DRAOn: A Distributed Reasoner for Aligned Ontologies

Chan Le Duc[1], Myriam Lamolle[1], Antoine Zimmermann[2], and Olivier Curé[3]

[1] LIASD Université Paris 8 - IUT de Montreuil, France
{chan.leduc, myriam.lamolle}@iut.univ-paris8.fr
[2] École Nationale Supérieure des Mines, FAYOL-ENSMSE, LSTI, F-42023 Saint-Étienne, France antoine.zimmermann@emse.fr
[3] LIGM Université Paris-Est, France
ocure@univ-mlv.fr

Abstract. DRAOn is a distributed reasoner which offers inference services for a network of OWL ontologies correlated by alignments. Reasoning with such networks of ontologies depends on the semantics we define for alignments with respect to ontologies. DRAOn supports two semantics for a network of ontologies: the standard Description Logics (DL) semantics for non-distributed reasoning, and the Integrated Distributed Description Logics (IDDL) semantics for distributed reasoning. Unlike the DL semantics where alignments are considered as inter-ontology axioms, the IDDL semantics interprets alignments as correspondences enabling to propagate non-emptiness (always satisfiable) and unsatisfiability of atomic concepts from an ontology to another one. Consequently, this makes distributed reasoning for a network of ontologies possible since consistency of the whole network can be decided from consistency of each ontology with axioms built from alignments such that these axioms ensure just necessary propagations of knowledge.

1 Introduction

We present DRAOn, a reasoner for a distributed network of aligned ontologies. The goal of DRAOn is to be able to reason on a set of independently developed ontologies that may overlap in concepts but are following different modelling perspectives, granularity, coverage, etc. To be able to do that, we assume that there exists explicit correspondences between different ontologies, that have been built buy automatic ontology matchers, by humans, or partly by both. Therefore, the structure that DRAOn is reasoning with is not a monolithical theory, but a network of aligned ontologies. Moreover, DRAOn is able to make use of existing reasoners as black boxes for some of its tasks.

The issues faced when reasoning with heterogeneous, distributed ontologies, even when they have been already matched, are twofold: (a) the semantics of cross-ontology correspondences can be tricky to define when ontologies have different modelling perspectives because a correspondence of terms from different ontologies does not necessarily translates directly to an axiom in the ontology language (or languages); and (b) the distribution of ontologies, and possibly of reasoning services associated with them may influence the design of the global reasoner.

2 Background and related work

In this paper, we assume that ontologies are all Description Logic theories and we assume some familiarity with DLs. We simply repeat that DL ontologies are interpreted according to an interpretation function $\cdot^{\mathcal{I}}$ over a domain $\Delta^{\mathcal{I}}$, and interpretations that satisfy all the axioms of an ontology are called *models* of the ontology. There are a number of existing OWL DL reasoners such as HermiT [1], Pellet [2] or FacT++ [3].

In order to use several ontologies for reasoning, we assume the existance of *correspondences* between ontologies of the form (e_i, e_j, r), where e_i is a term from one ontology, e_j a term from another ontology, and r denotes a fixed binary relation such as equality, subsumption, disjointness, etc. In this paper, only correspondences where r is $=$ (equality of individuals), $\in$ (class membership) or $\sqsubseteq$ (subsumption) will be considered. A set of correspondences between a pair of ontologies is an *ontology alignment* (or alignment for short). A set of ontologies together with their pairwise alignments is called a *network of aligned ontologies* (NAO). We will use bold face to denote sets, and therefore, the notation for an NAO will generally be $\langle \mathbf{O}, \mathbf{A} \rangle$.

Reasoning with NAOs strongly depends on the semantics of alignments. We say that a certain semantics of alignments defines a *distributed logic* (sometimes called "contextual logic" or even "modular ontology language"). In a distributed logic L, correspondences map to formula of the logic in question, so we will use a fonction τ_L to denote the logical formula associated to the correspondence. For instance, the simplest form of distributed logic is normal DL. An NAO can be interpreted as a DL ontology by mapping $(e_i, e_j, \sqsubseteq)$ (resp. $(e_i, e_j, =)$, $(e_i, e_j, \in)$) to $\tau_{\mathrm{DL}}(e_i, e_j, \sqsubseteq) = e_i \sqsubseteq e_j$ (resp. $e_i = e_j, e_j(e_i)$). Reasoning over an NAO is then the same as reasoning with an ontology formed by the union of all the ontologies in the NAO, and the axioms corresponding to the correspondences.

A number of authors, including ourselves, have argued that this is not satisfying when dealing with heterogeneous ontologies. Therefore, several other distributed logics were proposed. Here we focus on the ones that are adopting a Local Model Semantics (LMS [4]), among which we find Distributed Description Logics (DDL [5]), $\mathcal{E}$-connections [6], and IDDL [7]. LMS states that ontologies in an NAO have to be interpreted separately (*i.e.,* an interpretation of an NAO has a set of DL interpretations) and the NAO is satisfied if each ontology is locally satisfied, and some extra knowledge, such as alignments, can constrain further that satisfying models of the NAO. In DDL, a correspondence is interpreted as a *bridge rule* $\tau_{\mathrm{DDL}}(e_i, e_j, \sqsubseteq) = i{:}e_i \xrightarrow{\sqsubseteq} j{:}e_j$ which intuitively states that, from the view point of e_j's ontology, the term e_i is a subclass of the term e_j. In $\mathcal{E}$-connections, correspondences can take many more forms but can be translated into axioms using *links*, that are terms similar to DL roles, but which denote binary relations between elements of different interpretations domains. More precisely, $\tau_{\mathrm{Econn}}(e_i, e_j, \sqsubseteq) = e_i \sqsubseteq \exists \langle L \rangle e_j$, where L is a link, says that the elements of e_i in the first ontology are in relationship with the elements of e_j in the second ontology, via link L. Such a link axiom would be part of ontology e_i, which means that it represents a fact from e_i's ontology point of view.

As opposed to $\mathcal{E}$-connections and DDL, which treats correspondences as subjective views of an ontology wrt another ontology's terms, we proposed Integrated Distributed Description Logics (IDDL) as a distributed logic that assumes local ontologies

are agnostic to each others, and correspondences express knowledge of a mediation view that ties together the ontologies. In IDDL, $\tau_{\mathrm{DDL}}(e_i, e_j, \sqsubseteq) = i{:}e_i \xleftrightarrow{\sqsubseteq} j{:}e_j$ is simply called a correspondence. An IDDL interpretation of an NAO $\langle \mathbf{O}, \mathbf{A} \rangle$ is a structure $\langle \mathbf{I}, \boldsymbol{\epsilon} \rangle$ such that $\mathbf{I} = \{\mathcal{I}_o\}_{o \in \mathbf{O}}$ is a set of DL interpretations of the ontologies, and $\boldsymbol{\epsilon} = \{\epsilon_o : \Delta^{\mathcal{I}_o} \rightarrow \Delta\}_{o \in \mathbf{O}}$ is a set of functions (so called *equalising functions*) from local domains of interpretations to a set called the global domain. The composition of a local interpretation function $\cdot^{\mathcal{I}_i}$ with its corresponding equalising function ϵ_i defines the global interpretations of local terms. A correspondence $i{:}e_i \xleftrightarrow{\sqsubseteq} j{:}e_j$ is satisfied iff $\epsilon_i(e_i^{\mathcal{I}_i}) \subseteq \epsilon_j(e_j^{\mathcal{I}_j})$.[4]

There are implementations of DDL (Drago, a tableau-based peer-to-peer reasoner [8]) and $\mathcal{E}$-connections (embedded in an earlier version of Pellet. DRAOn, the reasoner described here, implements both the classical DL semantics of NAOs, and the IDDL semantics, according to our algorithm described in [9].

3 Implementation and Architecture

DRAOn implements the algorithm in [9] in Java, and provides a Java API. We first present an overview of the algorithm before giving more implementation details.

3.1 Algorithm

Let $\langle \mathbf{O}, \mathbf{A} \rangle$ be an NAO where $\mathbf{O}$ is a set of DL decidable ontologies, and $\mathbf{A}$ is a set of alignments between these ontologies. We assume that each ontology $o \in \mathbf{O}$ is attached to a reasoner that can be queried with a set X of DL axioms and answer whether $o \cup X$ is consistent. Then, checking the consistency of an NAO amounts to querying the local reasoners multiple times with a well chosen set of axioms. The main idea is that, when correspondences are restricted to cross-ontology subsumption, equality or membership, alignments can only influence the vacuity and non-vacuity of concepts or roles appearing in the alignments.

The main steps of the algorithm are the following:

1. build an *alignment ontology*, noted A, which corresponds to the translation of correspondences into OWL axioms (using τ_{DL} mentioned in Sect. 2);
2. if A is inconsistent then the NAO is inconsistent;
3. choose a subset S (resp. P) of the concepts (resp. roles) appearing in A to build a *global configuration* Ω consisting of axioms $C_i(x)$ (non-emptiness) with x a fresh individual name for each $C_i \in S$ and $C_j \sqsubseteq \bot$ (unsatisfiability) for each $C_j \notin S$ (resp., $R_i(x, y)$ for $R \in P$ and $\top \sqsubseteq \forall R_j.\bot$ instead);
4. for a given ontology o, define a *local configuration* wrt o as the subset of Ω which involves only terms from o, noted ω_o;
5. if $A \cup \Omega$ is inconsistent, then go back to step 3;
6. for each $o \in \mathbf{O}$, query the local reasoner attached to o using the axioms in ω_o; if all local reasoners answer positively, then the network is consistent. Otherwise, go back to step 3 until there are no more configuration available.

[4] For a set S, $\epsilon(S)$ has to be understood as the set $\{\epsilon(x) \mid x \in S\}$.

3.2 Architetcture

In terms of system architecture, DRAOn consists of a *global OWL reasoner* and several *local OWL reasoners*. The global reasoner is in charge of (i) computing global configurations Ω, (ii) checking consistency of $A \cup \Omega$ for each global configuration Ω and (iii) sending a local configuration to each local reasoner. Each local reasoner checks consistency of $o \cup \Omega$.

The distributed behavior of the algorithm results from a feature of IDDL which does not enforce a strong relationship between alignments and ontologies. Instead of merging ontologies and alignments into a unique ontology, DRAOn distributes reasoning tasks over ontologies to different local reasoners. In addition to early improvements, we implement optimizations for reducing the number of configurations to be considered and for improving communication protocol between global and local reasoners. The main ideas are:

- Global configurations are built in an incremental way, and results obtained from checking previous configurations are reused. For each configuration, we check entailment rather than consistency. This allows for putting forward eventual backtracking points.
- The communication protocol between global and local reasoners is parallelized, that means, configurations are sent to local reasoners in a broadcasting way rather than a sequential one. The global reasoner uses Java threads to manage communication with the local reasoners. We have to use sockets to establish the communication between global and local reasoners instead of OWLLink[5]. This is due to efficiency question and an intrinsic characteristic of IDDL.

Apart from checking consistency of an NAO, the current version of DRAOn implements entailment services under the IDDL semantics for some kinds of entailment concept axioms. Therefore, DRAOn offers the following inference services:

- Checking consistency of an NAO under the DL semantics in a non-distributed way;
- Checking entailment of an OWL axiom by an NAO under the DL semantics in a non-distributed way;
- Checking consistency of an NAO based on the IDDL semantics in a non-distributed way, that is, all reasoning over ontologies and alignments is sequentially performed at one site on a computer.
- Checking consistency of an NAO based on the IDDL semantics in a distributed way.
- Checking entailment of some kinds of OWL axioms by a network of aligned ontologies based on the IDDL semantics in a distributed way.
  Checking whether an NAO entails a local GCI axiom $i{:}C \sqsubseteq D$ can be done by checking the consistency of the NAO with an additional local axiom $i{:}(C \sqcap \neg D)(x)$. This can be done by sending this axiom in additional to the configuration when querying local reasoners.

[5] http://www.owllink.org/

4 Usage Reports et Experiments

In a Java programme, users need to import the following classes defined in the OWL API, the Alignment API and DRAOn itself.
• Using DL as underlying semantics. If users would like to use the DL semantics for a network of aligned ontologies, the following instructions should be put in the programme to create a IDDL reasoner object.

```
IDDLReasoner reasoner = new IDDLReasoner(Set<URI> uris);
IDDLReasoner reasoner =
                 new IDDLReasoner(List<Alignment> aligns);
```

where `uris` is a set of alignment URIs, `aligns` are alignment objects. This means that a network of aligned ontologies can be defined without declaring explicitely ontologies in that creation since the URIs of aligned ontologies can be extracted from alignments. DRAOn uses the DL semantics for the network by default. We can now check whether the defined network is consistent, or entailment of an OWL axiom `ax` by the network under the DL semantics.

```
boolean reasoner.isConsistent();
boolean reasoner.isEntailed(OWLAxiom ax);
```

