Description Logics: 
A Logical Foundation of the 
Semantic Web and its Applications

Volker Haarslev
Concordia University, Computer Science Department
1455 de Maisonneuve Blvd. W.
Montreal, Quebec H3G 1M8, Canada

http://www.cs.concordia.ca/~faculty/haarslev/
haarslev@cs.concordia.ca

Idea of the Semantic Web

- World Wide Web
  - medium of
    - documents for people rather than of
      - information that can be manipulated automatically
    - augment web pages with data targeted at computers
    - add documents solely for computers
      - called semantic markup
  - ...transforms into the Semantic Web

- Find meaning of semantic data by following
  - hyperlinks to definitions of key terms and
    - rules for reasoning about data logically

- Spur development of automated web services
  - highly functional agents

Tim Berners-Lee, James Hendler,
Ora Lassila: The Semantic Web
Typical Information Retrieval Example

- Suppose you are a salesperson, who wishes to find a Ms. Cook you met at a trade conference last year
  - you don’t remember her first name but
  - you remember she worked for one of your clients and
  - her daughter is a student of your alma mater
- An intelligent search agent can
  - ignore pages relating to cooks, cookies, Cook Islands, etc.
  - find pages of companies your clients are working for
  - follow links to or find private home pages
  - check whether a daughter is still in school
  - match with students from your alma mater
- If you already have the Semantic Web

Basic Web Technology

- Uniform Resource Identifier (URI)
  - foundation of the Web
  - identify items on the Web
  - uniform resource locator (URL): special form of URI
- Extensible Markup Language (XML)
  - send documents across the Web
  - allows anyone to design own document formats (syntax)
  - can include markup to enhance meaning of document’s content
  - machine readable
- Resource Description Framework (RDF)
  - make machine-processable statements
  - triple of URIs: subject, predicate, object
  - intended for information from databases
Schemas and Ontologies for the Web

- Usual assumption: data is nearly perfect
  - book rating with scale 1-10 instead of really_good,...,really_bad
  - conversion without meaning difficult
  - information newly tagged with has_author instead of creator_of
- Even worse: URIs have no meaning
- Solution: schemas and ontologies
- RDF Schemas: author is subclass of contributor
- DARPA Agent Markup Language with Ontology Inference Layer (DAML+OIL)
  - add semantics: has_author is the inverse relation of creator_of
  - now we understand the meaning of has_author
  - has_author(book,author) \equiv creator_of(author,book)

A Logical Foundation for the Semantic Web

- Systems can understand basic concepts such as
  - subclass
  - inverse relation, etc.
- Even better
  - state (any) logical principle
  - permit computers to reason (by inference) using these principles
  - an employee sells more than 100 items per day \Rightarrow bonus
  - follow semantic links to construct a proof for your conclusions
  - exchange proofs between agents (and human users)
- DAML+OIL is a syntactic variant of a well-known and very expressive description logic
Why Description Logics?

- Designed to represent knowledge
- Based on formal semantics
- Inference problems have to be decidable
- Probably the most thoroughly understood set of formalisms in all of knowledge representation
- Computational space has been thoroughly mapped out
- Wide variety of systems have been built
  - however, only very few highly optimized systems exist
- Wide range of logics developed
  - from very simple (no disjunction, no full negation)
  - to very expressive (comparable to DAML+OIL)
- Very tight coupling between theory and practice

Description Logics: Introduction (1)

- Origins
  - structured inheritance networks
  - frame-based representations
- Factual world
  - named individuals, e.g., charles, elizabeth
  - (binary) relationships between individuals, e.g., has_child
- Descriptions form hierarchical knowledge
  - two disjoint alphabets: concept and role names
  - roles denote binary descriptions, e.g., has_child(x,y)
  - concepts denote unary descriptions, e.g.,
    parent(x) = person(x) \land 3y : (has_child(x,y) \land person(y))
**Description Logics: Introduction (2)**

