Semantics

An interpretation \( \mathcal{I} \) for DL is a pair

\[ \langle D, I \rangle \]

where \( D \) is any set of objects called the *domain* of the interpretation

and

\( I \) is a mapping called the *interpretation mapping* from the non-logical symbols of DL to elements and relations over \( D \).

Extending I

- \( \mathcal{I}[\text{thing}] \)
- \( \mathcal{I}[\text{ALL} \; r \; d] \)
- \( \mathcal{I}[\text{EXIST} \; n \; r] \)
- \( \mathcal{I}[\text{FILLS} \; r \; c] \)
- \( \mathcal{I}[\text{AND} \; d_1 \ldots d_n] \)

Truth in an Interpretation

Given an interpretation \( \mathcal{I} \), we say that \( \alpha \) is true in \( \mathcal{I} \), or \( \mathcal{I} \models \alpha \) according to the following rules.

1. \( \mathcal{I} \models (c \rightarrow d) \) iff \( \mathcal{I}[c] \subseteq \mathcal{I}[d] \);
2. \( \mathcal{I} \models (d \sqsubseteq d') \) iff \( \mathcal{I}[d] \subseteq \mathcal{I}[d'] \);
3. \( \mathcal{I} \models (d \equiv d') \) iff \( \mathcal{I}[d] = \mathcal{I}[d'] \);

Assuming that \( d \) and \( d' \) are concepts, and \( c \) is a constant.
Entailment

\[ S \models \alpha \]

if for every \( I \)

\[ I \models S \]

then \( I \models \alpha \)

Examples

\[ \text{AND} \text{ Doctor Female} \subseteq \text{Doctor} \]

\[ \text{john} \rightarrow \text{Thing} \]

\[ \text{(Surgeon} \subseteq \text{Doctor} \]

\[ KB \models [(\text{AND} \text{ Doctor Female} \subseteq \text{Doctor})] \]

\[ \text{(Surgeon} \not\subseteq [\text{AND} \text{ Doctor} \text{ FILLS: Specialty surgery}]) \]

Computing Entailments

We want to be able to determine if \( KB \models \alpha \), for

sentences \( \alpha \) of the form:

\[ (c \rightarrow d) \]

\[ (d \subseteq e) \]

Simplifying the Knowledge Base

- Remove sentences of the form

\[ (c \rightarrow d) \]

- Left Hand side of \( \subseteq \) and \( \not\subseteq \) sentences must be

  atomic concepts other than \text{Thing} and each atom appears on

  the left hand side only once.

- Assume \( \subseteq \) and \( \not\subseteq \) sentences are acyclic.
Normalization

1. expand definitions:

\[(\text{Surgeon} \equiv [\text{AND} \text{ Doctor} [\text{FILLS} : \text{Specialty surgery}]])\]

\[[\text{AND} \ldots \text{Surgeon} \ldots]\]

expands to

\[[\text{AND} \ldots [\text{AND} \text{ Doctor} [\text{FILLS} : \text{Specialty surgery}]] \ldots]\]

2. flatten the AND operators:

\[\text{AND} \ldots [\text{AND} \ d_1 \ldots d_n] \ldots\]

can be simplified to:

\[\text{AND} \ldots d_1 \ldots d_n \ldots\]

3. combine the ALL operators:

\[\text{AND} \ldots [\text{ALL} \ r \ d_1] \ldots [\text{ALL} \ r \ d_2] \ldots\]

can be simplified to:

\[\text{AND} \ldots [\text{ALL} \ r \ [\text{AND} \ d_1 \ d_2]] \ldots\]

4. combine EXISTS operators:

\[\text{AND} \ldots [\text{EXISTS} \ n_1 \ r] \ldots [\text{EXISTS} \ n_2 \ r] \ldots\]

can be simplified to

\[\text{AND} \ldots [\text{EXISTS} \ n \ r] \ldots\]

where \(n\) is the maximum of \(n_1\) and \(n_2\).

Normalization (cont)

5. deal with Thing:

Remove vacuous concepts as arguments to AND

\[[\text{ALL} \ r \ \text{Thing}]\]

6. remove redundant expressions:

[\text{AND} \ a_1 \ldots a_m]

\[[\text{FILLS} \ r_1 \ c_1] \ldots [\text{FILLS} \ r_m' \ c_m']\]

\[[\text{EXISTS} \ n_1 \ s_1] \ldots [\text{EXISTS} \ n_m'' \ s_m'']\]

\[[\text{ALL} \ t_1 \ e_1] \ldots [\text{ALL} \ t_m''' \ e_{m'''}]\]

The Final Result
Structure Mapping

\[ KB \models (d \sqsubseteq e) \]

**IDEA:** For \( d \) to be subsumed by \( e \), the normalized \( d \) must account for each component of the normalized \( e \) in some way.

Structure Mapping Procedure

**Input:** Two normalized concepts \( d \) and \( e \) where \( d \) is of the form \([\text{AND} \ d_1 \ldots d_m]\) and \( e \) is of the form \([\text{AND} \ d'_1 \ldots d'_m]\).

**Output** yes or no, according to whether \( KB \models (d \sqsubseteq e) \).

Return yes iff for each component \( e_j \), there exists a component \( d_i \) such that \( d_i \) matches \( e_j \) as follows:

1. if \( e_j \) is an atomic concept, then either \( d_i \) is identical to \( e_j \), or there is a sentence of the form \((d_i \sqsubseteq d')\) in the KB, where recursively some component of \( d' \) matches \( e_j \);
2. if \( e_j \) is of the form \([\text{FILLS} \ r \ c]\), then \( d_i \) must be identical to it;
3. if \( e_j \) is of the form \([\text{EXISTS} \ n \ r]\), then the corresponding \( d_i \) must be of the form \([\text{EXISTS} \ n' \ r]\), for some \( n' \geq n \); if \( n = 1 \), then \( d_i \) may be of the form \([\text{FILLS} \ r \ c]\);
4. if \( e_j \) is of the form \([\text{ALL} \ r \ e']\), then \( d_i \) must be of the form \([\text{ALL} \ r \ d']\), where recursively \( d' \) is subsumed by \( e' \).

Computing Satisfaction

\[ KB \models (c \rightarrow e) \]

iff

\[ KB \models (d \sqsubseteq e) \]

where \( d \) is the AND of every concept \( d_i \) such that \((c \rightarrow d_i)\) is in the KB.

Taxonomies and Classification

given some query concept, \( q \), find all \( c \) in the KB such that

\[ KB \models (c \rightarrow q) \]

given a constant \( c \), find all the atomic concepts \( a \) such that

\[ KB \models (c \rightarrow a) \]

Partial Order, Classification
Computing Classification

Consider adding a sentence \((a \equiv d)\) to a taxonomy.

1. First calculate \(S\), the most specific subsumers of \(d\).

2. Next calculate \(G\), the most general subsumees of \(d\).

3. If there is a concept \(a'\) in \(S \cap G\), then the concept is already present.

4. Otherwise, insert \(a\).

Classification (cont)

1. Answering Questions
2. Taxonomies and Frame Hierarchies
3. Inheritance

Computing (cont)

1. Computing most specific subsumers.

2. Computing the most general subsumees.