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)

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): inherited properties cannot be overridden
Description Logics

- Previously also known as 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 and relate them to the introduced terminology (ABox),
  - 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
Possible Questions:
Charles
- Is a grandmother a parent?
- Is Diana a parent?
- Is William a man?
- Is Elizabeth a mother-wod?

Diana is a mother-wod
Diana has the child William
Atomic concepts and roles

- **Concept names:**
  - E.g., Grandmother, Male, ...(in the following usually capitalized)
  - We will use symbols such as A1, A2, ... for concept names
- **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:** $\mathcal{t}, \mathcal{t}1, \ldots$
  - **Semantics:** Binary relations $t(\cdot, \cdot)$ or set-theoretically $t^I \subseteq D \times D$.

Boolean operators

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

Concept and role description

- From (atomic) concept and role names, complex concept and role descriptions can be created
- In our example, e.g., “Human and Female.”
  - **Symbolically:** $C$ for concept descriptions and $r$ for role descriptions
- Which particular constructs are available depends on the chosen description logic!
  - **FOL semantics:** A concept description $C$ corresponds to a formula $C(x)$ with the free variable $x$.
    Similarly with role descriptions $r$: they correspond to formulae $r(x,y)$ with free variables $x,y$.
- **Set semantics:**
  - $C^I = \{d \in D : C(d) \text{ is true in } I\}$
  - $r^I = \{(d,e) \in D^2 : r(d,e) \text{ is true in } I\}$

Role restrictions

- **Motivation:**
  - Often we want to describe something by restricting the possible “fillers” of a role, e.g. Mother $\land$ father.
  - 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
  - Mother $\exists y \text{ has-child } \text{ Man}$
  - Woman $\exists y \text{ has-child } \text{ Parent}$
- **FOL 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:**
  - $(\exists r.C)^I = \{d \in D : \text{ there ex. some } e \text{ s.t. } (d,e) \in r^I \land e \in C^I\}$
  - $(\forall r.C)^I = \{d \in D : \text{ for each } e \text{ with } (d,e) \in r^I, e \in C^I\}$
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:
  - Mother $\cap \geq 3$ has-child
  - Mother $\cap \leq 2$ has-child
- **FOL semantics:**
  \[
  (\geq n \ r)(x) = \exists y_1 \ldots y_n (r(x,y_1) \land \ldots \land r(x,y_n) \land y_1 \neq y_2 \land \ldots \land y_{n-1} \neq y_n)
  \]
  \[
  (\leq n \ r)(x) = \neg (\geq n+1 \ r)(x)
  \]
- **Set semantics:**
  \[
  (\geq n \ r)^T = \{d \in D : |\{e \in D : r^T(d,e)\}| \geq n\}
  \]
  \[
  (\leq n \ r)^T = D \setminus (\geq n+1 \ r)^T
  \]

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$^{-1}$
- **Example:** $\exists$ has-child$^{-1}$ . Rich
- **FOL semantics:**
  \[
  r^{-1}(x,y) = r(y,x)
  \]
- **Set semantics:**
  \[
  (r^{-1})^T = \{(d,e) \in D^2 : (e,d) \in r^T\}
  \]

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)
  - $\text{has-child} \circ \text{has-child}$
- **FOL semantics:**
  \[
  (r \circ s)(x,y) = \exists z (r(x,z) \land s(z,y))
  \]
- **Set semantics:**
  \[
  (r \circ s)^T = \{(d,e) \in D^2 : \exists f \text{ s.t. } (d,f) \in r^T \land (f,e) \in s^T\}
  \]

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)
- **FOL semantics:**
  \[
  (r \sqsubseteq s)(x) = \forall y (r(x,y) \rightarrow s(x,y))
  \]
- **Set semantics:** Let $r^T(d) = \{e : r^T(d,e)\}$.
  \[
  (r \sqsubseteq s)^T = \{d \in D : r^T(d) \subseteq s^T(d)\}
  \]
- **Note:** Role value maps lead to undecidability of satisfiability testing of concept descriptions!
3 TBox and ABox

- Terminology Box
- Assertional Box
- Example

Terminology box

- In order to introduce new terms, we use two kinds of terminological axioms:
  - $A \equiv C$
  - $A \subseteq C$

  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 $A \sqsubseteq \forall r.B$,
    $B \sqsubseteq \exists s.A$

TBoxes: semantics

- TBoxes restrict the set of possible interpretations.

- FOL semantics:
  - $A \equiv C$ corresponds to $\forall x (A(x) \leftrightarrow C(x))$
  - $A \sqsubseteq C$ corresponds to $\forall x (A(x) \rightarrow C(x))$

- Set semantics:
  - $A \equiv C$ corresponds to $A^I = C^I$
  - $A \sqsubseteq C$ corresponds to $A^I \subseteq C^I$

- 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.
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.

FOL 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.
Some reasoning services

- Does a description $C$ make sense at all, i.e., is it satisfiable? A concept description $C$ is satisfiable, if 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 $C \sqsubseteq D$) if we have for all interpretations $C^I \subseteq D^I$.
- Is $a$ an instance of a concept $C$? $a$ is an instance of $C$ if 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.

