Decentralized Reasoning on a Network of Aligned Ontologies with Link Keys

J. Lhez, C. Le Duc, T. Dong, M.Lamolle

LIASD, *University Paris 8*, IUT of Montreuil

October 25, 2019

Outline

1. Network of aligned ontologies with link keys
2. Decentralized reasoning and existing approaches
3. Syntax and semantics with link keys
4. Decentralized reasoning with propagation
5. Implementation and evaluation
6. Conclusion

Outline

Ontology alignment example with link keys

Example

Ontology O_i**:**

(1) $DemoPaperPresenter(Anna)$

(2) $DemoPaperPresenter \sqsubseteq Participant$

Ontology O_j**:**

(3) $PhDStudent \sqsubseteq Researcher$

(4) $Researcher \sqsubseteq \neg Developer$

Alignment axioms A_{ij}**:**

(5) DemoPaperPresenter → Researcher

(6) DemoPaperPresenter → Developer

(7) $\{\langle present, registerTo\rangle\}\text{linkkey}\langle Participant, Researcher\rangle$

Reasoning:

$O_i \cup O_j \cup A_{ij} \models \{\langle present, registerTo\rangle\}\text{linkkey}$
$\langle DemoPaperPresenter, PhDStudent\rangle$?

Network description

- **Ontology network:** Set of ontologies $\{O_1, O_2, O_3 ...\}$ and their alignments $\{A_{12}, A_{13} ...\}$. An alignment is a set of axioms linking ontologies ($C_1 \rightarrow C_2 \in A_{12}$ with $C_1 \in O_1, C_2 \in O_2$).
- **Link key:** $\{\langle P_k, Q_k \rangle\}_{k=1}^{n}$linkkey$\langle C, D \rangle$ with $C, P_1 ... P_k \in O_1$ and $D, Q_1 ... Q_k \in O_2$.
- **Description logic** $\mathcal{ALC}$ (slightly extended)

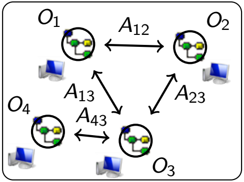
Outline

1. Network of aligned ontologies with link keys
2. Decentralized reasoning and existing approaches
3. Syntax and semantics with link keys
4. Decentralized reasoning with propagation
5. Implementation and evaluation
6. Conclusion

Decentralized approach

Different approaches

- **Centralized approach**
  Merge all axioms and elements of the network together.
- **Decentralized approach**
  Consider all ontologies separately.



Decentralized approach allows for distribution reasoning.
Using weakened semantics enhances further the performances.

Existing approaches

Several different existing systems:

- **Pellet** (E-connections) not distributed.
- **Somewhere** peer to peer, restricted to propositional logic.
- **Drago** (DDL), **early Draon** (IDDL) distributed tableau algorithm.

Performance issues

Approaches distributed, but reasoning requires a lot of exchanges.
No current system supports link keys.

Principle of distributed tableau algorithms

1. Build an alignment model M_1
   - Sent to local ontologies for checking
2. If M_1 is "incompatible" with a local ontology, the algorithm builds another model M_2
   - Another checking is processed with the ontologies
3. The algorithm loops until it finds a model M_i compatible with all local ontologies
   - Or all alignment models have been processed

The number of models is exponential in the size of the network.

Outline

1. Network of aligned ontologies with link keys
2. Decentralized reasoning and existing approaches
3. **Syntax and semantics with link keys**
4. Decentralized reasoning with propagation
5. Implementation and evaluation
6. Conclusion

Syntax

- $C \rightarrow D$ or $C \leftarrow D$ where $C \in \text{sub}(O_i)$ and $D \in \text{sub}(O_j)$. Such a correspondence is called *concept correspondence*.
- $a \approx b$ ($a \not\approx b$) where $a \in \text{Voc}_I(O_i)$ and $b \in \text{Voc}_I(O_j)$. Such a correspondence is called *individual correspondence*.
- a link key $\{\langle P_k, Q_k \rangle\}_{k=1}^{n} \text{linkkey} \langle C, D \rangle$ where $P_k \in \text{Voc}_R(O_i)$, $Q_k \in \text{Voc}_R(O_j)$ for $1 \leq k \leq n$ and $C \in \text{sub}(O_i)$ and $D \in \text{sub}(O_j)$. Such a correspondence is called *link key correspondence*.

