ECML 2007 PKDD
WARSAW POLAND

THE 18TH EUROPEAN CONFERENCE ON MACHINE LEARNING
AND
THE 11TH EUROPEAN CONFERENCE ON PRINCIPLES AND PRACTICE
OF KNOWLEDGE DISCOVERY IN DATABASES

THE CHALLENGES OF THE SEMANTIC WEB TO MACHINE LEARNING AND DATA MINING
TUTORIAL NOTES

---

presented by
Francesca A. Lisi

September 17, 2007

Warsaw, Poland

Prepared and presented by:
Francesca A. Lisi
LACAM Group, Dipatimento di Informatica,
Università degli Studi di Bari, Italy

The Challenges of the Semantic Web to Machine Learning and Data Mining

Francesca A. Lisi
lisi@di.uniba.it

LACAM group
Dipartimento di Informatica

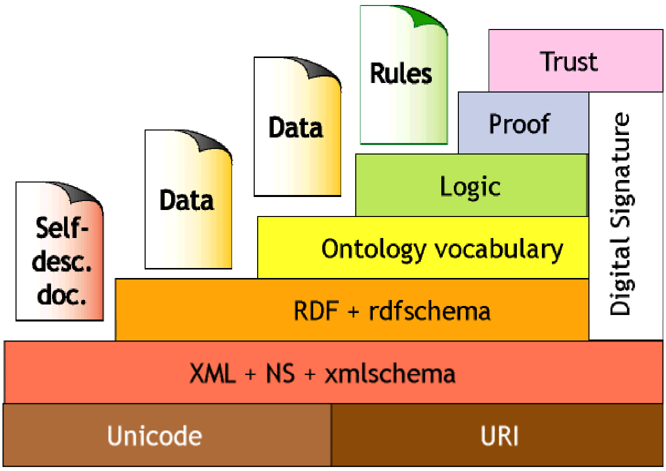
Università degli Studi di Bari
Via Orabona, 4 - 70126 Bari - Italy

The Semantic Web

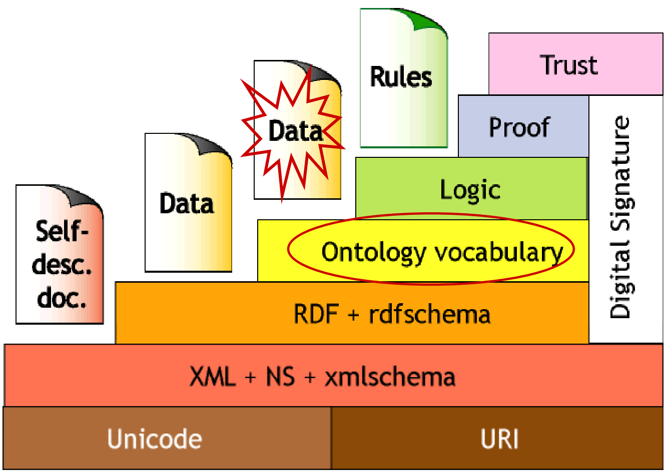
T. Berners-Lee, J. Hendler, and O. Lassila (2001). *The Semantic Web*. Scientific American, May 2001, pp. 34–43.

- Evolving extension of the World Wide Web (WWW) in which WWW content can be expressed not only in natural language, but also in a format that can be read and used by software agents, thus permitting them to find, share and integrate information more easily.
- Vision of the WWW as a universal medium for data, information, and knowledge exchange.

The Semantic Web: layered architecture



The Semantic Web: layer of ontologies



What is an ontology?

T. R. Gruber (1993). *A translation approach to portable ontologies*. Knowledge Acquisition, 5(2): 199-220.

An Ontology is a

formal specification ⇒ Executable

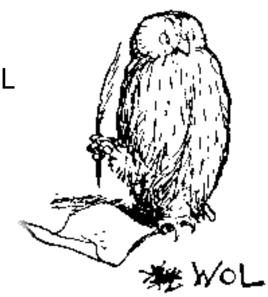
of a shared ⇒ Group of persons

conceptualization ⇒ About concepts

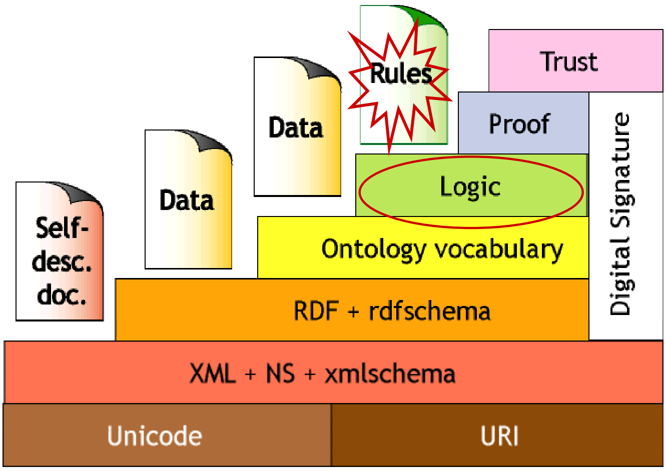
of a domain of interest ⇒ Between application and „unique truth“

OWL (Ontology Web Language)

- W3C **recommendation** (i.e., a standard) for Web ontologies
  - http://www.w3.org/2004/OWL/
- Developed by the W3C WebOnt Working Group
- Mark-up language
  - compatible with RDF/XML exchange format
  - based on earlier languages OIL and DAML+OIL



The Semantic Web: Rules on top of ontologies



SWRL (Semantic Web Rule Language)

- Submitted to W3C for standardization
  - http://www.w3.org/Submission/SWRL/
- Mark-up language
  - compatible with RDF/XML exchange format
  - integration of OWL and RuleML
- W3C RIF (Rule Interchange Format) Working Group

What the Semantic Web can do for ML/DM

1. Lots and lots of tools to describe and exchange data for later use by ML/DM methods in a canonical way!
2. Using ontological structures to improve the ML/DM tasks
3. Provide background knowledge to guide ML/DM systems
   - See PriCKLws@ECML/PKDD-07

What ML/DM can do for the Semantic Web

1. Learning Ontologies (even if not fully automatic)
2. Learning to map between ontologies
3. Deep Annotation: Reconciling databases and ontologies
4. Annotation by Information Extraction
5. Duplicate recognition

Tutorial focus

- ⌘The acquisition of ontologies and rules for the Semantic Web is a very demanding task
- ⌘The logical nature of ontology and rule languages for the Semantic Web should not be neglected when choosing ML/DM methods to be applied
- ⌘Inductive Logic Programming can be a source of solutions to the Knowledge Acquisition bottleneck of the Semantic Web

Tutorial overview

- ⌘Part I: "Logical Foundations of Ontology and Rule Languages for the Semantic Web" (1h 30m)
- ⌘Part II: "Logic-based ML/DM methods for the Semantic Web" (1h 30m)

The Challenges of the Semantic Web to Machine Learning and Data Mining

Part I: "Logical Foundations of Ontology and Rule Languages for the Semantic Web" (1h 30m)

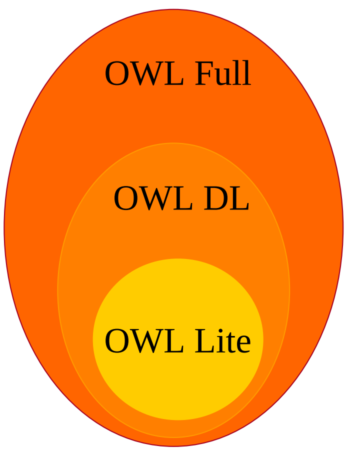
Part I: Overview

- KR systems based on Description Logics
- KR systems combining Description Logics and Horn Clausal Logic (fragments)

Part I: Overview

- *KR systems based on Description Logics*
- KR systems combining Description Logics and Horn Clausal Logic (fragments)

OWL



- OWL provide three levels of expressive power
- All three correspond to fragments of First Order Logic but
- OWL DL is based on a family of fragments with desirable computational properties: Description Logics!

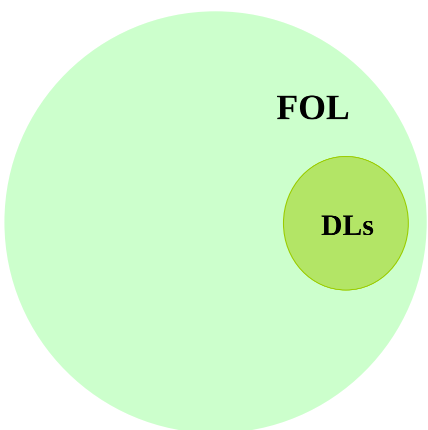
OWL DL

- Why Description Logics?
- It exploits results of 15+ years of KR&R research
  - Well defined (model theoretic) semantics
  - Formal properties well understood (complexity, decidability)
  - Known reasoning algorithms
  - Implemented systems (highly optimised)

Cerebra FaCT++ Racer Pellet

What are Description Logics?

F. Baader et al. (2003). *The Description Logic Handbook: Theory, Implementation, Applications.* Cambridge University Press, Cambridge, UK.



- DLs are decidable variable-free fragments of First Order Logic (FOL)
  - Describe domain in terms of concepts (classes), roles (properties, relationships) and individuals
- DLs provide a family of logic based formalisms for Knowledge Representation and Reasoning (KR&R)
  - Descendants of semantic networks and KL-ONE

DL Basics

- Concepts (unary predicates/formulae with one free variable)
  - E.g., Person, Doctor, HappyParent, (Doctor $\sqcup$ Lawyer)
- Roles (binary predicates/formulae with two free variables)
  - E.g., hasChild, loves, (hasBrother $\circ$ hasDaughter)
- Individuals (constants)
  - E.g., John, Mary, Italy
- Operators (for forming complex concepts and roles from atomic ones) restricted so that:
  - Satisfiability/subsumption is decidable and, *if possible*, of low complexity
  - No need for explicit use of variables
    - Restricted form of $\exists$ and $\forall$
    - Features such as counting can be succinctly expressed

The DL Family

- Smallest propositionally closed DL is $\mathcal{ALC}$ (Schmidt-Schauss and Smolka, 1991)
- $\mathcal{S}$ often used for $\mathcal{ALC}$ extended with transitive roles (R_+)
- Additional letters indicate other extensions, e.g.:
  - $\mathcal{H}$ for role hierarchy (e.g., hasDaughter $\sqsubseteq$ hasChild)
  - $\mathcal{O}$ for nominals/singleton classes (e.g., {Italy})
  - $\mathcal{I}$ for inverse roles (e.g., isChildOf $\equiv$ hasChild^{-})
  - $\mathcal{N}$ for number restrictions (e.g., $\geqslant$2hasChild, $\leqslant$3hasChild)
  - $\mathcal{Q}$ for qualified number restrictions (e.g., $\geqslant$2hasChild.Doctor)
  - $\mathcal{F}$ for functional number restrictions (e.g., $\leqslant$1hasMother)
- $\mathcal{S}$ + role hierarchy ($\mathcal{H}$) + inverse ($\mathcal{I}$) + QNR ($\mathcal{Q}$) = $\mathcal{SHIQ}$
- $\mathcal{SHIQ}$is the basis for OWL
  - OWL DL $\approx$ $\mathcal{SHIQ}$extended with nominals (i.e., $\mathcal{SHOIQ}$)
  - OWL Lite $\approx$ $\mathcal{SHIQ}$with only functional restrictions (i.e., $\mathcal{SHIF}$)

$\mathcal{ALC}$ syntax

atomic concept	A	Human
atomic role	R	likes
conjunction	C⊓D	Human⊓Male
disjunction	C⊔D	Nice⊔Rich
negation	¬C	¬Meat
existential restriction	∃R.C	∃hasChild.Human
value restriction	∀R.C	∀hasChild.Nice

- E.g., person all of whose children are either Doctors or have a child who is a Doctor:

Person ⊓ ∀hasChild.(Doctor ⊔∃hasChild.Doctor)

DL Semantics

Semantics given by standard FOL model theory:

Interpretation function $\mathcal{I}$ Interpretation domain $\Delta^{\mathcal{I}}$

Individuals $i^{\mathcal{I}} \in \Delta^{\mathcal{I}}$

John

Mary

Concepts $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$

Lawyer

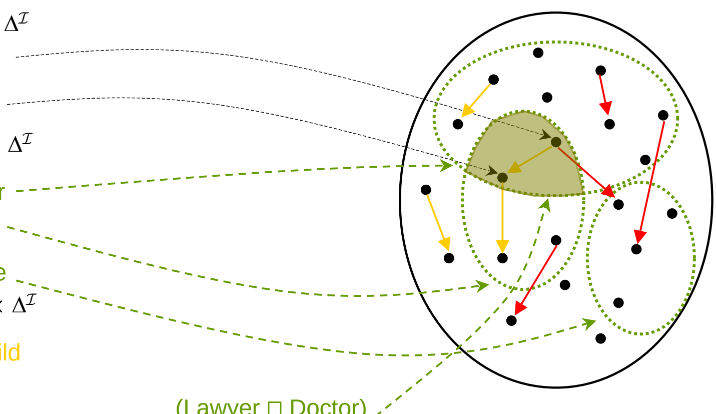
Doctor

Vehicle

Roles $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$

hasChild

owns



(Lawyer ⊓ Doctor)

DL Semantics:
Unique Names Assumption (UNA)

R. Reiter (1980). A logic for default reasoning. *Artificial Intelligence*, 13:81-132.

- $a^{\mathcal{I}} \neq b^{\mathcal{I}}$ if $a \neq b$

$\mathcal{ALC}$ semantics

atomic concept	A	$A^I \subseteq \Delta^I$
atomic role	R	$R^I \subseteq \Delta^I \times \Delta^I$
conjunction	$C \sqcap D$	$C^I \cap D^I$
disjunction	$C \sqcup D$	$C^I \cup D^I$
negation	$\neg C$	$\Delta^I \setminus C^I$
existential restriction	$\exists R.C$	$\{x \mid \exists y.\langle x,y\rangle \in R^I \wedge y \in C^I\}$
value restriction	$\forall R.C$	$\{x \mid \forall y.\langle x,y\rangle \in R^I \Rightarrow y \in C^I\}$

DL Deduction Rules

Tableau calculus

- Applies rules that correspond to DL constructors
  - E.g., John:(Person $\sqcap$ Doctor) $\rightarrow_{\sqcap}$ John:Person and John:Doctor
- 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

$\mathcal{ALC}$ Deduction Rules

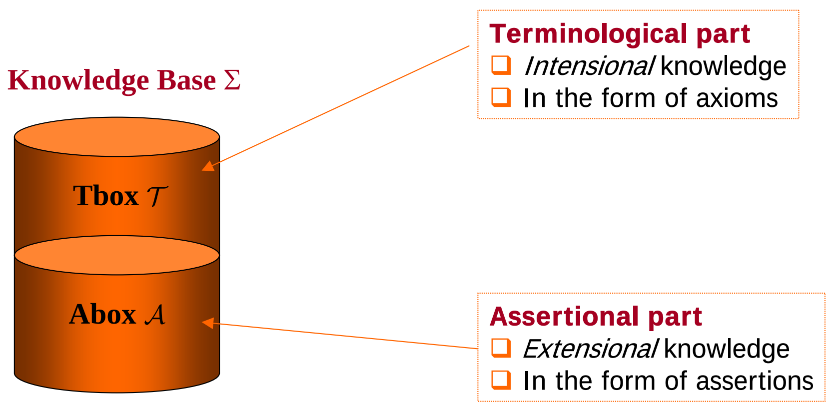
An algorithm based on **tableau calculus** for $\mathcal{ALC}$

- Tries to build a (tree) model $\mathcal{I}$ for input concept C
- Breaks down C syntactically, inferring constraints on elements in $\mathcal{I}$
- Applies inference rules corresponding to $\mathcal{ALC}$ constructors (e.g. $\rightarrow_{\exists}$)
- Works non-deterministically in PSpace
- Stops when a clash, i.e. a contradiction, occurs (C is inconsistent) or no other rule can be applied (C is consistent)

Mapping DLs to FOL

- **Most DLs are decidable fragments of FOL**
  - $\mathcal{ALC}$ is a fragment of FOL with two variables (L2)
- For mapping $\mathcal{ALC}$ to FOL introduce:
  - a unary predicate A for a concept name A
  - a binary relation R for a role name R
- Translate complex concepts C, D as follows:
  - $t_x(A)=A(x)$ $\quad$ $t_y(A)=A(x)$
  - $t_x(C\sqcap D)= t_x(C)\wedge t_x(D)$ $\quad$ $t_y(C\sqcap D)= t_y(C)\wedge t_y(D)$
  - $t_x(C\sqcup D)= t_x(C)\vee t_x(D)$ $\quad$ $t_y(C\sqcup D)= t_y(C)\vee t_y(D)$
  - $t_x(\exists R.C)= \exists y.R(x,y)\wedge t_y(C)$ $\quad$ $t_y(\exists R.C)= \exists y.R(x,y)\wedge t_x(C)$
  - $t_x(\forall R.C)= \forall y.R(x,y)\Rightarrow t_y(C)$ $\quad$ $t_y(\forall R.C)= \forall y.R(x,y)\Rightarrow t_x(C)$

DL Knowledge Bases



$\mathcal{ALC}$ Knowledge Bases: syntax

Tbox

- *equality axioms*
  - $A \equiv C$
  - Father $\equiv$ Man$\sqcap\exists$hasChild.Human
- *inclusion axioms*
  - $C \sqsubseteq D$
  - $\exists$favourite.Brewery $\sqsubseteq$ $\exists$drinks.Beer

ABox

- *concept assertions*
  - a:C
  - john:Father
- *role assertions*
  - <a,b>:R
  - < john,bill>: has-child

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!

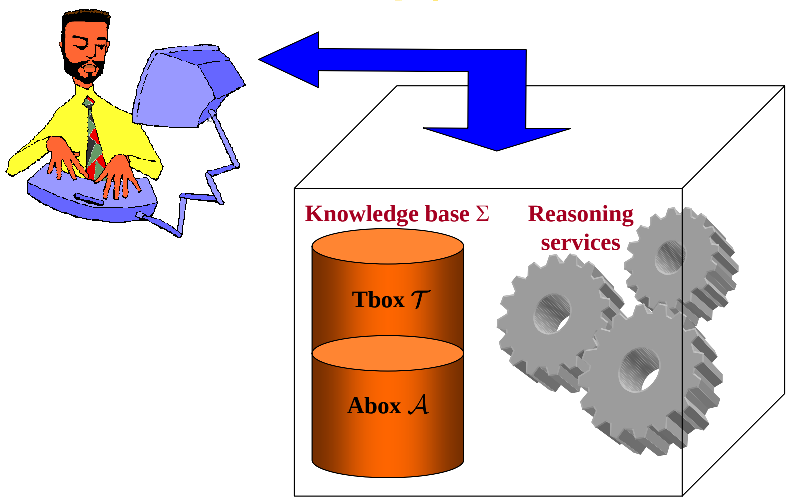
$\mathcal{ALC}$ Knowledge Bases: semantics

An interpretation $\mathcal{I}_O = (\Delta^{\mathcal{I}}, .^{\mathcal{I}})$ satisfies

- an equality axiom $A \equiv C$ iff $A^{\mathcal{I}} \equiv C^{\mathcal{I}}$
- an inclusion axiom $C \sqsubseteq D$ iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- a Tbox $\mathcal{T}$ iff $\mathcal{I}$ satisfies all axioms in $\mathcal{T}$

- a concept assertion $a{:}C$ iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$
- a role assertion $\langle a,b \rangle{:}R$ iff $\langle a^{\mathcal{I}},b^{\mathcal{I}} \rangle \in R^{\mathcal{I}}$
- a ABox $\mathcal{A}$ iff $\mathcal{I}$ satisfies all assertions in $\mathcal{A}$

DL-based KR&R systems



DL-based KR&R systems: standard reasoning tasks

Subsumption

- .. of concepts C and D ($C \sqsubseteq D$)
  - Is $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ in all interpretations $\mathcal{I}$?
- .. of concepts C and D w.r.t. a TBox $\mathcal{T}$ ($C \sqsubseteq_{\mathcal{T}} D$)
  - Is $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ in all models $\mathcal{I}$ of $\mathcal{T}$?

Consistency

- .. of a concept C w.r.t. a TBox $\mathcal{T}$
  - Is there a model $\mathcal{I}$ of $\mathcal{T}$ with $C^{\mathcal{I}} \neq \emptyset$?
- .. of a ABox $\mathcal{A}$
  - Is there a model $\mathcal{I}$ of $\mathcal{A}$?
- .. of a KB ($\mathcal{T}$, $\mathcal{A}$)
  - Is there a model $\mathcal{I}$ of both $\mathcal{T}$ and $\mathcal{A}$?

DL-based KR&R systems: standard reasoning tasks (2)

- Subsumption and consistency are closely related
  - $C \sqsubseteq_{\mathcal{T}} D$ iff $C \sqcap \neg D$ is inconsistent w.r.t. $\mathcal{T}$
  - C is consistent w.r.t. $\mathcal{T}$ iff not $C \sqsubseteq_{\mathcal{T}} A \sqcap \neg A$
- Algorithms for checking consistency w.r.t TBoxes suffice
  - Based on tableau calculus
  - Decidability is important
  - Complexity between P and ExpTime

Instance check

- .. of an individual a and a concept C w.r.t. a KB Σ
  - Is a:C derivable from Σ? Or equivalently,
  - Is $\Sigma \cup \{a{:}\neg C\}$ consistent?

$\mathcal{ALC}$-based KR&R systems: example of instance check

- Σ=DairyProduct$\sqsubseteq$Product, product11:DairyProduct, etc.
  - Is product11:Product derivable from Σ?

    Or equivalently
  - Is $\Sigma \cup \{$ product11:$\neg$ Product$\}$ consistent?

$$\underbrace{\text{product11}:\neg\text{Product} \qquad \text{DairyProduct}\sqsubseteq\text{Product}}_{\rightarrow_{\sqsubseteq}}$$

product11:$\neg$DairyProduct$\sqcup$Product

$\rightarrow_{\sqcup}$

$$\underbrace{\text{product11}:\neg\text{DairyProduct} \qquad \text{product11:DairyProduct}}_{\rightarrow_{\perp}}$$

product11:$\perp$

DL-based KR&R systems: non-standard reasoning tasks

Most Specific Concept (MSC)

Nebel, B. (1990). *Reasoning and Revision in Hybrid Representation Systems*. New York: Springer.

- Intuitively, the MSC of individuals in an ABox is a concept description that represents all the properties of the individuals including the concept assertions they occur in and their relationship to other individuals
- The existence of MSC is not guaranteed for all DLs
  - Approximation of MSC is possible!
- However, if the MSC exists, it is uniquely determined up to equivalence

DL-based KR&R systems: non-standard reasoning tasks (2)

Least Common Subsumer (LCS)

W.W. Cohen, A. Borgida, & H. Hirsh (1992). *Computing Least Common Subsumers in Description Logics*. Proc. of the Tenth National Conf. on Artificial Intelligence (AAAI92), pages 754-760. AAAI Press/MIT Press.

- The LCS of a given sequence of concept descriptions is
  - *Intuitively*, a concept description that represents the properties that all the elements of the sequence have in common
  - *More formally*, the MSC description that subsumes the given concept descriptions
- The existence of the LCS for a given sequence of concept descriptions is not guaranteed but ..
- .. if an LCS exists, then it is uniquely determined up to equivalence

Back to OWL DL: DL syntax

Constructor	DL Syntax	Example	FOL Syntax
intersectionOf	$C_1 \sqcap \ldots \sqcap C_n$	Human $\sqcap$ Male	$C_1(x) \wedge \ldots \wedge C_n(x)$
unionOf	$C_1 \sqcup \ldots \sqcup C_n$	Doctor $\sqcup$ Lawyer	$C_1(x) \vee \ldots \vee C_n(x)$
complementOf	$\neg C$	$\neg$Male	$\neg C(x)$
oneOf	$\{x_1\} \sqcup \ldots \sqcup \{x_n\}$	{john} $\sqcup$ {mary}	$x = x_1 \vee \ldots \vee x = x_n$
allValuesFrom	$\forall P.C$	$\forall$hasChild.Doctor	$\forall y.P(x,y) \rightarrow C(y)$
someValuesFrom	$\exists P.C$	$\exists$hasChild.Lawyer	$\exists y.P(x,y) \wedge C(y)$
maxCardinality	$\leqslant nP$	$\leqslant$1hasChild	$\exists^{\leqslant n} y.P(x,y)$
minCardinality	$\geqslant nP$	$\geqslant$2hasChild	$\exists^{\geqslant n} y.P(x,y)$

- C is a concept (class); P is a role (property); x is an individual name
- XMLS datatypes as well as classes in $\forall$P.C and $\exists$P.C
  - Restricted form of DL concrete domains

Back to OWL DL: DL syntax (2)

OWL Syntax	DL Syntax	Example
subClassOf	$C_1 \sqsubseteq C_2$	Human $\sqsubseteq$ Animal $\sqcap$ Biped
equivalentClass	$C_1 \equiv C_2$	Man $\equiv$ Human $\sqcap$ Male
subPropertyOf	$P_1 \sqsubseteq P_2$	hasDaughter $\sqsubseteq$ hasChild
equivalentProperty	$P_1 \equiv P_2$	cost $\equiv$ price
transitiveProperty	$P^+ \sqsubseteq P$	ancestor$^+$ $\sqsubseteq$ ancestor

OWL Syntax	DL Syntax	Example
type	$a : C$	John : Happy-Father
property	$\langle a, b \rangle : R$	$\langle$John, Mary$\rangle$: has-child

- OWL ontology equivalent to DL KB (Tbox + Abox)

Back to OWL DL: an example

- Dairy products are products

```
<owl:Class rdf:ID=“DairyProduct”>
  <rdfs:subClassOf rdf:about="#Product"/>
   </rdfs:subClassOf>
</owl:Class>
```

- European customers are customers living in European countries

```
<owl:Class rdf:ID=“EuropeanCustomer”>
 <owl:equivalentClass/>
   <owl:intersectionOf rdf:parseType=" collection">
     <owl:Class rdf:about="#Customer"/>
     <owl:restriction/>
       <owl:onProperty rdfResource=“#livesIn”/>
       <owl:allValuesFrom rdf:resource="#EuropeanCountry"/>
       </owl:allValuesFrom>
     </owl:restriction>
   </owl:intersectionOf>
 </owl:equivalentClass>
</owl:Class>
```

Description Logics: Bibliography (only the essential)

- F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, P. F. Patel-Schneider (2003). *The Description Logic Handbook: Theory, Implementation, Applications*. Cambridge University Press, Cambridge, UK.
- R. Kusters (2001). *Non-Standard Inferences in Description Logics*. Volume 2100 of Lecture Notes in Artificial Intelligence. Springer-Verlag.
- M. Schmidt-Schauß & G. Smolka (1991). *Attributive concept descriptions with complements*. Artificial Intelligence, 48 (1): 1-26.

- On-line material: http://dl.kr.org/courses.html

- C. Peltason (1991). The BACK system—an overview. *SIGART Bull.* 2, 3 (Jun. 1991), 114-119.
- CLASSIC: http://www.bell-labs.com/project/classic/

Part I: Overview

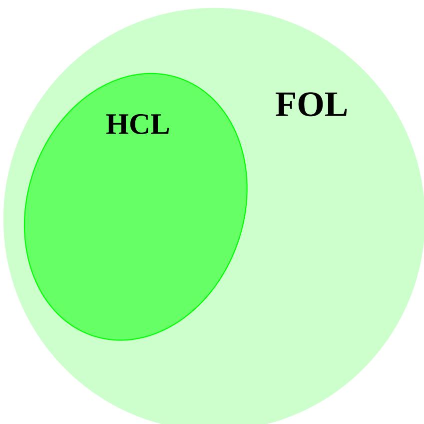
- KR systems based on Description Logics
- *KR systems combining Description Logics and Horn Clausal Logic (fragments)*

RuleML (Rule Markup Language)

- Developed to express both forward (bottom-up) and backward (top-down) rules in XML for deduction, rewriting, and further inferential-transformational tasks.
- Based on Datalog (function-free fragment of Horn Clausal Logic)
- Defined by the Rule Markup Initiative (an open network of individuals and groups from both industry and academia)
  - http://www.ruleml.org/

RuleML
Realize your Knowledge

What is Horn Clausal Logic?



- Horn clausal logic (HCL) is the FOL fragment that contains universally quantified disjunctions of literals with at most one positive literal
- It is at the basis of Logic Programming and Deductive Databases

HCL syntax

- Clausal language $\mathcal{L}$ = the set of constant, variable, functor and predicate symbols

- Term: Constant / Variable / Function applied to a term
- Atom: Predicate applied to n terms
- Literal: (negated) atom

- Horn Clause allows the two following equivalent notations
  - $\forall X\ \forall Y (p(X, Y) \vee \neg\, q(X, a) \vee \neg\, r(Y,f(a)))$
  - $p(X, Y) \leftarrow q(X, a), r(Y,f(a))$

- Definite clause (rule): only one literal in the head
- Unit clause (fact): rule without head

HCL Semantics

Herbrand model theory

- Herbrand universe U_H = the set of all ground terms that can be formed out from the constants and function symbols in $\mathcal{L}$

- Herbrand base B_H = the set of all ground atoms that can be formed out from terms in U_H and predicates in $\mathcal{L}$

- Herbrand interpretation I_H = subset of B_H containing all atoms that are true in I_H

HCL Deduction Rules

SLD-resolution

2 opposite literals (up to a substitution) : $l_i\theta_1 = \neg k_j\theta_2$

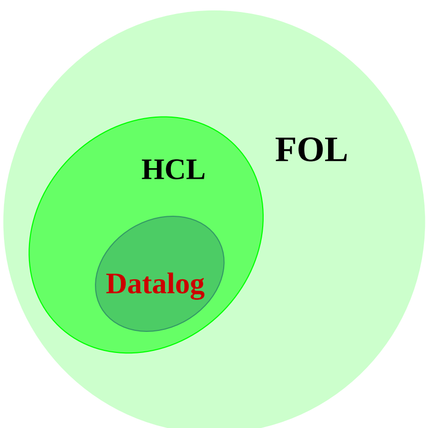
$$\frac{l_1 \vee \ldots \vee l_i \vee \ldots \vee l_n \qquad\qquad k_1 \vee \ldots \vee k_j \vee \ldots \vee k_m}{(l_1 \vee l_2 \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_n \vee k_1 \vee k_{j-1} \vee k_{j+1} \ldots \vee k_m)\ \theta_1\theta_2}$$

e.g., p(X) :- q(X) and q(X) :- r(X,Y) yield p(X) :- r(X,Y)
p(X) :- q(X) and q(a) yield p(a).

- complete by refutation!

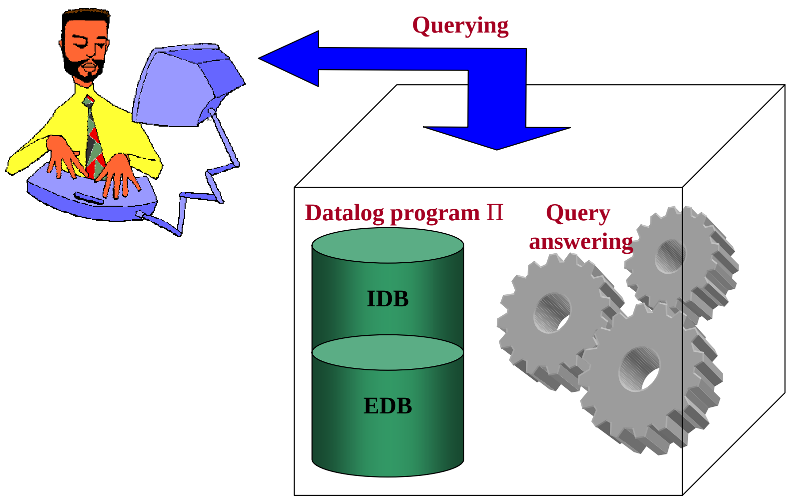
Datalog

S. Ceri, G. Gottlob, & L. Tanca (1990). *Logic Programming and Databases*. Springer.



- It is a function-free fragment of HCL (more precisely of definite clauses)
- It is used as logical language for relational databases
- Query answering by SLD-refutation

Deductive databases



Closed World Assumption (CWA)

- The information in a database is generally considered to be complete (*closed world*)
- A database instance represents exactly one interpretation, namely the one where classes and relations in the schema are interpreted by the objects and the tuples in the instance
- Negation As Failure: what is unknown is false

Datalog: example of query answering

⌘ Π= item(OrderID, ProductID) $\leftarrow$ orderDetail(OrderID, ProductID,_,_,_)
orderDetail(order10248, product11, '£14',12,0.00)
Etc.

- Is item(order10248, product11) derivable from Π?
- Is $\Pi \cup \{\neg$ item(order10248, product11)$\}$ consistent?

$\leftarrow$ item(order10248, product11)   item(OrderID, ProductID) $\leftarrow$ orderDetail(OrderID, ProductID,_,_,_)

{OrderID/order10248, ProductID/ product11 }

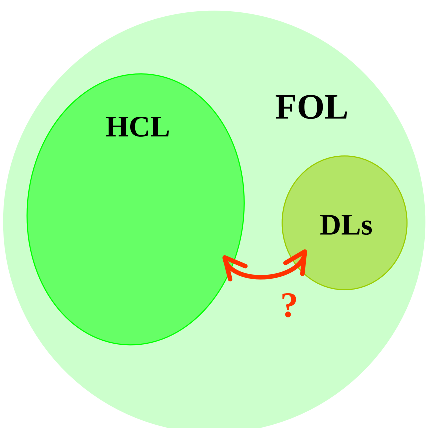
$\leftarrow$ orderDetail(order10248, product11,_,_,_)   orderDetail(order10248, product11, '£14',12,0.00)

{}

$\leftarrow$

DLs and HCL

A. Borgida (1996). On the relative expressiveness of Description Logics and Predicate Logics. *Artificial Intelligence*, 82: 353-367.



⌘ HCL and DLs can not be compared wrt expressive power
  - No relations of arbitrary arity or arbitrary joins between relations in DLs
  - No exist. quant. in HCL

⌘ Can they be combined?

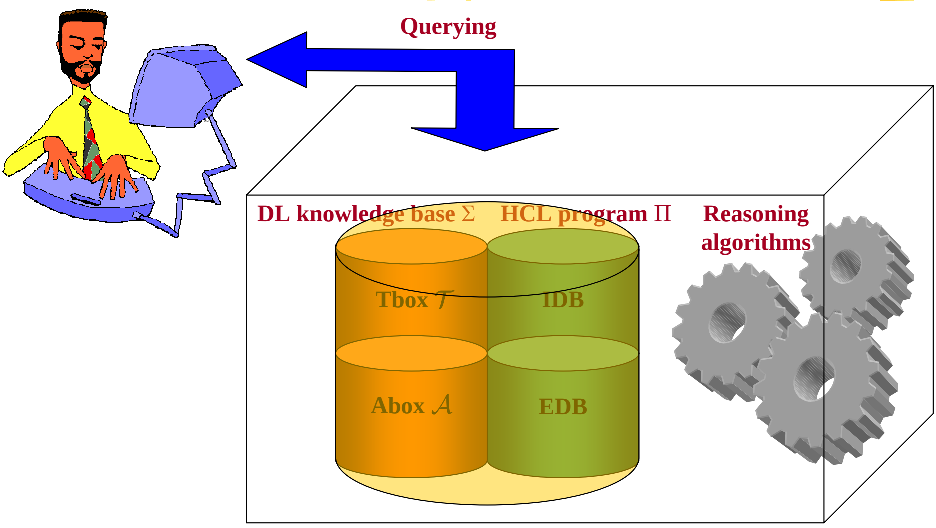
Hybrid DL-HCL KR&R Systems

Levy & M.-C. Rousset (1998). Combining Horn rules and Description Logics in CARIN. *Artificial Intelligence*, 104: 165-209.
F. Donini et al. (1998). $\mathcal{AL}$-log: Integrating Datalog and Description Logics. J. of Intelligent Systems, 10(3):227-252.

- It allows more expressive and deductive power
  - CARIN is a family of powerful hybrid languages
  - $\mathcal{AL}$-log is less powerful than CARIN

- It can easily lead to undecidability if unrestricted
  - Some CARIN languages are decidable
  - $\mathcal{AL}$-log is decidable

Hybrid DL-HCL KR&R systems



$\mathcal{AL}$-log syntax

$\mathcal{B}$ = < Σ , Π >

$\mathcal{ALC}$ knowledge base

constrained Datalog program

constrained Datalog clauses

$$\alpha_0 \leftarrow \alpha_1, \ldots, \alpha_m \ \&\ \gamma_1, \ldots, \gamma_n$$

where α_i are Datalog literals and γ_j are constraints ($\mathcal{ALC}$ concepts from Σ used as "typing constraints" for variables)

- item(OrderID, ProductID) ← orderDetail(OrderID, ProductID,_,_,_) & OrderID:Order, ProductID:Product
- Safeness conditions:
  - Only positive Datalog literals in the body
  - Only one Datalog literal in the head
  - Constraints must refer to variables occurring in the Datalog part
  - Variables in the Datalog part can be constrained

$\mathcal{AL}$-log semantics

$\mathcal{J}$=($\mathcal{I}_O$, $\mathcal{I}_H$)

interpretation for **Σ**

Herbrand interpretation for **Π_D**

- $\mathcal{J}$ satisfies $\mathcal{B}$ iff
  - it satisfies Σ, and
  - for each clause $\alpha_0 \leftarrow \alpha_1, \ldots, \alpha_m \ \&\ \gamma_1, \ldots, \gamma_n$, for each of its ground instances $\alpha'_0 \leftarrow \alpha'_1, \ldots, \alpha'_m \ \&\ \gamma'_1, \ldots, \gamma'_n$, either there exists one γ'_i, $1 \leq i \leq n$, that is not satisfied by $\mathcal{J}$ or $\alpha'_0 \leftarrow \alpha'_1, \ldots, \alpha'_m$ is satisfied by $\mathcal{J}$
- OWA of $\mathcal{ALC}$ and CWA of Datalog do not interfere (safeness)
- UNA holds for $\mathcal{ALC}$ and *ground* Datalog

$\mathcal{AL}$-log reasoning

Query answering

- Atomic queries (only Datalog)
- Constrained SLD-resolution= SLD-resolution (Datalog part) + tableau calculus ($\mathcal{ALC}$ part)
  - decidable
  - Sound and complete by refutation
- Queries are answered by constrained SLD-refutation
  - For each ground instance Q' of the query Q,
  - collect the set of all constrained SLD-derivations $d_1, d_2, .., d_m$ of bounded length (with $d_i = Q^i_0..Q^i_{n_i}$) for Q' in Σ
  - Then check whether $\Sigma \models disj(Q^1_{n_1}, .., Q^m_{n_m})$

$\mathcal{AL}$-log reasoning: example of query answering

← item(order10248, product11)

item(OrderID, ProductID) ← orderDetail(OrderID, ProductID,_,_,_) & OrderID:Order, ProductID:Product

{OrderID/order10248, ProductID/product11}

← orderDetail(order10248, Y,_,_,_) & order10248:Order, Y:Product

orderDetail(order10248, product11, '£14',12,0.00)

{Y/product11 }

← & order10248:Order, product11:Product

Assuming that this is the only SLD-derivation for the query, the existential entailment problem boils down to prove that

$\Sigma \cup$ { order10248:¬Order, product11:¬Product }

is unsatisfiable!

CARIN syntax and semantics

- Σ is based on any DL (but good results for $\mathcal{ALCNR}$)
- Π contain Horn rules, i.e. definite clauses, where DL literals:
  - can be built from either concept or role predicates
  - are allowed in rule heads
- The semantics naturally follows as in $\mathcal{AL}$-log

CARIN reasoning

Query answering

- Atomic queries (built from either concept, role or ordinary predicates)
- Constrained SLD-resolution= SLD-resolution (HCL part) + tableau calculus (DL part)
  - complete by refutation for non-recursive CARIN-$\mathcal{ALCNR}$
  - Decidable for the non-recursive case
  - Undecidable for the recursive case, unless weaken the DL part or impose rules to be role-safe

Back to SWRL

- SWRL is undecidable!
- Several decidable alternatives to SWRL recently proposed:
  - DL-safe rules (Motik et al., 2005)
  - r-hybrid KBs (Rosati, 2005)
  - $\mathcal{DL}$+log (Rosati, 2006)
  - hybrid MKNF KBs (Motik & Rosati, 2007)

Back to SWRL: an example

item(OrderID, ProductID) ← orderDetail(OrderID, ProductID,_,_,_)
& OrderID:Order, ProductID:Product

```
<ruleml:imp>
  <ruleml:_body>
  <swrlx:classAtom>
      <owlx:Class owlx:name="&Order" /> <ruleml:var> OrderID </ruleml:var>
  </swrlx:classAtom>
  <swrlx:classAtom>
      <owlx:Class owlx:name ="&Product" /> <ruleml:var> ProductID </ruleml:var>
  </swrlx:classAtom>
  <swrlx:individualPropertyAtom  swrlx:property="&orderDetail">
      <ruleml:var> OrderID </ruleml:var> <ruleml:var> ProductID </ruleml:var>.. <ruleml:var> .. </ruleml:var>
  </swrlx:individualPropertyAtom>
  </ruleml:_body>
  <ruleml:_head>
  <swrlx:individualPropertyAtom  swrlx:property="&item">
       <ruleml:var> OrderID </ruleml:var> <ruleml:var> ProductID </ruleml:var>
    </swrlx:individualPropertyAtom>
  </ruleml:_head>
</ruleml:imp>
```

Hybrid DL-HCL KR&R Systems: Bibliography

- A. Borgida (1996). *On the relative expressiveness of Description Logics and Predicate Logics*. Artificial Intelligence, 82: 353-367.
- F. Donini et al. (1998). *AL-log: Integrating Datalog and Description Logics*. J. of Intelligent Systems, 10(3):227-252.
- T. Eiter, T. Lukasiewicz, R. Schindlauer, H. Tompits (2004). *Combining Answer Set Programming with Description Logics for the Semantic Web*. KR 2004: 141-151
- T. Eiter, G. Ianni, A. Polleres, R. Schindlauer, H. Tompits (2006). *Reasoning with Rules and Ontologies*. Reasoning Web 2006: 93-127
- B.N. Grosof, I. Horrocks, R. Volz, S. Decker (2003). *Description logic programs: combining logic programs with description logic*. WWW 2003: 48-57.
- I. Horrocks, P.F. Patel-Schneider (2004). *A proposal for an OWL rules language*. WWW 2004: 723-731.

Hybrid DL-HCL KR&R Systems: Bibliography (2)

- I. Horrocks, P.F. Patel-Schneider, S. Bechhofer, D. Tsarkov (2005). *OWL rules: A proposal and prototype implementation*. J. Web Sem. 3(1): 23-40.
- A. Levy & M.-C. Rousset (1998). *Combining Horn rules and Description Logics in CARIN*. Artificial Intelligence, 104: 165-209.
- B. Motik, I. Horrocks, R. Rosati, & U. Sattler (2006). *Can OWL and Logic Programming Live Together Happily Ever After?* In I.F. Cruz et al. (eds), Proc. of the 5th Int. Semantic Web Conference (ISWC 2006), volume 4273 of LNCS, pages 501–514. Springer.
- B. Motik & R. Rosati (2007). *A Faithful Integration of Description Logics with Logic Programming*. In Proc. of the 20th Int. Joint Conference on Artificial Intelligence (IJCAI 2007), pp. 477–482.
- B. Motik, U. Sattler & R. Studer (2004). *Query Answering for OWL-DL with Rules*. In S. A. McIlraith, D. Plexousakis, & F. van Harmelen (eds), Proc. of the 3rd Int. Semantic Web Conference, volume 3298 of LNCS, pp. 549–563. Springer.

Hybrid DL-HCL KR&R Systems: Bibliography (3)

- R. Rosati (2005a). *On the decidability and complexity of integrating ontologies and rules*. J. Web Sem. 3(1): 61-73.
- R. Rosati (2005b). *Semantic and Computational Advantages of the Safe Integration of Ontologies and Rules*. PPSWR 2005: 50-64
- R. Rosati (2006). *DL+log: Tight Integration of Description Logics and Disjunctive Datalog*. KR 2006: 68-78

The Challenges of the Semantic Web to Machine Learning and Data Mining

Part II: “Acquisition of Ontologies and Rules for the Semantic Web with Inductive Logic Programming” (1h 30m)

Part II: Overview

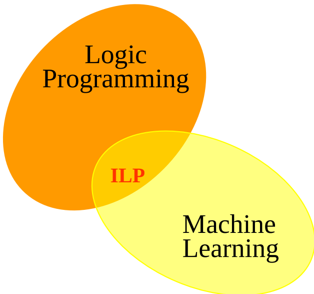
- Introduction to Inductive Logic Programming (ILP)
- ILP and DL representations
- ILP and hybrid DL-HCL representations
- ILP and the Semantic Web: Research directions

Part II: Overview

- *Introduction to ILP*
- ILP and DL representations
- ILP and hybrid DL-HCL representations
- ILP and the Semantic Web: Research directions

Inductive Logic Programming

S.-H. Nienhuys-Cheng & R. de Wolf (1997). *Foundations of Inductive Logic Programming*. LNAI Tutorial Series, Springer.



- *Originally* Induction of rules from examples and background knowledge within the HCL framework
  - Scope of induction: discrimination
  - Class of tasks: prediction

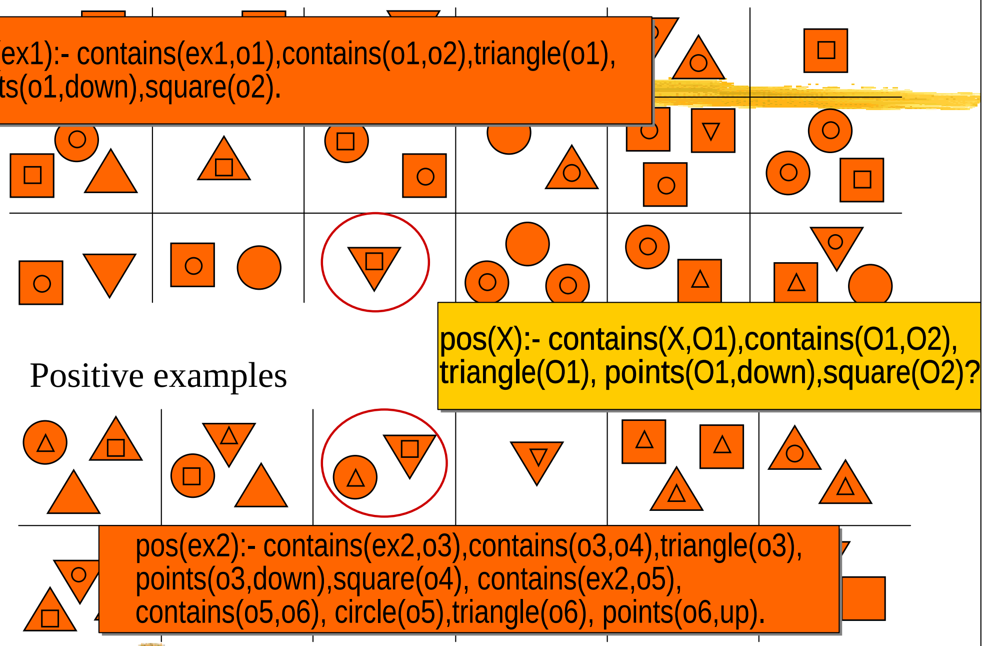
- *Currently* Induction of rules from observations and background knowledge within the framework of FOL (fragments)
  - scope of induction: discrimination/characterization
  - task: prediction/description

ILP Example: "Bongard problems"

- Simplified version of Bongard problems used as benchmarks in ILP
  - Bongard: a Russian scientist studying pattern recognition
  - Bongard problem: Given some pictures, find patterns in them

- E.g. we want to find a set of hypotheses (clausal theory) that is complete and consistent with the following set of (positive and negative) examples
  - Complete=covers all positive examples
  - Consistent=covers no negative example

Negative examples



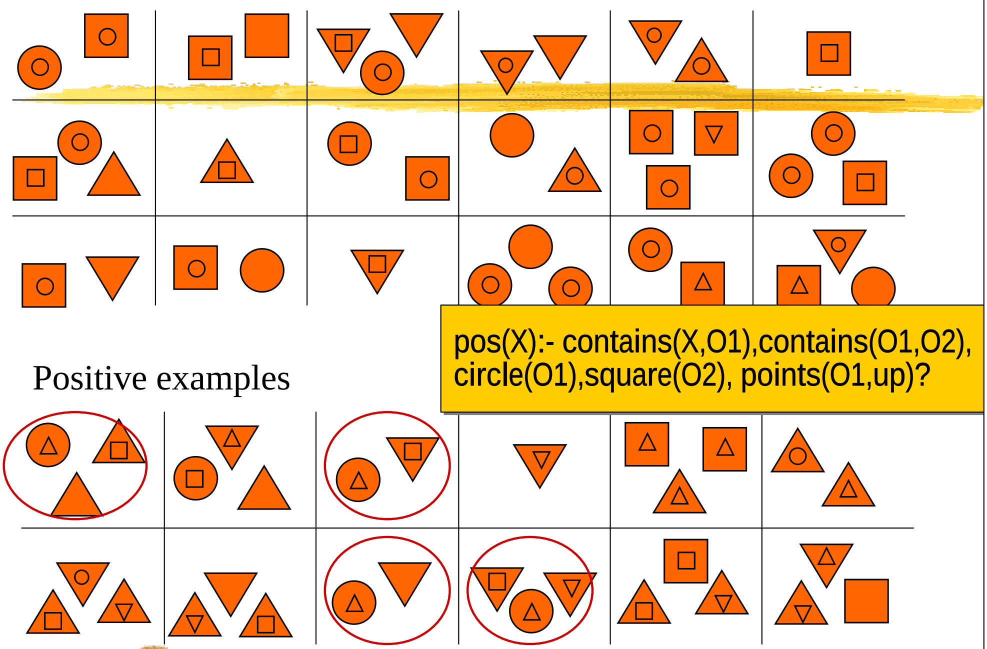
neg(ex1):- contains(ex1,o1),contains(o1,o2),triangle(o1),
points(o1,down),square(o2).

pos(X):- contains(X,O1),contains(O1,O2),
triangle(O1), points(O1,down),square(O2)?

Positive examples

pos(ex2):- contains(ex2,o3),contains(o3,o4),triangle(o3),
points(o3,down),square(o4), contains(ex2,o5),
contains(o5,o6), circle(o5),triangle(o6), points(o6,up).

Negative examples

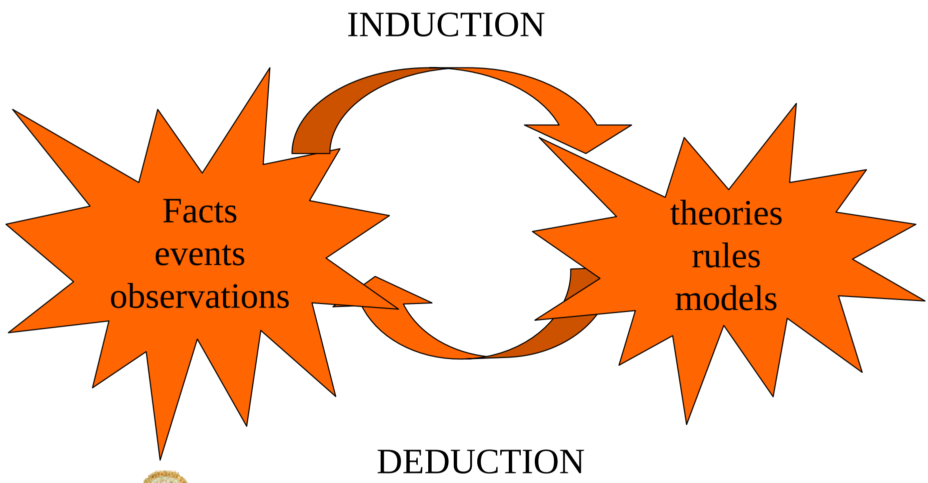


pos(X):- contains(X,O1),contains(O1,O2),
circle(O1),square(O2), points(O1,up)?

Positive examples

Induction in ILP

Induction as inverted deduction



Inverse resolution

S. Muggleton & W. Buntine (1988). *Machine invention of first-order predicates by inverting resolution*. Proc. of the 5th Int. Conf. On Machine Learning, pp. 339-352.

- Resolution implements |- for sets of clauses
- Inverting it allows to generalize a clausal theory
- Inverse resolution is much more difficult than resolution itself
  - different operators defined
  - no unique results

Inverse resolution (2)

- Properties of inverse resolution:
  - + in principle very powerful
  - - gives rise to huge search space
  - - result of inverse resolution not unique
    - e.g., father(j,p):-male(j) and parent(j,p) yields father(j,p):-male(j),parent(j,p) or father(X,Y):-male(X),parent(X,Y) or ...
- Need for a ordered hypothesis space

Induction in ILP (2)

Induction as generalization

- Exploits results obtained in Concept Learning (Mitchell, 1982)
  - Generalization = search through a partially ordered space of hypotheses with the goal of finding the hypothesis that best fits the training examples
- Provides a bunch of techniques for structuring, searching, and boundingthe space of hypotheses when the hypothesis language is defined over HCL

Generality orders: θ-subsumption

G. Plotkin (1970). A note on inductive generalization. *Machine Intelligence*, 5:153-163.
G. Plotkin (1971). A further note on inductive generalization. *Machine Intelligence*, 6:101-124.

- θ-subsumption implements |- for single clauses
- C_1 θ-subsumes C_2 (denoted $C_1 \leq_\theta C_2$) if and only if there exists a variable substitution θ such that $C_1\theta \subseteq C_2$
  - to check this, first write clauses as disjunctions
    - $a,b,c \leftarrow d,e,f \quad \Leftrightarrow \quad a \vee b \vee c \vee \neg d \vee \neg e \vee \neg f$
  - then try to replace variables with constants or other variables
- Most often used in ILP
- Syntactic generality!!

Generality orders: θ-subsumption (2)

Logical properties

- Sound: if c_1 θ-subsumes c_2 then $c_1 \models c_2$
- Incomplete: possibly $c_1 \models c_2$ without c_1 θ-subsuming c_2 (but only for recursive clauses)
  - c_1 : p(f(X)) :- p(X)
  - c_2 : p(f(f(X))) :- p(X)
- Checking θ-subsumption is decidable but NP-complete

Generality orders: θ-subsumption (3)

Algebraic properties

- It is a semi-order relation
  - I.e. transitive and reflexive, not anti-symmetric
- It generates equivalence classes
  - equivalence class: $c_1 \sim c_2$ iff $c_1 \leq_\theta c_2$ and $c_2 \leq_\theta c_1$
  - c_1 and c_2 are then called *syntactic variants*
  - c_1 is *reduced clause* of c_2 iff c_1 contains minimal subset of literals of c_2 that is still equivalent with c_2
  - each equivalence class represented by its reduced clause

Generality orders: θ-subsumption (4)

Algebraic properties (cont.)

- It generates a partial order on those equivalence classes
  - If c_1 and c_2 in different equivalence classes, either $c_1 \leq_\theta c_2$ or $c_2 \leq_\theta c_1$ or neither => anti-symmetry => partial order
- Thus, reduced clauses form a lattice
  - Least/greatest upper/lower bound of two clauses always exists and is unique
  - Infinite chains $c_1 \leq_\theta c_2 \leq_\theta c_3 \leq_\theta \ldots \leq_\theta c$ exist
- Looking for good hypothesis = traversing this lattice

Generality orders: generalized subsumption

W. Buntine (1988). Generalized subsumption and its applications to induction and redundancy. *Artificial Intelligence*, 36(2): 149-176.

- $\mathcal{B}$ background knowledge
- C_1, C_2 two definite clauses
- σ a Skolem substitution for C_2 w.r.t. $\{C_1\} \cup \mathcal{B}$

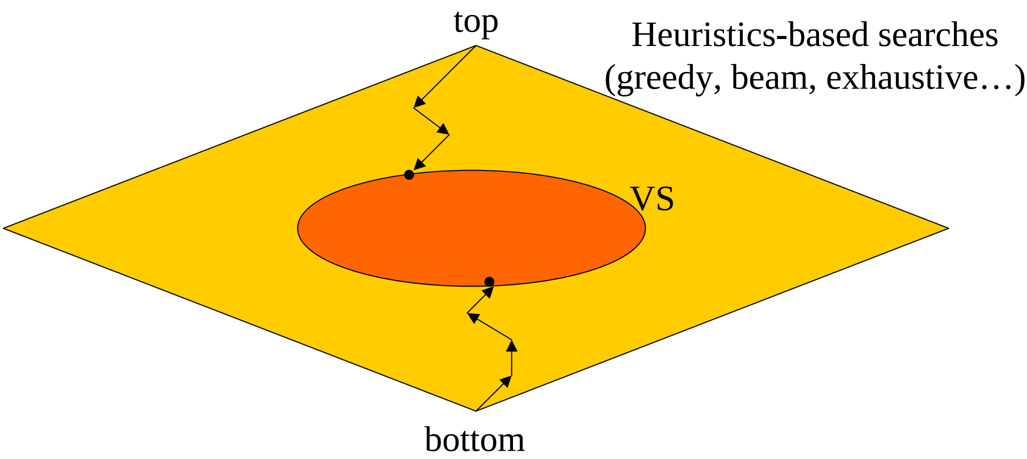
$C_1 \geq_{\mathcal{B}} C_2$ iff there exists a substitution θ for C_1 such that

- $head(C_1)\theta = head(C_2)$
- $\mathcal{B} \cup body(C_2)\sigma \mid\!- body(C_1)\theta\sigma$
- $body(C_1)\theta\sigma$ is ground.

Generality orders: generalized subsumption (2)

- Background knowledge $\mathcal{B}$
  - 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)
- Semantic generality!!
  - $C_1 \geq_{\mathcal{B}} C_2$
  - θ- subsumption fails

Refinement operators



Refinement operators: properties

- How to traverse hypothesis space so that
  - no hypotheses are generated more than once?
  - no hypotheses are skipped?
- Properties of refinement operators
  - globally complete: each point in lattice is reachable from top
  - locally complete: each point directly below c is in $\rho(c)$ (useful for greedy systems)
  - optimal: no point in lattice is reached twice (useful for exhaustive systems)
  - minimal, proper, …

Refinement operators: lgg

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

- Bottom-up search in clausal spaces
  - Starts from 2 clauses and compute least general generalisation (lgg)
  - i.e., given 2 clauses, return most specific single clause that is more general than both of them
- We shall consider only the case of clausal spaces ordered according to θ-subsumption
  - lgg under θ-subsumption

Refinement operators: lgg (2)

- Definition of **lgg of terms**:
  - (let s_i, t_j denote any term, V a variable)
  - $lgg(f(s_1,\ldots,s_n), f(t_1,\ldots,t_n)) = f(lgg(s_1,t_1),\ldots,lgg(s_n,t_n))$
  - $lgg(f(s_1,\ldots,s_n),g(t_1,\ldots,t_n)) = V$
- Definition of **lgg of literals**:
  - lgg(p(s1,…,sn),p(t1,…,tn)) = p(lgg(s1,t1),…,lgg(sn,tn))
  - lgg(¬p(…), ¬ p(…)) = ¬ lgg(p(…),p(…))
  - lgg(p(s1,…,sn),q(t1,…,tn)) is undefined
  - lgg(p(…), ¬p(…)) and lgg(¬p(…),p(…)) are undefined
- Definition of **lgg of clauses:**
  - $lgg(c_1,c_2) = \{lgg(l_1, l_2) \mid l_1 \in c_1, l_2 \in c_2 \text{ and } lgg(l_1,l_2) \text{ defined}\}$

Refinement operators: relative lgg

G. Plotkin (1971). A further note on inductive generalization. *Machine Intelligence*, 6:101-124.

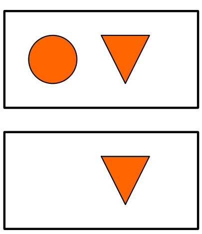
- relative to "background theory" B
  - assume B is a set of facts
- rlgg(e_1,e_2) = lgg(e_1 :- B, e_2 :- B)
- method to compute:
  - change facts into clauses with body B
  - compute lgg of clauses
  - remove B, reduce
- Used in in the ILP system Golem (Muggleton & Feng)

Refinement operators: example

- Given the following 2 simple Bongard configurations, find least general clause that would predict both to be positive

1

2



```
pos(1).                 pos(2).
contains(1,o1).         contains(2,o3).
contains(1,o2).
triangle(o1).           triangle(o3).
points(o1,down).        points(o3,down).
circle(o2).
```

Refinement operators: example

⌘Method 1: represent example by clause; compute lgg of examples

```
pos(1) :- contains(1,o1), contains(1,o2), triangle(o1),
          points(o1,down), circle(o2).
pos(2) :- contains(2,o3), triangle(o3), points(o3,down).
```

```
lgg(
(pos(1) :- contains(1,o1), contains(1,o2), triangle(o1), points(o1,down), circle(o2)) ,
(pos(2) :- contains(2,o3), triangle(o3), points(o3, down) )
= pos(X) :- contains(X,Y), triangle(Y), points(Y,down)
```


Refinement operators: example

⌘Method 2: represent class of example by fact, other properties in background; compute rlgg

Examples:

```
pos(1).
pos(2).
```

Background:

```
contains(1,o1).       contains(2,o3).
contains(1,o2).
triangle(o1).         triangle(o3).
points(o1,down).      points(o3,down).
circle(o2).
```

rlgg(pos(1), pos(2)) = ?   (exercise)

Refinement operators: Shapiro's specialization operator

E. Shapiro (1971). *An algorithm that infers theories from facts*. Proc. of the 7th Int. Conf.on Artificial Intelligence, pp. 446-451.

- Top down search in clausal spaces ordered according to theta-subsumption:
  - ρ(c) yields set of refinements of c
  - theory: $\rho(c) = \{c' \mid c'$ is a maximally general specialisation of c$\}$
  - practice: $\rho(c) \subseteq \{c \cup \{l\} \mid l$ is a literal$\} \cup \{c\theta \mid \theta$ is a substitution$\}$
- Used in many ILP systems

Declarative bias

C. Nedellec et al. (1996). *Declarative bias in ILP*. In L. De Raedt (ed.), Advances in Inductive Logic Programming, IOS Press.

- Language bias
  - Specifies and restricts the set of clauses or theories that are permitted (language of hypotheses)
- 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.

ILP logical settings

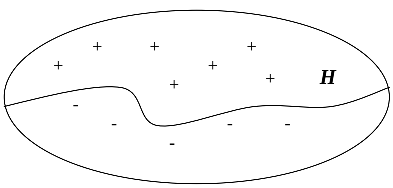
L. De Raedt, L. Dehaspe (1997). *Clausal Discovery*. Machine Learning 26(2-3): 99-146.

- Orthogonality of the following two dimensions
  - Scope of induction
    - discriminant vs. characteristic induction
  - Representation of the observations
    - learning from implications vs. learning from interpretations

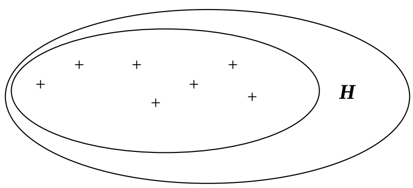
leads to 4 different logical settings for ILP

ILP logical settings: Predictive vs Descriptive ILP

Prediction



Description



ILP logical settings:
Learning from entailment

Examples:

pos(1).
pos(2).
:- pos(3).

Background knowledge:

contains(1,o1).
contains(1,o2).
contains(2,o3).
triangle(o1).
triangle(o3).
points(o1,down).
points(o3,down).
circle(o2).
contains(3,o4).
circle(o4).

Hypothesis:

pos(X) :- contains(X,Y),
triangle(Y), points(Y,down).

Example = a fact *e*
(or clause *e:-B*)

ILP logical settings:
Learning from interpretations

all information that intuitively belongs to the example, is represented in the example, not in the background knowledge!

Examples:

pos(1) :- contains(1,o1), contains(1,o2), triangle(o1),
points(o1,down), circle(o2).
pos(2) :- contains(2,o3), triangle(o3), points(o3,down).
:- pos(3), contains(3,o4), circle(o4).

Background knowledge:

polygon(X) :- triangle(X).
polygon(X) :- square(X).

knowledge concerning the domain,
not concerning specific examples!

Hypothesis:

pos(X) :- contains(X,Y),
triangle(Y), points(Y,down).

ILP logical settings: Learning from interpretations (3)

•Example as a set of facts (intepretation)
•CWA made *inside* interpretations

Examples:

pos: {contains(o1), contains(o2), triangle(o1),
points(o1,down), circle(o2)}
pos: {contains(o3), triangle(o3), points(o3,down)}
neg: {contains(o4), circle(o4)}

Background knowledge:

polygon(X) :- triangle(X).
polygon(X) :- square(X).

constraint on pos

∃Y:contains(Y),triangle(Y),points(Y,down).

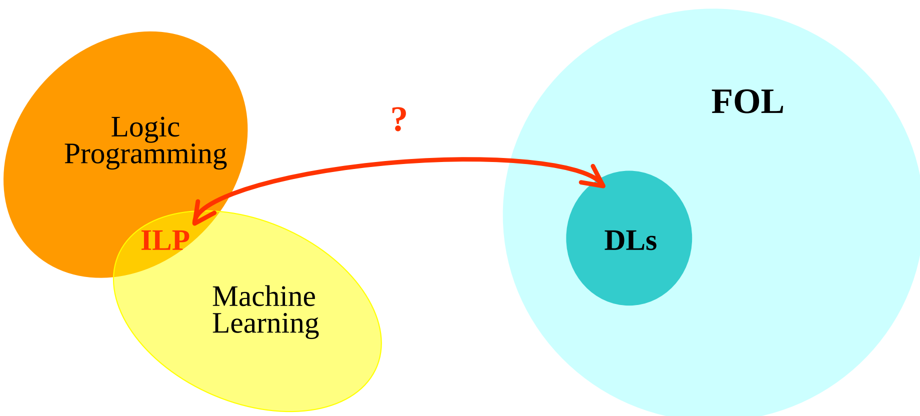
ILP logical settings: some remarks

- When learning from interpretations
  1. You can dispose of an “example identifier”
     - but can also use standard format
  2. You assume CWA for each example description
     - i.e., example description is assumed to be complete
  3. You have class of example related to information inside example + background information, NOT to information in other examples
- Because of 3rd property, more limited than learning from entailment
  - You cannot learn relations between examples, nor recursive clauses
- ... but also more efficient because of 2nd and 3rd property
  - positive PAC-learnability results (De Raedt and Džeroski, 1994), vs. negative results for learning from entailment

Part II: Overview

- Introduction to ILP
- *ILP and DL representations*
- ILP and hybrid DL-HCL representations
- ILP and the Semantic Web: Research directions

Learning in DLs



Learnability of DLs

W.W. Cohen & H. Hirsh (1992). *Learnability of Description Logics*. Proc. of the Fifth Annual Workshop on Computational Learning Theory (COLT92), pp. 116-127. ACM Press.

M. Frazier & L. Pitt (1994). *CLASSIC learning*. In Proc. of the Seventh Annual Conference on Computational Learning theory (COLT '94). ACM Press, New York, NY, 23-34.

- Learnability of sublanguages of CLASSIC w.r.t. the PAC learning model
- LCS used as a means for inductive learning from examples assumed to be concept descriptions

Learning in CLASSIC

W.W. Cohen & H. Hirsh (1994). *Learning the CLASSIC Description Logic: Theoretical and Experimental Results*. Proc. of the 4th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR94), pages 121-133.

- Supervised learning
  - Classified examples: ABox individuals
  - Goal: induce new concepts to be added to the TBox
- Search direction: bottom-up
- Algorithm: LCSLearn/LCSLearnDISJ
  1. Apply the MSC operator to compute the minimal Tbox generalizations of the examples
  2. Apply the LCS operator to generalize the MSC descriptions of examples
- Limits: overly specific concept definitions

Learning in BACK

J.-U. Kietz & K. Morik (1994). *A Polynomial Approach to the Constructive Induction of Structural Knowledge*. Machine Learning 14(1): 193-217.

- Unsupervised learning
  - Unclassified examples: ABox individuals
  - Goal: induce new concepts to be added to the TBox
- Search direction: bottom-up
- Algorithm: KLUSTER
  1. Cluster the ABox individuals into *n* mutually disjoint concepts so that *n* supervised learning problems are obtained
  2. Find a correct definition of each of these concepts as follows:
     1. Compute and evaluate the *most specific generalization* (MSG) of a concept by applying the MSC operator;
     2. Obtain the *most general discrimination* (MGD) of the concept by further generalizing the MSG.

Refinement operators for DLs

L. Badea & S.-H. Nienhuys-Cheng (2000). *A Refinement Operator for Description Logics*. In J. Cussens & A. Frisch (eds): Inductive Logic Programming, LNAI 1866, pp. 40-59

- Complete and proper refinement operator for $\mathcal{ALER}$
- No minimal refinement operators exist for $\mathcal{ALER}$
  - Minimality of all refinement steps can be achieved except for those introducing
- Complete refinement operators for $\mathcal{ALER}$ can not be locally finite
- An upward refinement operator can be obtained by inverting the arrows in the refinement rules of the downward one

Refinement operators for DLs (2)

J. Lehmann & P. Hitzler (2007b). *Foundations of Refinement Operators for Description Logics*. In: Proceedings of the 17th Int. Conf. on Inductive Logic Programming.

- Let $\mathcal{L}$ be a DL which allows to express $\top$, $\bot$, $\sqcap$, $\sqcup$, $\exists$ and $\forall$
  - E.g. $\mathcal{ALC}$
- Maximal sets of properties of $\mathcal{L}$ refinement operators
  1. {Weakly complete, complete, finite}
  2. { Weakly complete, complete, proper}
  3. { Weakly complete, non-redundant, finite}
  4. { Weakly complete, non-redundant, proper}
  5. { Non-redundant, finite, proper}

- Application: learning in $\mathcal{ALC}$ (Lehmann & Hitzler, 2007a)

Learning in $\mathcal{ALC}$

F. Esposito, N. Fanizzi, L. Iannone, I. Palmisano, & G. Semeraro (2004). *Knowledge-intensive induction of terminologies from metadata*. Proc. of the 3rd International Semantic Web Conference (ISWC04), volume 3298 of Springer LNCS, pp. 411-426.

- Supervised learning
  - Classified examples: ABox individuals
  - Goal: find a correct Tbox concept definition
- Search direction: bottom-up/top-down
- Algorithm: YinYang
  1. Apply the MSC operator to compute the minimal Tbox generalizations of the examples
  2. Apply *downward and upward refinement operators* for $\mathcal{ALC}$ to converge towards a correct concept definition
- http://www.di.uniba.it/~iannone/yinyang/

Learning in $\mathcal{ALC}$(2)

N. Fanizzi, L. Iannone, I. Palmisano, & G. Semeraro (2004). *Concept Formation in Expressive Description Logics*. In J.F. Boulicault et al. (eds.): Proc. of the 15th European Conference on Machine Learning, *ECML04*, pp. 99-110, Springer.

- Unsupervised learning
  - Unclassified examples: ABox individuals
  - Goal: induce new concepts to be added to the TBox
- Algorithm: CSKA
  1. Cluster the ABox individuals into *mutually disjoint concepts* (see KLUSTER)
  2. For each of these concepts find a correct concept definition by applying *downward and upward refinement operators* for $\mathcal{ALC}$ (see Yin/Yang)
- Application: ontology refinement

Learning in $\mathcal{ALC}$(3)

C. d'Amato, N. Fanizzi, & F. Esposito (2006). *Reasoning by Analogy in Description Logics through Instance-based Learning*. Proc. of the 3rd Italian Semantic Web Workshop.

- Algorithm: kNN-DL
  - instance-based learning system
  - based on structural/semantic *(dis)similarity measures*

N. Fanizzi, C. d'Amato, F. Esposito. *Instance Based Retrieval by Analogy*. SAC 2007 SDRC Track, 11-15 March 2007, Seoul, Korea

- Algorithm: DiVS-kNN
  - instance-based learning system
  - Based on *disjunctive version space*

Learning in $\mathcal{ALC}$(4)

N. Fanizzi & C. d'Amato (2006). *A Declarative Kernel for $\mathcal{ALC}$ Concept Descriptions.* ISMIS 2006: Lecture Notes in Computer Science 4203, pp. 322-331

- Task: classification
- From distances to kernels
  - Kernel is a similarity measure (can be obtained from distances)
  - Kernel machine = algorithm parameterized by kernels

Learning in DLs: bibliography

- J. Alvarez (1998). A Description Logic System for Learning in Complex Domains. Proc. of the 1998 Int. Workshop on Description Logics (DL'98).
- J. Alvarez (2000a). A Formal Framework for Theory Learning using Description Logics. Proc. of Int. Workshop on Inductive Logic Programming (ILP'00), work in progress track.
- J. Alvarez (2000b). TBox Acquisition and Information Theory. In: Proc. of the 2000 Int. Workshop on Description Logics (DL'00).
- L. Badea & S.-H. Nienhuys-Cheng (2000a). A Refinement Operator for Description Logics. ILP 2000: 40-59
- L. Badea & S.-H. Nienhuys-Cheng (2000b). Refining Concepts in Description Logics. Description Logics 2000: 31-44

Learning in DLs: bibliography (2)

- W.W. Cohen, A. Borgida, & H. Hirsh (1992). *Computing Least Common Subsumers in Description Logics*. Proc. of the Tenth National Conf. on Artificial Intelligence (AAAI92), pages 754-760. AAAI Press/MIT Press.
- W.W. Cohen & H. Hirsh (1992). *Learnability of Description Logics*. Proc. of the Fifth Annual Workshop on Computational Learning Theory (COLT92), pages 116-127. ACM Press.
- W.W. Cohen & H. Hirsh (1994a). *Learning the CLASSIC Description Logic: Theoretical and Experimental Results*. Proc. of the 4th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR94), pages 121-133.
- W.W. Cohen & H. Hirsh (1994b). *The Learnability of Description Logics with Equality Constraints*. Machine Learning, 17(2):169-199.

Learning in DLs: bibliography (3)

- C. d'Amato, N. Fanizzi, & F. Esposito (2006). *A dissimilarity measure for $\mathcal{ALC}$ concept descriptions*. SAC 2006: 1695-1699
- C. d'Amato & N. Fanizzi (2006). *Lazy Learning from Terminological Knowledge Bases*. Proc. 16th International Symposium on Methodologies for Intelligent Systems, 27-29 September 2006, Bari, Italy
- F. Esposito, N. Fanizzi, L. Iannone, I. Palmisano, G. Semeraro (2004). *Knowledge-Intensive Induction of Terminologies from Metadata*. International Semantic Web Conference 2004: 441-455
- F. Esposito, N. Fanizzi, L. Iannone, I. Palmisano, G. Semeraro (2005). *A Counterfactual-Based Learning Algorithm for Description Logic*. AI*IA 2005: 406-417
- F. Esposito, N. Fanizzi, L. Iannone, I. Palmisano, G. Semeraro: *Induction and Revision of Terminologies*. ECAI 2004: 1007-1008

Learning in DLs: bibliography (4)

- N. Fanizzi & C. d'Amato (2006). *A Declarative Kernel for $\mathcal{ALC}$ Concept Descriptions*. ISMIS: 322-331.
- N. Fanizzi, S. Ferilli, L. Iannone, I. Palmisano, & G. Semeraro (2004). *Downward Refinement in the $\mathcal{ALN}$ Description Logic*. HIS 2004: 68-73.
- N. Fanizzi, L. Iannone, I. Palmisano, G. Semeraro (2004). *Concept Formation in Expressive Description Logics*. ECML 2004: 99-110.
- M. Frazier & L. Pitt (1994). *CLASSIC learning*. In Proceedings of the Seventh Annual Conference on Computational Learning theory (COLT '94). ACM Press, New York, NY, 23-34.
- M. Frazier & L. Pitt (1996). *CLASSIC Learning*. Machine Learning, 25 (2-3): 151-193.

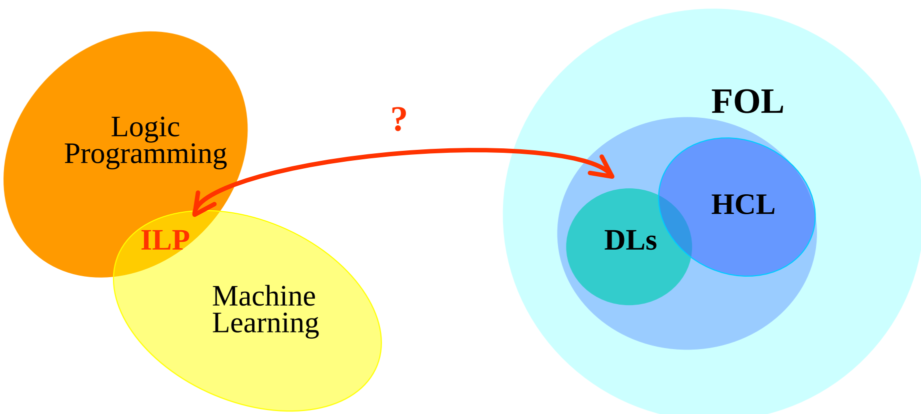
Learning in DLs: bibliography (5)

- L. Iannone, I. Palmisano and N. Fanizzi (2007). *An algorithm based on counterfactuals for concept learning in the Semantic Web*. Applied Intelligence, 26(2): 139-159.
- J. Lehmann & P. Hitzler (2007a). *A Refinement Operator Based Learning Algorithm for the $\mathcal{ALC}$ Description Logic*. In: Proceedings of the 17th International Conference on Inductive Logic Programming (ILP) 2007
- J. Lehmann & P. Hitzler (2007b). *Foundations of Refinement Operators for Description Logics*. In: Proceedings of the 17th International Conference on Inductive Logic Programming (ILP) 2007
- V. Ventos, P. Brézellec, H. Soldano, D. Bouthinon (1998). *Learning Concepts in C-CLASSIC(delta/epsilon)*. Description Logics 1998

Part II: Overview

- Introduction to ILP
- ILP and DL representations
- *ILP and hybrid DL-HCL representations*
- ILP and the Semantic Web: Research directions

Learning in DL-HCL



Learning in CARIN-$\mathcal{ALN}$

C. Rouveirol & V. Ventos (2000). *Towards learning in* CARIN-$\mathcal{ALN}$. In J. Cussens & A. Frisch (eds): Inductive Logic Programming, Springer LNAI 1866, 191-208.

- Scope of induction: prediction
- Logical setting: learning from interpretations
- Language of hypotheses: definite clauses in CARIN-$\mathcal{ALN}$
- Generality order: adaptation of Buntine's generalized subsumption to CARIN-$\mathcal{ALN}$
- Coverage relations: query answering in CARIN-$\mathcal{ALN}$

Learning in CARIN-$\mathcal{ALN}$ (2)

J.-U. Kietz (2003). *Learnability of description logic programs*. In S. Matwin and C. Sammut (Eds.), Inductive Logic Programming, Springer LNAI 2583, 117-132.

- Method for transforming CARIN-$\mathcal{ALN}$ into Datalog extended with numerical constraints
- Transfer of learnability results known for ILP to learning in CARIN-$\mathcal{ALN}$

Learning in $\mathcal{AL}$-log

F.A. Lisi (2005). *Principles of Inductive Reasoning on the Semantic Web: A Framework for Learning in AL-log*. In F. Fages and S. Soliman (Eds.), Principles and Practice of Semantic Web Reasoning, Springer LNCS 3703, 118-132.

- Scope of induction: prediction/description
- Logical setting: learning from interpretations/learning from implications
- Language of hypotheses: constrained Datalog clauses
- Generality order: adaptation of Buntine's generalized subsumption to $\mathcal{AL}$-log
- Coverage relations: query answering in $\mathcal{AL}$-log

Learning in DL-HCL: Bibliography

- A.M. Frisch (1991). *The Substitutional Framework for Sorted Deduction: Fundamental Results on Hybrid Reasoning*. Artif. Intell. 49(1-3): 161-198.
- A.M. Frisch (1999). *Sorted downward refinement: Building background knowledge into a refinement operator for inductive logic programming*. In S. Dzeroski and P.A. Flach (Eds.), Inductive Logic Programming, Springer LNAI 1634, 104-115 .
- J.-U. Kietz (2003). *Learnability of description logic programs*. In S. Matwin and C. Sammut (Eds.), Inductive Logic Programming, Springer LNAI 2583, 117-132.
- F.A. Lisi (2005). *Principles of Inductive Reasoning on the Semantic Web: A Framework for Learning in AL-log*. In F. Fages and S. Soliman (Eds.), Principles and Practice of Semantic Web Reasoning, Springer LNCS 3703, 118-132.

Learning in DL-HCL: Bibliography (2)

- F.A. Lisi (2006). *Practice of Inductive Reasoning on the Semantic Web.* In: J.J. Alferes, J. Bailey, W. May, U. Schwertel (Eds.), Principles and Practice of Semantic Web Reasoning, Springer LNCS 4187, 242-256.
- F.A. Lisi & F. Esposito (2004). *Efficient Evaluation of Candidate Hypotheses in $\mathcal{AL}$-log*. In R. Camacho, R. King, and A. Srinivasan (Eds.), Inductive Logic Programming, Springer LNAI 3194, 216-233.
- F.A. Lisi & F. Esposito (2006). *Two Orthogonal Biases for Choosing the Intensions of Emerging Concepts in Ontology Refinement*. In G. Brewka, S. Coradeschi, A. Perini & P. Traverso (Eds.): *ECAI 2006. Proc. of the 17th European Conf. on Artificial Intelligence*, IOS Press: Amsterdam, 765-766.
- F.A. Lisi & F. Esposito (2007). *On the Missing Link between Frequent Pattern Discovery and Concept Formation*. To appear in: S. Muggleton, R. Otero & A. Tamaddoni-Nezhad (Eds.), Inductive Logic Programming, Springer LNAI ?, ?-?.

Learning in DL-HCL: Bibliography (3)

- F.A. Lisi & D. Malerba (2003). *Ideal Refinement of Descriptions in $\mathcal{AL}$-log*. In T. Horvath and A. Yamamoto (Eds.), Inductive Logic Programming, LNAI 2835, 215-232, Springer: Berlin.
- F.A. Lisi & D. Malerba (2003). *Bridging the Gap between Horn Clausal Logic and Description Logics in Inductive Learning*. In A. Cappelli and F. Turini (Eds.), AI*IA 2003: Advances in Artificial Intelligence, LNAI 2829, 53-64, Springer: Berlin.
- F.A. Lisi & D. Malerba (2004). *Inducing Multi-Level Association Rules from Multiple Relations*. Machine Learning, 55:175-210.
- C. Rouveirol & V. Ventos (2000). *Towards learning in CARIN-$\mathcal{ALN}$*. In J. Cussens & A. Frisch (eds): Inductive Logic Programming, Springer LNAI 1866, 191-208.

Part II: Overview

- Introduction to ILP
- ILP and DL representations
- ILP and hybrid DL-HCL representations
- *ILP and the Semantic Web: Research directions*

ILP and the Semantic Web: research directions in theory

- ILP frameworks for learning/mining in more expressive DLs and DL-HCL hybridizations
  - closer to OWL and SWRL
- ILP frameworks for learning/mining under uncertainty and vagueness
  - closer to real-world ontologies
- ILP frameworks for learning/mining from multiple contexts
  - Closer to the real scenario of the Semantic Web

ILP and the Semantic Web: research directions in practice

- Efficient implementations
- Interfacing of ILP systems with specialized reasoners for the Semantic Web
  - (Fuzzy) OWL/SWRL reasoners
- Experimental work on big OWL/SWRL ontologies

ILP and the Semantic Web: applications for learning in DLs

- Ontology Refinement
- Ontology Matching
- Ontology Merging
- FOAF
- Semantic retrieval
- Etc.

ILP and the Semantic Web: applications for learning in DL-HCL

- Ontology Refinement
  - Some concepts are better defined with rules
- Ontology Mapping
- Semantic Web Services
- Business rules
- Policy rules
- Etc.

Potentially all RIF use cases!

Further resources

- Tutorials on the Semantic Web
  - http://www.w3.org/2001/sw/BestPractices/Tutorials
  - http://km.aifb.uni-karlsruhe.de/ws/prowl2006/
  - http://rease.semanticweb.org/
- Tutorials on Machine Learning for the Semantic Web
  - http://www.aifb.uni-karlsruhe.de/WBS/pci/OL_Tutorial_ECML_PKDD_05/
  - http://www.uni-koblenz.de/~staab/Research/Events/ICML05tutorial/icml05tutorial.pdf
  - http://www.smi.ucd.ie/Dagstuhl-MLSW/proceedings/
  - http://ingenieur.kahosl.be/projecten/swa2002/slides/hendrik%20blockeel/Blockeel.ppt