- Important syntactic feature: variable-free notation
  - Constructors: $\cap$, $\cup$, $\neg$, $\exists$, $\forall$
  - Standard description logic $\mathcal{ALC}$
- Description of concept **parent**
  - $\text{parent} \equiv \text{person} \cap \exists\text{has\_child}\cdot\text{person}$
- We add two concepts
  - $\text{woman} \equiv \text{female} \cap \text{person}$
  - $\text{mother} \equiv \text{female} \cap \text{parent}$
- What type of inferences are interesting?
  - Satisfiability of (named) concepts
  - Subsumption of (named) concepts

**Inference Service: Concept Satisfiability**

- The concepts **woman**, **mother**, **parent** are satisfiable
- However, the concept $\neg\text{woman} \cap \text{mother}$ is unsatisfiable
- Why? We unfold the definition of **woman** and **mother**
  - $\neg\text{woman} \cap \text{mother} \equiv$
  - $\neg(\text{female} \cap \text{person}) \cap \text{female} \cap \text{parent} \equiv$
  - $(\neg\text{female} \cup \neg \text{person}) \cap \text{female} \cap \text{parent} \equiv$
  - $(\neg\text{female} \cup \neg \text{person}) \cap \text{female} \cap \text{parent} \equiv$
  - $\neg\text{person} \cap \text{female} \cap \text{parent} \equiv$
  - $\neg\text{person} \cap \text{female} \cap \text{person} \cap \exists\text{has\_child}\cdot\text{person} \equiv$
  - $\neg\text{person} \cap \text{female} \cap \text{person} \cap \exists\text{has\_child}\cdot\text{person}$
  - $\neg$
- The conjunct $\neg\text{woman} \cap \text{mother}$ can never be satisfied
Inference Service: Concept Subsumption

Consider the question

*Is a mother always a woman?*

Subsumes the concept *woman* the concept *mother*?

Description logic reasoners offer the computation of a subsumption hierarchy (taxonomy) of all named concepts

![Diagram showing the hierarchy of concepts](image)

Yes, *woman* subsumes *mother* (see also proof on previous slide)

Description Logics: Semantics (1)

Translation to first-order predicate logic usually possible

Declarative and compositional semantics preferred

Standard Tarski-style interpretation \( \mathcal{J} = (\Delta^3, \cdot^3) \)

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A )</td>
<td>( A^3 \subseteq \Delta^3 ), ( A ) is a concept name</td>
</tr>
<tr>
<td>( \neg C )</td>
<td>( \Delta^3 \setminus C^3 )</td>
</tr>
<tr>
<td>( C \cap D )</td>
<td>( C^3 \cap D^3 )</td>
</tr>
<tr>
<td>( C \cup D )</td>
<td>( C^3 \cup D^3 )</td>
</tr>
<tr>
<td>( \forall R.C )</td>
<td>( { x \in \Delta^3 \mid \forall y: (x,y) \in R^3 \Rightarrow y \in C^3 } )</td>
</tr>
<tr>
<td>( \exists R.C )</td>
<td>( { x \in \Delta^3 \mid \exists y \in \Delta^3: (x,y) \in R^3 \land y \in C^3 } )</td>
</tr>
<tr>
<td>( R )</td>
<td>( R^3 \subseteq \Delta^3 \times \Delta^3 ), ( R ) is a role name</td>
</tr>
<tr>
<td>( C \sqsubseteq D )</td>
<td>( C^3 \sqsubseteq D^3 )</td>
</tr>
<tr>
<td>( C \equiv D )</td>
<td>( C^3 = D^3 )</td>
</tr>
</tbody>
</table>

Concepts

Roles

Axioms
Description Logics: Concept Examples

- woman ≡ person □ female
- parent ≡ person □
  ∃ has_child.person
- mother ≡ parent □ female
- mother_having_only_female_kids ≡ mother □
  ∀ has_child.female
- mother_having_only_daughters ≡ woman □
  parent □
  ∀ has_child.woman

\[\equiv\]

