Uncertainty Reasoning for the Semantic Web

Thomas Lukasiewicz

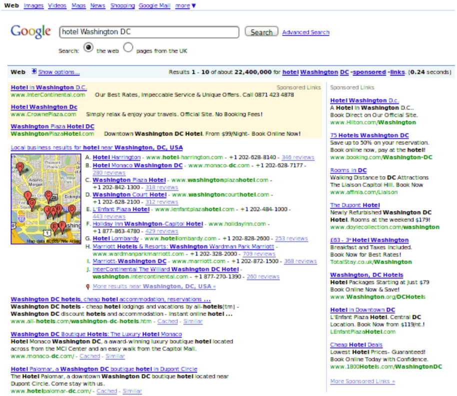
Computing Laboratory, University of Oxford, UK
thomas.lukasiewicz@comlab.ox.ac.uk

Outline

Outline

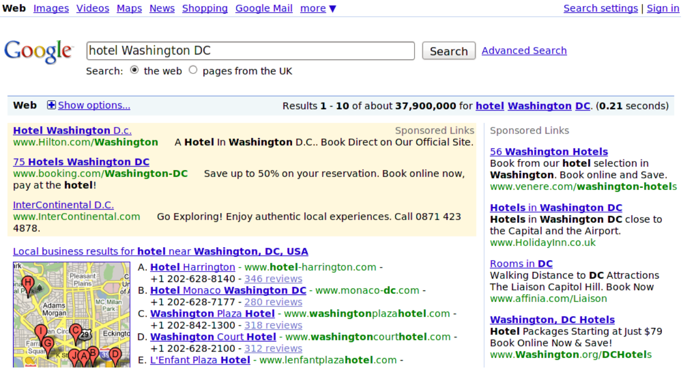
Web Search

Ranking of Web pages to be returned for a Web search query; e.g., via PageRank technique (based on statistical methods):



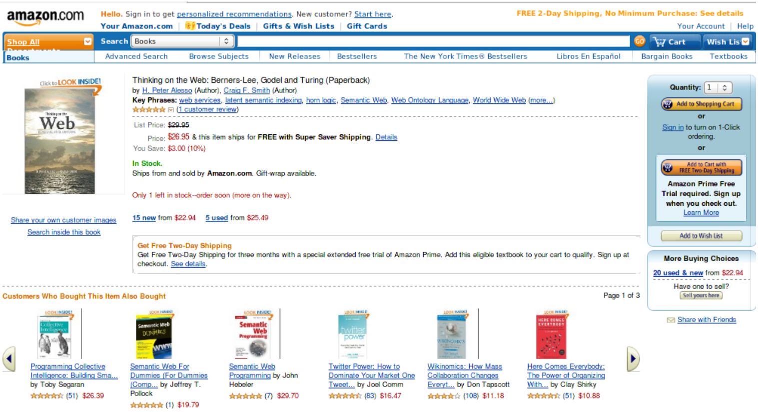
Computational Advertising

Find the best ad to present to a user in a given context, such as querying a search engine ("sponsored search"), reading a web page ("content match"), watching a movie, etc.



Recommender Systems

Present information items (movies, music, books, news, images, web pages, etc.) that may interest a user, e.g.,



Other Examples

- Web spam detection
- Information extraction
- Semantic annotation
- Trust and reputation
- User preference modeling
- Belief fusion and opinion pooling
- Machine translation
- Speech
- Natural language processing
- Computer vision
- ...

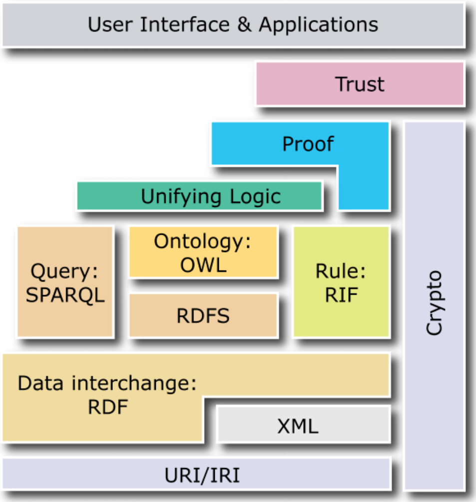
Outline

Key Ideas

- Evolution of the current Web in which the meaning of information and services on the Web is defined...
- ...making it possible to understand and satisfy the requests of people and machines to use the Web content.
- Vision of the Web as a universal medium for data, information, and knowledge exchange.
- Extension of the current Web by standards and technologies that help machines to understand the information on the Web to support richer discovery, data integration, navigation, and automation of tasks.