5 Outlook

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

Bernhard Nebel.  
**Reasoning and Revision in Hybrid Representation Systems.**  

---

## 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>$A^\mathcal{C}$</td>
</tr>
<tr>
<td>$A \cap B$</td>
<td>$A \cap B^\mathcal{C}$</td>
<td>$A \cap B^\mathcal{C}$</td>
</tr>
<tr>
<td>$A \cup B$</td>
<td>$A \cup B^\mathcal{C}$</td>
<td>$A \cup B^\mathcal{C}$</td>
</tr>
<tr>
<td>$\neg C$</td>
<td>$\neg C^\mathcal{C}$</td>
<td>$\neg C^\mathcal{C}$</td>
</tr>
<tr>
<td>$\forall C$</td>
<td>$\forall C^\mathcal{C}$</td>
<td>$\forall C^\mathcal{C}$</td>
</tr>
<tr>
<td>$\exists r$</td>
<td>$\exists r^\mathcal{C}$</td>
<td>$\exists r^\mathcal{C}$</td>
</tr>
<tr>
<td>$\leq n$</td>
<td>$\leq n^\mathcal{C}$</td>
<td>$\leq n^\mathcal{C}$</td>
</tr>
<tr>
<td>$r = s$</td>
<td>$r = s^\mathcal{C}$</td>
<td>$r = s^\mathcal{C}$</td>
</tr>
<tr>
<td>$r \neq s$</td>
<td>$r \neq s^\mathcal{C}$</td>
<td>$r \neq s^\mathcal{C}$</td>
</tr>
<tr>
<td>$r \subseteq s$</td>
<td>$r \subseteq s^\mathcal{C}$</td>
<td>$r \subseteq s^\mathcal{C}$</td>
</tr>
<tr>
<td>$g = h$</td>
<td>$g = h^\mathcal{C}$</td>
<td>$g = h^\mathcal{C}$</td>
</tr>
<tr>
<td>${i_1, \ldots, i_n}$</td>
<td>${i_1, \ldots, i_n}^\mathcal{C}$</td>
<td>${i_1, \ldots, i_n}^\mathcal{C}$</td>
</tr>
</tbody>
</table>

---

## Summary: Role Descriptions

<table>
<thead>
<tr>
<th>Abstract</th>
<th>Concrete</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\top$</td>
<td>$\top$</td>
<td>$\top^\mathcal{C}$</td>
</tr>
<tr>
<td>$\bot$</td>
<td>$\bot$</td>
<td>$\bot^\mathcal{C}$</td>
</tr>
<tr>
<td>$r \cap s$</td>
<td>$r \cap s^\mathcal{C}$</td>
<td>$r \cap s^\mathcal{C}$</td>
</tr>
<tr>
<td>$r ; s$</td>
<td>$r ; s^\mathcal{C}$</td>
<td>$r ; s^\mathcal{C}$</td>
</tr>
<tr>
<td>$r^{-1}$</td>
<td>$r^{-1}^\mathcal{C}$</td>
<td>$r^{-1}^\mathcal{C}$</td>
</tr>
<tr>
<td>$\text{transitive}$</td>
<td>$\forall r (r \circ r \circ r)$</td>
<td>$\forall r (r \circ r \circ r)$</td>
</tr>
<tr>
<td>$\text{reflexive}$</td>
<td>$\forall r (r \circ r)$</td>
<td>$\forall r (r \circ r)$</td>
</tr>
<tr>
<td>$\text{functional}$</td>
<td>$\forall r (\forall d_1 \forall d_2 (r(d_1) \land r(d_2) \rightarrow d_1 = d_2))$</td>
<td>$\forall r (\forall d_1 \forall d_2 (r(d_1) \land r(d_2) \rightarrow d_1 = d_2))$</td>
</tr>
<tr>
<td>$\text{reflexive and functional}$</td>
<td>$\forall r (\forall d_1 \forall d_2 (r(d_1) \land r(d_2) \rightarrow d_1 = d_2))$</td>
<td>$\forall r (\forall d_1 \forall d_2 (r(d_1) \land r(d_2) \rightarrow d_1 = d_2))$</td>
</tr>
<tr>
<td>$\text{left-total}$</td>
<td>$\forall d_1 \exists d_2 (r(d_1) \land r(d_2))$</td>
<td>$\forall d_1 \exists d_2 (r(d_1) \land r(d_2))$</td>
</tr>
<tr>
<td>$\text{right-total}$</td>
<td>$\forall d_1 \exists d_2 (r(d_1) \land r(d_2))$</td>
<td>$\forall d_1 \exists d_2 (r(d_1) \land r(d_2))$</td>
</tr>
<tr>
<td>$\text{left-functional}$</td>
<td>$\forall d_1 \forall d_2 (r(d_1) \land r(d_2) \rightarrow d_1 = d_2)$</td>
<td>$\forall d_1 \forall d_2 (r(d_1) \land r(d_2) \rightarrow d_1 = d_2)$</td>
</tr>
<tr>
<td>$\text{right-functional}$</td>
<td>$\forall d_1 \forall d_2 (r(d_1) \land r(d_2) \rightarrow d_1 = d_2)$</td>
<td>$\forall d_1 \forall d_2 (r(d_1) \land r(d_2) \rightarrow d_1 = d_2)$</td>
</tr>
</tbody>
</table>

---

References: