From propositions under :- to concepts under ⊑

from boolean values 0/1 to concepts

*Dogs are pets*

pet(X) :- dog(X).

dog ⊑ pet
From propositions under \( \neg \) to concepts under \( \sqsubseteq \)

from boolean values 0/1 to concepts

*Dogs are pets*

\[
\text{pet}(X) :- \text{dog}(X).
\]

\[
\text{dog} \sqsubseteq \text{pet}
\]

- avoid variables

*Fido is a pet*

\[
\text{pet}(\text{fido})
\]
From propositions under :- to concepts under ⊒

from boolean values 0/1 to concepts

*Dogs are pets*
pet(X) :- dog(X).
dog ⊑ pet

- avoid variables

*Fido is a pet*
pet(fido)

- connectives on concepts

*pet with 4 legs that barks*
[AND pet [EXISTS 4 :leg] [FILLS :VoiceCall bark]]

Brachman & Levesque (KRR chapter 9)
From propositions under :- to concepts under ⊑

from boolean values 0/1 to concepts

Dogs are pets
pet(X) :- dog(X).
dog ⊑ pet subsumption/subtyping (TBox)

- avoid variables

Fido is a pet
pet(fido) predication/typing (ABox)

- connectives on concepts (contra propositions)

pet with 4 legs that barks
[AND pet [EXISTS 4 :leg] [FILLS :VoiceCall bark]]

Brachman & Levesque (KRR chapter 9)
BL concepts

\[ \langle \text{concept} \rangle ::= \top \mid \langle \text{atomic-concept} \rangle \mid \]
\[ \text{[AND } \langle \text{concept} \rangle \cdots \langle \text{concept} \rangle \text{]} \mid \]
\[ \text{[ALL } \langle \text{role} \rangle \langle \text{concept} \rangle \text{]} \mid \]
\[ \text{[EXISTS } n \langle \text{role} \rangle \text{]} \mid \]
\[ \text{[FILLS } \langle \text{role} \rangle \langle \text{constant} \rangle \text{]} \]
BL concepts and their interpretations

\[ \langle \text{concept} \rangle ::= \top | \langle \text{atomic-concept} \rangle | \] \[\text{AND} \langle \text{concept} \rangle \cdots \langle \text{concept} \rangle | \] \[\text{ALL} \langle \text{role} \rangle \langle \text{concept} \rangle | \] \[\text{EXISTS} \ n \langle \text{role} \rangle | \] \[\text{FILLS} \langle \text{role} \rangle \langle \text{constant} \rangle \] \[I = \langle D, \phi, \pi \rangle \text{ with } [\top] = D \text{ and} \]

\[ [A] = \{ d \in D \mid \pi(A)(d) = \text{true} \} \quad \text{for atomic concept } A \]
\[ [R] = \{ (d, d') \in D \times D \mid \pi(R)(d, d') = \text{true} \} \quad \text{for role } R \]
BL concepts and their interpretations

\[
\langle\text{concept}\rangle ::= \top \mid \langle\text{atomic-concept}\rangle \mid \\
\text{[AND} \langle\text{concept}\rangle \cdots \langle\text{concept}\rangle\text{]} \mid \\
\text{[ALL} \langle\text{role}\rangle \langle\text{concept}\rangle\text{]} \mid \\
\text{[EXISTS} n \langle\text{role}\rangle\text{]} \mid \\
\text{[FILLS} \langle\text{role}\rangle \langle\text{constant}\rangle\text{]} \\
\]

\[I = \langle D, \phi, \pi \rangle \text{ with } \llbracket \top \rrbracket = D \text{ and }\]

\[
\llbracket A \rrbracket = \{ d \in D \mid \pi(A)(d) = \text{true} \} \quad \text{for atomic concept } A \\
\llbracket R \rrbracket = \{ (d, d') \in D \times D \mid \pi(R)(d, d') = \text{true} \} \quad \text{for role } R
\]