- Use ontologies for a precise definition of shared terms in Web resources, use KR technology for automated reasoning from Web resources, and apply cooperative agent technology for processing the information of the Web.
- Consists of several *hierarchical layers*, including
  - the Ontology layer: *OWL Web Ontology Language*: *OWL Lite* $\approx \mathcal{SHIF}(\mathbf{D})$, *OWL DL* $\approx \mathcal{SHOIN}(\mathbf{D})$, *OWL Full*; recent tractable fragments: OWL EL, OWL QL, OWL RL;
  - the Rules layer: Rule Interchange Format (RIF); current ongoing standardization;
  - the Logic and Proof layers, which should offer other sophisticated representation and reasoning capabilities.

Semantic Web Stack



Challenges (from Wikipedia)

Semantic Web - Wikipedia, the ...

Challenges

[edit]

Some of the challenges for the Semantic Web include vastness, vagueness, uncertainty, inconsistency and deceit. Automated reasoning systems will have to deal with all of these issues in order to deliver on the promise of the Semantic Web.

- Vastness: The World Wide Web contains at least 48 billion pages as of this writing (August 2, 2009). The SNOMED CT medical terminology ontology contains 370,000 class names, and existing technology has not yet been able to eliminate all semantically duplicated terms. Any automated reasoning system will have to deal with truly huge inputs.
- Vagueness: These are imprecise concepts like "young" or "tall". This arises from the vagueness of user queries, of concepts represented by content providers, of matching query terms to provider terms and of trying to combine different knowledge bases with overlapping but subtly different concepts. Fuzzy logic is the most common technique for dealing with vagueness.
- Uncertainty: These are precise concepts with uncertain values. For example, a patient might present a set of symptoms which correspond to a number of different distinct diagnoses each with a different probability. Probabilistic reasoning techniques are generally employed to address uncertainty.
- Inconsistency: These are logical contradictions which will inevitably arise during the development of large ontologies, and when ontologies from separate sources are combined. Deductive reasoning fails catastrophically when faced with inconsistency, because "anything follows from a contradiction". Defeasible reasoning and paraconsistent reasoning are two techniques which can be employed to deal with inconsistency.
- Deceit: This is when the producer of the information is intentionally misleading the consumer of the information. Cryptography techniques are currently utilized to ameliorate this threat.

This list of challenges is illustrative rather than exhaustive, and it focuses on the challenges to the "unifying logic" and "proof" layers of the Semantic Web. The World Wide Web Consortium (W3C) Incubator Group for Uncertainty

Uncertainty (and Vagueness) in the Semantic Web

- Uncertainty: statements are true or false. But, due to lack of knowledge we can only estimate to which probability / possibility / necessity degree they are true or false, e.g., “John wins in the lottery with the probability 0.01”.
- Vagueness: statements involve concepts for which there is no exact definition, such as tall, small, close, far, cheap, and expensive; statements are true to some degree, e.g., “Hotel Verdi is close to the train station to degree 0.83”.
- Uncertainty and vagueness are important in the SW; many existing proposals for extensions of SW languages (RDF, OWL, DLs, rules) by uncertainty and vagueness.

In the following, some own such proposals: probabilistic DLs, probabilistic dl-programs, probabilistic fuzzy dl-programs.

Outline

Probabilistic Ontologies

Generalization of classical ontologies by probabilistic knowledge.

Main types of encoded probabilistic knowledge:

- Terminological probabilistic knowledge about concepts and roles:

  "Birds fly with a probability of at least 0.95".

- Assertional probabilistic knowledge about instances of concepts and roles:

  "Tweety is a bird with a probability of at least 0.9".

