Principles of Knowledge Representation and Reasoning

Semantic Networks and Description Logics II: Description Logics – Terminology and Notation

Bernhard Nebel, Malte Helmert and Stefan Wölfl

Albert-Ludwigs-Universität Freiburg

July 11, 2008

Description Logics – Terminology and Notation

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)

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)

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

$\leadsto$ 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)

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)

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)

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)
  - . . . 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**

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**

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**

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**

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**

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**

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**

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.

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.

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.

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.

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?

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?

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?

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?

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?

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?

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 $A, A_1, \ldots$
  - **Semantics**: Monadic predicates $A(\cdot)$ or set-theoretically a subset of the universe $A^{\mathcal{I}} \subseteq \mathcal{D}$.
- **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 $t^{\mathcal{I}} \subseteq \mathcal{D} \times \mathcal{D}$.

Atomic Concepts and Roles

- **Concept names**:
  - E.g., `Grandmother`, `Male`, ... (in the following usually *capitalized*)
  - We will use **symbols** such as $A, A_1, \ldots$
  - **Semantics**: Monadic predicates $A(\cdot)$ or set-theoretically a subset of the universe $A^{\mathcal{I}} \subseteq \mathcal{D}$.
- **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 $t^{\mathcal{I}} \subseteq \mathcal{D} \times \mathcal{D}$.

Concept and Role Description

- Out of *concept* and *role names*, complex **descriptions** can be created
- In our example, e.g. "a `Human` <u>and</u> `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**:

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

Concept and Role Description

- Out of *concept* and *role names*, complex **descriptions** can be created
- In our example, e.g. "a `Human` <u>and</u> `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**:

$$
\begin{aligned}
C^{\mathcal{I}} &= \{d \mid C(d) \text{ "is true in" } \mathcal{I}\} \\
r^{\mathcal{I}} &= \{(d, e) \mid r(d, e) \text{ "is true in" } \mathcal{I}\}
\end{aligned}
$$

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**:

$$C^{\mathcal{I}} = \{d \mid C(d) \text{ "is true in" } \mathcal{I}\}$$
$$r^{\mathcal{I}} = \{(d, e) \mid r(d, e) \text{ "is true in" } \mathcal{I}\}$$

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**:

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

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

Concept and Role Description

- Out of *concept* and *role names*, complex **descriptions** can be created
- In our example, e.g. "a `Human` <u>and</u> `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**:

$$C^{\mathcal{I}} = \{d \mid C(d) \text{ "is true in" } \mathcal{I}\}$$
$$r^{\mathcal{I}} = \{(d, e) \mid r(d, e) \text{ "is true in" } \mathcal{I}\}$$

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**:

$$\begin{aligned} C^{\mathcal{I}} &= \{d \mid C(d) \text{ "is true in" } \mathcal{I}\} \\ r^{\mathcal{I}} &= \{(d, e) \mid r(d, e) \text{ "is true in" } \mathcal{I}\} \end{aligned}$$

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 ⊓ Female
  - Father ⊔ Mother
  - ¬ Female
- **Predicate logic semantics**: $C(x) \wedge D(x)$, $C(x) \vee D(x)$, $\neg C(x)$
- **Set semantics**: $C^{\mathcal{I}} \cap D^{\mathcal{I}}$, $C^{\mathcal{I}} \cup D^{\mathcal{I}}$, $\mathcal{D} - C^{\mathcal{I}}$

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**: $C(x) \wedge D(x)$, $C(x) \vee D(x)$, $\neg C(x)$
- **Set semantics**: $C^{\mathcal{I}} \cap D^{\mathcal{I}}$, $C^{\mathcal{I}} \cup D^{\mathcal{I}}$, $\mathcal{D} - C^{\mathcal{I}}$

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**: $C(x) \wedge D(x)$, $C(x) \vee D(x)$, $\neg C(x)$
- **Set semantics**: $C^{\mathcal{I}} \cap D^{\mathcal{I}}$, $C^{\mathcal{I}} \cup D^{\mathcal{I}}$, $\mathcal{D} - C^{\mathcal{I}}$

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**: $C(x) \wedge D(x)$, $C(x) \vee D(x)$, $\neg C(x)$
- **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`.
  - 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
  - $\texttt{Mother} \sqcap \forall \texttt{has-child.Man}$
  - $\texttt{Woman} \sqcap \exists \texttt{has-child.Parent}$
- **Predicate logic semantics**:

$$(\exists r.C)(x) = \exists y : (r(x,y) \wedge C(y))$$
$$(\forall r.C)(x) = \forall y : (r(x,y) \rightarrow C(y))$$