\[
\llbracket \text{[AND C1} \cdots \text{ Cn]} \rrbracket = \llbracket C1 \rrbracket \cap \cdots \cap \llbracket Cn \rrbracket \\
\llbracket \text{[ALL R} C \rrbracket = \{ d \mid \{ d' \mid d[R]d' \} \subseteq \llbracket C \rrbracket \} \\
\llbracket \text{[EXISTS} n R \rrbracket = \{ d \mid |\{ d' \mid d[R]d' \}| \geq n \} \\
\llbracket \text{[FILLS} R a \rrbracket = \{ d \mid d[R]a \} \]
\]
Structure matching

normalize concept $C$ to \{\text{\text{C-requirements}}\}

$C \sqsubseteq C' \iff$ each $C'$-requirement is matched by a $C$-requirement.
Structure matching

normalize concept $C$ to $\{C\text{-requirements}\}$

$C \sqsubseteq C' \iff$ each $C'$-requirement is matched by a $C$-requirement.

Normalization

- flatten $[\text{AND } C_1 [\text{AND } C_2 C_3]] \leadsto [\text{AND } C_1 C_2 C_3]$
- merge $[\text{ALL } R C], [\text{ALL } R C'] \leadsto [\text{ALL } R [\text{AND } C C']]$
Structure matching

normalize concept $C$ to $\{C$-requirements$\}$

$C \sqsubseteq C' \iff$ each $C'$-requirement is matched by a $C$-requirement.

Normalization

- flatten $[\text{AND } C_1 [\text{AND } C_2 C_3]] \leadsto [\text{AND } C_1 C_2 C_3]$
- merge $[\text{ALL } R C], [\text{ALL } R C'] \leadsto [\text{ALL } R [\text{AND } C C']]$

$C$ matches $C'$ iff $C$ is $C'$ or else one of the following

- $C$ is $[\text{EXISTS } n R]$ and $C'$ is $[\text{EXISTS } m R]$ where $n > m$
- $C$ is $[\text{FILLS } R a]$ and $C'$ is $[\text{EXISTS } 1 R]$
- $C$ is $[\text{ALL } R C'']$ and $C'$ is $[\text{ALL } R C'''$] where $C'' \sqsubseteq C'''$
B&L lacks negation

Encode subsumptions $C \sqsubseteq C_1 \ldots C \sqsubseteq C_n$ by replacing $C$ with

$$[\text{AND } C_1 \ldots C_n a_C]$$
B&L lacks negation

Encode subsumptions $C \sqsubseteq C_1 \ldots C \sqsubseteq C_n$ by replacing $C$ with

$$[\text{AND } C_1 \cdots C_n a_C]$$

No requirements for all inclusive atomic concept $\top$ (Thing)

$$C \sqsubseteq \top \text{ for every concept } C.$$ 

“Monster” model: $\top \sqsubseteq C$ for every (atomic) concept $C$
B&L lacks negation

Encode subsumptions $C \sqsubseteq C_1 \ldots C \sqsubseteq C_n$ by replacing $C$ with

$$[\text{AND } C_1 \cdots C_n a_C]$$

No requirements for all inclusive atomic concept $\top$ (Thing)

$$C \sqsubseteq \top \text{ for every concept } C.$$

“Monster” model: $\top \sqsubseteq C$ for every (atomic) concept $C$
- problematic with negation $\neg$ \quad ($\top \not\sqsubseteq \neg \top$)

Negation and disjunction (OR) in

$$[\text{AND } [\text{OR } C C'] \neg C] \sqsubseteq C'$$
B&L lacks negation

Encode subsumptions $C \sqsubseteq C_1 \ldots C \sqsubseteq C_n$ by replacing $C$ with

$$[\text{AND } C_1 \cdots C_n a_C]$$

No requirements for all inclusive atomic concept $\top$ (Thing)

$$C \sqsubseteq \top \text{ for every concept } C.$$ 

“Monster” model: $\top \sqsubseteq C$ for every (atomic) concept $C$

- problematic with negation $\neg$ $(\top \not\sqsubseteq \neg \top)$

Negation and disjunction (OR) in

$$[\text{AND } [\text{OR } C C'] \neg C] \sqsubseteq C'$$

- need $a \neq b$ for

$$[\text{AND } [\text{FILLS } R a] [\text{FILLS } R b]] \sqsubseteq [\text{EXISTS } 2 R]$$
Example of an interpretation