Use of Probabilistic Ontologies

- In medicine, biology, defense, astronomy, ...
- In the Semantic Web:
  - Quantifying the degrees of overlap between concepts, to use them in Semantic Web applications: information retrieval, personalization, recommender systems, ...
  - Information retrieval, for an increased recall (e.g., Udrea et al.: Probabilistic ontologies and relational databases. In *Proc. CoopIS/DOA/ODBASE-2005*).
  - Ontology matching (e.g., Mitra et al.: OMEN: A probabilistic ontology mapping tool. In *Proc. ISWC-2005*).
  - Probabilistic data integration, especially for handling ambiguous and inconsistent pieces of information.

In the following, I sketch the main ideas behind the probabilistic description logics P-$\mathcal{SHOQ}(\mathbf{D})$, P-$\mathcal{SHIF}(\mathbf{D})$, P-$\mathcal{SHOIN}(\mathbf{D})$ for representing probabilistic ontologies:

References:

- R. Giugno and T. Lukasiewicz. P-$\mathcal{SHOQ}(\mathbf{D})$: A probabilistic extension of $\mathcal{SHOQ}(\mathbf{D})$ for probabilistic ontologies in the Semantic Web. In *Proceedings JELIA-2002*, pp. 86-97, September 2002.
- T. Lukasiewicz. Expressive probabilistic description logics. *Artificial Intelligence*, 172(6/7), 852–883, April 2008.

Key Ideas

Description logics model a domain of interest in terms of concepts and roles, which represent classes of individuals and binary relations between classes of individuals, respectively.

A description logic knowledge base encodes in particular subset relationships between concepts, subset relationships between roles, the membership of individuals to concepts, and the membership of pairs of individuals to roles.

Here, 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).

Example

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) $\mathit{provides}(\mathit{ibm}, \mathit{pc\_ibm})$; $\mathit{provides}(\mathit{hp}, \mathit{pc\_hp})$.

Key Ideas

- Integration of (propositional) logic- and probability-based representation and reasoning formalisms.
- Reasoning from logical constraints and interval restrictions for conditional probabilities (also called *conditional constraints*).
- Reasoning from convex sets of probability distributions.
- Model-theoretic notion of logical entailment.

Example (Syntax of Probabilistic Knowledge Bases)

Probabilistic knowledge base $KB = (L, P)$:

- $L = \{bird \Leftarrow eagle\}$:

  "All eagles are birds".

- $P = \{(have\_legs \mid bird)[1, 1], (fly \mid bird)[0.95, 1]\}$:

  "All birds have legs".

  "Birds fly with a probability of at least 0.95".

Example (Semantics of Probabilistic KBs)

- Set of basic propositions $\Phi = \{bird, fly\}$.
- $\mathcal{I}_\Phi$ contains exactly the worlds I_1, I_2, I_3, and I_4 over Φ:

	fly	$\neg fly$
$bird$	I_1	I_2
$\neg bird$	I_3	I_4

- Some probabilistic interpretations:

Pr_1	fly	$\neg fly$
$bird$	19/40	1/40
$\neg bird$	10/40	10/40

Pr_2	fly	$\neg fly$
$bird$	0	1/3
$\neg bird$	1/3	1/3

- $\mathrm{Pr}_1(fly \wedge bird) = 19/40$ and $\mathrm{Pr}_1(bird) = 20/40$.
- $\mathrm{Pr}_2(fly \wedge bird) = 0$ and $\mathrm{Pr}_2(bird) = 1/3$.
- $\neg fly \Leftarrow bird$ is false in Pr_1, but true in Pr_2.
- $(fly \mid bird)[.95, 1]$ is true in Pr_1, but false in Pr_2.

Example (Satisfiability and Logical Entailment)

- Probabilistic knowledge base:

$$KB = (\{bird \Leftarrow eagle\},\ \{(have\_legs \mid bird)[1,1], (fly \mid bird)[0.95,1]\}).$$

- KB is satisfiable, since

  Pr with $Pr(bird \wedge eagle \wedge have\_legs \wedge fly) = 1$ is a model.

