Logic-based Rule Learning for the Web of Data

Francesca A. Lisi

Department of Computer Science
Lab of Knowledge Acquisition and Machine Learning (LACAM)

francesca.lisi@uniba.it

July 12, 2017

Overview

Introduction I

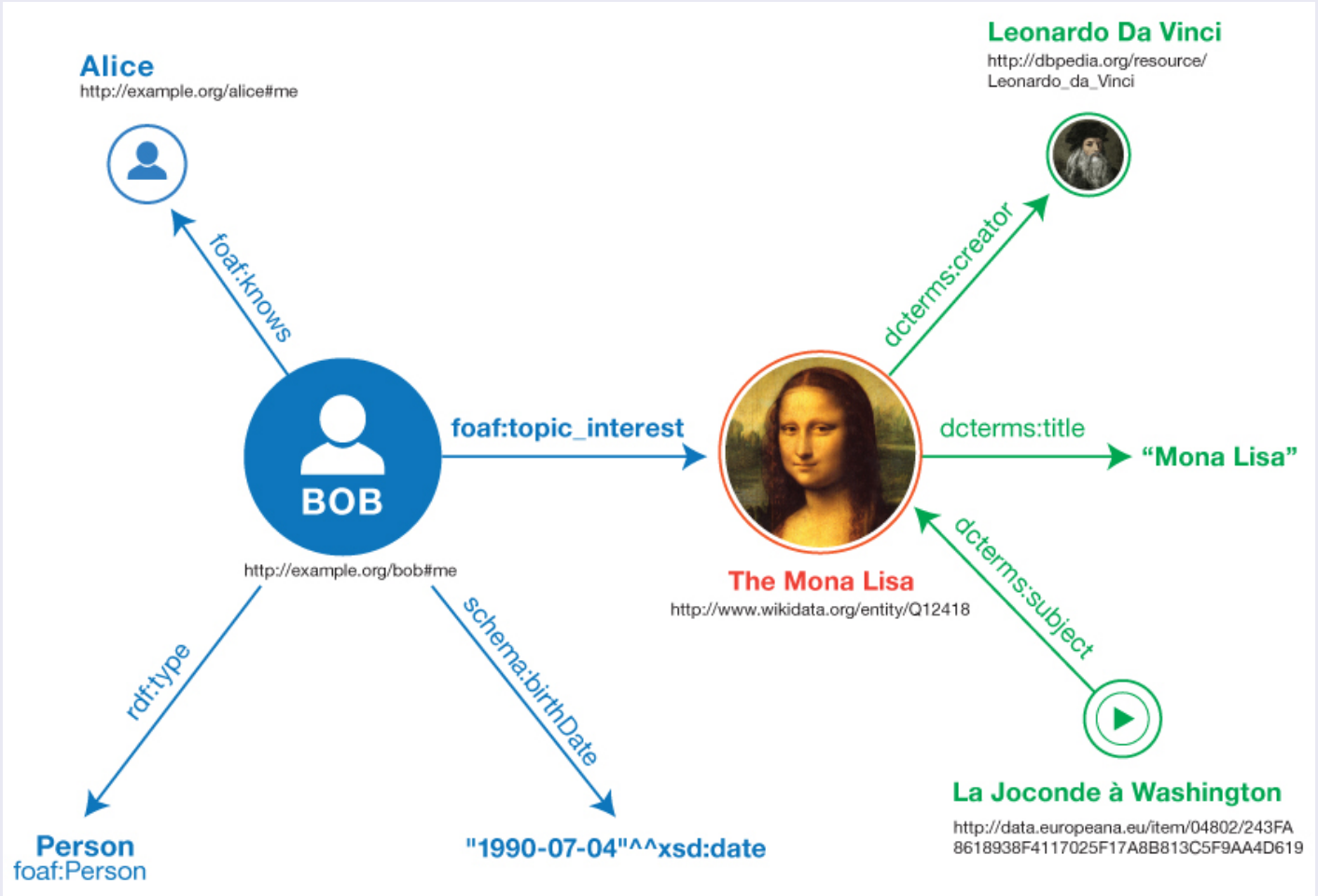
The Web of Data

- Feature: builds upon the WWW infrastructure to represent and interrelate data (aka *Linked Data*),
- Aim: transforming the Web from a distributed file system into a *distributed database system*.
- The foundational *standards* of the Web of Data include:

  URI used to identify resources
  RDF used to relate resources

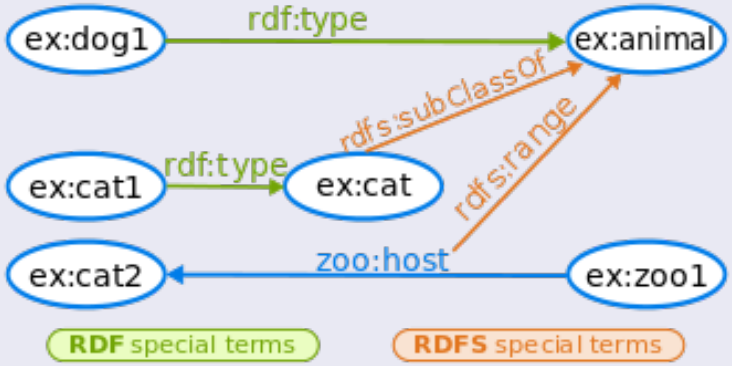
Introduction II

Resource Description Framework (RDF)
`https://www.w3.org/RDF/`



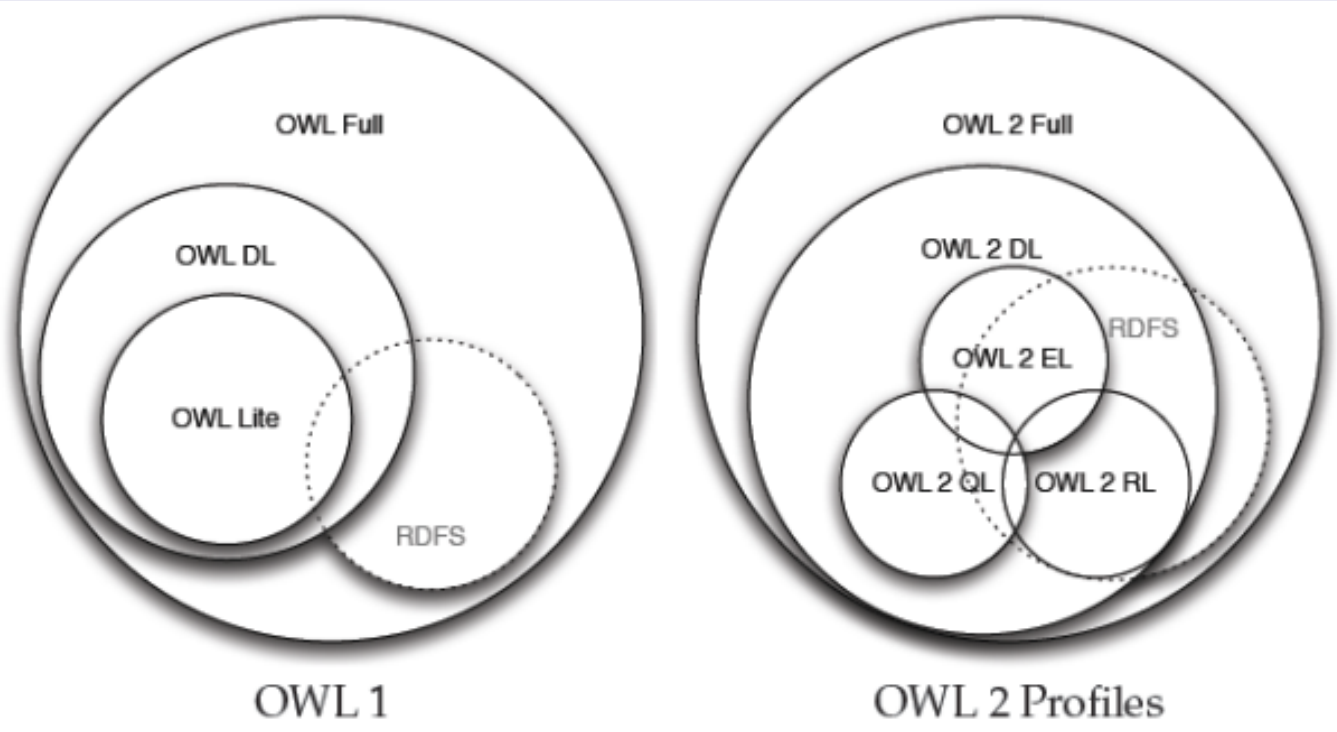
Introduction III

RDF Schema
https://www.w3.org/TR/rdf-schema/



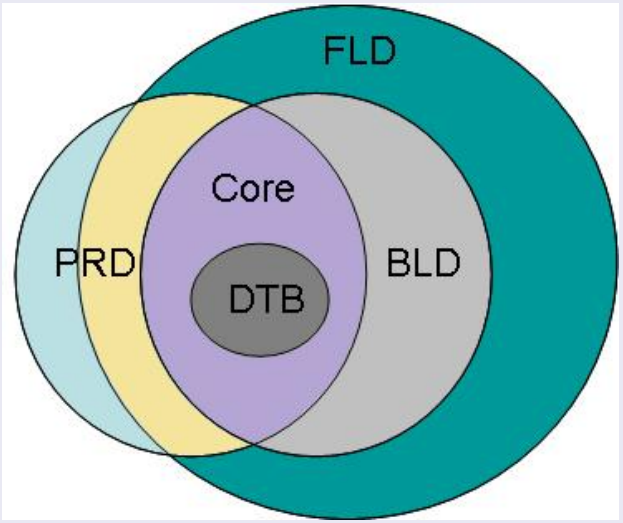
Introduction IV

Ontology Web Language (OWL)
`https://www.w3.org/OWL/`



Introduction V

Rule Interchange Format (RIF)
http://www.w3.org/2005/rules/wiki/RIF_Working_Group



Part I: Basics of Logic-based Rule Learning

- Why do we need rule learning algorithms?
- What are the foundations of rule learning?
- How can we cast rule learning within the framework of logic programming?

Motivation I

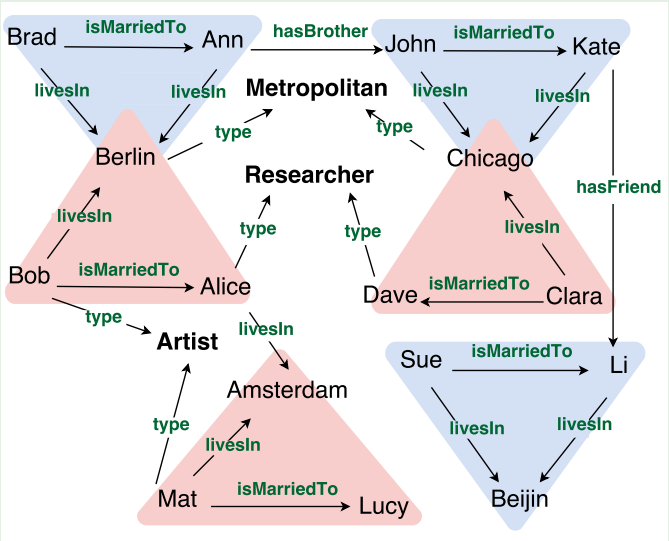
The curation of large RDF graphs

- *Knowledge graphs* (KGs) are huge sets of RDF triples
  - see, *e.g.*, DBpedia [a]
- KGs are inherently *incomplete* since they are automatically constructed by applying information extraction techniques.
- The task of *completion* (aka *link prediction*) is of crucial importance for the curation of KGs.
- Data mining algorithms can be exploited to automatically build rules able to make predictions on missing links.