Alignment semantics

- If $a \approx b$ is in A_{ij} then $a^{\mathcal{I}} = a^{\mathcal{J}}$.
- If $a \not\approx b$ is in A_{ij} then $a^{\mathcal{I}} \neq a^{\mathcal{J}}$.
- **If $C \rightarrow D$ is in A_{ij} then $D^{\mathcal{J}} = \emptyset$ implies $C^{\mathcal{I}} = \emptyset$.**

Example 1

DemoPaperPresenter → Researcher
DemoPaperPresenter → Developer

- If $\{\langle P_k, Q_k\rangle\}_{k=1}^{n}$linkkey$\langle C, D\rangle$ is in A_{ij} then $(a_k^i)^{\mathcal{I}} = (a_k^j)^{\mathcal{J}}$, $\langle a^{\mathcal{I}}, (a_k^i)^{\mathcal{I}}\rangle \in P_k^{\mathcal{I}}$, $\langle b^{\mathcal{J}}, (a_k^j)^{\mathcal{J}}\rangle \in Q_k^{\mathcal{J}}$ for all $1 \leq k \leq n$, $a^{\mathcal{I}} \in C^{\mathcal{I}}$, $b^{\mathcal{J}} \in D^{\mathcal{J}}$ imply $a^{\mathcal{I}} = b^{\mathcal{J}}$.

Example 2

$\{\langle \mathit{presentsDemo}, \mathit{makesPresentation}\rangle, \langle \mathit{hasName}, \mathit{named}\rangle\}$linkkey
$\langle \mathit{DemoPaperPresenter}, \mathit{Researcher}\rangle$

Consistency checking

Let $N = \langle \{O_i\}_{i=1}^n, \{A_{ij}\}_{i,j=1,i\neq j}^n \rangle$ be a network of aligned ontologies. N is consistent if there is a model $\mathcal{I} = \{\mathcal{I}_i\}_{i=1}^n$ of O_i for all $1 \leq i \leq n$ such that:

1. For each correspondence $a \approx b$ (resp. $a \not\approx b$) in A_{ij} with $i < j$, $a^{\mathcal{I}_i} = b^{\mathcal{I}_j}$ (resp. $a^{\mathcal{I}_i} \neq b^{\mathcal{I}_j}$).
2. There is no pair of correspondences $a \approx b, a \not\approx b$ in A_{ij} (A_{ij} is clash-free)
3. For each correspondence $C \rightarrow D$ in A_{ij} with $i < j$, if $D^{\mathcal{I}_j} = \emptyset$ then $C^{\mathcal{I}_i} = \emptyset$.
4. For each correspondence $\{\langle P_k, Q_k \rangle\}_{k=1}^n \text{linkkey}\langle C, D \rangle$ in A_{ij} with $i < j$, if $(a_k^i)^{\mathcal{I}_i} = (a_k^j)^{\mathcal{I}_j}$, $\langle a^{\mathcal{I}_i}, (a_k^i)^{\mathcal{I}_i} \rangle \in P_k^{\mathcal{I}_i}$, $\langle b^{\mathcal{I}_j}, (a_k^j)^{\mathcal{I}_j} \rangle \in Q_k^{\mathcal{I}_j}$ for all $1 \leq k \leq n$, $a^{\mathcal{I}_i} \in C^{\mathcal{I}_i}$, $b^{\mathcal{I}_j} \in D^{\mathcal{I}_j}$ then $a^{\mathcal{I}_i} = b^{\mathcal{I}_j}$.

Entailment reduction to consistency

Definition

- N *entails* a link key α ($N \models \alpha$) if every model $\mathcal{I}$ of N satisfies α.

