Principles of Knowledge Representation and Reasoning

5. Semantic Networks and Description Logics

5.3 Description Logics – Terminology and Notation

Bernhard Nebel

- Introduction
- Concepts and Roles
- TBoxes and ABoxes
- Reasoning Services
- Outlook
Motivation

• Main problem with semantic networks and frames

⇒ The lack of formal semantics!
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
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
Structural Inheritance Networks

- Concepts are defined/described using a small set of well-defined operators.

- Distinction between conceptual and object-related knowledge.
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*
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)
Systems and Applications

- **Systems:**
  - **KL-ONE**: First implementation of the ideas (1978)
• **Systems:**
  
  - **KL-ONE**: First implementation of the ideas (1978)
  - ... then **NIKL, KL-TWO, KRYPTON, KANDOR, CLASSIC, BACK, KRIS, YAK, CRACK** ...
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
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
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,
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,
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*
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
Description Logics

- Previously also *KL-ONE-alike languages, frame-based languages, terminological logics, concept languages*
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*,
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*),
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,
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.
Male is: the opposite of female
A human is a kind of: living entity
A woman is: a human and female
A man is: a human and 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
Informal Example

Male is: the opposite of female
A human is a kind of: living entity
A woman is: a human and female
A man is: a human and 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
Informal Example

Male is: the opposite of female
A human is a kind of: living entity
A woman is: a human and female
A man is: a human and 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
Informal Example

**Male is:** the opposite of female

A human is a kind of: living entity

A woman is: a human and female

A man is: a human and 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?
Informal Example

Male is: the opposite of female
A human is a kind of: living entity
A woman is: a human and female
A man is: a human and 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?
Informal Example

**Male is:** the opposite of female

A **human** is a kind of: living entity

A **woman** is: a human and female

A **man** is: a human and 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?
Informal Example

Male is: the opposite of female
A human is a kind of: living entity
A woman is: a human and female
A man is: a human and 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:**
  - In our example, e.g., Grandmother, Male, ...(usually *capitalized* names)
  - We will use **symbols** such as $A, A_1, \ldots$
  - **Semantics:** Monadic predicates $A(\cdot)$ or set-theoretically a subset of the universe $A^I \subseteq \mathcal{D}$. 
Atomic Concepts and Roles

- **Concept names:**
  - In our example, e.g., Grandmother, Male, ...(usually *capitalized* names)
  - We will use *symbols* such as $A, A_1, \ldots$
  - **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 (usually *lowercase* names).
  - 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
Out of concept and role names, complex descriptions can be created.

In our example, e.g. “a human and female.”
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.
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.
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 $x, y$. 
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:**
  
  \[
  C^I = \{d \mid C(d) \text{ “is true in” } I\}
  \]
  \[
  r^I = \{(d,e) \mid r(d,e) \text{ “is true in” } I\}
  \]
Boolean Operators

• **Syntax**: let $C$ and $D$ be concept descriptions, then the following are also concept descriptions:
  - $C \cap D$ (Concept conjunction)
  - $C \sqcup D$ (Concept disjunction)
  - $\neg C$ (Concept negation)
Boolean Operators

- **Syntax**: let $C$ and $D$ be concept descriptions, then the following are also concept descriptions:
  - $C \sqcap D$ (Concept conjunction)
  - $C \sqcup D$ (Concept disjunction)
  - $\neg C$ (Concept negation)

- **Examples**:
  - human $\sqcap$ female
  - father $\sqcup$ mother
  - $\neg$ female
Boolean Operators

- **Syntax**: let $C$ and $D$ be concept descriptions, then the following are also concept descriptions:
  - $C \sqcap D$ (Concept conjunction)
  - $C \sqcup D$ (Concept disjunction)
  - $\neg C$ (Concept negation)

- **Examples**:
  - human $\sqcap$ female
  - father $\sqcup$ mother
  - $\neg$ female

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

- **Syntax**: let $C$ and $D$ be concept descriptions, then the following are also concept descriptions:
  - $C \sqcap D$ (Concept conjunction)
  - $C \sqcup D$ (Concept disjunction)
  - $\neg C$ (Concept negation)

- **Examples**:
  - human $\sqcap$ female
  - father $\sqcup$ mother
  - $\neg$ female

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

- **Set semantics**: $C^\mathcal{I} \cap D^\mathcal{I}$, $C^\mathcal{I} \cup D^\mathcal{I}$, $\mathcal{D} - C^\mathcal{I}$
Role Restrictions

- **Motivation:**
  - Often we want to describe something by *restricting* the possible “fillers” of a role, e.g. *mother-wod.*
Role Restrictions

