A Framework for Representing Ontology Mappings under Probabilities and Inconsistency

Andrea Calì [1] Thomas Lukasiewicz [2,3] Livia Predoiu [4]
Heiner Stuckenschmidt [4]

[1] Computing Laboratory, University of Oxford, UK
andrea.cali@comlab.ox.ac.uk

[2] DIS, Sapienza Università di Roma, Italy
lukasiewicz@dis.uniroma1.it

[3] Institut für Informationssysteme, TU Wien, Austria
lukasiewicz@kr.tuwien.ac.at

[4] Institut für Informatik, Universität Mannheim, Germany
{heiner,livia}@informatik.uni-mannheim.de

Outline

One of the major challenges of the Semantic Web: aligning heterogeneous ontologies via semantic mappings.

Mappings are automatically produced by matching systems.

Automatically created mappings often contain uncertain hypotheses and errors:

- mapping hypotheses are often oversimplifying;
- there may be conflicts between different hypotheses for semantic relations;
- semantic relations are only given with a degree of confidence in their correctness.

In this paper, we present a logic-based language (close to semantic web languages) for representing, combining, and reasoning about such ontology mappings.

- Ontologies are encoded in L (here: OWL DL or OWL Lite).
- $Q(O)$ denotes the matchable elements of the ontology O.
- Matching: Given two ontologies O and O', determine correspondences between $Q(O)$ and $Q(O')$.
- Correspondences are 5-tuples (id, e, e', r, n) such that
  - id is a unique identifier;
  - $e \in Q(O)$ and $e' \in Q(O')$;
  - $r \in R$ is a semantic relation (here: implication);
  - n is a degree of confidence in the correctness.

- Tight integration of mapping and ontology language
- Support for mappings refinement
- Support for repairing inconsistencies
- Representation and combination of confidence
- Decidability and efficiency of instance reasoning

Description logic knowledge bases in $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$ (which are the DLs behind OWL Lite and OWL DL, respectively).

Description logic knowledge base L for an online store:

(1) $\mathit{Textbook} \sqsubseteq \mathit{Book}$; (2) $\mathit{PC} \sqcup \mathit{Laptop} \sqsubseteq \mathit{Electronics}$; $\mathit{PC} \sqsubseteq \neg\mathit{Laptop}$;
(3) $\mathit{Book} \sqcup \mathit{Electronics} \sqsubseteq \mathit{Product}$; $\mathit{Book} \sqsubseteq \neg\mathit{Electronics}$;
(4) $\mathit{Sale} \sqsubseteq \mathit{Product}$;
(5) $\mathit{Product} \sqsubseteq\ \geqslant 1\, \mathit{related}$; (6) $\geqslant 1\, \mathit{related} \sqcup \geqslant 1\, \mathit{related}^- \sqsubseteq \mathit{Product}$;
(7) $\mathit{related} \sqsubseteq \mathit{related}^-$; $\mathit{related}^- \sqsubseteq \mathit{related}$;
(8) $\mathit{Textbook}(\mathit{tb\_ai})$; $\mathit{Textbook}(\mathit{tb\_lp})$; (9) $\mathit{related}(\mathit{tb\_ai}, \mathit{tb\_lp})$;
(10) $\mathit{PC}(\mathit{pc\_ibm})$; $\mathit{PC}(\mathit{pc\_hp})$; (11) $\mathit{related}(\mathit{pc\_ibm}, \mathit{pc\_hp})$;
(12) $\mathit{provides}(\mathit{ibm}, \mathit{pc\_ibm})$; $\mathit{provides}(\mathit{hp}, \mathit{pc\_hp})$.

Disjunctive program P for an online store:

(1) $pc(pc_1)$; $pc(pc_2)$; $pc(obj_3) \vee laptop(obj_3)$;
(2) $brand\_new(pc_1)$; $brand\_new(obj_3)$;
(3) $vendor(dell, pc_1)$; $vendor(dell, pc_2)$;
(4) $avoid(X) \leftarrow camera(X), not\ sale(X)$;
(5) $sale(X) \leftarrow electronics(X), not\ brand\_new(X)$;
(6) $provider(V) \leftarrow vendor(V, X), product(X)$;
(7) $provider(V) \leftarrow provides(V, X), product(X)$;
(8) $similar(X, Y) \leftarrow related(X, Y)$;
(9) $similar(X, Z) \leftarrow similar(X, Y), similar(Y, Z)$;
(10) $similar(X, Y) \leftarrow similar(Y, X)$;
(11) $brand\_new(X) \vee high\_quality(X) \leftarrow expensive(X)$.

Syntax

- Sets $\mathbf{A}$, $\mathbf{R}_A$, $\mathbf{R}_D$, $\mathbf{I}$, and $\mathbf{V}$ of atomic concepts, abstract roles, datatype roles, individuals, and data values, respectively.
- Finite sets Φ_p and Φ_c of constant and predicate symbols with: (i) Φ_p not necessarily disjoint to $\mathbf{A}$, $\mathbf{R}_A$, and $\mathbf{R}_D$, and (ii) $\Phi_c \subseteq \mathbf{I} \cup \mathbf{V}$.
- A tightly integrated disjunctive dl-program $KB = (L, P)$ consists of a description logic knowledge base L and a disjunctive program P.