- Some conclusions under logical entailment:

$$KB \Vdash (have\_legs \mid bird)[0.3,1], \quad KB \Vdash (fly \mid bird)[0.6,1].$$

- Tight conclusions under logical entailment:

$$KB \models_{tight} (have\_legs \mid bird)[1,1], \quad KB \models_{tight} (fly \mid bird)[0.95,1],$$
$$KB \models_{tight} (have\_legs \mid eagle)[1,1], \quad KB \models_{tight} (fly \mid eagle)[0,1].$$

Towards Stronger Notions of Entailment

Problem: Inferential weakness of logical entailment.

Solutions:

- Probabilistic default reasoning: Adding the inheritance of probabilistic properties along subconcept relationships and a mechanism for resolving local inconsistencies.
- Probabilistic independencies: Adding explicit or implicit probabilistic independencies.
  Special case: Bayesian networks
- Probability selection techniques: Perform inference from a representative distribution (e.g., of maximum entropy or in the center of mass) of the encoded (convex) set of distributions rather than the whole set.

Logical vs. Lexicographic Entailment

Probabilistic knowledge base:

$$KB = (\{bird \Leftarrow eagle\},\ \{(have\_legs \mid bird)[1,1], (fly \mid bird)[0.95,1]\}).$$

Tight conclusions under logical entailment:

$$KB \models_{tight} (have\_legs \mid bird)[1,1], \quad KB \models_{tight} (fly \mid bird)[0.95,1],$$
$$KB \models_{tight} (have\_legs \mid eagle)[1,1], \quad KB \models_{tight} (fly \mid eagle)[0,1].$$

Tight conclusions under probabilistic lexicographic entailment:

$$KB \mathrel{|\!\!\sim}^{lex}_{tight} (have\_legs \mid bird)[1,1], \quad KB \mathrel{|\!\!\sim}^{lex}_{tight} (fly \mid bird)[0.95,1],$$
$$KB \mathrel{|\!\!\sim}^{lex}_{tight} (have\_legs \mid eagle)[1,1], \quad KB \mathrel{|\!\!\sim}^{lex}_{tight} (fly \mid eagle)[0.95,1].$$

Probabilistic knowledge base:

$$KB = (\{bird \Leftarrow penguin\}, \{(have\_legs \mid bird)[1,1], (fly \mid bird)[1,1], (fly \mid penguin)[0,0.05]\})\,.$$

Tight conclusions under logical entailment:

$$KB \models_{tight} (have\_legs \mid bird)[1,1],\ KB \models_{tight} (fly \mid bird)[1,1],$$
$$KB \models_{tight} (have\_legs \mid penguin)[1,0],\ KB \models_{tight} (fly \mid penguin)[1,0]\,.$$

Tight conclusions under probabilistic lexicographic entailment:

$$KB \Vdash\!\sim^{lex}_{tight} (have\_legs \mid bird)[1,1],\ KB \Vdash\!\sim^{lex}_{tight} (fly \mid bird)[1,1],$$
$$KB \Vdash\!\sim^{lex}_{tight} (have\_legs \mid penguin)[1,1],\ KB \Vdash\!\sim^{lex}_{tight} (fly \mid penguin)[0,0.05].$$

Probabilistic knowledge base:

$$KB = (\{bird \Leftarrow penguin\}, \{(have\_legs \mid bird)[0.99, 1], (fly \mid bird)[0.95, 1], (fly \mid penguin)[0, 0.05]\}).$$

Tight conclusions under logical entailment:

$$KB \models_{tight} (have\_legs \mid bird)[0.99, 1],\ KB \models_{tight} (fly \mid bird)[0.95, 1],$$
$$KB \models_{tight} (have\_legs \mid penguin)[0, 1],\ KB \models_{tight} (fly \mid penguin)[0, 0.05].$$

Tight conclusions under probabilistic lexicographic entailment:

$$KB \Vdash\!\sim^{lex}_{tight} (have\_legs \mid bird)[0.99, 1],\ KB \Vdash\!\sim^{lex}_{tight} (fly \mid bird)[0.95, 1],$$
$$KB \Vdash\!\sim^{lex}_{tight} (have\_legs \mid penguin)[0.99, 1],\ KB \Vdash\!\sim^{lex}_{tight} (fly \mid penguin)[0, 0.05].$$