- **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*
Role Restrictions

- **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
  - Mother $\sqsubseteq \forall \text{has-child}.\text{Man}$
  - Woman $\sqsubseteq \exists \text{has-child}.\text{Parent}$
Role Restrictions

- **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
  - Mother $\sqcap \forall \text{has-child. Man}$
  - Woman $\sqcap \exists \text{has-child. Parent}$

- **Predicate logic semantics:**
  \[
  (\exists r.C)(x) = \exists y : (r(x, y) \land C(y)) \quad (\forall r.C)(x) = \forall y : (r(x, y) \rightarrow C(y))
  \]
Role Restrictions

• **Motivation:**
  o Often we want to describe something by *restricting* the possible “fillers” of a role, e.g. mother → wod.
  o 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
  o Mother \( \sqcap \forall \text{has-child.Man} \)
  o Woman \( \sqcap \exists \text{has-child.Parent} \)

• **Predicate logic semantics:**

\[
(\exists r.C)(x) = \exists y : (r(x, y) \land C(y)) \quad (\forall r.C)(x) = \forall y : (r(x, y) \to C(y))
\]

• **Set semantics:**

\[
(\exists r.C)^I = \{d | \exists e : (d, e) \in r^I \land e \in C^I\} \quad (\forall r.C)^I = \{d | \forall e : (d, e) \in r^I \to e \in C^I\}
\]
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.
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 $\cap (\geq 3 \text{has-child})$
  - Mother $\cap (\leq 2 \text{has-child})$
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 $\sqcap (\geq 3\text{has-child})$
  - Mother $\sqcap (\leq 2\text{has-child})$

- **Predicate logic semantics:**
  
  \[
  (\geq n\ r)(x) = \exists y_1 \ldots y_n : \left( r(x, y_1) \land \ldots \land r(x, y_n) \land y_1 \neq y_2 \land \ldots \land y_{n-1} \neq y_n \right)
  \]
  
  \[
  (\leq n\ r)(x) = \neg (\geq n + 1\ r)(x)
  \]
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 $\cap (\geq 3\text{has-child})$
  - Mother $\cap (\leq 2\text{has-child})$

- **Predicate logic semantics:**
  - $(\geq n \ r)(x) = \exists y_1 \ldots y_n : (r(x, y_1) \wedge \ldots \wedge r(x, y_n) \wedge y_1 \neq y_2 \wedge \ldots \wedge y_{n-1} \neq y_n)$
  - $(\leq n \ r)(x) = \neg (\geq n + 1 \ r)(x)$

- **Set semantics:**
  - $(\geq n \ r)^I = \{d \mid |\{e \mid r^I(d, e)\}| \geq n\}$
  - $(\leq n \ r)^I = \mathcal{D} - (\geq n + 1 \ r)^I$
Inverse Roles

• **Motivation:**
  - How can we describe the concept “children of rich parents”?
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)
  - $\text{has-child}^{-1}$
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:** $\exists$has-child$^{-1}$.Rich
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:**

\[
r^{-1}(x, y) = r(y, x)
\]
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:** $\exists$ has-child$^{-1}$. Rich

- **Predicate logic semantics:**
  $$r^{-1}(x, y) = r(y, x)$$

- **Set semantics:**
  $$\overline{(r^{-1})} = \{(d, e) | (e, d) \in r\}$$
Role Composition

- **Motivation:**
  - How can we define the role `has-grandchild` given the role `has-child`?
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` ◦ `has-child`
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 ∘ has-child`

- **Predicate logic semantics:**
  \[(r ∘ s)(x, y) = \exists z : (r(x, z) \land s(z, y))\]
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 ∘ has-child

- **Predicate logic semantics:**
  \[
  (r \circ s)(x, y) = \exists z : (r(x, z) \land s(z, y))
  \]

- **Set semantics:**
  \[
  (r \circ s)^I = \{ (d, e) | \exists f : (d, f) \in r^I \land (f, e) \in s^I \}\]
Role Value Maps

- **Motivation:**
  - How do we express the concept “women, who know all the friends of their children”
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 \(\sqcap (\text{has-child} \circ \text{has-friend} \sqsubseteq \text{knows})\)
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 $\sqcap (\text{has-child} \circ \text{has-friend} \sqsubseteq \text{knows})$

- **Predicate logic semantics:**
  \[
  (r \sqsubseteq s)(x) = \forall y: \left( r(x, y) \rightarrow s(x, y) \right)
  \]
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 \( \sqcap \left( \text{has-child} \circ \text{has-friend} \sqsubseteq \text{knows} \right) \)