Person ⊨ ∃gives.(Talk ⊨ ∀topic.DL)

Person ⊨ ∀gives.(Talk ⊨ ∃topic.DL)
Negation for $\mathcal{ALC}$ ($\sqcap, \forall, \neg$)

$$\llbracket \neg C \rrbracket = \{ d \in D \mid d \notin \llbracket C \rrbracket \}$$

Write

$$C \sqcap C' \text{ for } \llbracket \text{AND } C \ C' \rrbracket \text{ and } \forall R.C \text{ for } \llbracket \text{ALL } R \ C \rrbracket$$

and add negation to express

$$C \sqcup C' \text{ as } \neg(\neg C \sqcap \neg C') \text{ and } \exists R.C \text{ as } \neg(\forall R.\neg C).$$
Negation for $\mathcal{ALC}$ $(\sqcap, \forall, \neg)$

$$\llbracket \neg C \rrbracket = \{ d \in D \mid d \notin \llbracket C \rrbracket \}$$

Write

$$C \sqcap C' \text{ for } [\text{AND } C \ C'] \text{ and } \forall R.C \text{ for } [\text{ALL } R \ C]$$

and add negation to express

$$C \sqcup C' \text{ as } \neg(\neg C \sqcap \neg C') \text{ and } \exists R.C \text{ as } \neg(\forall R.\neg C).$$

Subsumption $C \sqsubseteq C'$ reduces to instances

$$C \sqsubseteq C' \iff \text{every } C\text{-instance must be a } C'\text{-instance}$$

$$\iff C(a), \neg C'(a) \text{ leads to a contradiction (for fresh } a)$$
Negation for $\mathcal{ALC}$ ($\cap$, $\forall$, $\neg$)

\[ \llbracket \neg C \rrbracket = \{ d \in D \mid d \not\in \llbracket C \rrbracket \} \]

Write

$C \cap C'$ for $[\text{AND } C \ C']$ and $\forall R \ C$ for $[\text{ALL } R \ C]$ 

and add negation to express

$C \sqcup C'$ as $\neg(\neg C \cap \neg C')$ and $\exists R \ C$ as $\neg(\forall R. \neg C)$.

Subsumption $C \sqsubseteq C'$ reduces to instances

$C \sqsubseteq C' \iff$ every $C$-instance must be a $C'$-instance

$\iff C(a), \neg C'(a)$ leads to a contradiction (for fresh $a$)

E.g. $C \sqsubseteq C$ gives contradiction $C(a), \neg C(a)$.
Negation for $\mathcal{ALC}$ $(\sqcap, \forall, \neg)$

$$[\neg C] = \{d \in D \mid d \notin [C]\}$$

Write

$$C \sqcap C' \text{ for } [\text{AND } C C'] \text{ and } \forall R.C \text{ for } [\text{ALL } R C]$$

and add negation to express

$$C \sqcup C' \text{ as } \neg(\neg C \sqcap \neg C') \text{ and } \exists R.C \text{ as } \neg(\forall R.\neg C).$$

Subsumption $C \sqsubseteq C'$ reduces to instances

$$C \sqsubseteq C' \iff \text{ every } C\text{-instance must be a } C'\text{-instance}$$

$$\iff C(a), \neg C'(a) \text{ leads to a contradiction (for fresh } a)$$

E.g. $C \sqsubseteq C$ gives contradiction $C(a), \neg C(a)$.

Consider all instantiations/assertions (contra structure matching)
Decomposing instantiations (for a model)

\[
(C \cap C')(a) \leadsto C(a), C'(a)
\]
\[
(\forall R.C)(a), R(a, b) \leadsto C(b)
\]
\[
(\exists R.C)(a) \leadsto R(a, b), C(b) \text{ for fresh } b
\]
Decomposing instantiations (for a model)

\[(C \sqcap C')(a) \leadsto C(a), C'(a)\]
\[(\forall R.C)(a), R(a, b) \leadsto C(b)\]
\[(\exists R.C)(a) \leadsto R(a, b), C(b) \text{ for fresh } b\]

\[(C \sqcup C')(a) \leadsto \begin{cases} C(a) \\ C'(a) \end{cases}\]
Decomposing instantiations (for a model)

\[(C \cap C')(a) \Rightarrow C(a), C'(a)\]

\[(\forall R.C)(a), R(a, b) \Rightarrow C(b)\]

\[(\exists R.C)(a) \Rightarrow R(a, b), C(b)\] for fresh \(b\)

\[(C \sqcup C')(a) \Rightarrow \begin{cases} C(a) \\ C'(a) \end{cases}\]