Semantics

- An interpretation I is any subset of the Herbrand base HB_Φ.
- I is a model of P is defined as usual.
- I is a model of L iff $L \cup I \cup \{\neg a \mid a \in HB_\Phi - I\}$ is satisfiable.
- I is a model of KB iff I is a model of both L and P.
- The Gelfond-Lifschitz reduct of $KB = (L, P)$ w.r.t. $I \subseteq HB_\Phi$, denoted KB^I, is defined as the disjunctive dl-program (L, P^I), where P^I is the standard Gelfond-Lifschitz reduct of P w.r.t. I.
- $I \subseteq HB_\Phi$ is an answer set of KB iff I is a minimal model of KB^I.
- KB is consistent iff it has an answer set.
- A ground atom $a \in HB_\Phi$ is a cautious (resp., brave) consequence of a disjunctive dl-program KB under the answer set semantics iff every (resp., some) answer set of KB satisfies a.

Examples

A disjunctive dl-program $KB = (L, P)$ is given by the above description logic knowledge base L and disjunctive program P.

Another disjunctive dl-program $KB' = (L', P')$ is obtained from KB by adding to L the axiom $\geqslant 1\, similar \sqcup \geqslant 1\, similar^{-} \sqsubseteq Product$, which expresses that only products are similar:

The predicate symbol *similar* in P' is also a role in L', and it freely occurs in both rule bodies and rule heads in P'.

Properties

Every answer set of a disjunctive program *KB* is also a minimal model of *KB*, and the converse holds when *KB* is positive.

The answer set semantics of disjunctive dl-programs faithfully extends its ordinary counterpart and the first-order semantics of description logic knowledge bases.

The tight integration of ontologies and rules semantically behaves very differently from the loose integration: $KB = (L, P)$, where

$$\begin{aligned} L &= \{person(a),\ person \sqsubseteq male \sqcup female\} \text{ and} \\ P &= \{client(X) \leftarrow male(X),\ client(X) \leftarrow female(X)\}\,, \end{aligned}$$

implies $client(a)$, while $KB' = (L', P')$, where

$$\begin{aligned} L' &= \{person(a),\ person \sqsubseteq male \sqcup female\} \text{ and} \\ P' &= \{client(X) \leftarrow DL[male](X),\ client(X) \leftarrow DL[female](X)\}\,, \end{aligned}$$

does *not* imply $client(a)$.

Basics

Tightly integrated disjunctive dl-programs $KB = (L, P)$ can be used for representing (possibly inconsistent) mappings (without confidence values) between two ontologies.

Intuitively, L encodes the union of the two ontologies, while P encodes the mappings between the ontologies.

Here, disjunctions in rule heads and nonmonotonic negations in rule bodies in P can be used to resolve inconsistencies.

Example

The following two mappings have been created by the hmatch system for mapping the CRS Ontology (O_1) on the EKAW Ontology (O_2):

$$\mathit{EarlyRegisteredParticipant}(X) \leftarrow \mathit{Participant}(X)\,;$$
$$\mathit{LateRegisteredParticipant}(X) \leftarrow \mathit{Participant}(X)\,.$$

L is the union of two description logic knowledge bases L_1 and L_2 encoding the ontologies O_1 resp. O_2, while P encodes the mappings.

However, we cannot directly use the two mapping relationships as two rules in P, since this would introduce an inconsistency in KB.

Resolving Inconsistencies

By disjunctions in rule heads:

$$\mathit{EarlyRegisteredParticipant}(X) \vee \mathit{LateRegisteredParticipant}(X) \leftarrow \mathit{Participant}(X)\,.$$

By nonmonotonic negations in rule bodies (using additional background information):

$$\mathit{EarlyRegisteredParticipant}(X) \leftarrow \mathit{Participant}(X) \wedge \mathit{RegisterdbeforeDeadline}(X)\,;$$
$$\mathit{LateRegisteredParticipant}(X) \leftarrow \mathit{Participant}(X) \wedge \mathit{not}\ \mathit{RegisteredbeforeDeadline}(X)\,.$$

Syntax and Semantics

Tightly integrated probabilistic dl-program $KB = (L, P, C, \mu)$:

- description logic knowledge base L,
- disjunctive program P with values of random variables $A \in C$ as "switches" in rule bodies,
- probability distribution μ over all joint instantiations B of the random variables $A \in C$.

They specify a set of probability distributions over first-order models: Every joint instantiation B of the random variables along with the generalized normal program specifies a set of first-order models of which the probabilities sum up to $\mu(B)$.