[a] `http://wiki.dbpedia.org/`

Motivation II

An example of rule mining for KG completion



New facts, *e.g.*, $livesIn(alice, berlin)$, $livesIn(dave, chicago)$ and $livesIn(lucy, amsterdam)$, can be derived from the following mined rule:

$$r1: \ isMarriedTo(x, y), livesIn(x, z) \Rightarrow livesIn(y, z) \qquad (1)$$

and used to complete the KG [Tran et al., 2017].

Motivation III

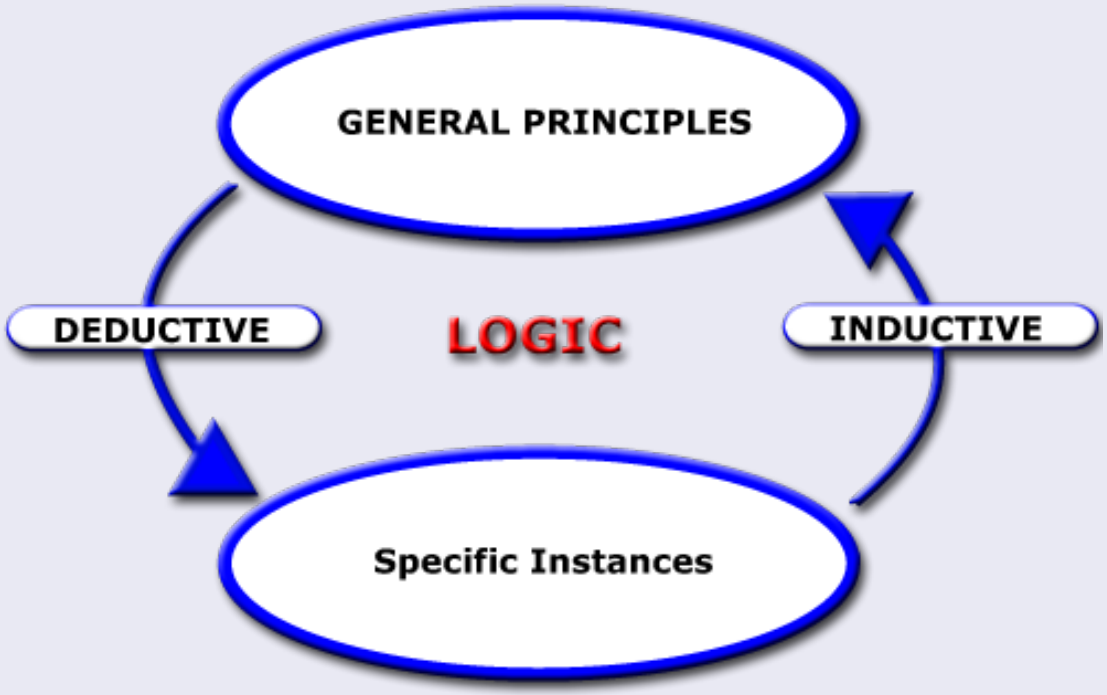
Rule acquisition needs automation

- Demanding knowledge engineering activity for very large KBs
  - Rule authoring task more error-prone than rule reviewing
  - Partial automation by applying rule learning algorithms
- In the Web of Data learned rules might take ontologies into account

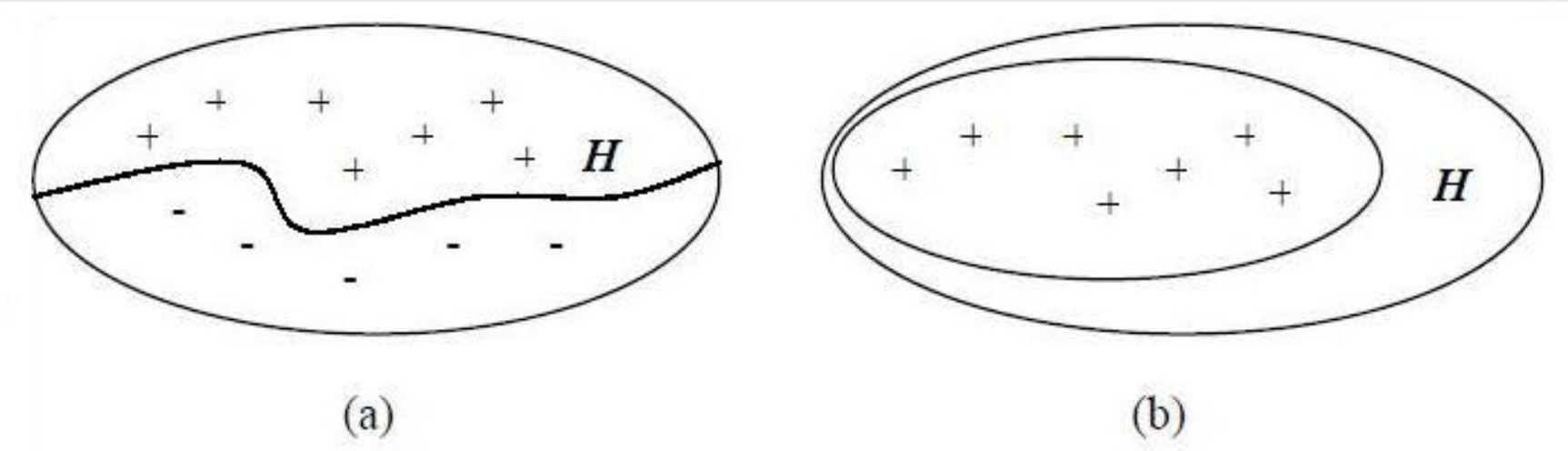
Rule Learning and Mining

- Inductive inference
- Generalization
- Classification rule learning
- Association rule mining

Inductive inference



Scope of induction



(a) Discriminate for making predictions on unseen data (*e.g.*, classification)

(b) Characterize for understanding the available data (*e.g.*, clustering)

Rule Learning and Mining III

Generalization

- It proceeds from a premise about a sample to a conclusion about the population.
  - The proportion Q of the sample has attribute A.
  - Therefore:
  - The proportion Q of the population has attribute A.
- While the conclusion of a deductive argument is *certain*, the truth of the conclusion of an inductive argument is *probable*, based upon the evidence given.
- Instead of being valid or invalid, inductive arguments are either *strong* or *weak*, which describes how probable it is that the conclusion is true.
  - In a *strong* inductive generalization, the sample must represent the population.

Rule Learning and Mining IV

What kind of rules?

Classification rule $IF\ f_1\ AND\ f_2\ AND\ \ldots\ AND\ f_L\ THEN\ Class = c_i$

Association rule $X \Rightarrow Y(s\%, c\%)$ (see Baralis' invited talk of this morning!)

Rule Learning and Mining V

Classification rule learning

Given:

- a *data description language*, defining the form of data,
- a *hypothesis description language*, defining the form of rules,
- a *coverage function* $Covered(r, e)$, defining whether rule r covers example e,
- a *class attribute* **C**, and
- a set E of *training examples*, instances for which the class labels are known, described in the data description language.

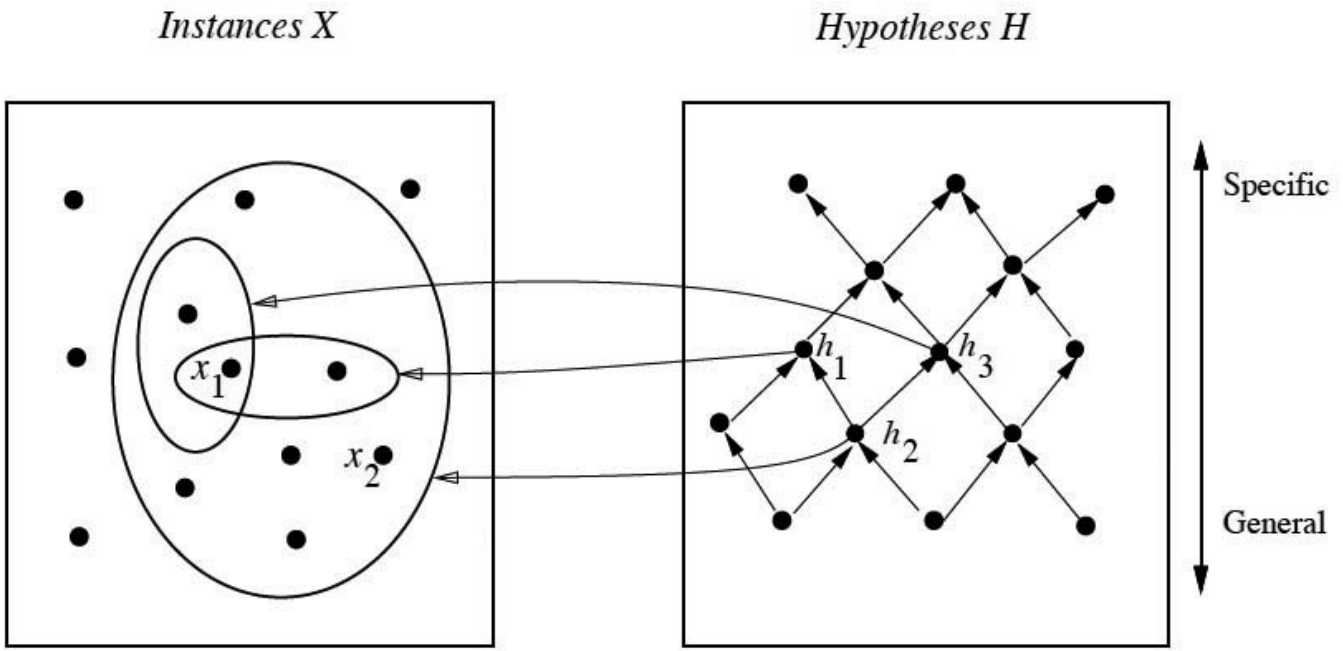
Find: a *hypothesis* in the form of a rule set R formulated in the hypothesis description language which is

- *complete*, *i.e.*, it covers all the examples, and
- *consistent*, *i.e.*, it predicts the correct class for all the examples.