Set semantics:

$$(\exists r.C)^{\mathcal{I}} = \{d | \, \exists e : (d,e) \in r^{\mathcal{I}} \wedge e \in C^{\mathcal{I}}\}$$
$$(\forall r.C)^{\mathcal{I}} = \{d | \, \forall e : (d,e) \in r^{\mathcal{I}} \rightarrow e \in C^{\mathcal{I}}\}$$

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

$$(\exists r.C)^{\mathcal{I}} = \{d | \, \exists e \, : \, (d,e) \in r^{\mathcal{I}} \land e \in C^{\mathcal{I}}\}$$
$$(\forall r.C)^{\mathcal{I}} = \{d | \, \forall e \, : \, (d,e) \in r^{\mathcal{I}} \rightarrow e \in C^{\mathcal{I}}\}$$

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

$$(\exists r.C)^{\mathcal{I}} = \{d \mid \exists e : (d,e) \in r^{\mathcal{I}} \land e \in C^{\mathcal{I}}\}$$
$$(\forall r.C)^{\mathcal{I}} = \{d \mid \forall e : (d,e) \in r^{\mathcal{I}} \rightarrow e \in C^{\mathcal{I}}\}$$

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
  - $\texttt{Mother} \sqcap \forall \texttt{has-child.Man}$
  - $\texttt{Woman} \sqcap \exists \texttt{has-child.Parent}$
- **Predicate logic semantics**:

$$(\exists r.C)(x) = \exists y : (r(x,y) \wedge C(y))$$
$$(\forall r.C)(x) = \forall y : (r(x,y) \rightarrow C(y))$$

Set semantics:

$$(\exists r.C)^{\mathcal{I}} = \{d | \exists e : (d,e) \in r^{\mathcal{I}} \wedge e \in C^{\mathcal{I}}\}$$
$$(\forall r.C)^{\mathcal{I}} = \{d | \forall e : (d,e) \in r^{\mathcal{I}} \rightarrow e \in C^{\mathcal{I}}\}$$

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

$$(\exists r.C)^{\mathcal{I}} = \{d \mid \exists e : (d, e) \in r^{\mathcal{I}} \land e \in C^{\mathcal{I}}\}$$
$$(\forall r.C)^{\mathcal{I}} = \{d \mid \forall e : (d, e) \in r^{\mathcal{I}} \rightarrow e \in C^{\mathcal{I}}\}$$

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:
  - $\texttt{Mother} \sqcap (\geq 3\, \texttt{has-child})$
  - $\texttt{Mother} \sqcap (\leq 2\, \texttt{has-child})$
- **Predicate logic semantics**:

$$\begin{aligned}(\geq n\ r)(x) &= \exists y_1 \ldots y_n : \big(r(x, y_1) \wedge \ldots \wedge r(x, y_n) \wedge \\ &\quad y_1 \neq y_2 \wedge \ldots \wedge y_{n-1} \neq y_n\big) \\ (\leq n\ r)(x) &= \neg(\geq n+1\ r)(x)\end{aligned}$$

- **Set semantics**:

$$(\geq n\ r)^{\mathcal{I}} = \{d \mid\ |\{e | r^{\mathcal{I}}(d,e)\}| \geq n\}$$
$$(\leq n\ r)^{\mathcal{I}} = \mathcal{D} - (\geq n+1\ r)^{\mathcal{I}}$$

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:
  - $\texttt{Mother} \sqcap (\geq 3\, \texttt{has-child})$
  - $\texttt{Mother} \sqcap (\leq 2\, \texttt{has-child})$
- **Predicate logic semantics**:

$$
\begin{aligned}
(\geq n\ r)(x) &= \exists y_1 \ldots y_n : \big(r(x, y_1) \land \ldots \land r(x, y_n) \land \\
&\qquad y_1 \neq y_2 \land \ldots \land y_{n-1} \neq y_n\big) \\
(\leq n\ r)(x) &= \neg(\geq n + 1\ r)(x)
\end{aligned}
$$