- grandma ≡ woman □ ∃ has_child.parent
- great_grandma ≡ woman □
  ∃ has_child.∃ has_child.parent
Description Logics: Semantics (2)

- Interpretation domain can be chosen arbitrarily
- Distinguishing features of description logics
  - domain can be infinite
  - open world assumption
- A concept $C$ is satisfiable iff there exists an interpretation $I$ such that $C^I \neq \emptyset$
  - $I$ is called a model of $C$
- Subsumption can be reduced to satisfiability
  - $\text{subsumes}(C, D) \iff \neg \text{sat}(\neg C \sqcap D)$
  - denoted as $C \sqsupseteq D$ or $D \sqsubseteq C$

Description Logics: TBox

- A collection of concept axioms is called a TBox (Terminological Box)
- Satisfiability of concepts defined w.r.t. a TBox $\mathcal{T}$
- Inference services
  - TBox coherence: List all unsatisfiable concept names in $\mathcal{T}$
  - compute subsumption hierarchy (taxonomy) of concept names in $\mathcal{T}$
- Why emphasize concept names?
  - ontological decisions of users
  - important concepts will be named
Example Taxonomy

![Diagram of a taxonomy hierarchy]

Description Logics: Individuals

- How can we assert knowledge about individuals?
- Assertional axioms
  - concept assertion for an individual $a$
    - $a : C$ satisfied iff $a^I \in C^I$
    - example: elizabeth : mother
  - role assertion for two individuals $a$ and $b$
    - $(a,b) : R$ satisfied iff $(a^I, b^I) \in R^I$
    - example: (elizabeth,charles) : has_child
- Unique name assumption
  - Different names denote different individuals
  - $a^I \neq b^I$
Description Logics: ABox (1)

- A collection of assertional axioms is called an ABox (Assertional Box)
- Satisfiability of assertions defined w.r.t.
  - ABox \( \mathcal{A} \)
  - TBox \( \mathcal{T} \)
- Inference services
  - ABox satisfiability: Is the collection \( \mathcal{A} \) of assertions satisfiable?
  - Instance checking: \( \text{instance}(a, C, \mathcal{A}) \)
    Is \( a \) an instance of concept \( C \) or subsumes \( C \) the individual \( a \)?
  - ABox realization: compute for all individuals in \( \mathcal{A} \) their most-specific concept names w.r.t. TBox \( \mathcal{T} \)

Description Logics: ABox (2)

- New basic inference service: ABox satisfiability
  - \( \text{asat}(\mathcal{A}) \)
- All other inference services can be reduced to \( \text{asat} \)
  - instance checking:
    \( \text{instance}(a, C, \mathcal{A}) = \neg \text{asat}(\mathcal{A} \cup \{a: \neg C\}) \)
  - concept satisfiability:
    \( \text{sat}(C) = \text{asat}({a:C}) \)
  - concept subsumption:
    \( \text{subsumes}(C, D) = \neg \text{sat}(\neg C \sqcap D) = \neg \text{asat}({a: \neg C \sqcap D}) \)

Open world assumption
- \( \mathcal{A} = \{\text{andrew: male}, (\text{charles, andrew}): \text{has child}\} \)
- Does \( \text{instance}(\text{charles}, \forall \text{has child: male}, \mathcal{A}) \) hold?
  No.
  Why?
  (See later)
Description Logics: ABox Example

- (male ⊑ ¬female)  
  additional axiom ensuring disjointness

- queen_mum : woman
- (queen_mum,elizabeth) : has_child
- elizabeth : woman
- (elizabeth,charles) : has_child
- (elizabeth,anne) : has_child
- charles : parent □ male
- anne : woman
- (charles,andrew) : has_child
- andrew : person □ male

