Introduction

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)

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
  - DAML+OIL, now OWL

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

Elizabeth is a woman
Elizabeth has the child Charles
Charles is a man
Diana is a mother-wod
Diana has the child William

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

Atomic Concepts and Roles

- **Concept names:**
  - E.g., Grandmother, Male, ...(in the following usually capitalized)
  - We will use symbols such as A, A₁, ...
  - Semantics: Monadic predicates A(·) 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, t₁, ...
  - Semantics: Dyadic predicates t(·, ·) 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 descriptions $C$ 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:

\[
C^I = \{ d \mid C(d) \text{ “is true in” } \mathcal{I} \}
\]

\[
r^I = \{ (d,e) \mid r(d,e) \text{ “is true in” } \mathcal{I} \}
\]

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 $\sqcup$ Mother
- $\neg$ Female

- Predicate logic semantics: $C(x) \land D(x)$, $C(x) \lor D(x)$, $\neg C(x)$

Set semantics: $C^I \cap D^I$, $C^I \cup D^I$, $D - C^I$

Role Restrictions

- Motivation:
  - Often we want to describe something by restricting the possible “fillers” of a role, e.g. Mother-vod.
  - 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
  - $\exists r.C$ has-child $\exists r.C$ Father $\exists r.C$ Woman

- 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:

\[
(\exists r.C)^I = \{ d \mid \exists e : (d, e) \in r^I \land e \in C^I \}
\]

\[
(\forall r.C)^I = \{ d \mid \forall e : (d, e) \in r^I \rightarrow e \in C^I \}
\]

Cardinality Restriction

- Motivation:
  - Often we want to describe something by restricting the number of possible “fillers” of a role, e.g., a Mother with at least 3 children or at most 2 children.

- Idea: We restrict the cardinality of the role filler sets:
  - $\exists r.C$ has-child $\geq 3$
  - $\exists r.C$ has-child $\leq 2$

- Predicate logic semantics:

\[
(\geq n r)(x) = \exists y_1 \ldots y_n : (r(x, y_1) \land \ldots \land r(x, y_n) \land y_1 \neq y_2 \land \ldots \land y_{n-1} \neq y_n)
\]

\[
(\leq n r)(x) = \neg (\exists y_1 \ldots y_n : (r(x, y_1) \land \ldots \land r(x, y_n) \land y_1 \neq y_2 \land \ldots \land y_{n-1} \neq y_n))
\]

Set semantics:

\[
(\geq n r)^I = \{ d \mid |\{ e \mid \neg e^I(d, e) \} | \geq n \}
\]

\[
(\leq n r)^I = D - (\geq n r)^I
\]
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:**
  \((r^{-1})^T = \{(d, e) | (e, d) \in r^T\}\)

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:**
  \((r \circ s)^T = \{(d, e) | \exists f : (d, f) \in r^T \land (f, e) \in s^T\}\)

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 \(r^T(d) = \{e | r^T(d, e)\}\).
  \((r \sqsubseteq s)^T = \{d | r^T(d) \subseteq s^T(d)\}\)

- **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*:
  - \(A \equiv C\)
  - \(A \sqsubseteq C\)

  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 \(A \equiv C\), \(A \sqsubseteq D\)
  - no cyclic definitions (even not indirectly), such as \(A \equiv \forall r.B, B \equiv \exists s.A\)
**TBox and ABox Terminology Box**

**TBoxes: Semantics**

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

**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:**
  - \( a^T \in D \)
  - \( a : C \) corresponds to \( a^T \in C^T \)
  - \( (a, b) : r \) corresponds to \( (a^T, b^T) \in r^T \)
- **Models** of an ABox and of ABox+TBox can be defined analogously to models of a TBox.

