INFR11215 Knowledge Graphs

Description Logics

Jeff Z. Pan

http://knowledge-representation.org/j.z.pan/

[Reading: Baader et al., Chapters 1 and Sections 2.1 and 2.2]

Mind the Syntax

• All rich men love Jane
  – $\forall x[\text{Rich}(x) \land \text{Man}(x) \rightarrow \text{love}(x, \text{Jane})]$
  – $\forall x[\text{Rich}(x) \land \text{Man}(x) \supset \text{love}(x, \text{Jane})]$ //KR book
  – $\forall x[\text{Rich}(x) \land \text{Man}(x) \Rightarrow \text{love}(x, \text{Jane})]$ //DL book

– Rich $\sqcap$ Man $\sqsubseteq \exists \text{love.}(\text{Jane})$
– $\text{SubClassOf (}
\text{intersectionOf(Rich, Man),}
\text{restriction(love someValueFrom(oneOf(Jane)))}
\text{)}$
Mind the Syntax

• Jane loves both John and Jim
  – love(Jane,John) ∧ love(Jane,Jim)
  – Jane: \(\exists\text{love,}\{\text{John}\} \cup \{\text{Jim}\}\) //complex fact
  – or simply (Jane,John):love, (Jane,Jim):love //simple fact

  – Individual (Jane
    value(love, John) value(love,Jim)
  ) //simple fact
  – Individual (Jane
    type(restriction(love someValueFrom(oneOf(John,Jim))))
  ) //complex fact

Mind the Syntax

• Jane loves either John or Jim
  – love(Jane,John) ∨ love(Jane,Jim)
  – Jane: \(\exists\text{love,}\{\text{John}\} \cup \{\text{Jim}\}\) \(\cap =1\)love

  – Individual (Jane
    type(intersectionOf
      restriction(love someValueFrom(oneOf(John,Jim))))
      restriction(love maxCardinality(1))
      restriction(love minCardinality(1))
  )
Lecture Outline

- Motivation
- Overview of Description Logics (DLs)
- Semantics of DLs and Reasoning in DLs
- Practical

What is Knowledge
How to Classify them

- **Knowledge: verified beliefs and practical skills** (e.g. operating an instrument)
- How to represent knowledge? How to classify knowledge?

- Entity level knowledge (fact)
- Conceptual knowledge (schema)
KG vs Database

- Data in database can be seen as basic facts

<table>
<thead>
<tr>
<th>Student ID</th>
<th>Name</th>
<th>take-course</th>
</tr>
</thead>
<tbody>
<tr>
<td>p001</td>
<td>John</td>
<td>cs3019</td>
</tr>
<tr>
<td>p002</td>
<td>Tom</td>
<td>cs3023</td>
</tr>
</tbody>
</table>

- [csd:p001 rdf:type csd:Student .]
- [csd:p002 rdf:type csd:Student .]
- [csd:p001 csd:name "John" .]
- [csd:p002 csd:name "Tom" .]
- [csd:p001 csd:take-course csd:cs3019 .]
- [csd:p002 csd:take-course csd:cs3023 .]

Schema in a Database System

- A database system includes some schema constraints, such as the foreign key constraint

<table>
<thead>
<tr>
<th>Student ID</th>
<th>Name</th>
<th>take-course</th>
</tr>
</thead>
<tbody>
<tr>
<td>p001</td>
<td>John</td>
<td>cs3015</td>
</tr>
<tr>
<td>p002</td>
<td>Tom</td>
<td>cs3025</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Course ID</th>
<th>Title</th>
<th>coordinator</th>
</tr>
</thead>
<tbody>
<tr>
<td>cs3017</td>
<td>AIS</td>
<td>AS</td>
</tr>
<tr>
<td>cs3025</td>
<td>KBS</td>
<td>JP</td>
</tr>
</tbody>
</table>
Schema in a Knowledge Graph

1) Allow schema constraints, such as DisjointClasses (UndgStudent MastStudent)

<table>
<thead>
<tr>
<th>UndgStudent ID</th>
<th>Name</th>
<th>take-course</th>
</tr>
</thead>
<tbody>
<tr>
<td>csd:p001</td>
<td>John</td>
<td>csd:cs3014</td>
</tr>
<tr>
<td>csd:p002</td>
<td>Tom</td>
<td>csd:cs3025</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MastStudent ID</th>
<th>Name</th>
<th>take-course</th>
</tr>
</thead>
<tbody>
<tr>
<td>csd:p008</td>
<td>Yuan</td>
<td>csd:cs5010</td>
</tr>
<tr>
<td>csd:p002</td>
<td>Tom</td>
<td>csd:cs5017</td>
</tr>
</tbody>
</table>

2) Allow some reasoning based on axioms (open world assumption), such as SubClassOf (MastStudent Student)