- **Predicate logic semantics:**
  \[
  (r \sqsubseteq s)(x) = \forall y : \left( r(x, y) \rightarrow s(x, y) \right)
  \]

- **Set semantics:** Let \( r^\mathcal{I}(d) = \{ e | r^\mathcal{I}(d, e) \} \).
  \[
  (r \sqsubseteq s)^\mathcal{I} = \{ d | r^\mathcal{I}(d) \subseteq s^\mathcal{I}(d) \}
  \]
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 ⊓ (has-child ◦ has-friend ⊑ knows)

- **Predicate logic semantics:**
  \[
  (r \sqsubseteq s)(x) = \forall y : (r(x, y) \rightarrow s(x, y))
  \]

- **Set semantics:** Let \( r^I(d) = \{ e \mid r^I(d, e) \} \).
  \[
  (r \sqsubseteq s)^I = \{ d \mid r^I(d) \subseteq s^I(d) \}
  \]

- **Note:** Role value maps lead to undecidability of satisfiability of concept descriptions!
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.
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:
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 $A \equiv C$, $A \sqsubseteq D$
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 $A \equiv C, A \subseteq D$
- no cyclic definitions (even not indirectly), such as $A \equiv \forall r.B, B \equiv \exists s.A$
TBoxes: Semantics

- TBoxes restrict the set of possible interpretations.
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 $\forall x : (A(x) \rightarrow C(x))$
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 $A^I = C^I$
  - $A \sqsubseteq C$ corresponds to $A^I \subseteq C^I$
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 $A^I = C^I$
  - $A \sqsubseteq C$ corresponds to $A^I \subseteq C^I$

- Non-empty interpretations which satisfy all terminological axioms are called **models** of the TBox.
• 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.
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.
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.
• **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 $r(a, b)$
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 $(a^I, b^I) \in r^I$
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.
Example TBox

\[ \text{Male} \doteq \neg \text{Female} \]
\[ \text{Human} \sqsubseteq \text{Living\_entity} \]
\[ \text{Woman} \doteq \text{Human} \sqcap \text{Female} \]
\[ \text{Man} \doteq \text{Human} \sqcap \text{Male} \]
\[ \text{Mother} \doteq \text{Woman} \sqcap \exists \text{has\_child\_Human} \]
\[ \text{Father} \doteq \text{Man} \sqcap \exists \text{has\_child\_Human} \]
\[ \text{Parent} \doteq \text{Father} \uplus \text{Mother} \]
\[ \text{Grandmother} \doteq \text{Woman} \sqcap \exists \text{has\_child\_Parent} \]
\[ \text{Mother\_without\_daughter} \doteq \text{Mother} \sqcap \forall \text{has\_child\_Male} \]
\[ \text{Mother\_with\_many\_children} \doteq \text{Mother} \sqcap (\geq 3 \text{has\_child}) \]
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
Does a description $C$ make sense at all, i.e., is it **satisfiable**?

A concept description $C$ is satisfiable iff there exists an interpretation $\mathcal{I}$ such that $C^\mathcal{I} \neq \emptyset$. 
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 $C^I \subseteq D^I$. 
Some Reasoning Services

- Does a description $C$ make sense at all, i.e., is it **satisfiable**?

  $\leadsto$ A concept description $C$ is satisfiable iff there exists an interpretation $\mathcal{I}$ such that $C^\mathcal{I} \neq \emptyset$.

- Is one concept a specialization of another one, is it **subsumed**?

  $\leadsto$ $C$ is **subsumed by** $D$, in symbols $C \sqsubseteq D$ iff we have for all interpretations $C^\mathcal{I} \subseteq D^\mathcal{I}$.

- Is $a$ an **instance** of a concept $C$?

  $\leadsto$ $a$ is an instance of $C$ iff for all interpretations, we have $a^\mathcal{I} \in C^\mathcal{I}$.
Some Reasoning Services

• Does a description \( C \) make sense at all, i.e., is it **satisfiable**?

\[ \leadsto \text{A concept description } C \text{ is satisfiable iff there exists an interpretation } I \text{ such that } C^I \neq \emptyset. \]

• Is one concept a specialization of another one, is it **subsumed**?

\[ \leadsto C \text{ is subsumed by } D, \text{ in symbols } C \sqsubseteq D \text{ iff we have for all interpretations } C^I \subseteq D^I. \]

• Is \( a \) an **instance** of a concept \( C \)?

\[ \leadsto a \text{ is an instance of } C \text{ iff for all interpretations, we have } a^I \in C^I. \]

→ **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?
Outlook

- Can we *reduce* the reasoning services to perhaps just one problem?
- What could be *reasoning algorithms*?
Outlook

- Can we reduce the reasoning services to perhaps just one problem?
- What could be reasoning algorithms?
- What about complexity and decidability?
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?
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?