- **Set semantics**:

$$
(\geq n\ r)^{\mathcal{I}} = \{d \mid |\{e | r^{\mathcal{I}}(d, e)\}| \geq n\}
$$

$$
(\leq n\ r)^{\mathcal{I}} = \mathcal{D} - (\geq n + 1\ r)^{\mathcal{I}}
$$

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:
  - $\texttt{Mother} \sqcap (\geq 3\, \texttt{has-child})$
  - $\texttt{Mother} \sqcap (\leq 2\, \texttt{has-child})$
- **Predicate logic semantics**:

$$\begin{aligned}
(\geq n\ r)(x) &= \exists y_1 \ldots y_n : \big(r(x, y_1) \wedge \ldots \wedge r(x, y_n) \wedge \\
&\qquad y_1 \neq y_2 \wedge \ldots \wedge y_{n-1} \neq y_n\big) \\
(\leq n\ r)(x) &= \neg(\geq n + 1\ r)(x)
\end{aligned}$$

- **Set semantics**:

$$(\geq n\ r)^{\mathcal{I}} = \{d \mid\ |\{e | r^{\mathcal{I}}(d, e)\}| \geq n\}$$

$$(\leq n\ r)^{\mathcal{I}} = \mathcal{D} - (\geq n + 1\ r)^{\mathcal{I}}$$

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:
  - $\texttt{Mother} \sqcap (\geq 3\, \texttt{has-child})$
  - $\texttt{Mother} \sqcap (\leq 2\, \texttt{has-child})$
- **Predicate logic semantics**:

$$
\begin{aligned}
(\geq n\ r)(x) &= \exists y_1 \ldots y_n : \big(r(x,y_1) \wedge \ldots \wedge r(x,y_n) \wedge \\
&\qquad y_1 \neq y_2 \wedge \ldots \wedge y_{n-1} \neq y_n\big) \\
(\leq n\ r)(x) &= \neg(\geq n+1\ r)(x)
\end{aligned}
$$

- **Set semantics**:

$$(\geq n\ r)^{\mathcal{I}} = \{d \,\big|\, |\{e | r^{\mathcal{I}}(d,e)\}| \geq n\}$$
$$(\leq n\ r)^{\mathcal{I}} = \mathcal{D} - (\geq n+1\ r)^{\mathcal{I}}$$

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)
  - $\texttt{has-child}^{-1}$
- **Application**: $\exists \texttt{has-child}^{-1}.\texttt{Rich}$
- **Predicate logic semantics**:

$$r^{-1}(x,y) \quad = \quad r(y,x)$$

- **Set semantics**:

$$(r^{-1})^{\mathcal{I}} \quad = \quad \{(d,e) \mid (e,d) \in r^{\mathcal{I}}\}$$

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)
  - $\texttt{has-child}^{-1}$
- **Application**: $\exists\texttt{has-child}^{-1}.\texttt{Rich}$
- **Predicate logic semantics**:

$$r^{-1}(x,y) \quad = \quad r(y,x)$$

- **Set semantics**:

$$(r^{-1})^{\mathcal{I}} \quad = \quad \{(d,e) \mid (e,d) \in r^{\mathcal{I}}\}$$

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)
  - $\texttt{has-child}^{-1}$
- **Application**: $\exists\texttt{has-child}^{-1}.\texttt{Rich}$
- **Predicate logic semantics**:

$$r^{-1}(x, y) \quad = \quad r(y, x)$$

- **Set semantics**:

$$(r^{-1})^{\mathcal{I}} \quad = \quad \{(d, e) \mid (e, d) \in r^{\mathcal{I}}\}$$

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)
  - $\texttt{has-child}^{-1}$
- **Application**: $\exists\texttt{has-child}^{-1}.\texttt{Rich}$
- **Predicate logic semantics**:

$$r^{-1}(x,y) \quad = \quad r(y,x)$$

- **Set semantics**:

$$(r^{-1})^{\mathcal{I}} \quad = \quad \{(d,e) \mid (e,d) \in r^{\mathcal{I}}\}$$

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)
  - $\texttt{has-child}^{-1}$
- **Application**: $\exists\texttt{has-child}^{-1}.\texttt{Rich}$
- **Predicate logic semantics**:

$$r^{-1}(x,y) \quad = \quad r(y,x)$$

- **Set semantics**:

$$(r^{-1})^{\mathcal{I}} \quad = \quad \{(d,e) \mid (e,d) \in r^{\mathcal{I}}\}$$

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**:

$$(r \circ s)(x, y) \quad = \quad \exists z : (r(x, z) \wedge s(z, y))$$

- **Set semantics**:

$$(r \circ s)^{\mathcal{I}} \quad = \quad \{(d, e) \mid \exists f : (d, f) \in r^{\mathcal{I}} \wedge (f, e) \in s^{\mathcal{I}}\}$$

