Motivation

- Main problem with semantic networks and frames
  - The lack of formal semantics!
  - Disadvantage of simple inheritance networks
  - Concepts are atomic and do not have any structure

→ Brachman’s structural inheritance networks (1977)

Description Logics – Terminology and Notation

Introduction

Concept and Roles

TBox and ABox

Reasoning Services

Outlook

Structural Inheritance Networks

- Concepts are defined/described using a small set of well-defined operators
- Distinction between conceptual and object-related knowledge
- Computation of subconcept relation and of instance relation
- Strict inheritance (of the entire structure of a concept)
Introduction

Systems and Applications

- **Systems:**
  - KL-ONE: First implementation of the ideas (1978)
  - ... then NIKL, KL-TWO, KRYPTON, KANDOR, CLASSIC, BACK, KRIS, YAK, CRACK ...
  - ... currently FaCT, DLP, RACER 1998

- **Applications:**
  - First, natural language understanding systems
  - ... then configuration systems,
  - ... information systems,
  - ... currently, it is one tool for the semantic web

Description Logics

- Previously also *KL-ONE*-alike languages, frame-based languages, terminological logics, concept languages

- **Description Logics (DL)** allow us
  - to describe concepts using complex descriptions,
  - to introduce the terminology of an application and to structure it (TBox),
  - to introduce objects (ABox) and relate them to the introduced terminology,
  - and to reason about the terminology and the objects.

Informal Example

**Male is:** the opposite of female
A human is a kind of: living entity
A woman is: a human and a female
A man is: a human and a male
A mother is: a woman with at least one child that is a human
A father is: a man with at least one child that is a human
A parent is: a mother or a father
A grandmother is: a woman, with at least one child that is a parent
A mother-wod is: a mother with only male children

Possible Questions:
- Is a grandmother a parent?
- Is Diana a parent?
- Is William a man?
- Is Elizabeth a mother-wod?

Concept and Roles

**Concept names:**
- E.g., Grandmother, Male,... (in the following usually capitalized)
- We will use symbols such as \( A, \ldots \)

**Semantics:** Monadic predicates \( A(\cdot) \) or set-theoretically a subset of the universe \( A^I \subseteq D \).

**Role names:**
- In our example, e.g., child. Often we will use names such as has-child or something similar (in the following usually lowercase).
- Role names are disjoint from concept names

**Symbolically:** \( t, t_1, \ldots \)

**Semantics:** Dyadic predicates \( t(\cdot, \cdot) \) or set-theoretically \( t^I \subseteq D \times D \).
Concept and Role Description

- Out of concept and role names, complex descriptions can be created.
- In our example, e.g. “a Human and Female.”
- Symbolically: $C$ for concept descriptions and $r$ for role descriptions.
- Which particular constructs are available depends on the chosen description logic.
- Predicate logic semantics: A concept description $C$ corresponds to a formula $C(x)$ with the free variable $x$. Similarly with $r$: It corresponds to formula $r(x, y)$ with free variables $x, y$.
- Set semantics:
  \[ C^I = \{ d \mid C(d) \text{ is true in } I \} \]
  \[ r^I = \{ (d, e) \mid r(d, e) \text{ is true in } I \} \]

Boolean Operators

- **Syntax**: let $C$ and $D$ be concept descriptions, then the following are also concept descriptions:
  - $C \cap D$ (Concept conjunction)
  - $C \cup D$ (Concept disjunction)
  - $\neg C$ (Concept negation)
- **Examples**:
  - Human $\cap$ Female
  - Father $\cup$ Mother
  - $\neg$ Female
- Predicate logic semantics: $C(x) \land D(x), C(x) \lor D(x), \neg C(x)$
- Set semantics: $C^I \cap D^I, C^I \cup D^I, D - C^I$

Cardinality Restriction

- **Motivation**:
  - Often we want to describe something by restricting the possible “fillers” of a role, e.g. Mother-wod.
  - Sometimes we want to say that there is at least a filler of a particular type, e.g. Grandmother.
- **Idea**: Use quantifiers that range over the role-fillers
  - $\text{Mother} \cap \forall \text{has-child.Man}$
  - $\text{Woman} \cap \exists \text{has-child.Parent}$
- **Predicate logic semantics**:
  \[
  (\exists r.C)(x) = \exists y : (r(x, y) \land C(y))
  \\
  (\forall r.C)(x) = \forall y : (r(x, y) \rightarrow C(y))
  \]
- Set semantics:
  \[
  (\exists r.C)^I = \{ d \mid \exists e : (d, e) \in r^I \land e \in C^I \} \\
  (\forall r.C)^I = \{ d \mid \forall e : (d, e) \in r^I \rightarrow e \in C^I \}
  \]
Inverse Roles