The special case of Concept Learning

- Unique class (called target concept)
- Examples split into positive and negative for the given class
- Generalization as a search in the space of hypotheses structured according to some generality order [Mitchell, 1982]
- Learning involves searching this space with the goal of finding a hypothesis that best fits the examples

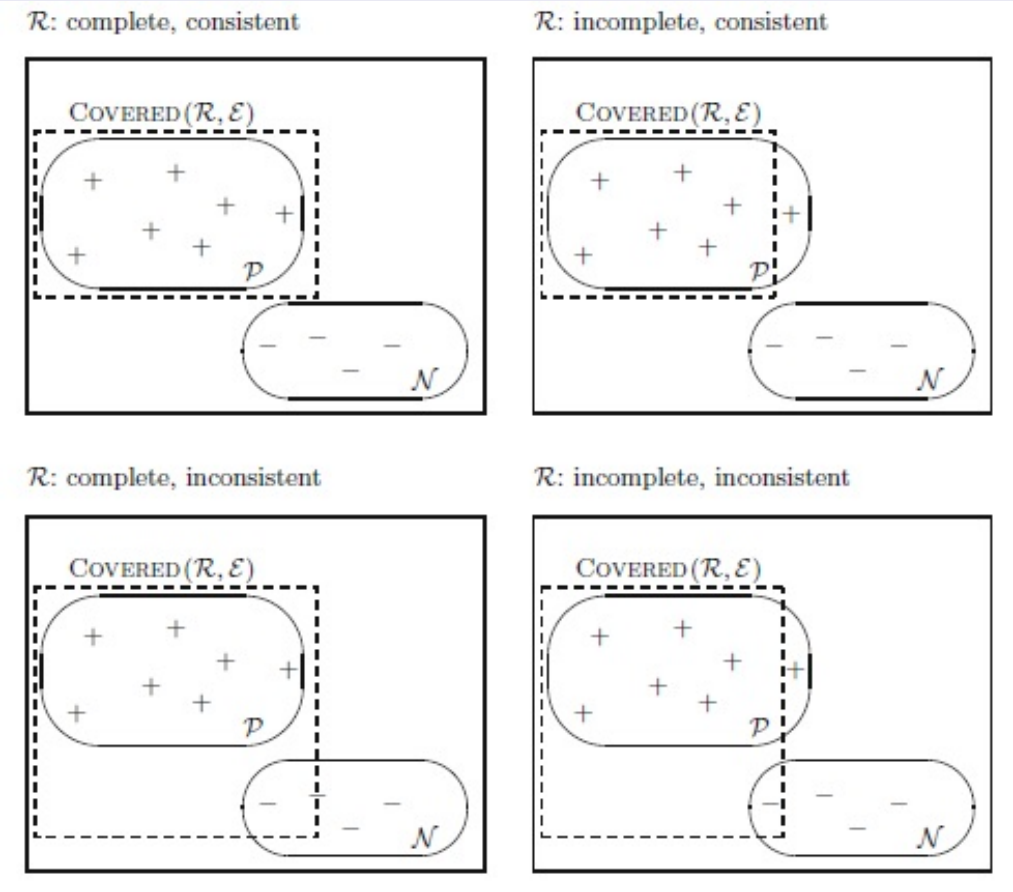
Example of generalization as search [Mitchell, 1997]



x_1= <Sunny, Warm, High, Strong, Cool, Same>
x_2= <Sunny, Warm, High, Light, Warm, Same>

h_1= <Sunny, ?, ?, Strong, ?, ?>
h_2= <Sunny, ?, ?, ?, ?, ?>
h_3= <Sunny, ?, ?, ?, Cool, ?>

Coverage function [Fürnkranz et al., 2012]

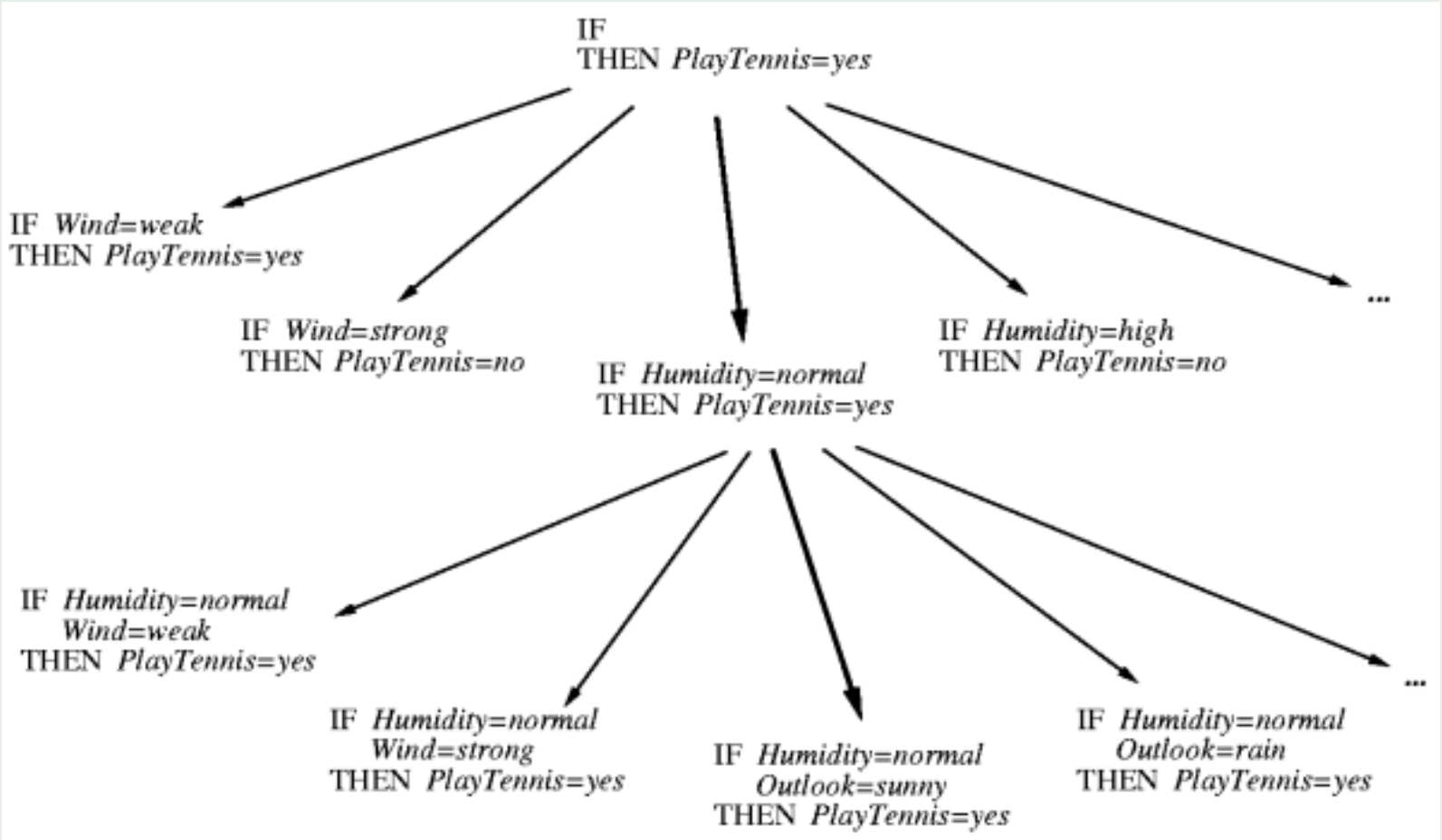


Rule Learning and Mining IX

Sequential covering method

- The learning process carries on inducing rules until all positive examples are covered.
- When a rule is induced, the positive examples covered by the rule are removed from $\mathcal{E}$.
- In order to induce a rule, the learning process starts with the most general rule and refines it by applying some specialization operator.
- The iterated specialization of the rule continues until the rule does not cover any negative example and its *confidence degree* is greater than a fixed threshold (θ).
- The confidence degree of rules being generated with the specialization operator allows for evaluating the *information gain* obtained on each refinement step.

Example of learning one rule [Mitchell, 1997]



Rule Learning and Mining XI

Association rule mining [Agrawal et al., 1993]

- It consists of two tasks:
  1. Discovery of *frequent* association patterns
  2. Derivation of *strong* association rules from frequent patterns (*trivial!*)
- Two thresholds to be set up for evaluating the goodness of rules
  - frequency measured w.r.t. the given *minimum support*
  - strength measured w.r.t. the given *minimum confidence*
- Many variants and extensions (*see Baralis' invited talk of this morning!*)

Frequent pattern discovery [Mannila and Toivonen, 1997]

Given

- a data set **r**,
- a language $\mathcal{L}$,
- a minimum support threshold *minsup*

the task of *frequent pattern discovery* is to find the set $\mathcal{F}$ of all $P \in \mathcal{L}$ that are frequent in **r**.

Evaluation function

A pattern $P \in \mathcal{L}$ with $supp(P, \mathbf{r}) = s$ is *frequent* in $\mathbf{r}$ iff $s \geq minsup$.

Levelwise search method

If a generality order $\succeq$ over $\mathcal{L}$ can be defined such that $\succeq$ is monotone w.r.t. *supp*, the lattice $(\mathcal{L}, \succeq)$ can be searched with a breadth-first strategy level by level of depth.

Levelwise search method

`Inputs:` $\mathbf{r}$, $(\mathcal{L}, \succeq)$, *minsup*

`Outputs:` $\mathcal{F}$

1. $\mathcal{F} \leftarrow \emptyset$
2. $k \leftarrow 1$
3. $\mathcal{C}_k \leftarrow \{P \in \mathcal{L} | \forall Q \in \mathcal{L} : P \succ Q\}$
4. **While** $\mathcal{C}_k \neq \emptyset$ **do**
5. $\quad \mathcal{F}_k \leftarrow \{P \in \mathcal{C}_k | freq(\mathbf{r}, P)\}$
6. $\quad \mathcal{F} \leftarrow \mathcal{F} \cup \mathcal{F}_k$
7. $\quad k \leftarrow k + 1$
8. $\quad \mathcal{C}_k \leftarrow \{P \in \mathcal{L} | \forall Q \succ P : Q \in \mathcal{F}\} \setminus \bigcup_{j \leq k} \mathcal{C}_j$
9. **Return** $\mathcal{F}$

Inductive Logic Programming in a nutshell

- Major logic-based approach to rule learning [Muggleton, 1990]
  - Concept Learning within the LP framework
  - Use of background knowledge (BK)
- Bunch of techniques for structuring, searching and bounding the hypothesis space [Nienhuys-Cheng and de Wolf, 1997]
- Several extensions beyond LP as well as beyond concept learning

Techniques of Inductive Logic Programming

- Generality orders
- Refinement operators
- Declarative bias

ILP Techniques I

Generality orders

Two kinds of comparison between clauses

- Syntactic if it is based on the structure of the clauses
- Semantic if it looks at the implications of the clauses

Definition (θ-subsumption [Plotkin, 1970])

Let c_1 and c_2 be two clauses. We say that c_1 θ-subsumes c_2 if there exists a substitution θ, such that $c_1\theta \subseteq c_2$.

Example of comparison under θ-subsumption

- $c_1 = father(X, Y) :- parent(X, Y)$
- $c_2 = father(X, Y) :- parent(X, Y), male(X)$
- $c_3 = father(bob, Y) :- parent(bob, Y)$
- c_1 θ-subsumes c_2 for $\theta = \{\}$
- c_1 θ-subsumes c_3 for $\theta = \{X/bob\}$
- c_2 and c_3 do not θ-subsume one another (incomparable clauses)

ILP Techniques III

Properties of θ-subsumption [Plotkin, 1970]

Logical

- if c_1 θ-subsumes c_2 then $c_1 \models c_2$ (soundness)
- possibly $c_1 \models c_2$ without c_1 θ-subsuming c_2 (but only for recursive clauses) (incompleteness)
- Checking θ-subsumption is decidable but NP-complete

Algebraic

- It is a semi-order relation
- It generates equivalence classes
- It generates a partial order on those equivalence classes
- Thus, reduced clauses form a lattice

A further example showing the limits of θ-subsumption

Background knowledge B

- $pet(X) : -cat(X)$
- $pet(X) : -dog(X)$
- $small(X) : -cat(X)$

Clauses:

- $c_1 = cuddlypet(X) : -small(X), pet(X)$
- $c_2 = cuddlypet(X) : -cat(X)$

Intuitively c_1 is more general than c_2

Formally Is there any way of proving that under θ-subsumption?