Key Ideas

- probabilistic generalization of the description logics $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$ behind OWL Lite and OWL DL, respectively
- terminological probabilistic knowledge about concepts and roles
- assertional probabilistic knowledge about instances of concepts and roles
- terminological probabilistic inference based on lexicographic entailment in probabilistic logic (stronger than logical entailment)
- assertional probabilistic inference based on lexicographic entailment in probabilistic logic (for combining assertional and terminological probabilistic knowledge)
- terminological and assertional probabilistic inference problems reduced to sequences of linear optimization problems

Medical Example

- Terminological default knowledge:
  - “generally, heart patients suffer from high blood pressure”,
  - “generally, pacemaker patients don’t suffer from high blood pressure”.
- Terminological probabilistic knowledge:
  - “generally, pacemaker patients are male with prob. $\geqslant 0.4$”,
  - “generally, heart patients have a private insurance with probability $\geqslant 0.9$”.
- Assertional probabilistic knowledge
  - “Tom is a pacemaker patient with probability $\geqslant 0.8$”,
  - “Mary has the symptom breathing difficulties with probability $\geqslant 0.6$”,
  - “Mary has the symptom chest pain with probability $\geqslant 0.9$”.

Computational Complexity

- Consistency of probabilistic TBoxes (PTCON)
- Consistency of probabilistic KBs (PKBCON)
- Tight lexicographically entailed intervals for (terminological and assertional) conditional concept statements

	P-*DL-Lite*	P-$\mathcal{SHIF}$(**D**)	P-$\mathcal{SHOIN}$(**D**)
PTCON	NP-complete	EXP-complete	NEXP-complete
PKBCON	NP-complete	EXP-complete	NEXP-complete

	P-*DL-Lite*	P-$\mathcal{SHIF}$(**D**)	P-$\mathcal{SHOIN}$(**D**)
TLEXENT	$\mathrm{FP}^{\mathrm{NP}}$-complete	FEXP-complete	in $\mathrm{FP}^{\mathrm{NEXP}}$

Outline

Overview

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 the following, I survey a logic-based language (close to semantic web languages) for representing, combining, and reasoning about such ontology mappings.

References:

- A. Calì, T. Lukasiewicz, L. Predoiu, H. Stuckenschmidt. Tightly coupled probabilistic description logic programs for the Semantic Web. *Journal on Data Semantics*, 12, 95–130, June 2009.
- T. Lukasiewicz, L. Predoiu, H. Stuckenschmidt. Tightly integrated probabilistic description logic programs for representing ontology mappings. Submitted for journal publication, March 2009.

Basics

- 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.

Representation Requirements

- 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 Logics

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 Programs

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 $\geqslant 1\, \mathit{similar} \sqcup \geqslant 1\, \mathit{similar}^{-} \sqsubseteq \mathit{Product}$, 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):

$$
\begin{aligned}
&\mathit{EarlyRegisteredParticipant}(X) \leftarrow \mathit{Participant}(X)\,;\\
&\mathit{LateRegisteredParticipant}(X) \leftarrow \mathit{Participant}(X)\,.
\end{aligned}
$$

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

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

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 $\mu^\star$: $\mathit{sel\_hmatch}_1, \mathit{sel\_falcon}_1 \mapsto 0.55, 0.45$.

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.

- 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.
- Well-founded semantics for normal case, with first-order rewritable special cases for first-order rewritable DLs.

Outline

In the following, I describe the main ideas behind an approach to probabilistic fuzzy dl-programs, used for a shopping agent application, from:

T. Lukasiewicz and U. Straccia. Description logic programs under probabilistic uncertainty and fuzzy vagueness. *International Journal of Approximate Reasoning*, 50(6), 837–853, June 2009.

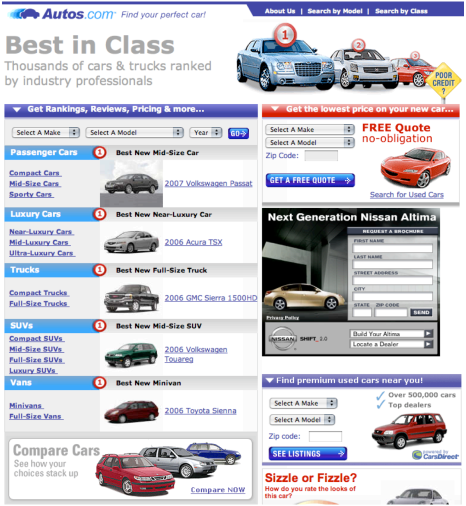
Example

Suppose a person would like to buy “a sports car that costs at most about 22 000 EUR and has a power of around 150 HP”.