**Example TBox**

- Male \( \equiv \neg \text{Female} \)
- Human \( \sqsubseteq \text{Living entity} \)
- Woman \( \equiv \text{Human} \sqcap \text{Female} \)
- Man \( \equiv \text{Human} \sqcap \text{Male} \)
- Mother \( \equiv \text{Woman} \sqcap \exists \text{has-child.Human} \)
- Father \( \equiv \text{Man} \sqcap \exists \text{has-child.Human} \)
- Parent \( \equiv \text{Father} \sqcup \text{Mother} \)
- Grandmother \( \equiv \text{Woman} \sqcap \exists \text{has-child.Parent} \)
- Mother-without-daughter \( \equiv \text{Mother} \sqcap \neg \text{has-child.Male} \)
- Mother-with-many-children \( \equiv \text{Mother} \sqcap (\geq 3 \text{has-child}) \)
### 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?
- 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?
- $C$ 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 $C$?
- $a$ is an instance of $C$ iff for all interpretations, we have $a^I \in C^I$.
- 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</td>
<td>A</td>
<td>A²</td>
</tr>
<tr>
<td>C ∩ D</td>
<td>(and C D)</td>
<td>C² ∩ D²</td>
</tr>
<tr>
<td>C ∪ D</td>
<td>(or C D)</td>
<td>C² ∪ D²</td>
</tr>
<tr>
<td>¬C</td>
<td>(not C)</td>
<td>D - C²</td>
</tr>
<tr>
<td>∀r.C</td>
<td>(all r C)</td>
<td>{d ∈ D : r²(d) ⊆ C²}</td>
</tr>
<tr>
<td>∃r.C</td>
<td>(some r C)</td>
<td>{d ∈ D : r²(d) ≠ ∅}</td>
</tr>
<tr>
<td>≥ n r</td>
<td>(atleast n r)</td>
<td>{d ∈ D :</td>
</tr>
<tr>
<td>≤ n r</td>
<td>(atmost n r)</td>
<td>{d ∈ D :</td>
</tr>
<tr>
<td>∃r.C</td>
<td>(some r C)</td>
<td>{d ∈ D : r²(d) ∩ C² ≠ ∅}</td>
</tr>
<tr>
<td>≥ n r.C</td>
<td>(atleast n r C)</td>
<td>{d ∈ D :</td>
</tr>
<tr>
<td>≤ n r.C</td>
<td>(atmost n r C)</td>
<td>{d ∈ D :</td>
</tr>
<tr>
<td>r = s</td>
<td>(eq r s)</td>
<td>{d ∈ D : r²(d) = s²(d)}</td>
</tr>
<tr>
<td>r ≠ s</td>
<td>(neq r s)</td>
<td>{d ∈ D : r²(d) ≠ s²(d)}</td>
</tr>
<tr>
<td>r ⊆ s</td>
<td>(subset r s)</td>
<td>{d ∈ D : r²(d) ⊆ s²(d)}</td>
</tr>
<tr>
<td>g = h</td>
<td>(eq g h)</td>
<td>{d ∈ D : g²(d) = h²(d)}</td>
</tr>
<tr>
<td>g ≠ h</td>
<td>(neq g h)</td>
<td>{d ∈ D : g²(d) ≠ h²(d)}</td>
</tr>
<tr>
<td>{q₁, q₂, . . . , qₙ}</td>
<td>(oneof q₁ . . . qₙ)</td>
<td>{q₁², q₂², . . . , qₙ²}</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>t</td>
<td>f²(, functional role)</td>
</tr>
<tr>
<td>r ∩ a</td>
<td>(and r a)</td>
<td>r² ∩ a²</td>
</tr>
<tr>
<td>r ∪ a</td>
<td>(or r a)</td>
<td>r² ∪ a²</td>
</tr>
<tr>
<td>¬r</td>
<td>(not r)</td>
<td>D × D - r²</td>
</tr>
<tr>
<td>r⁻¹</td>
<td>(inverse r)</td>
<td>{(d, d') : (d', d) ∈ r²}</td>
</tr>
<tr>
<td>r</td>
<td>C</td>
<td>(restr r C)</td>
</tr>
<tr>
<td>r°</td>
<td>(trans r)</td>
<td>r²</td>
</tr>
<tr>
<td>r ∘ s</td>
<td>(compose r s)</td>
<td>r² ∩ s²</td>
</tr>
<tr>
<td>1</td>
<td>self</td>
<td>{(d, d) : d ∈ D}</td>
</tr>
</tbody>
</table>