• Using IDDL as underlying semantics with distributed feature. Before creating reasoner object for this setting, we must define a set of nodes each of which corresponds to an ontology of the network. A local reasoner (HermiT) is associated with each node. The following code shows how to create such nodes:

```
LocalNode node1 =
      new LocalNode(String host1, URI uri1, String port1);
LocalNode node2 =
      new LocalNode(String host2, URI uri2, String port2);
Set<LocalNode> nodes = new HashSet<LocalNode>();
nodes.add(node1);
nodes.add(node2);
```

where the constructor of `LocalNode` expects a host name, the URI of an ontology and a port number. From these nodes, we can now create a global reasoner that calls reasoning services offered by local reasoners located at different machines.

```
IDDLReasoner reasoner = new IDDLReasoner(
  Set<URI> uris, Set<LocalNode> nodes, Semantics.IDDL);
IDDLReasoner reasoner = new IDDLReasoner(
  List<Alignment> als,Set<LocalNode> nodes,Semantics.IDDL);
```

We refer the reader to the DRAOn's web site[6] for a tutorial.

We have do some experiments with well-known ontologies and alignment methods. The tableau following presents empirical results obtained from the current version of

[6] http://iddl.gforge.inria.fr/tutorial.html

DRAOn which uses HermiT as local reasoner. These experiments were performed on a MAC with 4Gb RAM, 2.4 GHz Intel Core i5. The result for the distributed IDDL reasoning was obtained with a computer network consisting of the MAC and a DELL with 2Gb RAM, 1.06GHz Intel i2.

Ontology 1	Ontology 2	Alignment	DL	non-distr. IDDL	distr. IDDL
Small NCI (10,000 axioms, 6,500 entities)	Small FMA (3,800 axioms, 3,700 entities)	Alcomo Map. (2,800 corr.)	7,5s	46s	30s
Humain (5,500 axioms, 3,300 entities)	Mouse (4,500 axioms, 2,750 entities)	Ref. Map. (1516 corr.)	6s	4.5s	4s

IDDL reasoning performances depend on the two services `OWLReasoner.getTypes(.)` and `OWLReasoner.getUnsatisfiableClasses()` which are called for checking unsatisfiability and non-emptiness of concepts involved in alignments. In addition, it might be required to perform more experiments to establish relationships between kinds of aligned ontologies and IDDL reasoning performances.

5 Conclusion

We have presented an OWL-based distributed reasoner, called DRAOn, which offers inference services for a network of aligned OWL ontologies. The distributed behavior of DRAOn is based on a feature of the IDDL semantics which allows a reasoner to distribute reasoning tasks over ontologies to different reasoners. In terms of performance, DRAOn is considerably penalized by unavailability of specific services of an OWL reasoner, for instance, returning all unsatisfiable concepts which belong to a given set of concepts; or returning all non-empty concepts which belong to a given set of concepts. Availability of such services in a future release of OWL reasoners may improve dramatically the performance of DRAOn.

We intend to extend DRAOn to reasoning on a network with disjointness correspondences in alignments. This may require us to develop new optimizations since the current algorithm for such a network is highly intractable.

References

1. Shearer, R., Motik, B., Horrocks, I.: HermiT: A Highly-Efficient OWL Reasoner. In: Proc. of the OWLED 2008. (2008)
2. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: a pratical OWL-DL reasoner. Journal of Web Semantics **5**(2) (2007) 51–53
3. Tsarkov, D., Horrocks, I.: FaCT++ description logic reasoner: System description. In: Proc. of IJCAR 2006. Volume 4130., Springer (2006) 292–297
4. and: Local Models Semantics, or contextual reasoning=Locality+Compatibility. **127**(2) (2001) 221–259
5. Borgida, A., Serafini, L.: Distributed description logics : Assimilating information from peer sources. Journal Of Data Semantics (1) (2003) 153–184

6. Kutz, O., Lutz, C., Wolter, F., Zakharyaschev, M.: E-connections of abstract description systems. Artif. Intell. **156**(1) (2004) 1–73
7. Zimmermann, A.: Integrated distributed description logics. In: Proceedings of the International Workshop on Description Logics. (2007)
8. Serafini, L., Tamilin, A.: Drago: Distributed reasoning architecture for the semantic web. In: Proceedings of the Europeen Semantic Web Conference. (2005) 361–376
9. Zimmermann, A., Le Duc, C.: Reasoning with a network of aligned ontologies. In: RR. (2008) 43–57