Disjunctive Normal Form \(\approx\) tableau \(\{A_1, \ldots, A_n\}\)

\[\varphi \Rightarrow \bigvee_{i=1}^{n} \bigwedge A_i\]

where \(A_i\) consists of atomic and negated atomic formulas (ABox)
- describes a model if there are no contradictory pairs.

Tableau rules decompose a formula into subformulas
Tableau rules for $\mathcal{ALC}$

(⊓-rule) If $\mathcal{A}$ contains $(C \sqcap C')(a)$ but not both $C(a)$ and $C'(a)$,

$$\mathcal{A} := \mathcal{A} \cup \{C(a), C'(a)\}.$$ 

(⊔-rule) If $\mathcal{A}$ contains $(C \sqcup C')(a)$ but neither $C(a)$ nor $C'(a)$,

$$\mathcal{A}_1 := \mathcal{A} \cup \{C(a)\}$$

$$\mathcal{A}_2 := \mathcal{A} \cup \{C'(a)\}.$$
Tableau rules for $\mathcal{ALC}$

(⊓-rule) If $\mathcal{A}$ contains $(C \cap C')(a)$ but not both $C(a)$ and $C'(a)$,

$$\mathcal{A} := \mathcal{A} \cup \{C(a), C'(a)\}.$$ 

(⊔-rule) If $\mathcal{A}$ contains $(C \sqcup C')(a)$ but neither $C(a)$ nor $C'(a)$,

$$\mathcal{A}_1 := \mathcal{A} \cup \{C(a)\}$$
$$\mathcal{A}_2 := \mathcal{A} \cup \{C'(a)\}.$$ 

E.g. $(C \sqcup C') \cap \neg C' \sqsubseteq C$
initialize $\mathcal{A} = \{((C \sqcup C') \cap \neg C')(a), \neg C(a)\}$
Tableau rules for $\mathcal{ALC}$

(⊓-rule) If $\mathcal{A}$ contains $(C \sqcap C')(a)$ but not both $C(a)$ and $C'(a)$,

$\mathcal{A} := \mathcal{A} \cup \{C(a), C'(a)\}$.

(⊔-rule) If $\mathcal{A}$ contains $(C \sqcup C')(a)$ but neither $C(a)$ nor $C'(a)$,

$\mathcal{A}_1 := \mathcal{A} \cup \{C(a)\}$
$\mathcal{A}_2 := \mathcal{A} \cup \{C'(a)\}$.

E.g. $(C \sqcup C') \sqcap \neg C' \sqsubseteq C$
initialize $\mathcal{A} = \{((C \sqcup C') \sqcap \neg C')(a), \neg C(a)\}$

(∃-rule) If $\mathcal{A}$ contains $(\exists R.C)(a)$ but no $b$ s.t. $R(a, b)$ and $C(b)$,

$\mathcal{A} := \mathcal{A} \cup \{R(a, b), C(b)\}$ where $b$ is fresh.

(∀-rule) If $\mathcal{A}$ contains $(\forall R.C)(a)$ and $R(a, b)$ but not $C(b)$,

$\mathcal{A} := \mathcal{A} \cup \{C(b)\}$.
$C \subseteq C'$ via tableaux

**Step 1.** Express counter-example (towards a contradiction or counter-model)

$C(a), \neg C'(a)$ (for fresh $a$)

**Step 2.** Push negations inside via De Morgan’s laws

$\neg(\forall R. C) \leadsto (\exists R) \neg C$

$\neg(\exists R. C) \leadsto (\forall R) \neg C$

$\neg(C \cap C') \leadsto (\neg C) \cup \neg C'$

$\neg(C \cup C') \leadsto (\neg C) \cap \neg C'$

**Step 3.** Decompose concepts using tableaux rules

**Outcome:** Yes ($C \subseteq C'$) if there is a contradictory pair

No (counter-model) otherwise

**Example**  $(\forall R. C) \cap (\forall R. C') \subseteq \forall R.(C \cap C')$ on board
Counter-model

$$(\exists R. C) \cap (\exists R. C') \subseteq \exists R. (C \cap C')$$
Counter-model

\((\exists R. C) \cap (\exists R. C') \subseteq \exists R. (C \cap C') \) ??