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**:

$$(r \circ s)(x, y) \quad = \quad \exists z : (r(x, z) \land s(z, y))$$

- **Set semantics**:

$$(r \circ s)^{\mathcal{I}} \quad = \quad \{(d, e) \mid \exists f : (d, f) \in r^{\mathcal{I}} \land (f, e) \in s^{\mathcal{I}}\}$$

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**:

$$(r \circ s)(x, y) \quad = \quad \exists z : (r(x, z) \wedge s(z, y))$$

- **Set semantics**:

$$(r \circ s)^{\mathcal{I}} \quad = \quad \{(d, e) \mid \exists f : (d, f) \in r^{\mathcal{I}} \wedge (f, e) \in s^{\mathcal{I}}\}$$

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**:

$$(r \circ s)(x, y) \quad = \quad \exists z : (r(x, z) \wedge s(z, y))$$

- **Set semantics**:

$$(r \circ s)^{\mathcal{I}} \quad = \quad \{(d, e) \mid \exists f : (d, f) \in r^{\mathcal{I}} \wedge (f, e) \in s^{\mathcal{I}}\}$$

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
  - $\texttt{Woman} \sqcap (\texttt{has-child} \circ \texttt{has-friend} \sqsubseteq \texttt{knows})$
- Predicate logic semantics:

$$(r \sqsubseteq s)(x) \quad = \quad \forall y : \big(r(x,y) \rightarrow s(x,y)\big)$$

- Set semantics: Let $r^{\mathcal{I}}(d) = \{e \mid r^{\mathcal{I}}(d,e)\}$.

$$(r \sqsubseteq s)^{\mathcal{I}} \quad = \quad \{d | r^{\mathcal{I}}(d) \subseteq s^{\mathcal{I}}(d)\}$$

- Note: Role value maps lead to undecidability of satisfiability of concept descriptions!

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` $\sqcap$ (`has-child` $\circ$ `has-friend` $\sqsubseteq$ `knows`)
- **Predicate logic semantics**:

$$(r \sqsubseteq s)(x) \quad = \quad \forall y : \big(r(x,y) \rightarrow s(x,y)\big)$$

- **Set semantics**: Let $r^{\mathcal{I}}(d) = \{e \mid r^{\mathcal{I}}(d,e)\}$.

$$(r \sqsubseteq s)^{\mathcal{I}} \quad = \quad \{d | r^{\mathcal{I}}(d) \subseteq s^{\mathcal{I}}(d)\}$$

- **Note**: Role value maps lead to undecidability of satisfiability of concept descriptions!

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
  - $\texttt{Woman} \sqcap (\texttt{has-child} \circ \texttt{has-friend} \sqsubseteq \texttt{knows})$
- **Predicate logic semantics**:

$$(r \sqsubseteq s)(x) \quad = \quad \forall y : \big(r(x,y) \rightarrow s(x,y)\big)$$

- **Set semantics**: Let $r^{\mathcal{I}}(d) = \{e \mid r^{\mathcal{I}}(d,e)\}$.

$$(r \sqsubseteq s)^{\mathcal{I}} \quad = \quad \{d | r^{\mathcal{I}}(d) \subseteq s^{\mathcal{I}}(d)\}$$

- **Note**: Role value maps lead to undecidability of satisfiability of concept descriptions!

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
  - $\texttt{Woman} \sqcap (\texttt{has-child} \circ \texttt{has-friend} \sqsubseteq \texttt{knows})$
- **Predicate logic semantics**:

$$(r \sqsubseteq s)(x) \quad = \quad \forall y : \big(r(x,y) \rightarrow s(x,y)\big)$$

- **Set semantics**: Let $r^{\mathcal{I}}(d) = \{e \mid r^{\mathcal{I}}(d,e)\}$.

$$(r \sqsubseteq s)^{\mathcal{I}} \quad = \quad \{d | r^{\mathcal{I}}(d) \subseteq s^{\mathcal{I}}(d)\}$$

- **Note**: Role value maps lead to undecidability of satisfiability of concept descriptions!