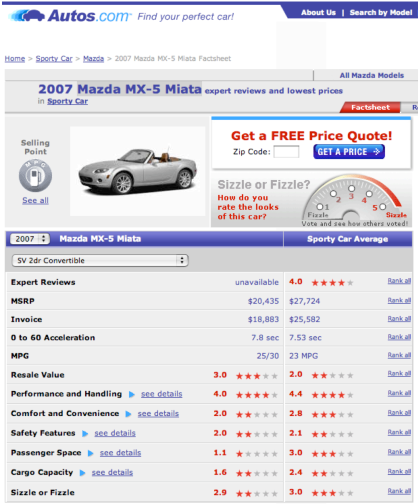
In todays Web, the buyer has to *manually*

- search for car selling web sites, e.g., using Google;
- select the most promising sites;
- browse through them, query them to see the cars that each site sells, and match the cars with the requirements;
- select the offers in each web site that match the requirements; and
- eventually merge all the best offers from each site and select the best ones.

Soft Shopping Agent



Soft Shopping Agent



2007 Mazda MX-5 Miata		Sporty Car Average	
Expert Reviews	unavailable	4.0	Rank all
MSRP	$20,435	$27,724	Rank all
Invoice	$18,883	$25,582	Rank all
0 to 60 Acceleration	7.8 sec	7.53 sec	Rank all
MPG	25/30	23 MPG	Rank all
Resale Value	3.0	2.0	Rank all
Performance and Handling see details	4.0	4.4	Rank all
Comfort and Convenience see details	2.0	2.8	Rank all
Safety Features see details	2.0	2.1	Rank all
Passenger Space see details	1.1	3.0	Rank all
Cargo Capacity see details	1.6	2.4	Rank all
Sizzle or Fizzle	2.9	3.0	Rank all

A *shopping agent* may support us, *automatizing* the whole process once it receives the request/query q from the buyer:

- The agent selects some sites/resources S that it considers as *relevant* to q (represented by probabilistic rules).
- For the top-k selected sites, the agent has to reformulate q using the terminology/ontology of the specific car selling site (which is done using probabilistic rules).
- The query q may contain many so-called *vague/fuzzy* concepts such as "the prize is around 22 000 EUR or less", and thus a car may *match* q to a *degree*. So, a resource returns a ranked list of cars, where the ranks depend on the degrees to which the cars match q.
- Eventually, the agent integrates the ranked lists (using probabilities) and shows the top-n items to the buyer.

Key Ideas

Description logics model a domain of interest in terms of concepts and roles, which represent classes of individuals and binary relations between classes of individuals, respectively.

A description logic knowledge base encodes in particular subset relationships between concepts, subset relationships between roles, the membership of individuals to concepts, and the membership of pairs of individuals to roles.

In fuzzy description logics, these relationships and memberships then have a degree of truth in $[0, 1]$.

Example

$$
\begin{aligned}
&\mathit{Cars} \sqcup \mathit{Trucks} \sqcup \mathit{Vans} \sqcup \mathit{SUVs} \sqsubseteq \mathit{Vehicles}\\
&\mathit{PassengerCars} \sqcup \mathit{LuxuryCars} \sqsubseteq \mathit{Cars}\\
&\mathit{CompactCars} \sqcup \mathit{MidSizeCars} \sqcup \mathit{SportyCars} \sqsubseteq \mathit{PassengerCars}
\end{aligned}
$$

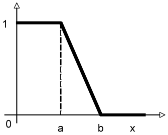
$$
\begin{aligned}
\mathit{Cars} \sqsubseteq\ & (\exists \mathit{hasReview}.\mathit{Integer}) \sqcap (\exists \mathit{hasInvoice}.\mathit{Integer})\\
& \sqcap (\exists \mathit{hasResellValue}.\mathit{Integer}) \sqcap (\exists \mathit{hasMaxSpeed}.\mathit{Integer})\\
& \sqcap (\exists \mathit{hasHorsePower}.\mathit{Integer}) \sqcap \ldots
\end{aligned}
$$

