* **65**, 126-146.

Virtuoso (2006) Advances in virtuoso RDF triple storage (bitmap indexing), http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main.

Wannous, R., Malki, J., Bouju, A. and Vincent, C. (2013) Time integration in semantic trajectories using an ontological modelling approach. In: *New Trends in Databases and Information Systems* **185**, 187-198. Springer Berlin/Heidelberg.

Yan, Z., Parent, C., Spaccapietra, S. and Chakraborty, D. (2010) A hybrid model and computing platform for spatio-semantic trajectories. In: *The Semantic Web: Research and Applications* **6088**, 60-75. Springer Berlin/Heidelberg.

Yan, Z., Chakraborty, D., Parent, C., Spaccapietra, S. and Aberer, K. (2011) SeMiTri: A framework for semantic annotation of heterogeneous trajectories. In: *Proceedings of the 14th ACM International Conference on Extending Database Technology*, 259-270.