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
  - $\texttt{Woman} \sqcap (\texttt{has-child} \circ \texttt{has-friend} \sqsubseteq \texttt{knows})$
- **Predicate logic semantics**:

$$(r \sqsubseteq s)(x) \quad = \quad \forall y : \big(r(x,y) \rightarrow s(x,y)\big)$$

- **Set semantics**: Let $r^{\mathcal{I}}(d) = \{e \mid r^{\mathcal{I}}(d,e)\}$.

$$(r \sqsubseteq s)^{\mathcal{I}} \quad = \quad \{d | r^{\mathcal{I}}(d) \subseteq s^{\mathcal{I}}(d)\}$$

- **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**:
  - $A \doteq C$
  - $A \sqsubseteq C$

  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 $A \doteq C$, $A \sqsubseteq D$
  - no cyclic definitions (even not indirectly), such as $A \doteq \forall r.B$, $B \doteq \exists s.A$

Terminology Box

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

  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 $A \doteq C$, $A \sqsubseteq D$
  - no cyclic definitions (even not indirectly), such as $A \doteq \forall r.B$, $B \doteq \exists s.A$

Terminology Box

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

  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 $A \doteq C$, $A \sqsubseteq D$
  - no cyclic definitions (even not indirectly), such as $A \doteq \forall r.B$, $B \doteq \exists s.A$

Terminology Box

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

  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 $A \doteq C$, $A \sqsubseteq D$
  - no cyclic definitions (even not indirectly), such as $A \doteq \forall r.B$, $B \doteq \exists s.A$

TBoxes: Semantics

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

TBoxes: Semantics

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

TBoxes: Semantics

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

TBoxes: Semantics

- TBoxes restrict the set of possible interpretations.
- **Predicate logic semantics**:
  - $A \doteq C$ corresponds to $\forall x : \big(A(x) \leftrightarrow C(x)\big)$
  - $A \sqsubseteq C$ corresponds to $\forall x : \big(A(x) \rightarrow C(x)\big)$
- **Set semantics**:
  - $A \doteq C$ corresponds to $A^{\mathcal{I}} = C^{\mathcal{I}}$
  - $A \sqsubseteq C$ corresponds to $A^{\mathcal{I}} \subseteq C^{\mathcal{I}}$
- 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.

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

Example TBox

$$
\begin{aligned}
\texttt{Male} &\doteq \neg\texttt{Female} \\
\texttt{Human} &\sqsubseteq \texttt{Living\_entity} \\
\texttt{Woman} &\doteq \texttt{Human} \sqcap \texttt{Female} \\
\texttt{Man} &\doteq \texttt{Human} \sqcap \texttt{Male} \\
\texttt{Mother} &\doteq \texttt{Woman} \sqcap \exists\texttt{has-child.Human} \\
\texttt{Father} &\doteq \texttt{Man} \sqcap \exists\texttt{has-child.Human} \\
\texttt{Parent} &\doteq \texttt{Father} \sqcup \texttt{Mother} \\
\texttt{Grandmother} &\doteq \texttt{Woman} \sqcap \exists\texttt{has-child.Parent} \\
\texttt{Mother-without-daughter} &\doteq \texttt{Mother} \sqcap \forall\texttt{has-child.Male} \\
\texttt{Mother-with-many-children} &\doteq \texttt{Mother} \sqcap (\geq 3\,\texttt{has-child})
\end{aligned}
$$

Example ABox

CHARLES:	Man
EDWARD:	Man
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

DIANA:	Woman
ELIZABETH:	Woman

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 $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$.
- Is one concept a specialization of another one, is it **subsumed**?
- 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?
- a is an instance of C iff for all interpretations, we have $a^{\mathcal{I}} \in C^{\mathcal{I}}$.
- **Note**: These questions can be posed with or without a TBox that restricts the possible interpretations.

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 $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$.
- Is one concept a specialization of another one, is it **subsumed**?
- 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?
- a is an instance of C iff for all interpretations, we have $a^{\mathcal{I}} \in C^{\mathcal{I}}$.
- **Note**: These questions can be posed with or without a TBox that restricts the possible interpretations.

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 $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$.
- Is one concept a specialization of another one, is it **subsumed**?
- 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?
- a is an instance of C iff for all interpretations, we have $a^{\mathcal{I}} \in C^{\mathcal{I}}$.
- **Note**: These questions can be posed with or without a TBox that restricts the possible interpretations.

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 $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$.
- Is one concept a specialization of another one, is it **subsumed**?
- 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?
- a is an instance of C iff for all interpretations, we have $a^{\mathcal{I}} \in C^{\mathcal{I}}$.
- **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**?

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**?

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**?

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**?