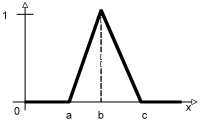
$$
\begin{aligned}
&\mathit{MazdaMX5Miata} \colon \mathit{SportyCar} \sqcap (\exists \mathit{hasInvoice}.18883)\\
&\qquad \sqcap (\exists \mathit{hasHorsePower}.166) \sqcap \ldots\\
&\mathit{MitsubishiEclipseSpyder} \colon \mathit{SportyCar} \sqcap (\exists \mathit{hasInvoice}.24029)\\
&\qquad \sqcap (\exists \mathit{hasHorsePower}.162) \sqcap \ldots
\end{aligned}
$$

We may now encode “costs at most about 22 000 EUR” and “has a power of around 150 HP” in the buyer’s request through the following concepts C and D, respectively:

$$C = \exists \mathit{hasInvoice}.\mathit{LeqAbout22000} \text{ and}$$
$$D = \exists \mathit{hasHorsePower}.\mathit{Around150HP},$$

where $\mathit{LeqAbout22000} = L(22000, 25000)$ and $\mathit{Around150HP} = \mathit{Tri}(125, 150, 175)$.





Syntax

A *normal fuzzy rule* r *is of the form (with atoms* $a, b_1, \ldots, b_m$*):*

$$
\begin{aligned}
a \leftarrow_{\otimes_0} b_1 \wedge_{\otimes_1} b_2 \wedge_{\otimes_2} \cdots \wedge_{\otimes_{k-1}} b_k \wedge_{\otimes_k} & \\
not_{\ominus_{k+1}} b_{k+1} \wedge_{\otimes_{k+1}} \cdots \wedge_{\otimes_{m-1}} not_{\ominus_m} b_m \geqslant v, &
\end{aligned}
\tag{1}
$$

A normal fuzzy program *P* is a finite set of normal fuzzy rules.

A *dl-query* $Q(\mathbf{t})$ *is of one of the following forms:*

- *a concept inclusion axiom* F *or its negation* $\neg F$;
- $C(t)$ *or* $\neg C(t)$, *with a concept* C *and a term* t;
- $R(t_1, t_2)$ *or* $\neg R(t_1, t_2)$, *with a role* R *and terms* t_1, t_2.

A *fuzzy dl-rule* r *is of form (1), where any* $b \in B(r)$ *may be a dl-atom, which is of form* $DL[S_1 op_1 p_1, \ldots, S_m op_m\, p_m; Q](\mathbf{t})$.

A *fuzzy dl-program* $KB = (L, P)$ *consists of a fuzzy description logic knowledge base* L *and a finite set of fuzzy dl-rules* P.

Example

The following fuzzy dl-rule encodes the buyer's request "a sports car that costs at most about 22 000 EUR and that has a power of around 150 HP".

$$\begin{aligned} query(x) \;\leftarrow_{\otimes}\; & DL[SportyCar](x) \wedge_{\otimes} \\ & DL[hasInvoice](x, y_1) \wedge_{\otimes} \\ & DL[LeqAbout22000](y_1) \wedge_{\otimes} \\ & DL[hasHorsePower](x, y_2) \wedge_{\otimes} \\ & DL[Around150HP](y_2) \geqslant 1\,. \end{aligned}$$

Here, $\otimes$ is the Gödel t-norm (that is, $x \otimes y = \min(x, y)$).