- **Motivation:**
  - How can we describe the concept "children of rich parents"?
- **Idea:** Define the "inverse" role for a given role (the converse relation)
  - has-child\(^{-1}\)
- **Application:** \(\exists \text{has-child}^{-1}.\text{Rich}\)
- **Predicate logic semantics:**
  \[ r^{-1}(x, y) = r(y, x) \]
- **Set semantics:**
  \[ (r^{-1})^T = \{(d, e) \mid (e, d) \in r^T\} \]

Role Composition

- **Motivation:**
  - How can we define the role has-grandchild given the role has-child?
- **Idea:** Compose roles (as one can compose binary relations)
  - has-child \(\circ\) has-child
- **Predicate logic semantics:**
  \[ (r \circ s)(x, y) = \exists z : (r(x, z) \land s(z, y)) \]
- **Set semantics:**
  \[ (r \circ s)^T = \{(d, e) \mid \exists f : (d, f) \in r^T \land (f, e) \in s^T\} \]

Role Value Maps

- **Motivation:**
  - How do we express the concept "women, who know all the friends of their children"?
- **Idea:** Relate role filler sets to each other
  - Woman \(\cap\) \{(has-child \(\circ\) has-friend \(\sqsubseteq\) knows\}
- **Predicate logic semantics:**
  \[ (r \sqsubseteq s)(x) = \forall y : (r(x, y) \rightarrow s(x, y)) \]
- **Set semantics:** Let \(r^T(d) = \{e \mid r^T(d, e)\}\).
  \[ (r \sqsubseteq s)^T = \{d \mid r^T(d) \subseteq s^T(d)\} \]
- **Note:** Role value maps lead to undecidability of satisfiability of concept descriptions!

Terminology Box

- In order to introduce new terms, we use two kinds of terminological axioms:
  - \(A \sqsubseteq C\)
  - \(A \sqsubseteq C\)
  where \(A\) is a concept name and \(C\) is a concept description.
- A terminology or TBox is a finite set of such axioms with the following additional restrictions:
  - no multiple definitions of the same symbol such as \(A \sqsubseteq C, A \sqsubseteq D\)
  - no cyclic definitions (even not indirectly), such as \(A \sqsubseteq \forall r.B, B \sqsubseteq \exists s.A\)
**TBoxes: Semantics**

- TBoxes restrict the set of possible interpretations.
  - **Predicate logic semantics:**
    - \(A \equiv C\) corresponds to \(\forall x : (A(x) \iff C(x))\)
    - \(A \subseteq C\) corresponds to \(\forall x : (A(x) \rightarrow C(x))\)
  - **Set semantics:**
    - \(A \equiv C\) corresponds to \(A^I = C^I\)
    - \(A \subseteq C\) corresponds to \(A^I \subseteq C^I\)
  - Non-empty interpretations which satisfy all terminological axioms are called **models** of the TBox.

**ABoxes: Semantics**

- **Individual names** are interpreted as elements of the universe under the **unique-name-assumption**, i.e., different names refer to different objects.
- **Assertions** express that an object is an instance of a concept or that two objects are related by a role.
  - **Predicate logic semantics:**
    - \(a : C\) corresponds to \(C(a)\)
    - \((a, b) : r\) corresponds to \(r(a, b)\)
  - **Set semantics:**
    - \(a^I \in D\)
    - \(a : C\) corresponds to \(a^I \in C^I\)
    - \((a, b) : r\) corresponds to \((a^I, b^I) \in r^I\)
- **Models** of an ABox and of ABox+TBox can be defined analogously to models of a TBox.

**Assertional Box**

- In order to state something about objects in the world, we use two forms of **assertions**:
  - \(a : C\)
  - \((a, b) : r\)
  - where \(a\) and \(b\) are **individual names** (e.g., ELIZABETH, PHILIP), \(C\) is a **concept description**, and \(r\) is a **role description**.
- An **ABox** is a finite set of assertions.

**Example TBox**

- Male \(\equiv \neg\text{Female}\)
- Human \(\subseteq\) Living_entity
- Woman \(\equiv\) Human \(\cap\) Female
- Man \(\equiv\) Human \(\cap\) Male
- Mother \(\equiv\) Woman \(\cap\) has-child.Human
- Father \(\equiv\) Man \(\cap\) has-child.Human
- Parent \(\equiv\) Father \(\cup\) Mother
- Grandmother \(\equiv\) Woman \(\cap\) has-child.Parent
- Mother-without-daughter \(\equiv\) Mother \(\cap\) has-child.Male
- Mother-with-many-children \(\equiv\) Mother \(\cap\) (\(\geq 3\) has-child)
Example ABox

CHARLES: Man  DIANA: Woman
EDWARD: Man  ELIZABETH: Woman
ANDREW: Man
DIANA: Mother-without-daughter
(ELIZABETH, CHARLES): has-child
(ELIZABETH, EDWARD): has-child
(ELIZABETH, ANDREW): has-child
(DIANA, WILLIAM): has-child
(CHARLES, WILLIAM): has-child

Some Reasoning Services

- Does a description $C$ make sense at all, i.e., is it satisfiable?
  $\iff$ A concept description $C$ is satisfiable iff there exists an interpretation $I$ such that $C^I \neq \emptyset$.

- Is one concept a specialization of another one, is it subsumed?
  $\iff$ $C$ is subsumed by $D$, in symbols $C \subseteq D$ iff we have for all interpretations $C^I \subseteq D^I$.

- Is $a$ an instance of a concept $C$?
  $\iff$ $a$ is an instance of $C$ iff for all interpretations, we have $a^I \in C^I$.

$\textbf{Note:}$ These questions can be posed with or without a TBox that restricts the possible interpretations.

Outlook

- Can we reduce the reasoning services to perhaps just one problem?
- What could be reasoning algorithms?
- What about complexity and decidability?
- What has all that to do with modal logics?
- How can one build efficient systems?

Literature


### Summary: Concept Descriptions

<table>
<thead>
<tr>
<th>Abstract</th>
<th>Concrete</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A \cap D )</td>
<td>( A \cap D )</td>
<td>( D^2 \cap D^2 )</td>
</tr>
<tr>
<td>( C \cup D )</td>
<td>( C \cup D )</td>
<td>( C^2 \cup D^2 )</td>
</tr>
<tr>
<td>( -C )</td>
<td>( -C )</td>
<td>( D - C^2 )</td>
</tr>
<tr>
<td>( \exists r \ C )</td>
<td>( \exists r \ C )</td>
<td>( { d \in D \mid r^2(d) \subseteq C^2 } )</td>
</tr>
<tr>
<td>( \geq n r \ C )</td>
<td>( \geq n r \ C )</td>
<td>( { d \in D \mid r^2(d) \geq n } )</td>
</tr>
<tr>
<td>( \leq n r \ C )</td>
<td>( \leq n r \ C )</td>
<td>( { d \in D \mid r^2(d) \leq n } )</td>
</tr>
<tr>
<td>( \not \equiv s )</td>
<td>( \not \equiv s )</td>
<td>( { d \in D \mid r^2(d) \neq s^2(d) } )</td>
</tr>
<tr>
<td>( r \subseteq s )</td>
<td>( r \subseteq s )</td>
<td>( { d \in D \mid r^2(d) \subseteq s^2(d) } )</td>
</tr>
<tr>
<td>( g = h )</td>
<td>( g = h )</td>
<td>( { d \in D \mid g^2(d) = h^2(d) } )</td>
</tr>
<tr>
<td>( f(1, 2, \ldots, 2) )</td>
<td>( f(1, 2, \ldots, 2) )</td>
<td>( { d \in D^2 \mid \exists } )</td>
</tr>
</tbody>
</table>

### Summary: Role Descriptions

<table>
<thead>
<tr>
<th>Abstract</th>
<th>Concrete</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f )</td>
<td>( f )</td>
<td>( f^2 ) (functional role)</td>
</tr>
<tr>
<td>( r \cap s )</td>
<td>( r \cap s )</td>
<td>( r^2 \cap s^2 )</td>
</tr>
<tr>
<td>( r \cup s )</td>
<td>( r \cup s )</td>
<td>( r^2 \cup s^2 )</td>
</tr>
<tr>
<td>( r^{-1} )</td>
<td>( r^{-1} )</td>
<td>( D \times D - r^2 )</td>
</tr>
<tr>
<td>( \text{rest} \ r )</td>
<td>( \text{rest} \ r )</td>
<td>( { (d, d') \mid (d', d) \in r } )</td>
</tr>
<tr>
<td>( \text{trans} \ r )</td>
<td>( \text{trans} \ r )</td>
<td>( { (d, d') \in r^2 \mid d' \in C^2 } )</td>
</tr>
<tr>
<td>( \text{compose} \ r )</td>
<td>( \text{compose} \ r )</td>
<td>( r^2 \circ s )</td>
</tr>
<tr>
<td>( \text{self} )</td>
<td>( \text{self} )</td>
<td>( { (d, d) \mid d \in D } )</td>
</tr>
</tbody>
</table>