<table>
<thead>
<tr>
<th>Student ID</th>
<th>Name</th>
<th>take-course</th>
</tr>
</thead>
<tbody>
<tr>
<td>csd:p001</td>
<td>John</td>
<td>csd:cs3015</td>
</tr>
<tr>
<td>csd:p002</td>
<td>Tom</td>
<td>csd:cs3025</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MastStudent ID</th>
<th>Name</th>
<th>take-course</th>
</tr>
</thead>
<tbody>
<tr>
<td>csd:p008</td>
<td>Yuan</td>
<td>csd:cs5010</td>
</tr>
<tr>
<td>csd:p002</td>
<td>Tom</td>
<td>csd:cs5017</td>
</tr>
</tbody>
</table>

Thus all the students include csd:p001, csd:p002, and csd:p008
Motivations

- Description Logics are the underpinning of the standard Web Ontology Language (OWL)
  - OWL v2 family
    - OWL 2 DL
    - OWL 2 EL, OWL 2 QL, OWL 2 RL
- OWL provides more expressive power than RDF (modern standard of semantic network) for
  - both terminological axioms (TBox)
  - and assertions (ABox)
- RDFa (HTML version of RDF) is e.g. used by schema.org
Lecture Outline

- Motivation
- Overview of Description Logics (DLs)
- Semantics of DLs and Reasoning in DLs
- Practical

Description Logics (DLs)

- Description
  - comes from class description, a formal expression that determines a set of objects with common properties
- Logic
  - semantics of class descriptions can be defined using logic
DLs as KR Languages

- **Formalism**: well defined syntax and formal semantics
- **High-level description**: only relevant aspect represented; others left out
- **Adequate expressive power**: trade-off between expressiveness and complexity
- **Intelligent applications**: must be able to provide reasoning services given requirements from applications
- **Effectively used**: need for scalable and efficient implementations

Syntax

- Provide an explicit symbolic representation of knowledge
- **not** just implicit, as e.g. neural networks
Semantics and Reasoning

• **Declarative semantics**
  – mapping of the symbolic expressions to an abstraction of the “world” (interpretation)
  – allow ones to determine whether a symbolic expression is true in the given world (model)

• Reasoning result should depend only on the semantics and not on the syntactic representation

• Not prededual semantics
  – Should not be defined by how certain programs using the symbolic representation behave

Reasoning Procedures

• The procedure should be a decision procedure for reasoning problems
  – **soundness**: positive answers are correct
  – **completeness**: negative answers are correct
  – **termination**: always give an answer in finite time

• First Order Logic (FOL)
  – Satisfiability does not have a decision procedure
  – Thus FOL is not an appropriate KR formalism

• Propositional Logic (0th order Logic)
  – Satisfiability is NP-complete; however, there are highly optimised SAR solvers
  – Expressive power is not sufficient
Description Logic History

- **Phase 1**: incomplete structural subsumption algorithms
- **Phase 2**: tableau algorithms (with complexity results) and optimisations
  - Systems: Kris, Crack
- **Phase 3**: tableau algorithms for very expressive DLs
  - Systems: FaCT, Racer, HermiT, Konclude, ...
- **Phase 4**: OWL standard (on top of RDF), lightweight language and approximate reasoning
  - Systems: CEL, TrOWL, Ontop, Mastro,...

RDF in Description Logics

- Class assertions C(e)
  - \[ e \text{ rdf:type } C . \]
- Property assertion r(e1,e2)
  - \[ e1 r e2 . \]
- SubClassOf axiom: C1 ⊆ C2
  - \[ C1 \text{ rdfs:subClassOf } C2 . \]
- SubPropertyOf Axiom r1 ⊆ r2
  - \[ r1 \text{ rdfs:SubPropertyOf } r2 . \]
- Property Domain (Range) axioms \( \exists p \subseteq D \) \( (\exists p \subseteq R ) \)
  - \[ p \text{ rdfs:domain } D . \] \( (p \text{ rdfs:range } R .) \)
Lecture Outline

• Motivation
• Overview of Description Logics (DLs)
• Semantics of DLs and Reasoning in DLs
• Practical

ALC: A Basic yet Expressive DL

• ALC: Attributed Language with Complement [Schmidt-Schauss & Smolka, 1991]
  • It is a basic language
    – Reasoning complexity is EXPTIME-complete
  • Naming scheme:
    – foundation language AL
    – can be further extended with constructors whose “letter” can be added after AL
    – C for complement (¬)
    – H for property subsumption (r1 ⊆ r2)
    – I for inverse property (r\textsuperscript{-1})
    – O for one of ({e})
    – Q for number restrictions (≥r.C, ≤r.C, =r.C)
**ALC Syntax**

Let $C$ and $R$ be disjoint sets of concept names and role names, respectively.