Reduction

- Let $\langle \{O_1, O_2\}, A_{12} \rangle$ be a network of aligned ontologies in $\mathcal{ALC}$. It holds that:

$$\langle \{O_1, O_2\}, A_{12} \rangle \models (\{\langle P_i, Q_i \rangle\}_{i=1}^{m} \; \textit{linkkey} \; \langle C, D \rangle) \text{ iff}$$
$$\langle \{O'_1, O'_2\}, A'_{12} \rangle \text{ is inconsistent}$$

with $O'_1 = O_1 \cup \{C(x)\} \cup \{P_i(x, z_i)\}_{i=1}^{n}$,
$O'_2 = O_2 \cup \{D(y)\} \cup \{Q_i(y, z'_i)\}_{i=1}^{n}$,
$A'_{12} = A_{12} \cup \{z_i \approx z'_i\}_{i=1}^{n} \cup \{x \not\approx y\}$, $x, z_1, \cdots, z_n$ are new individuals in O_1 and $y, z'_1, \cdots, z'_n$ are new individuals in O_2.

Outline

Propagation of individual equalities

Equivalence between individuals may lead to propagations:

- If there is $\{a_i^1 \approx a_j^1, a_i^2 \approx a_j^2\} \in A_{ij}$ then for each $O_k \models a_k^m \approx a_k^h$, $k \in \{i, j\}$, $m, h \in \{1, 2\}$, $m \neq h$ we can propagate $\{a_i^h \approx a_j^m, a_i^m \approx a_j^h\}$ in A_{ij} and $a_k^1 \approx a_k^2$ in O_k.

Link keys can trigger a propagation. For each link key $\{\langle P_k, Q_k\rangle\}_{k=1}^n$linkkey$\langle C, D\rangle$:

- If $O_i \cap \{C(a), \sim C(a)\} = \emptyset$ then add $(C \sqcup \sim C)(a)\}$ to O_i
- If $O_j \cap \{D(b), \sim D(b)\} = \emptyset$ then add $(D \sqcup \sim D)(b)\}$ to O_j
- If $P_i(a, x_i) \in O_i$, $Q_i(b, x_i) \in O_j$ then add $a \approx b$ to A_{ij}

Propagation of concept unsatisfiability

Subsumption transitivity may also lead to some propagation:

- If there is $\{C_i^1 \rightarrow C_j^1, C_i^2 \leftarrow C_j^2\} \in A_{ij}$, then for each $O_j \models C_j^1 \sqsubseteq C_j^2$, we add $\{C_i^1 \rightarrow C_j^2, C_i^2 \leftarrow C_j^1\}$ to A_{ij}
- If there is $\{C_i^1 \leftarrow C_j^1, C_i^2 \rightarrow C_j^2\} \in A_{ij}$, then for each $O_i \models C_i^1 \sqsubseteq C_i^2$, we add $\{C_i^1 \rightarrow C_j^2, C_i^2 \leftarrow C_j^1\}$ to A_{ij}

This allows the propagation of unsatisfiability through the network:

- For each axiom $D \rightarrow C \in A_{ij}$, if $O_j \models C \sqsubseteq \bot$, then $D \sqsubseteq \bot$ is added to O_i
- For each axiom $D \leftarrow C \in A_{ij}$, if $O_i \models D \sqsubseteq \bot$, then $C \sqsubseteq \bot$ is added to O_j

Propagation over the network

```
function PROPAGATEOVERNETWORK($\langle \{O_i\}_{i=1}^n, \{A_{ij}\}_{i,j=1,i\neq j}^n \rangle$)
    while $O_i, O_j, A_{ij}$ are unstationary for all $1 \leq i < j \leq n$ do
        for $1 \leq i < j \leq n$ do
            while $O_i, O_j, A_{ij}$ are unstationary do
                if propagateEqual($O_i, O_j, A_{ij}$) returns false then
                    return false
                end if
                if propagateUnsat($O_i, O_j, A_{ij}$) returns false then
                    return false
                end if
            end while
        end for
    end while
    return true
end function
```

Remark

A stationary state is reached when there is nothing left to propagate. One local inconsistency leads to global inconsistency

Example: use of the algorithm

O_i	$Alignment$	O_j
$a \approx c$	$a \approx b$	$D(f)$
$C(e)$	$c \approx d$	$Q(f, b)$
$P(e, a)$	$\{\langle P, Q\rangle\}\text{linkkey}\langle C, D\rangle$	$F \sqsubseteq \bot$
$E \sqsubseteq \bot$	$H \rightarrow F$	
	$E \leftarrow G$	
$H \sqsubseteq \bot$	$a \approx d, c \approx b$	$b \approx d$
	$e \approx f$	$G \sqsubseteq \bot$

Propagation through individual correspondences

Propagation through link key correspondences

Propagation through concept correspondences

This is only one step of propagation.