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

- Baader, F., D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, *The Description Logic Handbook: Theory, Implementation, Applications*, Cambridge University Press, Cambridge, UK, 2003.
- Ronald J. Brachman and James G. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2):171–216, April 1985.
- Franz Baader, Hans-Jürgen Bürckert, Jochen Heinsohn, Bernhard Hollunder, Jürgen Müller, Bernhard Nebel, Werner Nutt, and Hans-Jürgen Profitlich. Terminological Knowledge Representation: A proposal for a terminological logic. Published in Proc. *International Workshop on Terminological Logics*, 1991, DFKI Document D-91-13.
- Bernhard Nebel. *Reasoning and Revision in Hybrid Representation Systems*, volume 422 of *Lecture Notes in Artificial Intelligence*. Springer-Verlag, Berlin, Heidelberg, New York, 1990.

Summary: Concept Descriptions

Abstract	Concrete	Interpretation
A	A	$A^{\mathcal{I}}$
$C \sqcap D$	(and C D)	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
$C \sqcup D$	(or C D)	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
$\neg C$	(not C)	$\mathcal{D} - C^{\mathcal{I}}$
$\forall r.C$	(all r C)	$\left\{d \in \mathcal{D} : r^{\mathcal{I}}(d) \subseteq C^{\mathcal{I}}\right\}$
$\exists r$	(some r)	$\left\{d \in \mathcal{D} : r^{\mathcal{I}}(d) \neq \emptyset\right\}$
$\geq n\ r$	(atleast n r)	$\left\{d \in \mathcal{D} : \lvert r^{\mathcal{I}}(d)\rvert \geq n\right\}$
$\leq n\ r$	(atmost n r)	$\left\{d \in \mathcal{D} : \lvert r^{\mathcal{I}}(d)\rvert \leq n\right\}$
$\exists r.C$	(some r C)	$\left\{d \in \mathcal{D} : r^{\mathcal{I}}(d) \cap C^{\mathcal{I}} \neq \emptyset\right\}$
$\geq n\ r.C$	(atleast n r C)	$\left\{d \in \mathcal{D} : \lvert r^{\mathcal{I}}(d) \cap C^{\mathcal{I}}\rvert \geq n\right\}$
$\leq n\ r.C$	(atmost n r C)	$\left\{d \in \mathcal{D} : \lvert r^{\mathcal{I}}(d) \cap C^{\mathcal{I}}\rvert \leq n\right\}$
$r \doteq s$	(eq r s)	$\left\{d \in \mathcal{D} : r^{\mathcal{I}}(d) = s^{\mathcal{I}}(d)\right\}$
$r \neq s$	(neq r s)	$\left\{d \in \mathcal{D} : r^{\mathcal{I}}(d) \neq s^{\mathcal{I}}(d)\right\}$
$r \sqsubseteq s$	(subset r s)	$\left\{d \in \mathcal{D} : r^{\mathcal{I}}(d) \subseteq s^{\mathcal{I}}(d)\right\}$
$g \doteq h$	(eq g h)	$\left\{d \in \mathcal{D} : g^{\mathcal{I}}(d) = h^{\mathcal{I}}(d) \neq \emptyset\right\}$
$g \neq h$	(neq g h)	$\left\{d \in \mathcal{D} : \emptyset \neq g^{\mathcal{I}}(d) \neq h^{\mathcal{I}}(d) \neq \emptyset\right\}$
$\{i_1, i_2, \ldots, i_n\}$	(oneof $i_1 \ldots i_n$)	$\{i_1^{\mathcal{I}}, i_2^{\mathcal{I}}, \ldots, i_n^{\mathcal{I}}\}$

Summary: Role Descriptions

Abstract	Concrete	Interpretation
t	t	$t^{\mathcal{I}}$
f	f	$f^{\mathcal{I}}$, *(functional role)*
$r \sqcap s$	(and r s)	$r^{\mathcal{I}} \cap s^{\mathcal{I}}$
$r \sqcup s$	(or r s)	$r^{\mathcal{I}} \cup s^{\mathcal{I}}$
$\neg r$	(not r)	$\mathcal{D} \times \mathcal{D} - r^{\mathcal{I}}$
r^{-1}	(inverse r)	$\left\{(d, d') : (d', d) \in r^{\mathcal{I}}\right\}$
$r\vert_C$	(restr r C)	$\left\{(d, d') \in r^{\mathcal{I}} : d' \in C^{\mathcal{I}}\right\}$
r^+	(trans r)	$(r^{\mathcal{I}})^+$
$r \circ s$	(compose r s)	$r^{\mathcal{I}} \circ s^{\mathcal{I}}$
$\mathbf{1}$	self	$\{(d, d) : d \in \mathcal{D}\}$