TBox Taxonomy plus Individuals

- queen_mum
- elizabeth
- charles
- anne
- andrew

Diagram with Taxonomy and Individuals
Open World Assumption

- Can we prove that instance?(charles, ∀has_child.male, A) holds?
- No. Although the ABox contains only knowledge about one male child, it is unknown whether additional information about a female child might be added later.
- In order to prevent this, we could add
  - charles : ∀has_child.male or
  - assert that information about a second child will not be added in the future, i.e., close a role for an individual
- Not possible in the logic ALC since we need so-called number restrictions

More Description Logics Constructors

- Number restrictions on roles (N resp. Q)
  - simple: Ǝ≥2has_child or Ǝ≤5has_child
  - qualified: Ǝ≥2has_child.male or Ǝ≤1has_child.female
- Role hierarchies (H)
  - has_son ⊆ has_child, has_daughter ⊆ has_child
  - Ǝ≥2has_son ∩ Ǝ≥2has_daughter ∩ Ǝ≤4has_child
- Transitive roles (R+)
  - has_ancestors declared as transitive: ∀has_ancestors.human
  - has_parent ⊆ has_ancestors
- Inverse roles (I): has_parent = has_child
- Terminological cycles: human ⊆ Ǝ≥2has_parent.human
- General axioms
  - woman ∩ Ǝhas_child.Ǝhas_child.person ⊆ grandma

12
Tableau Methods

- How can we prove the satisfiability of a concept?
  - Achieved by applying tableau methods
    - set of completion rules operating on constraint sets or tableaux
    - clash triggers
  - Proof procedure
    - transform all concepts into negation normal form, e.g.,
      \[ \neg(C \cap D) \rightarrow \neg C \cup \neg D, \neg \exists R.C \rightarrow \forall R.\neg C \]
    - apply completion rules in arbitrary order as long as possible
      - application of rules
        - stops in case of a clash
        - terminates if no completion rule is applicable
      - satisfiable iff a clash-free tableau can be derived

Completion Rules for the Logic $ALC$

**Clash trigger**
\[ \{a:C, a:\neg C\} \subseteq A \]

**Conjunction rule**
if 1. $a:C \cap D \in A$, and
2. $\{a:C, a:D\} \not\subseteq A$
then $A' = A \cup \{a:C, a:D\}$

**Disjunction rule**
if 1. $a:C \cup D \in A$, and
2. $\{a:C, a:D\} \cap A = \emptyset$
then $A' = A \cup \{a:C\}$ or
$A' = A \cup \{a:D\}$

**Role exists restriction rule**
if 1. $a:\exists R.C \in A$, and
2. $\neg \exists b \in O: \{(a,b):R, b:C\} \subseteq A$
then $A' = A \cup \{(a,b):R, b:C\}$ with $b$ fresh in $A$

**Role value restriction rule**
if 1. $a:\forall R.C \in A$, and
2. $\exists b \in O: (a,b):R \in A$, and
3. $\{b:C\} \not\subseteq A$
then $A' = A \cup \{b:C\}$
Proof for Concept Satisfiability

- Subsumes the concept \textit{woman} the concept \textit{mother}?  
- Is the concept \textit{\neg woman \sqcap mother} unsatisfiable?  
- Application of completion rules
  - $\mathcal{A}_0 = \{a: (\neg \text{female} \sqcup \neg \text{person}) \sqcap \text{female} \sqcap \text{person} \sqcap \ldots\}$ (conjunction rule)  
  - $\mathcal{A}_1 = \{a: \neg \text{female}, a: \text{female}, a: \text{person}, \ldots\}$ (disjunction rule)  
  - $\mathcal{A}_2 = \{a: \neg \text{female} \sqcup \neg \text{person}, a: \text{female}, a: \text{person}, \ldots, a: \neg \text{female}\}$  
  - $\checkmark$ (clash between $a: \text{female}$ and $a: \neg \text{female}$ detected)  
  - $\mathcal{A}_3 = \{a: \neg \text{female} \sqcup \neg \text{person}, a: \text{female}, a: \text{person}, \ldots\}$ (disjunction rule)  
  - $\mathcal{A}_4 = \{a: \neg \text{female} \sqcup \neg \text{person}, a: \text{female}, a: \text{person}, \ldots, a: \neg \text{person}\}$  
  - $\checkmark$ (clash between $a: \text{person}$ and $a: \neg \text{person}$ detected)  
- The concept $\neg \text{woman} \sqcap \text{mother}$ is unsatisfiable  
- The concept \textit{woman} subsumes the concept \textit{mother}  