Theorem 1 : reducing global consistency to local consistencies

Let $\langle \{O_i\}_{i=1}^{n}, \{A_{ij}\}_{i,j=1,i\neq j}^{n} \rangle$ be a network of aligned ontologies. $\widehat{O_i}$ $(1 \leq i \leq n)$ denotes the resulting consistent ontologies obtained by calling *propagateOverNetwork*$(\langle \{O_i\}_{i=1}^{n}, \{A_{ij}\}_{i,j=1,i\neq j}^{n} \rangle)$.
It holds that $\widehat{O_i}$ is consistent for all $1 \leq i \leq n$ and $\widehat{A_{ij}}$ is clash-free for all $1 \leq i < j \leq n$ iff the network $\langle \{O_i\}_{i=1}^{n}, \{A_{ij}\}_{i,j=1,i\neq j}^{n} \rangle$ is consistent.

Theorem 2 : Complexity

Let $\langle \{O_i\}_{i=1}^n, \{A_{ij}\}_{i,j=1,i\neq j}^n \rangle$ be a network of aligned ontologies. The algorithm *propagateOverNetwork*$(\langle \{O_i\}_{i=1}^n, \{A_{ij}\}_{i,j=1,i\neq j}^n \rangle)$ runs in polynomial time in the size of the network if each check of entailment or consistency occurring in the algorithms is considered as an oracle.

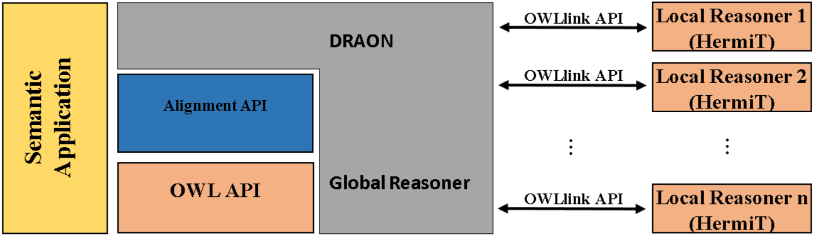
Remark

Our approach adds axioms and assertions to ontologies and alignments monotonously.
Each call to a local reasoner considered as oracle.

Outline

1. Network of aligned ontologies with link keys
2. Decentralized reasoning and existing approaches
3. Syntax and semantics with link keys
4. Decentralized reasoning with propagation
5. Implementation and evaluation
6. Conclusion

Implementation



API used

- **Alignment API:** for the alignment and the link keys
- **OWL API:** for the global ontology after propagation
- **OWLLink API:** for the decentralized ontologies

Evaluation

Ontology 1	Ontology 2	Alignment	IDDL	**APPROX**
Iasted	Sigkdd	iasted-sigkdd (without link keys)	3.5s	**9 ms**
Conference	Ekaw	conference-ekaw (without link keys)	7.5s	**11 ms**
Cmt	Edas	cmt-edas (without link keys)	7.5s	**16 ms**
FMA	SNOMED	FMA-SNOMED (without link keys)	$>$ 15 minutes	**81 s**
FMA	NCI	FMA-NCI (without link keys)	$>$ 15 minutes	**10 s**

Table: Execution time for checking consistency of ontology networks according to different semantics

Ontology 1	Ontology 2	Alignment	**Consistency in APPROX**
Iasted	Sigkdd	iast-sigkdd (with link keys)	**9 ms**
Conference	Ekaw	conference-ekaw (with link keys)	**11 ms**
Cmt	Edas	cmt-edas (with link keys)	**17 ms**

Table: Execution time (in milliseconds) for checking consistency of ontology networks with link keys

Outline

Conclusion

- Reasoning over a network of aligned ontologies leads to performance issues
- Decentralized approaches, even distributed, require an exponential amount of exchanges
- Using a weakened semantics decreases the complexity
- Our approach consists in propagating relevant knowledge for consistency checking
- The implementation is based on several APIs and gives some gain of performances

Future Work

- Better datasets for testing (larger, more complete)
- Treatment of correspondences with:
    - Atomic/complex concept
    - Add role correspondences
- Adding data properties (link keys)

Thank you

This work has been partially supported by the ANR project ELKER (ANR-17-CE23-0007-01)