Example

Probabilistic rules in P along with the probability μ on the choice space C of a probabilistic dl-program $KB = (L, P, C, \mu)$:

- $avoid(X) \leftarrow Camera(X), not\ offer(X), avoid\_pos$;
- $offer(X) \leftarrow Electronics(X), not\ brand\_new(X), offer\_pos$;
- $buy(C, X) \leftarrow needs(C, X), view(X), not\ avoid(X), v\_buy\_pos$;
- $buy(C, X) \leftarrow needs(C, X), buy(C, Y), also\_buy(Y, X), a\_buy\_pos$.

μ: $avoid\_pos, avoid\_neg \mapsto 0.9\,, 0.1$; $offer\_pos, offer\_neg \mapsto 0.9\,, 0.1$;
$v\_buy\_pos, v\_buy\_neg \mapsto 0.7\,, 0.3$; $a\_buy\_pos, a\_buy\_neg \mapsto 0.7\,, 0.3$.

$\{avoid\_pos, offer\_pos, v\_buy\_pos, a\_buy\_pos\} : 0.9 \times 0.9 \times 0.7 \times 0.7, \ldots$

Probabilistic query: $\exists(buy(john, ixus500))[L, U]$

Basics

Tightly integrated probabilistic dl-programs $KB = (L, P, C, \mu)$ can be used for representing (possibly inconsistent) mappings with confidence values between two ontologies.

Intuitively, L encodes the union of the two ontologies, while P, C, and μ encode the mappings between the ontologies.

Here, confidence values can be encoded as error probabilities, and inconsistencies can also be resolved via trust probabilities (in addition to using disjunctions and negations in P).

Example

Mapping the publication ontology in test 101 (O_1) on the ontology of test 302 (O_2) of the Ontology Alignment Evaluation Initiative:

Encoding two mappings produced by hmatch:

$$\begin{array}{l} \mathit{Book}(X) \leftarrow \mathit{Collection}(X) \wedge \mathit{hmatch}_1\,; \\ \mathit{Proceedings}_2(X) \leftarrow \mathit{Proceedings}_1(X) \wedge \mathit{hmatch}_2\,. \end{array}$$

$C = \{\{\mathit{hmatch}_i, \mathit{not\_hmatch}_i\} \mid i \in \{1, 2\}\}$
$\mu(\mathit{hmatch}_1) = 0.62$ and $\mu(\mathit{hmatch}_2) = 0.73$.

Encoding two mappings produced by falcon:

$$\begin{array}{l} \mathit{InCollection}(X) \leftarrow \mathit{Collection}(X) \wedge \mathit{falcon}_1\,; \\ \mathit{Proceedings}_2(X) \leftarrow \mathit{Proceedings}_1(X) \wedge \mathit{falcon}_2\,. \end{array}$$

$C' = \{\{\mathit{falcon}_i, \mathit{not\_falcon}_i\} \mid i \in \{1, 2\}\}$
$\mu'(\mathit{falcon}_1) = 0.94$ and $\mu'(\mathit{falcon}_2) = 0.96$.

Merging the two encodings:

$$\begin{aligned} &\mathit{Book}(X) \leftarrow \mathit{Collection}(X) \wedge \mathit{hmatch}_1 \wedge \mathit{sel\_hmatch}_1\,; \\ &\mathit{InCollection}(X) \leftarrow \mathit{Collection}(X) \wedge \mathit{falcon}_1 \wedge \mathit{sel\_falcon}_1\,; \\ &\mathit{Proceedings}_2(X) \leftarrow \mathit{Proceedings}_1(X) \wedge \mathit{hmatch}_2\,; \\ &\mathit{Proceedings}_2(X) \leftarrow \mathit{Proceedings}_1(X) \wedge \mathit{falcon}_2\,. \end{aligned}$$

$C'' = C \cup C' \cup \{\mathit{sel\_hmatch}_1, \mathit{sel\_falcon}_1\}$
$\mu'' = \mu \cdot \mu' \cdot \mu^{\star}$, where $\mu^{\star}\colon \mathit{sel\_hmatch}_1, \mathit{sel\_falcon}_1 \mapsto 0.55, 0.45$.

Any randomly chosen instance of *Proceedings* of O_1 is also an instance of *Proceedings* of O_2 with the probability 0.9892.

Probabilistic query $Q = \exists(\mathit{Book}(\mathit{pub}))[R, S]$:

The tight answer θ to Q is $\theta = \{R/0, S/0\}$ (resp., $\theta = \{R/0.341, S/0.341\}$), if *pub* is not (resp., is) an instance of *Collection* in O_1.

Summary:

- Tightly integrated probabilistic (disjunctive) dl-programs for representing ontology mappings.
- Resolving inconsistencies via disjunctions in rule heads and nonmonotonic negations in rule bodies.
- Explicitly representing numeric confidence values as error probabilities, resolving inconsistencies via trust probabilities, and reasoning about these on a numeric level.
- Expressive, well-integrated with description logic ontologies, still decidable, and data-tractable subsets.

Outlook:

- Implementation and experiments.
- More general tractability results?
- Efficient top-k query technique?