ILP Techniques V

Definition (generalized subsumption [Buntine, 1988])

Given two definite clauses c_1 and c_2 standardized apart and a definite program $\mathcal{K}$, we say that c_1 subsumes c_2 w.r.t. $\mathcal{K}$ iff there exists a ground substitution θ for c_1 such that:

1. $head(c_1)\theta = head(c_2)\sigma$, and
2. $\mathcal{K} \cup body(c_2)\sigma \models body(c_1)\theta$ where σ is a Skolem substitution for c_2 with respect to $\{c_1\} \cup \mathcal{K}$.

Example of comparison under generalized subsumption

Background knowledge B

- $pet(X) : -cat(X)$
- $pet(X) : -dog(X)$
- $small(X) : -cat(X)$

Clauses:

- $c_1 = cuddlypet(X) : -small(X), pet(X)$
- $c_2 = cuddlypet(X) : -cat(X)$

Intuitively c_1 is more general than c_2

Formally It can be proved under under generalized subsumption!

Definition (refinement operator)

Given a quasi-ordered search space $(\Sigma, \preceq)$

- a *downward refinement operator* is a mapping $\rho : \Sigma \rightarrow 2^{\Sigma}$ such that

$$\forall \alpha \in \Sigma \quad \rho(\alpha) \subseteq \{\beta \in \Sigma \mid \beta \preceq \alpha\}$$

- an *upward refinement operator* is a mapping $\delta : \Sigma \rightarrow 2^{\Sigma}$ such that

$$\forall \alpha \in \Sigma \quad \delta(\alpha) \subseteq \{\beta \in \Sigma \mid \alpha \preceq \beta\}$$

ILP Techniques VIII

Shapiro's specialization operator

- Top down search in clausal spaces ordered according to θ-subsumption
- Specialization by either adding literals to the body of a clause or by applying some substitution
- Used in many ILP systems

ILP Techniques IX

Least general generalization (lgg) [Plotkin, 1970]

An upward refinement operator for $(\mathcal{L}, \preceq_\theta)$

- $c_1 : f(t, a) : -p(t, a), m(t), f(a)$
- $c_2 : f(j, p) : -p(j, p), m(j), m(p)$
- $lgg(c_1, c_2) = f(X, Y) : -p(X, Y), m(X), m(Z)$

ILP Techniques X

Properties of refinement operators

Local finiteness only a finite set of specializations of each clause

Properness only proper specializations (*i.e.*, belonging to different equivalence classes)

Global completeness each point in lattice is reachable from top

Ideality Local finiteness + properness + completeness

Local completeness each point directly below c is in $\rho(c)$ (useful for greedy systems)

Optimality no point in lattice is reached twice (useful for exhaustive systems)

ILP Techniques XI

Declarative bias

Language bias Specifies and restricts the set of clauses or theories that are permitted

Search bias Concerns the way the system searches through the hypothesis space

Validation bias Determines when the learned theory is acceptable, so when the learning process may stop

Learning and mining rules with ILP

- Two representative ILP algorithms:
  1. FOIL [Quinlan, 1990]
  2. WARMR [Dehaspe and Toivonen, 1999]
- Obtained from previous propositional learning algorithms by following the upgrading methodology [De Raedt, 2008]

Learning and mining rules with ILP I

FOIL: general features

- Task: classification rule learning
- Method: sequential covering (aka separate-and-conquer)
- BK: only extensional
- Knowledge representation formalism: DATALOG.
- Upgrade from: AQ (Michalski, 1977).

FOIL: Algorithm for learning sets of rules

function FOIL-LEARN-SETS-OF-RULES(H, $\mathcal{E}^+$, $\mathcal{E}^-$, $\mathcal{K}$): $\mathcal{H}$
begin
1. $\mathcal{H} \leftarrow \emptyset$;
2. **while** $\mathcal{E}^+ \neq \emptyset$ **do**
3. $\quad r \leftarrow$ FOIL-LEARN-ONE-RULE(H, $\mathcal{E}^+$, $\mathcal{E}^-$, $\mathcal{K}$);
4. $\quad \mathcal{H} \leftarrow \mathcal{H} \cup \{r\}$;
5. $\quad \mathcal{E}^+_r \leftarrow \{e \in \mathcal{E}^+ | \mathcal{K} \cup r \models e\}$;
6. $\quad \mathcal{E}^+ \leftarrow \mathcal{E}^+ \setminus \mathcal{E}^+_r$;
7. **endwhile**
8. **return** $\mathcal{H}$

end

FOIL: Algorithm for learning one rule

function FOIL-LEARN-ONE-RULE(H, $\mathcal{E}^+$, $\mathcal{E}^-$, $\mathcal{K}$): r
begin
1. $B(\vec{x}) \leftarrow \top$; 2. $r \leftarrow \{B(\vec{x}) \rightarrow H(\vec{x})\}$; 3. $\mathcal{E}_r^- \leftarrow \mathcal{E}^-$;
4. **while** $cf(r) < \theta$ **and** $\mathcal{E}_r^- \neq \emptyset$ **do**
5. $\quad B_{best}(\vec{x}) \leftarrow B(\vec{x})$; 6. $maxgain \leftarrow 0$;
7. $\quad$ **foreach** $l \in \mathcal{K}$ **do**
8. $\quad\quad gain \leftarrow \text{GAIN}(B(\vec{x}) \wedge l(\vec{x}) \rightarrow H(\vec{x}), B(\vec{x}) \rightarrow H(\vec{x}))$;
9. $\quad\quad$ **if** $gain \geq maxgain$ **then**
10. $\quad\quad\quad maxgain \leftarrow gain$;
11. $\quad\quad\quad B_{best}(\vec{x}) \leftarrow B(\vec{x}) \wedge l(\vec{x})$;
12. $\quad\quad$ **endif**
13. $\quad$ **endforeach**
14. $\quad r \leftarrow \{B_{best}(\vec{x}) \rightarrow H(\vec{x})\}$;
15. $\quad \mathcal{E}_r^- \leftarrow \mathcal{E}_r^- \setminus \{e \in \mathcal{E}^- | \mathcal{K} \cup r \models e\}$;
16. **endwhile**
17. **return** r
end

FOIL: The GAIN function

$$\text{GAIN}(r', r) = p * (log_2(cf(r')) - log_2(cf(r))) \ , \tag{2}$$

where p is the number of distinct positive examples covered by the rule r that are still covered by r'.

FOIL: The computation of confidence degree

Given a Horn clause $B(\vec{x}) \rightarrow H(\vec{x})$, the confidence degree is given by:

$$cf(B(\vec{x}) \rightarrow H(\vec{x})) = P(B(\vec{x}) \wedge H(\vec{x}))/P(B(\vec{x})) \ . \tag{3}$$

Confidence degrees are computed in the spirit of domain probabilities [Halpern, 1990].

WARMR: general features

- Task: association rule mining
- Method: levelwise search
- BK: also intensional
- Knowledge representation formalism: DATALOG.
- Upgrade from: APRIORI [Agrawal et al., 1993].

WARMR vs APRIORI

	WARMR	APRIORI
Search space	DATALOG query lattice	itemset lattice
Generality relation	θ-subsumption relation	subset relation
Loading unit from database	sub-database	tuple
Candidate evaluation technique	coverage test	subset check

WARMR: discovering frequent relational patterns

`Inputs:`
a DATALOG database $\mathbf{r}$; a WRMODE language $\mathcal{L}$; a threshold *minsup*
`Outputs:` the set $\mathcal{F}$ of all queries $Q \in \mathcal{L}$ with $\sigma(Q, \mathbf{r}) \geq$ *minsup*

1. $\mathcal{I} \leftarrow \emptyset$
2. $\mathcal{F} \leftarrow \emptyset$
3. $k \leftarrow 1$
4. $\mathcal{C}_k \leftarrow \{? - key\}$
5. **While** $\mathcal{C}_k \neq \emptyset$ **do**
6. $\quad \mathcal{F}_k \leftarrow$ WARMR-EVAL$(\mathbf{r}, \mathcal{C}_k, minsup)$
7. $\quad \mathcal{F} \leftarrow \mathcal{F} \cup \mathcal{F}_k$
8. $\quad k \leftarrow k + 1$
9. $\quad \mathcal{C}_k \leftarrow$ WARMR-GEN$(\mathcal{F}_{k-1}, \mathcal{F}, \mathcal{I}, \mathcal{L})$
10. **Return** $\mathcal{F}$

WARMR: candidate generation

`Inputs:`
the WRMODE language $\mathcal{L}$; the set $\mathcal{I}$ of infrequent queries;
the set $\mathcal{F}$ of frequent queries; the set $\mathcal{F}_k$ of frequent queries for level k
`Outputs:` the set $\mathcal{C}_{k+1}$ of candidate queries for level $k+1$

1. $\mathcal{C}_{k+1} \leftarrow \emptyset$
2. **Foreach** query $Q_j \in \mathcal{F}_k$ **do**
3. **Foreach** immediate refinement $Q'_j \in \mathcal{L}$ of Q_j **do**
4. Add Q'_j to $\mathcal{C}_{k+1}$, unless:
5. (i) Q'_j is more specific than some query in $\mathcal{I}$, or
6. (ii) Q'_j is equivalent to some query in $\mathcal{F}_k \cup \mathcal{F}$
7. **Endforeach**
8. **Endforeach**

Return $\mathcal{C}_{k+1}$

Part II: Extensions of Logic-based Rule Learning

- What are the logics underlying the Web of Data?
- How can ILP be extended for dealing with incomplete knowledge?
- How can ILP be extended for addressing imprecision and vagueness?

Logics for the Web of Data I

Description Logics

- Family of logic-based KR formalisms [Baader et al., 2007]
  - Structured incomplete knowledge
  - Focus on entities rather than on relationships
  - Descendants of semantic networks and KL-ONE
- Mapping to decidable FOL fragments [Borgida, 1996]
  - Variable-free syntax
- Killer application: ontologies and Semantic Web

Description Logics I

Basics

Atomic concepts Unary predicates/formulae with one free variable (A)

- *e.g.*, *Person*, *Doctor*, *HappyParent*

Atomic roles Binary predicates/formulae with two free variables (R)