Semantics

An *interpretation I is a mapping $I\colon HB_P \rightarrow [0,1]$.*

The truth value of $a = DL[S_1 \uplus\ p_1, \ldots, S_m \uplus p_m; Q](\mathbf{c})$ under L, denoted $I_L(a)$, is defined as the maximal truth value $v \in [0,1]$ such that $L \cup \bigcup_{i=1}^{m} A_i(I) \models Q(\mathbf{c}) \geqslant v$, where

$$A_i(I) = \{S_i(\mathbf{e}) \geqslant I(p_i(\mathbf{e})) \mid I(p_i(\mathbf{e})) > 0,\ p_i(\mathbf{e}) \in HB_P\}\,.$$

I is a model of a ground fuzzy dl-rule *r* of the form (1) under *L*, denoted $I \models_L r$, iff

$$\begin{aligned} I_L(a) \geqslant v \otimes_0 I_L(b_1) \otimes_1 I_L(b_2) \otimes_2 \cdots \otimes_{k-1} I_L(b_k) \otimes_k \\ \ominus_{k+1} I_L(b_{k+1}) \otimes_{k+1} \cdots \otimes_{m-1} \ominus_m I_L(b_m), \end{aligned}$$

I is a *model of a fuzzy dl-program $KB = (L, P)$, denoted $I \models KB$, iff $I \models_L r$ for all $r \in ground(P)$.*

Stratified Fuzzy DL-Programs

Stratified fuzzy dl-programs are composed of hierarchic layers of positive fuzzy dl-programs linked via default negation:

A *stratification of* $KB = (L, P)$ with respect to DL_P is a mapping $\lambda : HB_P \cup DL_P \to \{0, 1, \ldots, k\}$ such that

- $\lambda(H(r)) \geqslant \lambda(a)$ (resp., $\lambda(H(r)) > \lambda(a)$) for each $r \in ground(P)$ and $a \in B^+(r)$ (resp., $a \in B^-(r)$), and
- $\lambda(a) \geqslant \lambda(a')$ for each input atom a' of each $a \in DL_P$,

where $k \geqslant 0$ is the *length of* λ*. A fuzzy dl-program* $KB = (L, P)$ *is* stratified iff it has a stratification λ of some length $k \geqslant 0$.

Theorem: Every stratified fuzzy dl-program *KB* is satisfiable and has a canonical minimal model via a finite number of iterative least models (which does not depend on the stratification of *KB*).

Example

The buyer's request, but in a "different" terminology:

$$query(x) \leftarrow_{\otimes} SportsCar(x) \wedge_{\otimes} hasPrize(x, y_1) \wedge_{\otimes} hasPower(x, y_2) \wedge_{\otimes} DL[LeqAbout22000](y_1) \wedge_{\otimes} DL[Around150HP](y_2) \geqslant 1$$

Ontology alignment mapping rules:

$$\begin{aligned} SportsCar(x) &\leftarrow_{\otimes} DL[SportyCar](x) \wedge_{\otimes} sc_{pos} \geqslant 1 \\ hasPrize(x) &\leftarrow_{\otimes} DL[hasInvoice](x) \wedge_{\otimes} hi_{pos} \geqslant 1 \\ hasPower(x) &\leftarrow_{\otimes} DL[hasHorsePower](x) \wedge_{\otimes} hhp_{pos} \geqslant 1 , \end{aligned}$$

Probability distribution μ:

$$\begin{array}{ll} \mu(sc_{pos}) = 0.91 & \mu(sc_{neg}) = 0.09 \\ \mu(hi_{pos}) = 0.78 & \mu(hi_{neg}) = 0.22 \\ \mu(hhp_{pos}) = 0.83 & \mu(hhp_{neg}) = 0.17 . \end{array}$$

The following are some tight consequences:

$$\begin{aligned} KB &\Vdash\!\sim_{tight} (\mathbf{E}[q(\mathit{MazdaMX5Miata})])[0.21, 0.21] \\ KB &\Vdash\!\sim_{tight} (\mathbf{E}[q(\mathit{MitsubishiEclipseSpyder})])[0.19, 0.19]\,. \end{aligned}$$

Informally, the expected degree to which *MazdaMX5Miata* matches the query q is 0.21, while the expected degree to which *MitsubishiEclipseSpyder* matches the query q is 0.19,

Thus, the shopping agent ranks the retrieved items as follows:

rank	item	degree
1.	*MazdaMX5Miata*	0.21
2.	*MitsubishiEclipseSpyder*	0.19

- Description logic programs that allow for dealing with probabilistic uncertainty and fuzzy vagueness.
- Semantically, probabilistic uncertainty can be used for data integration and ontology mapping, and fuzzy vagueness can be used for expressing vague concepts.
- Query processing based on fixpoint iterations.