Reasoning with Description Logics

- **RACER**: Reasoner for ABoxes and Concept Expressions Renamed  
  - Based on sound and complete algorithms  
  - Worst case complexity for many description logics  
    - PSpace, e.g., the logic $ALC$  
    - ExpTime, e.g., the logic $ALC$ with general axioms  
    - NexpTime  
      - the logic $ALCQ_{HIR}((D))$ supported by RACER  
      - the DAML+OIL logic  
  - Highly optimized reasoners required  
    - average complexity usually much better  
  - RACER is still the only reasoner for ABoxes
RACER System

- First system for $ALCQHI_{R^+}$ with ABoxes
  - sublogic of DAML+OIL
- Multiple TBoxes, multiple ABoxes
- Standalone server versions available for Linux and Windows (with Java interface)
- Newly added: concrete domains
  - represent constraints with linear inequations over the Reals
  - for instance: the relationship between the Celsius and Fahrenheit scales
- Almost finished
  - XML / RDF / DAML+OIL interface
- Standardized interface (API) is being developed

Selected Optimization Techniques

- State of the art optimization techniques employed
- Novel optimization techniques for
  - SAT reasoning
    - dependency-directed backtracking
    - semantic branching
    - caching
    - process qualified number restrictions with Simplex procedure
  - TBox reasoning
    - transformation of general axioms
    - classification order / clustering of nodes
    - fast test for non-subsumption: sound but incomplete
  - ABox reasoning
    - graph transformation
    - fast test for non-subsumption
    - data-flow techniques for realization
    - dependency-driven divide-and-conquer for instance checks
Application: UML Verification

XML representation created by UML Editor or Tool

Agent

Ship ⊑ ∃₁what_location_where. Port
ContainerShip ⊑ Ship

Application: Ontology Engineering

- UMLS thesaurus (Unified Medical Language System)
- Transformation into logic $ALCNH$
  - TBox with cycles, role hierarchy, and simple number restrictions
- UMLS knowledge bases
  - 200,000 concept names, 80,000 role names
- Optimization of TBox classification
  - topological sorting
    - achieving smart ordering for classification of concept names
  - dealing with domain and range restrictions of roles
    - transformation of special kind of general axioms
  - clustering of nodes in the taxonomy
  - speed up from several days to ~10 hours
  - new processors: ~3 hours
TBox Classification: Inserting a Concept

- Insert new concept $D$ into existing taxonomy w.r.t subsumption relationship

1. Top-search phase
   - traverse from top
   - determine parents of $D$
     - $C_1$ and $C_2$
     - $\text{SAT}(\neg C_1 \sqcap D), \ldots, \text{SAT}(\neg C_n \sqcap D)$

2. Bottom-search phase
   - traverse from bottom
   - determine children of $D$
     - $C_3$ and $C_4$
     - $\text{SAT}(C_1 \sqcap \neg D), \ldots, \text{SAT}(C_n \sqcap \neg D)$
Application: Distributed Agents

- Specialized reasoner for TV programs
- Specialized reasoner for data from Geographical Information Systems (GIS)
- Broker agent as mediator

Spatial Reasoning with Description Logics

- Binary predicates for qualitative spatial reasoning (RCC theory)