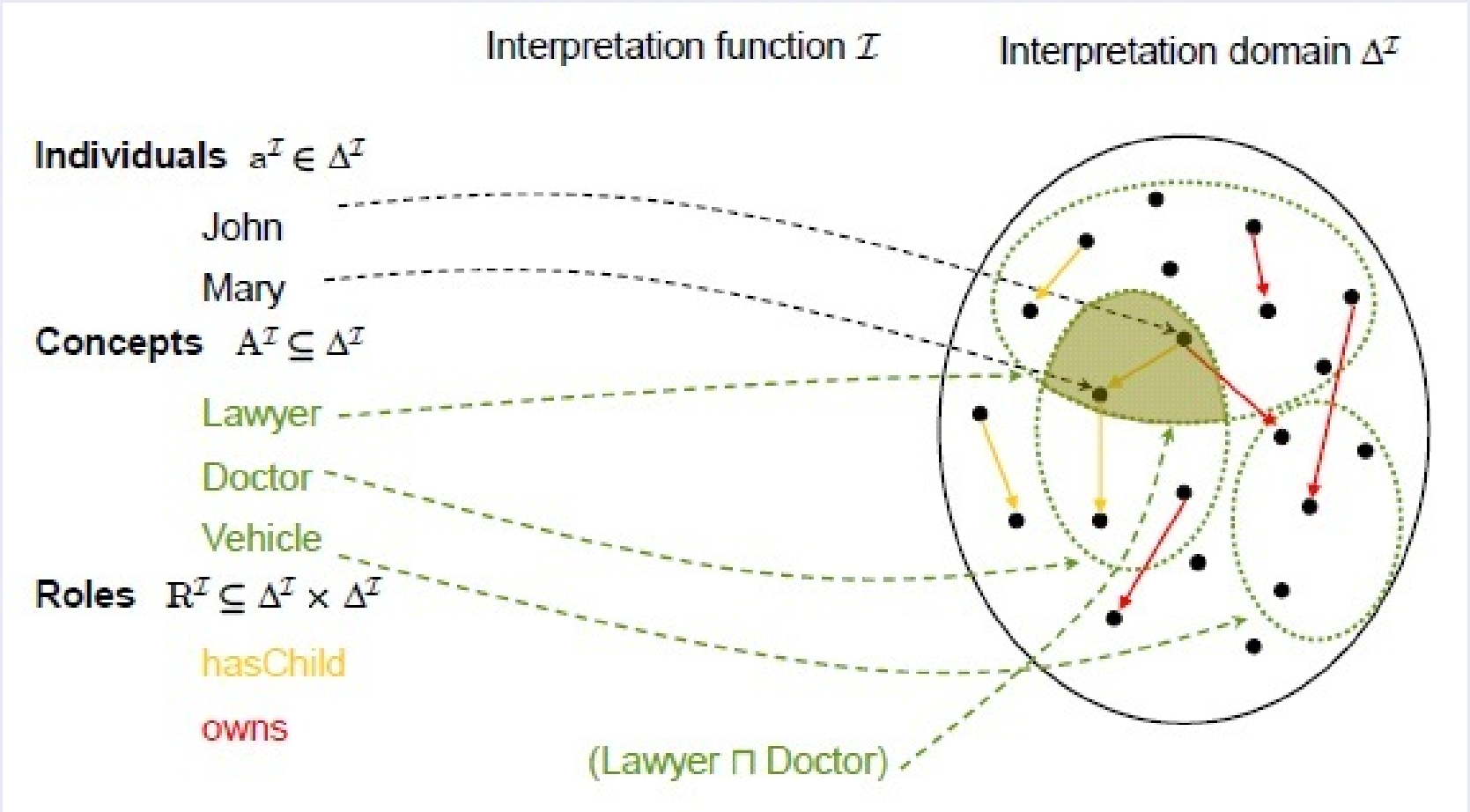
- *e.g.*, *hasChild*, *loves*

Individuals Constants (a)

- *e.g.*, *John*, *Mary*, *Italy*

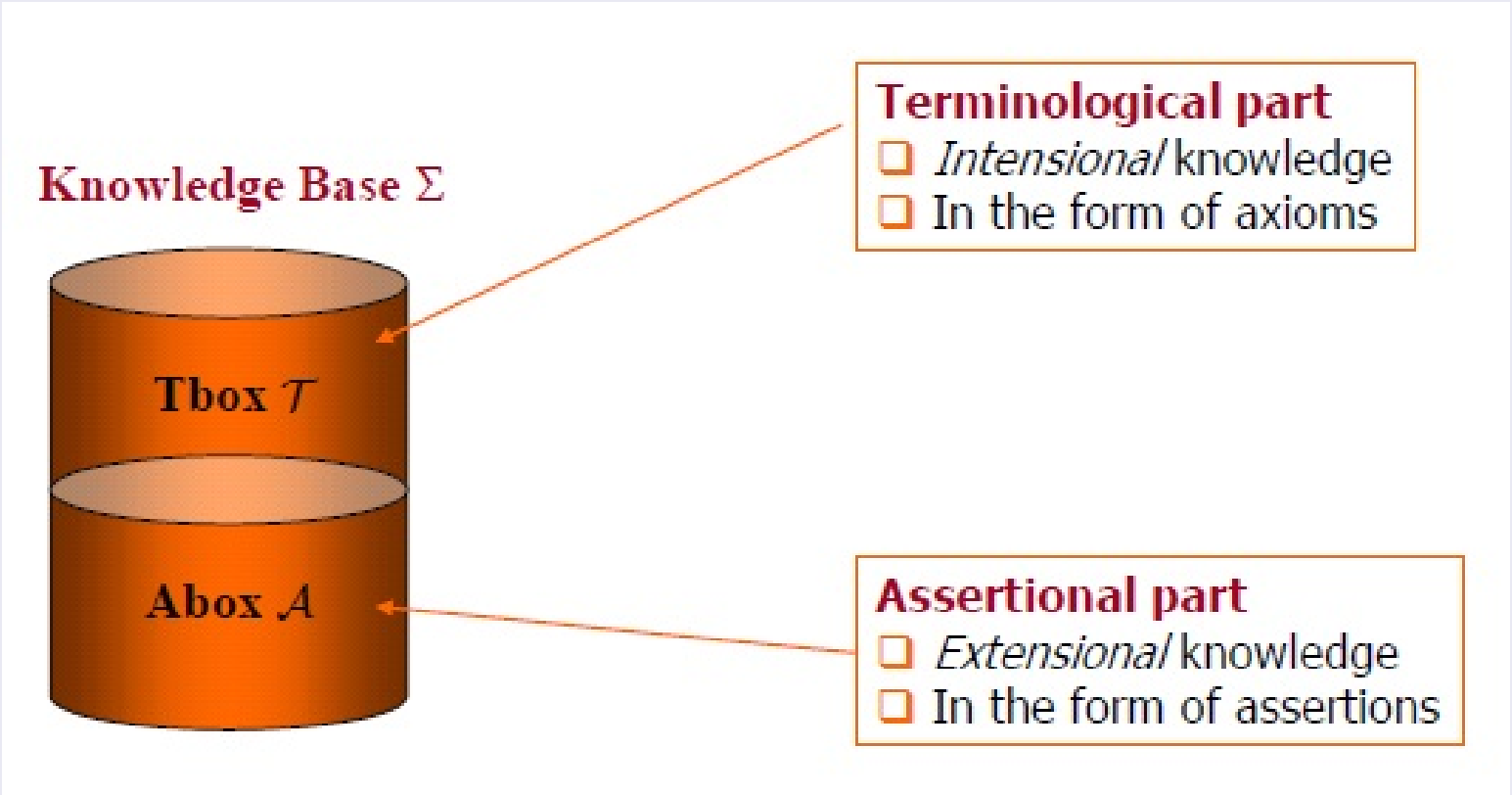
Constructors Operators (*e.g.*, $\sqcap$) for forming complex concepts and roles from atomic ones restricted so that satisfiability/subsumption is decidable and, if possible, of low complexity

Direct semantics



Description Logics III

Knowledge bases



Description Logics IV

Open World Assumption (OWA)

- The information in an Abox is generally considered to be incomplete (*open world*)
- An Abox represents possibly infinitely many interpretations, namely its models
- Query answering requires nontrivial reasoning
- Classical negation!

Description Logics V

Main reasoning tasks

Consistency check Is the KB satisfiable? *i.e.*, is there a model that satisfies both $\mathcal{T}$ and $\mathcal{A}$?

Subsumption check Is $C \sqsubseteq D$ satisfied in all models of $\mathcal{T}$?

Instance check Is $C(a)$ derivable from Σ?

Description Logics VI

Reasoning algorithms based on tableau calculi

Tableau calculus

1. Applies rules that correspond to DL constructors
   - E.g., $John : (Person \sqcap Doctor) \rightarrow_{\sqcap} John : Person$ and $John : Doctor$
2. Stops when no more rules applicable or clash occurs
   - Clash is an obvious contradiction, e.g., $A(x)$, $\neg A(x)$

- Some rules are nondeterministic (e.g., $\sqcup$, $\exists$)
  - In practice, this means *search*
- Cycle check (blocking) often needed to ensure termination

Description Logics VII

Naming conventions for DLs

$\mathcal{F}$	Functional restrictions
$\mathcal{E}$	Full existential qualification
$\mathcal{U}$	Concept union
$\mathcal{C}$	Complex concept negation
$\mathcal{H}$	Role hierarchy
$\mathcal{R}$	Limited complex role inclusion axioms; reflexivity; role disjointness
$\mathcal{O}$	Nominals
$\mathcal{I}$	Inverse properties
$\mathcal{N}$	Number restrictions
$\mathcal{Q}$	Qualified number restrictions

Notable examples of the DL family

- Smallest expressive DL is $\mathcal{ALC}$ [Schmidt-Schauss and Smolka, 1991]
- $\mathcal{SHIQ}$ [Horrocks et al., 2000] is a very expressive DL
  - $\mathcal{S}$ is an abbreviation for $\mathcal{ALC}$ with transitive roles
  - DL underlying the W3C Ontology Web Language OWL [Horrocks et al., 2003]
- Efficient query answering in DL-Lite [Calvanese et al., 2007]

Description Logics VIII

Syntax and semantics of $\mathcal{ALC}$ KBs

bottom concept	$\bot$	$\emptyset$
top concept	$\top$	$\Delta^{\mathcal{I}}$
atomic concept	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
role	R	$R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
individual	a	$a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$
concept negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
concept conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
concept disjunction	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
value restriction	$\forall R.C$	$\{x \in \Delta^{\mathcal{I}} \mid \forall y\ (x,y) \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$
existential restriction	$\exists R.C$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y\ (x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
equivalence axiom	$C \equiv D$	$C^{\mathcal{I}} = D^{\mathcal{I}}$
subsumption axiom	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
concept assertion	$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
role assertion	$R(a,b)$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$

Description Logics IX

Example of an $\mathcal{ALC}$ KB $\mathcal{K}$

A1 $DairyProduct \sqsubseteq Product$

F1 $DairyProduct(mozzarella)$

Example of instance check in $\mathcal{ALC}$

Is $Product(mozzarella)$ derivable from the KB $\mathcal{K}$? or equivalently, is $\mathcal{K} \cup \{\neg Product(mozzarella)\}$ consistent?

Description Logics X

Extensions

- Fuzzy DLs
  - See [Straccia, 2015] for a tutorial lecture
- Probabilistic DLs, *e.g.*, [Lukasiewicz, 2008]
  - See Zese's tutorial talk at RuleML+RR 2017 on Thursday 13 July morning
- DLs extended/integrated with LP
  - *Focus of this talk!*

Logic Programming and DLs

- What are the differences between LP and DLs?
- Is there an overlap between the two?
- Can we combine them?