$\mathcal{ALC}$-concept descriptions are defined by induction:

- If $A \in C$, then $A$ is an $\mathcal{ALC}$-concept description.
- If $C, D$ are $\mathcal{ALC}$-concept descriptions, and $r \in R$, then the following are $\mathcal{ALC}$-concept descriptions:
  - $C \sqcap D$ (conjunction)
  - $C \sqcup D$ (disjunction)
  - $\neg C$ (negation)
  - $\forall r. C$ (value restriction)
  - $\exists r. C$ (existential restriction)

**Abbreviations:**
- $\top := A \sqcup \neg A$ (top)
- $\bot := A \sqcap \neg A$ (bottom)
- $C \Rightarrow D := \neg C \sqcup D$ (implication)

---

**DL Interpretations**

- An interpretation $I$ is written as $(\Delta^I, \cdot^I)$
  - $\Delta^I$ is the **non-empty domain** (similar to universal set)
  - $\cdot^I$ is the **interpretation function**
    - all individuals (inc. unnamed ones) are members of the domain: $\cdot^I \in \Delta^I$
    - all classes are subsets of the domain $A^I \subseteq \Delta^I$
      - e.g., Employee$^I$ = $\{E1, E2, E3, E4\}$
    - all properties are subsets $R^I \subseteq \Delta^I \times \Delta^I$
      - e.g., Works-for$^I$ = $\{<E1,P1>, <E2,P1>, <E2,P2>, <E3,P1>, <E3,P2>, <E4,P2>\}$
  - Interpretation function allows us to consider all possible assignment of class and property memberships
    - all possible databases for the given schema
Excercise: DL Interpretations

![Knowledge Graph Example]

Knowledge Graphs
Jeff Z. Pan

Excercise: DL Interpretations (II)

- $\Delta' = \{\text{Elvis, Holger, ...}\}$
- Named objects
  - $\text{Elvis}' = \text{Elvis}$
  - $\text{Holger}' = \text{Holger}$
  - ...
- Named classes
  - $\text{Animal}' = \{\text{Flipper, Rudolph}\}$
  - $\text{Person}' = \{\text{Elvis, Holger, Kylie, Hai, S.Claus}\}$
  - $\text{Country}' = \{\text{Belgium, Paraguay, Latvia, China}\}$
- Named properties
  - $\text{has\_pet}' = \{<\text{Hai, Flipper}>, <\text{S.Claus, Rudolph}>\}$
  - $\text{lives\_in}' = \{<\text{Elvis, Belgium}>, <\text{Kylie, Paraguay}>, <\text{Hai, China}>\}$
**ALC Semantics**

An interpretation $I = (\Delta^I, \mathcal{E})$ consists of a non-empty domain $\Delta^I$ and an extension mapping $\mathcal{E}$:

- $A^I \subseteq \Delta^I$ for all $A \in C$.
- $(r^I)^\subseteq \Delta^I \times \Delta^I$ for all $r \in R$.

The extension mapping is extended to complex ALC-concept descriptions as follows:

- $(C \sqcap D)^I := C^I \cap D^I$
- $(C \sqcup D)^I := C^I \cup D^I$
- $(\neg C)^I := \Delta^I \setminus C^I$
- $(\forall r.C)^I := \{ d \in \Delta^I \mid \text{for all } e \in \Delta^I : (d, e) \in r^I \text{ implies } e \in C^I \}$
- $(\exists r.C)^I := \{ d \in \Delta^I \mid \text{there is } e \in \Delta^I : (d, e) \in r^I \text{ and } e \in C^I \}$

---

**Excercise: DL Interpretations (III)**

- Suppose we extend the vocabulary with
  - Young

- Given the following interpretation of Young:
  - Young$^I$ = \{Holger, Hai, Kylie, Flipper\}

- How about the interpretation of the OWL class description?
  - Young$^I$\,Person = \{Holger, Hai, Kylie\}
  - $\exists$has_pet\,Young$^I$ = \{Hai\}
Interpretations Axioms

- Axioms are used to “filter out” invalid interpretations from valid ones
  - An interpretation I is a model for an ontology O if it satisfies all its axioms
  - An ontology O is consistent if it has some model (valid interpretation).
Interpretations of Assertions

- Class assertions
  - An interpretation $I$ satisfies a class assertion $a:C$ if $a' \in C^I$
- Property assertions
  - An interpretation $I$ satisfies a property assertion $<a,b>:_R$ if $<a',b'> \in R^I$

Interpretations of Ontologies

- An ontology $O$ is called **consistent** if there exists (at least) **one** interpretation that satisfies $O$
- A class $C$ is **satisfiable** (w.r.t an ontology $O$) if there exists **one** interpretation $I$ of $O$, such that $C^I$ is not empty
- **Entailment ($|=\$)**: given an axiom $\alpha$, we say an ontology $O$ entails the axiom $\alpha$ if and only if **all** interpretation $I$ of $O$ satisfy $\alpha$. 
Entailments of Axioms

- **Entailment (|=)**: given an axiom $\alpha$, we say an ontology $O$ entails the axiom $\alpha$ if and only if all interpretation $I$ of $O$ satisfy $\alpha$.

Lecture Outline

- **Motivation**: DLs are the underpinning of the standard KG schema language OWL
- **Introduction**: Syntax and semantics of DL
- **Focus**: The ALC DL
- **Exercises (Next time we introduce reasoning in Description Logics)**
  - Formulate ALC concepts:
    - Young pet owner
    - Pet owner only have cats
  - Subsumption checking: are the following statements correct?
    - Young $\cap$ Person is subsumed by Person
    - Young $\cup$ Person is subsumed by Person