<table>
<thead>
<tr>
<th>Predicate</th>
<th>Diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>connected</td>
<td><img src="image" alt="connected" /></td>
</tr>
<tr>
<td>disjoint</td>
<td><img src="image" alt="disjoint" /></td>
</tr>
<tr>
<td>g_contains</td>
<td><img src="image" alt="g_contains" /></td>
</tr>
<tr>
<td>g_inside</td>
<td><img src="image" alt="g_inside" /></td>
</tr>
<tr>
<td>g_overlapping</td>
<td><img src="image" alt="g_overlapping" /></td>
</tr>
<tr>
<td>t_contains</td>
<td><img src="image" alt="t_contains" /></td>
</tr>
<tr>
<td>s_contains</td>
<td><img src="image" alt="s_contains" /></td>
</tr>
<tr>
<td>equal</td>
<td><img src="image" alt="equal" /></td>
</tr>
<tr>
<td>t_inside</td>
<td><img src="image" alt="t_inside" /></td>
</tr>
<tr>
<td>s_inside</td>
<td><img src="image" alt="s_inside" /></td>
</tr>
<tr>
<td>touching</td>
<td><img src="image" alt="touching" /></td>
</tr>
<tr>
<td>s_overlapping</td>
<td><img src="image" alt="s_overlapping" /></td>
</tr>
<tr>
<td>t_contains</td>
<td><img src="image" alt="t_contains" /></td>
</tr>
<tr>
<td>s_contains</td>
<td><img src="image" alt="s_contains" /></td>
</tr>
</tbody>
</table>
Example: Paradise Cottage (1)

- A paradise cottage
  - it is a cottage
  - suitable for fishing
    - located in the immediate vicinity of a river
    - simplification: estate touches a river
  - located in a mosquito-free forest
    - simplification: a mosquito-free forest does not overlap with a river
- Specification with \( ALCRP(D) \)
  - \( \text{fishing} \_ \text{cottage} = \text{cottage} \land \exists \text{is} \_ \text{touching} \_ \text{river} \)
  - \( \text{mosquito} \_ \text{free} \_ \text{forest} = \text{forest} \land \forall \text{is} \_ \text{connected} \_ \neg \text{river} \)
  - \( \text{paradise} \_ \text{cottage} = \text{fishing} \_ \text{cottage} \land \exists \text{g} \_ \text{inside} \_ \text{forest} \land \forall \text{g} \_ \text{inside} \_ \text{mosquito} \_ \text{free} \_ \text{forest} \)
- What is your opinion: dream or reality?

Example: Paradise Cottage (2)

- A situation, where a region \( r_1 \) (cottage) is located inside another region \( r_2 \) (forest) and the region \( r_1 \) touches a third region \( r_3 \) (river), implies that \( r_2 \) must be connected with \( r_3 \)
  \[ \text{g} \_ \text{inside}(r_1, r_2) \land \text{touching}(r_1, r_3) \Rightarrow \text{connected}(r_2, r_3) \]
- The concept paradise_cottage is unfortunately unsatisfiable due to induced spatial constraints
  - a mosquito-free forest is not allowed to be spatially connected with a river
  - only detectable with the logic \( ALCRP(D) \)
Future Research (1)

- Integration of spatial reasoning into description logics
  - bioinformatics
  - (semantics of) spatial queries
  - geographical information systems
- Extend support for very expressive description logics
  - integration of individuals into concept descriptions
  - concrete domains
    - non-linear, multivariate systems of inequations
- Development of new optimization techniques
  - inverse roles
  - individuals in concept descriptions
  - complex (and very large) knowledge bases

Future Research (2)

- Support of Semantic Web
- Support for databases
  - schemas
  - query subsumption
  - database integration
- Development of (industrial) applications
  - geographical information systems
  - telecommunication systems / mobile systems
  - computer vision
  - matchmaking of services
  - natural language understanding
  - ...
Other Areas of Interest

- Diagrammatic reasoning
- Visual languages / notations
- Knowledge management / engineering
- Software engineering (for AI)
- Object-oriented design
- Programming languages / paradigms
- ...

21