## 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>$A^I$</td>
</tr>
<tr>
<td>$C \cap D$</td>
<td>(and $C \cap D$)</td>
<td>$C^I \cap D^I$</td>
</tr>
<tr>
<td>$C \cup D$</td>
<td>(or $C \cup D$)</td>
<td>$C^I \cup D^I$</td>
</tr>
<tr>
<td>$\neg C$</td>
<td>(not $C$)</td>
<td>$D - C^I$</td>
</tr>
<tr>
<td>$\forall r.C$</td>
<td>(all $r.C$)</td>
<td>${d \in D \mid r^I(d) \subseteq C^I}$</td>
</tr>
<tr>
<td>$\exists r.C$</td>
<td>(some $r.C$)</td>
<td>${d \in D \mid r^I(d) \neq \emptyset}$</td>
</tr>
<tr>
<td>$\geq n \cdot r$</td>
<td>(atleast $n \cdot r$)</td>
<td>${d \in D \mid</td>
</tr>
<tr>
<td>$\leq n \cdot r$</td>
<td>(atmost $n \cdot r$)</td>
<td>${d \in D \mid</td>
</tr>
<tr>
<td>$\exists r.C$</td>
<td>(some $r.C$)</td>
<td>${d \in D \mid r^I(d) \cap C^I \neq \emptyset}$</td>
</tr>
<tr>
<td>$\geq n \cdot r.C$</td>
<td>(atleast $n \cdot r.C$)</td>
<td>${d \in D \mid</td>
</tr>
<tr>
<td>$\leq n \cdot r.C$</td>
<td>(atmost $n \cdot r.C$)</td>
<td>${d \in D \mid</td>
</tr>
<tr>
<td>$r = s$</td>
<td>(eq $r = s$)</td>
<td>${d \in D \mid r^I(d) = s^I(d)}$</td>
</tr>
<tr>
<td>$r \neq s$</td>
<td>(neq $r \neq s$)</td>
<td>${d \in D \mid r^I(d) \neq s^I(d)}$</td>
</tr>
<tr>
<td>$r \subseteq s$</td>
<td>(subset $r \subseteq s$)</td>
<td>${d \in D \mid r^I(d) \subseteq s^I(d)}$</td>
</tr>
<tr>
<td>$g \equiv h$</td>
<td>(eq $g \equiv h$)</td>
<td>${d \in D \mid g^I(d) = h^I(d) \neq \emptyset}$</td>
</tr>
<tr>
<td>$g \neq h$</td>
<td>(neq $g \neq h$)</td>
<td>${d \in D \mid \emptyset \neq g^I(d) \neq h^I(d) \neq \emptyset}$</td>
</tr>
<tr>
<td>${i_1, i_2, \ldots, i_n}$</td>
<td>(oneof $i_1 \ldots i_n$)</td>
<td>${i_1^I, i_2^I, \ldots, i_n^I}$</td>
</tr>
</tbody>
</table>
# Role Descriptions

<table>
<thead>
<tr>
<th>Abstract</th>
<th>Concrete</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t$</td>
<td>$t$</td>
<td>$t^\mathcal{I}$</td>
</tr>
<tr>
<td>$f$</td>
<td>$f$</td>
<td>$f^\mathcal{I}$, (functional role)</td>
</tr>
<tr>
<td>$r \cap s$</td>
<td>(and $r \cdot s$)</td>
<td>$r^\mathcal{I} \cap s^\mathcal{I}$</td>
</tr>
<tr>
<td>$r \cup s$</td>
<td>(or $r \cdot s$)</td>
<td>$r^\mathcal{I} \cup s^\mathcal{I}$</td>
</tr>
<tr>
<td>$\neg r$</td>
<td>(not $r$)</td>
<td>$D \times D - r^\mathcal{I}$</td>
</tr>
<tr>
<td>$r^{-1}$</td>
<td>(inverse $r$)</td>
<td>{$(d, d')</td>
</tr>
<tr>
<td>$r</td>
<td>C$</td>
<td>(restr $r \cdot C$)</td>
</tr>
<tr>
<td>$r^+$</td>
<td>(trans $r$)</td>
<td>$(r^\mathcal{I})^+$</td>
</tr>
<tr>
<td>$r \circ s$</td>
<td>(compose $r \cdot s$)</td>
<td>$r^\mathcal{I} \circ s^\mathcal{I}$</td>
</tr>
<tr>
<td>$1$</td>
<td>self</td>
<td>{$(d, d)</td>
</tr>
</tbody>
</table>