LP vs. DLs

A summary

1. CWA vs OWA
2. Single vs. multiple models
3. Negation as failure vs. classical negation
4. Strong negation vs. classical negation
5. Treatment of equality
   - Unique Names Assumption (UNA) [Reiter, 1980] might not hold in DLs
6. Existential quantification
   - Recent work on $\text{DATALOG}^{\pm}$ [Calì et al., 2009]
7. Decidability

Intersection between LP and DLs

Exploring the overlap

A partial overlap exists between LP and DLs which allows the extension and/or adaptation of known results in LP to DLs and viceversa

- Horn fragment of DLs
  [Krötzsch et al., 2008, Krötzsch et al., 2013, Krötzsch et al., 2015]
- Efficient DL reasoning with LP
  [Hustadt et al., 2007, Lukácsy and Szeredi, 2009]

Integration between LP and DLs I

Exploiting the power of combination

- Combination is more than the sum of the parts
- Very expressive FOL languages as an outcome
- Solutions to the semantic mismatch between LP and DLs

Integration between LP and DLs II

Approaches

Non-hybrid Combination within a *homogeneous* semantic framework

- description logic programs [Grosof et al., 2003]
- DL-safe rules [Motik et al., 2005]
- Semantic Web Rule Language (SWRL) [Horrocks and Patel-Schneider, 2004]

Hybrid Combination within a *heterogeneous* semantic framework

- *Focus of this talk!*

Integration between LP and DLs III

DL-safety [Motik et al., 2005]

- Syntactic restriction to the form of rules
- Helps avoiding unwanted interactions between the LP and the DL parts

Integration between LP and DLs IV

Schemes for hybrid integration

Loose coupling The LP part and the DL part are treated as separate and independent components

- *e.g.*, dl-programs [Eiter et al., 2008] rely on safe interface between an ASP engine and a DL reasoner

Tight integration Separation between the two vocabularies but notion of an integrated model which satisfies both parts

- *Focus of this talk!*

Full integration No separation between the two vocabularies

- *e.g.*, MKNF-knowledge bases [Motik and Rosati, 2007]

Tight integration of LP and DLs I

Overview

	CARIN [Levy and Rousset, 1998]	$\mathcal{AL}$-LOG [Donini et al., 1998]	$\mathcal{DL}$+LOG$^\vee$ [Rosati, 2006]
DL language	any DL	$\mathcal{ALC}$	any DL
CL language	Horn clauses	DATALOG clauses	DATALOG$^\vee$ clauses
DL-safety	no	yes	weakly DL-safe
rule head literals	DL/Horn	DATALOG	DL/DATALOG
rule body literals	DL/Horn	$\mathcal{ALC}$/DATALOG (no roles)	DL/DATALOG
semantics	Herbrand models + DL models	idem	stable models + DL models
reasoning	SLD-resolution + tableau calculus	idem	stable model computation + Boolean CQ/UCQ containment
decidability	only for some instantiations	yes	for all instantiations with DLs for which the Boolean CQ/UCQ containment is decidable
implementation	yes, *e.g.*[Goasdoué et al., 2000]	yes, *e.g.*[Ruckhaus et al., 2006]	unknown

Tight integration of LP and DLs II

Example: a $\mathcal{DL}$+LOG$^{\neg\vee}$ KB $\mathcal{K} = (\Sigma, \Pi)$

- The DL component Σ

[$A1$] PERSON $\sqsubseteq \exists$ FATHER$^-$.MALE
[$A2$] MALE $\sqsubseteq$ PERSON
[$A3$] FEMALE $\sqsubseteq$ PERSON
[$A4$] FEMALE $\sqsubseteq \neg$MALE
MALE(Bob)
PERSON(Mary)
PERSON(Paul)
FATHER(John,Paul)

- The DATALOG component Π

[$R1$] boy(X) $\leftarrow$ enrolled(X,c1,ft), PERSON(X), not girl(X)
[$R2$] girl(X) $\leftarrow$ enrolled(X,c2,ft), PERSON(X)
[$R3$] boy(X)$\vee$ girl(X) $\leftarrow$ enrolled(X,c3,ft), PERSON(X)
[$R4$] FEMALE(X) $\leftarrow$ girl(X)
[$R5$] MALE(X) $\leftarrow$ boy(X)
[$R6$] man(X) $\leftarrow$ enrolled(X,c3,pt), FATHER(X,Y)
enrolled(Paul,c1,ft)
enrolled(Mary,c1,ft)
enrolled(Mary,c2,ft)
enrolled(Bob,c3,ft)
enrolled(John,c3,pt)

- Notice that $\mathcal{K} \models_{NM}$ FEMALE(Mary), while $\Sigma \not\models_{FOL}$ FEMALE(Mary).

Learning hybrid rules with ILP I

Overview

	Learning CARIN-$\mathcal{ALN}$ rules [Rouveirol and Ventos, 2000, Kietz, 2003]	Learning $\mathcal{AL}$-LOG rules [Lisi, 2008]	Learning $\mathcal{SHIQ}$+LOG rules [Lisi, 2010]
prior knowledge	CARIN-$\mathcal{ALN}$ KB	$\mathcal{AL}$-LOG KB	$\mathcal{SHIQ}$+LOG KB
ontology language	$\mathcal{ALN}$	$\mathcal{ALC}$	$\mathcal{SHIQ}$
rule language	HCL	DATALOG	DATALOG
hypothesis language	CARIN-$\mathcal{ALN}$ non-recursive rules	$\mathcal{AL}$-LOG non-recursive rules	$\mathcal{SHIQ}$+LOG non-recursive rules
target predicate	Horn predicate	DATALOG predicate	$\mathcal{SHIQ}$/DATALOG predicate
logical setting	interpretations	interpretations/entailment	entailment
scope of induction	prediction	prediction/description	prediction/description
generality order	generalized subsumption	generalized subsumption	generalized subsumption
coverage test	CARIN query answering	$\mathcal{AL}$-LOG query answering	$\mathcal{DL}$+LOG$^\vee$ query answering
ref. operators	n.a.	downward	downward/upward
implementation	unknown	yes, see [Lisi, 2011]	no
application	no	yes, see [Lisi and Malerba, 2004]	no

Learning hybrid rules with ILP II

Applications

- Spatial data mining in $\mathcal{AL}$-LOG [Lisi and Malerba, 2004]
- Semantic Web Mining in $\mathcal{AL}$-LOG [Lisi, 2011]
- Learning $\mathcal{DL}$+LOG$^{\vee}$ rules for database design [Lisi, 2010]
- Learning $\mathcal{DL}$+LOG rules for ontology maintenance [Lisi, 2014]

Learning hybrid rules with ILP III

$\mathcal{AL}$-QuIn: general features [Lisi, 2011]

- Task: multi-level association rule mining
- Method: levelwise search
- BK: hybrid (relational DB + ontology)
- Knowledge representation formalism: $\mathcal{AL}$-log.
- Upgrade from: Warmr.

Learning hybrid rules with ILP IV

$\mathcal{AL}$-QuIn: problem statement

Given

- a relational data set Π
- a taxonomic ontology Σ
- a multi-grained language $\mathcal{L} = \{\mathcal{L}^l\}_{1 \leq l \leq maxG}$
- a set $\{minsup^l\}_{1 \leq l \leq maxG}$ of support thresholds

the problem of *frequent pattern discovery in* Π *at* l *levels of description granularity w.r.t.* Σ, $1 \leq l \leq maxG$, is to find the set $\mathcal{F}$ of all the patterns $P \in \mathcal{L}^l$ frequent in $\mathcal{B} = (\Pi, \Sigma)$, namely P's with support s such that (i) $s \geq minsup^l$ and (ii) all ancestors of P w.r.t. Σ are frequent.

Learning hybrid rules with ILP V

$\mathcal{AL}$-QuIn: discovering frequent onto-relational patterns

$\mathcal{AL}$-**QuIn**($\mathcal{B}$, *maxG*, *maxD*,$\{\mathcal{L}^l, minsup^l\}_{1 \leq l \leq maxG}$)

1. $\mathcal{F} \leftarrow \emptyset$;
2. $Q_t \leftarrow \{\text{generateTrivialPattern}(\mathcal{L})\}$;
3. $l \leftarrow 1$;
4. **while** $l \leq maxG$ **do**
5. $\quad \mathcal{I}^l \leftarrow \emptyset$;
6. $\quad k \leftarrow 1$;
7. $\quad \mathcal{C}^l_k \leftarrow \{Q_t\}$;
8. $\quad \mathcal{F}^l_k \leftarrow \emptyset$;
9. $\quad$ **while** $k \leq maxD$ **and** $\mathcal{C}^l_k \neq \emptyset$ **do**
10. $\quad\quad \mathcal{F}^l_k \leftarrow \text{evaluateCandidates}(\mathcal{B}, \mathcal{C}^l_k, \mathcal{I}^l, minsup^l)$;
11. $\quad\quad \mathcal{F} \leftarrow \mathcal{F} \cup \mathcal{F}^l_k$;
12. $\quad\quad k \leftarrow k + 1$;
13. $\quad\quad \mathcal{C}^l_k \leftarrow \text{generateCandidates}(\mathcal{F}^l_{k-1}, \mathcal{L}^l, \mathcal{I}^l)$;
14. $\quad$ **endwhile**
15. $\quad l \leftarrow l + 1$;
16. **endwhile**

return $\mathcal{F}$

Learning hybrid rules with ILP VI

Example of onto-relational association rule mining

Task: discovery of frequent patterns in $\mathcal{B}_{\texttt{CIA}}$ that describe Middle East countries w.r.t. the religions believed and the languages spoken at three levels of granularity ($maxG = 3$).
$\mathcal{L}_{\texttt{CIA}}$ is the set of $\mathcal{O}$-queries with $C_{ref} = \texttt{MiddleEastCountry}$ that can be generated from the alphabet $A = \{\texttt{believes/2}, \texttt{speaks/2}\}$ of binary DATALOG predicate names, and the alphabets

$\Gamma^1 = \{\texttt{Language}, \texttt{Religion}\}$
$\Gamma^2 = \{\texttt{IndoEuropeanLanguage}, \ldots, \texttt{MonotheisticReligion}, \ldots\}$
$\Gamma^3 = \{\texttt{IndoIranianLanguage}, \ldots, \texttt{MuslimReligion}, \ldots\}$

of $\mathcal{ALC}$ concept names for $1 \leq l \leq 3$, up to $maxD = 5$.

Learning inclusion axioms with ILP I

Inclusion axioms vs. rules

$C \sqsubseteq A_t$

Learning inclusion axioms with ILP II

Concept Learning in DLs: A brief history

- [Cohen and Hirsh, 1992] study the learnability of DLs
- ...
- [Badea and Nienhuys-Cheng, 2000] define the first refinement operator for DLs
- ...
- [Lehmann and Hitzler, 2008] study the properties of refinement operators for DLs and propose a refinement-based method for learning in DLs
  - Set-up of the DL-Learner system
- ...
- [Lisi and Straccia, 2014] present a FOIL-like algorithm for learning fuzzy $\mathcal{EL}(\mathbf{D})$ GCIs

Learning inclusion axioms with ILP III

FOIL-$\mathcal{DL}$: general features

- Task: classification rule mining
- Method: sequential covering
- BK: $\mathcal{DL}$ KB
- Hypothesis description language: fuzzy $\mathcal{EL}(\mathbf{D})$.
- Upgrade from: FOIL.

Learning inclusion axioms with ILP IV

FOIL-$\mathcal{DL}$: problem statement

Given:

- a consistent $\mathcal{DL}$ KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ (the *background theory*);
- an atomic concept A_t (the *target concept*);
- a set $\mathcal{E} = \mathcal{E}^+ \cup \mathcal{E}^-$ of crisp $\mathcal{DL}$ concept assertions labelled as either positive or negative examples for H (the *training set*);
- a set $\mathcal{L}_\mathcal{H}$ of fuzzy $\mathcal{EL}(\mathbf{D})$ GCIs of the form $C \sqsubseteq A_t$ (the *language of hypotheses*) where C is a complex concept

Find: a set $\mathcal{H} \subset \mathcal{L}_\mathcal{H}$ (a *hypothesis*) such that:

Completeness. $\forall e \in \mathcal{E}^+, \mathcal{K} \cup \mathcal{H} \models_+ e$, and

Consistency. $\forall e \in \mathcal{E}^-, \mathcal{K} \cup \mathcal{H} \not\models_+ e$.

Learning inclusion axioms with ILP V

Fuzzy $\mathcal{EL}(\mathbf{D})$ [Straccia, 2005]

- Complex concepts are built according to the following syntactic rules:

$$C \rightarrow \top \mid \bot \mid A \mid C_1 \sqcap C_2 \mid \exists R.C \mid \exists T.\mathbf{d}$$

  where
  - **d** can be one of the membership functions of fuzzy sets
  - $\sqcap$ and $\exists$ are interpreted as truth combination functions
- Concepts are interpreted as fuzzy sets
- Axioms are graded, *i.e.* have a truth degree α (if omitted, $\alpha = 1$)
  - e.g., $\mathcal{I}$ satisfies an axiom $\langle a{:}C, \alpha \rangle$ if $C^{\mathcal{I}}(a^{\mathcal{I}}) \geq \alpha$
- The *best entailment degree* of an axiom τ w.r.t. $\mathcal{K}$ is defined as

$$bed(\mathcal{K}, \tau) = \sup\{\alpha \mid \mathcal{K} \models \langle \tau, \alpha \rangle\} \ . \tag{4}$$

For a crisp axiom τ, we also write $\mathcal{K} \models_+ \tau$ iff $bed(\mathcal{K}, \tau) > 0$.

Learning inclusion axioms with ILP VI

FOIL-$\mathcal{DL}$: Learning sets of GCIs

function LEARN-SETS-OF-AXIOMS($\mathcal{K}$, A_t, $\mathcal{E}^+$, $\mathcal{E}^-$, $\mathcal{L}_\mathcal{H}$): $\mathcal{H}$
begin
1. $\mathcal{H} := \emptyset$; **D** = INITIALISEFUZZYCONCRETEDOMAIN($\mathcal{K}$);
2. **while** $\mathcal{E}^+ \neq \emptyset$ **do**
3. $\quad \phi :=$ LEARN-ONE-AXIOM($\mathcal{K}$, A_t, $\mathcal{E}^+$, $\mathcal{E}^-$, $\mathcal{L}_\mathcal{H}$);
4. $\quad \mathcal{H} := \mathcal{H} \cup \{\phi\}$;
5. $\quad \mathcal{E}^+_\phi := \{e \in \mathcal{E}^+ \mid \mathcal{K} \cup \{\phi\} \models_+ e\}$;
6. $\quad \mathcal{E}^+ := \mathcal{E}^+ \setminus \mathcal{E}^+_\phi$;
7. **endwhile**
8. **return** $\mathcal{H}$

end

Learning inclusion axioms with ILP VII

FOIL-$\mathcal{DL}$: Learning one GCI

function LEARN-ONE-AXIOM($\mathcal{K}$, A_t, $\mathcal{E}^+$, $\mathcal{E}^-$, $\mathcal{L}_\mathcal{H}$): ϕ
begin
1. $C := \top$; 2. $\phi := C \sqsubseteq A_t$; 3. $\mathcal{E}^-_\phi := \mathcal{E}^-$;
4. **while** $cf(\phi) < \theta$ **or** $\mathcal{E}^-_\phi \neq \emptyset$ **do**
5. $\quad C_{best} := C$;
6. $\quad maxgain := 0$;
7. $\quad \Phi := \text{SPECIALIZE}(\phi, \mathcal{L}_\mathcal{H}, \mathcal{K})$
8. $\quad$ **foreach** $\phi' \in \Phi$ **do**
9. $\quad\quad gain := \text{GAIN}(\phi', \phi)$;
10. $\quad\quad$ **if** $gain \geq maxgain$ **then**
11. $\quad\quad\quad maxgain := gain$;
12. $\quad\quad\quad C_{best} := C'$;
13. $\quad\quad$ **endif**
14. $\quad$ **endforeach**
15. $\quad \phi := C_{best} \sqsubseteq A_t$;
16. $\quad \mathcal{E}^-_\phi := \{e \in \mathcal{E}^- \mid \mathcal{K} \cup \{\phi\} \models_+ e\}$;
17. **endwhile**
18. **return** ϕ
end

Learning inclusion axioms with ILP VIII

FOIL-$\mathcal{DL}$: Initializing fuzzy concrete domains

The function INITIALISEFUZZYCONCRETEDOMAIN returns

$$\mathbf{D} = \bigcup_{T \text{ concrete role occurring in } \mathcal{K}} \{VeryLow_T, Low_T, Fair_T, High_T, VeryHigh_T\}$$

where, for each concrete role T occurring in $\mathcal{K}$,

1. determine, by relying on a crisp $\mathcal{DL}$ reasoner, the values $min_T = \min\{v \mid \mathcal{K} \models a{:}\exists T. \leq_v\}$ and $max_T = \max\{v \mid \mathcal{K} \models a{:}\exists T. \geq_v\}$.
2. partition the interval $[min_T, max_T]$ into four uniform subintervals and, for $k = (max_T - min_T)/4$, build the fuzzy concrete domain predicates $VeryLow_T = ls(min_T, min_T + k)$, $Low_T = tri(min_T, min_T + k, min_T + 2k)$, $Fair_T = tri(min_T + k, min_T + 2k, min_T + 3k)$, $High_T = tri(min_T + 2k, min_T + 3k, max_T)$ and $VeryHigh_T = rs(min_T + 3k, max_T)$.

FOIL-$\mathcal{DL}$: Refining fuzzy $\mathcal{EL}(\mathbf{D})$ GCIs

The function SPECIALIZE implements a *downward refinement* operator $\rho_{\mathcal{K}}$ which actually exploits the background theory $\mathcal{K}$ in order to avoid the generation of redundant or useless hypotheses:

$$\text{SPECIALIZE}(\phi, \mathcal{L}_{\mathcal{H}}, \mathcal{K}) = \{\phi' \in \mathcal{L}_{\mathcal{H}} \mid \phi' \in \rho_{\mathcal{K}}(\phi)\} \ . \tag{5}$$

The refinement operator $\rho_{\mathcal{K}}$ acts only upon the left-hand-side of a GCI:

$$\rho_{\mathcal{K}}(\phi) = \rho_{\mathcal{K}}(C \sqsubseteq A_t) = \{C' \sqsubseteq A_t \mid C' \in \rho_{\mathcal{K}}^{\mathcal{C}}(C)\} \tag{6}$$

by either adding a new conjunct or replacing an already existing conjunct with a more specific one.

Learning inclusion axioms with ILP X

FOIL-$\mathcal{DL}$: Evaluating fuzzy $\mathcal{EL}(\mathbf{D})$ GCIs

The function GAIN implements an information-theoretic criterion for selecting the best candidate at each refinement step

$$\text{GAIN}(\phi', \phi) = p * (log_2(cf(\phi')) - log_2(cf(\phi))) \ , \tag{7}$$

$$cf(\phi) = cf(C \sqsubseteq A_t) = \frac{\sum_{a \in \text{Ind}^+_\phi(\mathcal{A})} bed(\mathcal{K}, a{:}C)}{|\text{Ind}_\phi(\mathcal{A})|} \tag{8}$$

where $\text{Ind}^+_\phi(\mathcal{A})$ (resp., $\text{Ind}_\phi(\mathcal{A})$) is the subset of $\text{Ind}(\mathcal{A})$ containing those individuals a involved in $\mathcal{E}^+_\phi$ (resp., $\mathcal{E}^+_\phi \cup \mathcal{E}^-_\phi$) such that $bed(\mathcal{K}, a{:}C) > 0$.

Learning inclusion axioms with ILP XI

FOIL-$\mathcal{DL}$: The implementation

- Variant implemented in the *fuzzyDL-Learner* system[a]
  - Works under OWA and CWA
  - Vagueness dealt with Gödel logic
  - DL reasoner + specialized code
- Optimizations
  - Beam-search instead of greedy search
  - Backtracking
  - Informed application of the refinement operator
- Two GUIs
  - stand-alone Java application
  - tab widget plug-in for the ontology editor Protégé (release 4.2)

[a] `http://straccia.info/software/FuzzyDL-Learner/`

Related work I

Crisp DL learning

- DL-Learner [Lehmann, 2009] is the state-of-the-art system in DL concept learning, which offers a suite of algorithms such as
  - CELOE [Hellmann et al., 2009] for efficient learning of class expressions in OWL DL.
  - ELTL [Lehmann and Haase, 2010] for concept learning in $\mathcal{EL}$
- $\mathcal{DL}$-Foil [Fanizzi et al., 2008] adapts Foil to learn crisp OWL DL equivalence axioms.
- [Chitsaz et al., 2012] combine a refinement operator for $\mathcal{EL}^{++}$ with a reinforcement learning algorithm.

Related work II

Fuzzy DL Learning

- [Konstantopoulos and Charalambidis, 2010] propose an ad-hoc translation of fuzzy Łukasiewicz $\mathcal{ALC}$ DL constructs into LP in order to apply a conventional ILP method for rule learning.
  - Unsound method
- [Iglesias and Lehmann, 2011] propose to interface CELOE with the *fuzzyDL* reasoner
  - Uncomparable method
- [Lisi and Straccia, 2013] present a method for learning fuzzy $\mathcal{EL}$ GCI axioms from fuzzy DL-Lite KBs.
  - Unimplemented method

Conclusions

Summary

- Rules with only unary and binary predicates
- Taxonomies easily represented with RDFS

Open issues

- Scalability
- Language bias

Questions?

Thanks for your attention!

References I

Agrawal, R., Imielinski, T., and Swami, A. (1993).

Mining Association Rules between Sets of Items in Large Databases.
In Buneman, P. and Jajodia, S., editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216. ACM Press.

Baader, F., Calvanese, D., McGuinness, D., Nardi, D., and Patel-Schneider, P., editors (2007).

The Description Logic Handbook: Theory, Implementation and Applications (2nd ed.).
Cambridge University Press.

Badea, L. and Nienhuys-Cheng, S. (2000).

A refinement operator for description logics.
In Cussens, J. and Frisch, A., editors, *Inductive Logic Programming*, volume 1866 of *Lecture Notes in Artificial Intelligence*, pages 40–59. Springer-Verlag.

Borgida, A. (1996).

On the relative expressiveness of description logics and predicate logics.
Artificial Intelligence, 82(1–2):353–367.

Buntine, W. (1988).

Generalized subsumption and its application to induction and redundancy.
Artificial Intelligence, 36(2):149–176.

Calì, A., Gottlob, G., and Lukasiewicz, T. (2009).

Datalog$^{\pm}$: a unified approach to ontologies and integrity constraints.
In Fagin, R., editor, *Database Theory - ICDT 2009, 12th International Conference, St. Petersburg, Russia, March 23-25, 2009, Proceedings*, volume 361 of *ACM International Conference Proceeding Series*, pages 14–30. ACM.

References II

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., and Rosati, R. (2007).

Tractable Reasoning and Efficient Query Answering in Description Logics: The *dl-lite* Family.
Journal of Automated Reasoning, 39(3):385–429.

Chitsaz, M., Wang, K., Blumenstein, M., and Qi, G. (2012).

Concept learning for $\mathcal{EL}^{++}$ by refinement and reinforcement.
In Anthony, P., Ishizuka, M., and Lukose, D., editors, *PRICAI 2012: Trends in Artificial Intelligence - 12th Pacific Rim International Conference on Artificial Intelligence, Kuching, Malaysia, September 3-7, 2012. Proceedings*, Lecture Notes in Artificial Intelligence, pages 15–26. Springer-Verlag.

Cohen, W. W. and Hirsh, H. (1992).

Learnability of description logics.
In Haussler, D., editor, *Proceedings of the Fifth Annual ACM Conference on Computational Learning Theory, COLT 1992, Pittsburgh, PA, USA, July 27-29, 1992*. ACM.

De Raedt, L. (2008).

Logical and Relational Learning.
Springer.

Dehaspe, L. and Toivonen, H. (1999).

Discovery of frequent DATALOG patterns.
Data Mining and Knowledge Discovery, 3:7–36.

Donini, F. M., Lenzerini, M., Nardi, D., and Schaerf, A. (1998).

$\mathcal{AL}$-log: Integrating Datalog and Description Logics.
Journal of Intelligent Information Systems, 10(3):227–252.

References III

Eiter, T., Ianni, G., Lukasiewicz, T., Schindlauer, R., and Tompits, H. (2008).
Combining answer set programming with description logics for the semantic web.
Artificial Intelligence, 172(12-13):1495–1539.

Fanizzi, N., d'Amato, C., and Esposito, F. (2008).
DL-FOIL concept learning in description logics.
In Zelezný, F. and Lavrač, N., editors, *Inductive Logic Programming, 18th International Conference, ILP 2008, Prague, Czech Republic, September 10-12, 2008, Proceedings*, volume 5194 of *Lecture Notes in Computer Science*, pages 107–121. Springer.

Fürnkranz, J., Gamberger, D., and Lavrač, N. (2012).
Foundations of Rule Learning.
Springer.

Goasdoué, F., Lattès, V., and Rousset, M.-C. (2000).
The Use of CARIN Language and Algorithms for Information Integration: The PICSEL System.
International Journal of Cooperative Information Systems, 9(4):383–401.

Grosof, B. N., Horrocks, I., Volz, R., and Decker, S. (2003).
Description logic programs: combining logic programs with description logic.
In *Proceedings of the 12th International World Wide Web Conference*, pages 48–57. ACM.

Halpern, J. Y. (1990).
An Analysis of First-Order Logics of Probability.
Artificial Intelligence, 46(3):311–350.

Hellmann, S., Lehmann, J., and Auer, S. (2009).
Learning of OWL Class Descriptions on Very Large Knowledge Bases.
International Journal on Semantic Web and Information Systems, 5(2):25–48.

References IV

Horrocks, I. and Patel-Schneider, P. F. (2004).

A Proposal for an OWL Rules Language.
In *Proc. of the 13th International World Wide Web Conference*, pages 723–731. ACM.

Horrocks, I., Patel-Schneider, P. F., and van Harmelen, F. (2003).

From $\mathcal{SHIQ}$ and RDF to OWL: The Making of a Web Ontology Language.
Journal of Web Semantics, 1(1):7–26.

Horrocks, I., Sattler, U., and Tobies, S. (2000).

Practical reasoning for very expressive description logics.
Logic Journal of the IGPL, 8(3):239–263.

Hustadt, U., Motik, B., and Sattler, U. (2007).

Reasoning in description logics by a reduction to disjunctive datalog.
Journal of Automated Reasoning, 39(3):351–384.

Iglesias, J. and Lehmann, J. (2011).

Towards integrating fuzzy logic capabilities into an ontology-based inductive logic programming framework.
In *Proc. of the 11th Int. Conf. on Intelligent Systems Design and Applications*. IEEE Press.

Kietz, J.-U. (2003).

Learnability of description logic programs.
In Matwin, S. and Sammut, C., editors, *Inductive Logic Programming, 12th International Conference, ILP 2002, Sydney, Australia, July 9-11, 2002. Revised Papers*, volume 2583 of *Lecture Notes in Computer Science*, pages 117–132. Springer.

Konstantopoulos, S. and Charalambidis, A. (2010).

Formulating description logic learning as an inductive logic programming task.
In *Proc. of the 19th IEEE Int. Conf. on Fuzzy Systems*, pages 1–7. IEEE Press.

References V

Krötzsch, M., Rudolph, S., and Hitzler, P. (2008).

Description logic rules.

In Ghallab, M., Spyropoulos, C. D., Fakotakis, N., and Avouris, N. M., editors, *ECAI 2008*, volume 178 of *Frontiers in Artificial Intelligence and Applications*, pages 80–84. IOS Press.

Krötzsch, M., Rudolph, S., and Hitzler, P. (2013).

Complexities of horn description logics.

ACM Trans. Comput. Log., 14(1):2.

Krötzsch, M., Rudolph, S., and Schmitt, P. H. (2015).

A closer look at the semantic relationship between datalog and description logics.

Semantic Web, 6(1):63–79.

Lehmann, J. (2009).

DL-Learner: Learning Concepts in Description Logics.

Journal of Machine Learning Research, 10:2639–2642.

Lehmann, J. and Haase, C. (2010).

Ideal Downward Refinement in the $\mathcal{EL}$ Description Logic.

In De Raedt, L., editor, *Inductive Logic Programming, 19th International Conference, ILP 2009, Leuven, Belgium, July 02-04, 2009. Revised Papers*, volume 5989 of *Lecture Notes in Computer Science*, pages 73–87. Springer.

Lehmann, J. and Hitzler, P. (2008).

Foundations of Refinement Operators for Description Logics.

In Blockeel, H., Ramon, J., Shavlik, J. W., and Tadepalli, P., editors, *Inductive Logic Programming*, volume 4894 of *Lecture Notes in Artificial Intelligence*, pages 161–174. Springer.

References VI

Levy, A. Y. and Rousset, M.-C. (1998).
Combining Horn rules and description logics in CARIN.
Artificial Intelligence, 104:165–209.

Lisi, F. A. (2008).
Building Rules on Top of Ontologies for the Semantic Web with Inductive Logic Programming.
Theory and Practice of Logic Programming, 8(03):271–300.

Lisi, F. A. (2010).
Inductive Logic Programming in Databases: From Datalog to $\mathcal{DL}$+log.
Theory and Practice of Logic Programming, 10(3):331–359.

Lisi, F. A. (2011).
$\mathcal{AL}$-QuIn: An Onto-Relational Learning System for Semantic Web Mining.
International Journal on Semantic Web and Information Systems, 7(3):1–22.

Lisi, F. A. (2014).
Learning onto-relational rules with inductive logic programming.
In Lehmann, J. and Völker, J., editors, *Perspectives on Ontology Learning*, volume 18 of *Studies on the Semantic Web*, pages 93–111. IOS Press/AKA.

Lisi, F. A. and Malerba, D. (2004).
Inducing Multi-Level Association Rules from Multiple Relations.
Machine Learning, 55:175–210.

Lisi, F. A. and Straccia, U. (2013).
A logic-based computational method for the automated induction of fuzzy ontology axioms.
Fundamenta Informaticae, 124(4):503–519.

References VII

Lisi, F. A. and Straccia, U. (2014).

A FOIL-like Method for Learning under Incompleteness and Vagueness.
In Zaverucha, G., Santos Costa, V., and Paes, A., editors, *Inductive Logic Programming - 23rd International Conference, ILP 2013, Rio de Janeiro, Brazil, August 28-30, 2013, Revised Selected Papers*, volume 8812 of *Lecture Notes in Computer Science*, pages 123–139. Springer.

Lukácsy, G. and Szeredi, P. (2009).

Efficient description logic reasoning in prolog: The dlog system.
Theory and Practice of Logic Programming, 9(3):343–414.

Lukasiewicz, T. (2008).

Expressive probabilistic description logics.
Artificial Intelligence, 172(6-7):852–883.

Mannila, H. and Toivonen, H. (1997).

Levelwise search and borders of theories in knowledge discovery.
Data Mining and Knowledge Discovery, 1(3):241–258.

Mitchell, T. M. (1982).

Generalization as search.
Artificial Intelligence, 18:203–226.

Mitchell, T. M. (1997).

Machine Learning.
McGraw Hill.

Motik, B. and Rosati, R. (2007).

A faithful integration of description logics with logic programming.
In Veloso, M., editor, *IJCAI 2007, Proc. of the 20th Int. Joint Conf. on Artificial Intelligence*, pages 477–482.

References VIII

Motik, B., Sattler, U., and Studer, R. (2005).
Query Answering for OWL-DL with Rules.
Journal on Web Semantics, 3(1):41–60.

Muggleton, S. H. (1990).
Inductive logic programming.
In Arikawa, S., Goto, S., Ohsuga, S., and Yokomori, T., editors, *Proceedings of the 1st Conference on Algorithmic Learning Theory*. Springer/Ohmsma.

Nienhuys-Cheng, S.-H. and de Wolf, R. (1997).
Foundations of Inductive Logic Programming, volume 1228 of *Lecture Notes in Artificial Intelligence*.
Springer.

Plotkin, G. (1970).
A note on inductive generalization.
Machine Intelligence, 5:153–163.

Quinlan, J. R. (1990).
Learning logical definitions from relations.
Machine Learning, 5:239–266.

Reiter, R. (1980).
Equality and domain closure in first order databases.
Journal of ACM, 27:235–249.

Rosati, R. (2006).
$\mathcal{DL}$+log: Tight Integration of Description Logics and Disjunctive Datalog.
In Doherty, P., Mylopoulos, J., and Welty, C. A., editors, *Proc. of Tenth International Conference on Principles of Knowledge Representation and Reasoning*, pages 68–78. AAAI Press.

References IX

Rouveirol, C. and Ventos, V. (2000).

Towards Learning in CARIN-$\mathcal{ALN}$.

In Cussens, J. and Frisch, A. M., editors, *Inductive Logic Programming, 10th International Conference, ILP 2000, London, UK, July 24-27, 2000, Proceedings*, volume 1866 of *Lecture Notes in Artificial Intelligence*, pages 191–208. Springer.

Ruckhaus, E., Kolovski, V., Parsia, B., and Cuenca Grau, B. (2006).

Integrating Datalog with OWL: Exploring the $\mathcal{AL}$-log Approach.

In Etalle, S. and Truszczynski, M., editors, *Logic Programming, 22nd International Conference, ICLP 2006, Seattle, WA, USA, August 17-20, 2006, Proceedings*, volume 4079 of *Lecture Notes in Computer Science*, pages 455–456. Springer.

Schmidt-Schauss, M. and Smolka, G. (1991).

Attributive concept descriptions with complements.

Artificial Intelligence, 48(1):1–26.

Straccia, U. (2005).

Description logics with fuzzy concrete domains.

In *UAI '05, Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence, Edinburgh, Scotland, July 26-29, 2005*, pages 559–567. AUAI Press.

Straccia, U. (2015).

All about fuzzy description logics and applications.

In Faber, W. and Paschke, A., editors, *Reasoning Web. Web Logic Rules - 11th International Summer School 2015, Berlin, Germany, July 31 - August 4, 2015, Tutorial Lectures*, volume 9203 of *Lecture Notes in Computer Science*, pages 1–31. Springer.

Tran, H. D., Stepanova, D., Gad-Elrab, M., Lisi, F. A., and Weikum, G. (2017).

Towards nonmonotonic relational learning from knowledge graphs.

In Cussens, J. and Russo, A., editors, *ILP 2016*, volume 10326 of *Lecture Notes in Computer Science*, pages 1–14. Springer.