\(\neg \exists R. (C \cap C') \leadsto \forall R. \neg (C \cap C') \leadsto \forall R. (\neg C \cup \neg C')\)
Counter-model

\[(\exists R. C) \cap (\exists R. C') \subseteq \exists R. (C \cap C') \quad ??\]

\[-\exists R. (C \cap C') \leadsto \forall R. \neg (C \cap C') \leadsto \forall R. (\neg C \sqcup \neg C')\]

\[\mathcal{A} := (((\exists R. C) \cap (\exists R. C'))(a), (\forall R. (\neg C \sqcup \neg C'))(a))\]

\[\leadsto \{ \ldots, (\exists R. C)(a), (\exists R. C')(a) \} \quad \text{by } \cap\text{-rule}\]

\[\leadsto \{ \ldots, R(a, b), C(b) \} \quad \text{by } \exists\text{-rule}\]

\[\leadsto \{ \ldots, (\neg C \sqcup \neg C')(b) \} \quad \text{by } \forall\text{-rule}\]

\[\leadsto \begin{cases} \{ \ldots, (\neg C)(b) \} & \text{discard: contradicts } C(b) \quad \text{by } \sqcup\text{-rule} \\ \{ \ldots, (\neg C')(b) \} & \text{keep: no contradiction} \end{cases}\]
Counter-model

\[(\exists R. C) \cap (\exists R. C') \subseteq \exists R. (C \cap C') \quad ??\]

\[-\exists R. (C \cap C') \leadsto \forall R. \neg (C \cap C') \leadsto \forall R. (\neg C \cup \neg C')\]

\[\mathcal{A} := \{(((\exists R. C) \cap (\exists R. C'))(a), (\forall R. (\neg C \cup \neg C'))(a)\}\]

\[\leadsto \{\ldots, (\exists R. C)(a), (\exists R. C')(a)\} \text{ by } \cap\text{-rule}\]

\[\leadsto \{\ldots, R(a, b), C(b)\} \text{ by } \exists\text{-rule}\]

\[\leadsto \{\ldots, (\neg C \cup \neg C')(b)\} \text{ by } \forall\text{-rule}\]

\[\leadsto \left\{ \begin{array}{l}
\{\ldots, (\neg C)(b)\} \text{ discard: contradicts } C(b) \\
\{\ldots, (\neg C')(b)\} \text{ keep: no contradiction}
\end{array} \right. \text{ by } \sqcup\text{-rule}\]

Similarly, \((\exists R. C')(a)\) leads to \{\ldots, R(a, c), C'(c), \ldots, (\neg C)(c)\}.
Counter-model

\[(\exists R.C) \cap (\exists R.C') \subseteq \exists R.(C \cap C') \text{ ??}\]

\[\neg \exists R.(C \cap C') \leadsto \forall R.\neg(C \cap C') \leadsto \forall R.\neg(C \sqcup \neg C')\]

\[\mathcal{A} := \{((\exists R.C) \cap (\exists R.C'))(a), (\forall R.\neg(C \sqcup \neg C'))(a)\}\]

\[\leadsto \{\ldots, (\exists R.C)(a), (\exists R.C')(a)\} \text{ by } \cap\text{-rule}\]

\[\leadsto \{\ldots, R(a, b), C(b)\} \text{ by } \exists\text{-rule}\]

\[\leadsto \{\ldots, (\neg C \sqcup \neg C')(b)\} \text{ by } \forall\text{-rule}\]

\[\leadsto \left\{\begin{array}{l}
\{\ldots, (\neg C)(b)\} \text{ discard: contradicts } C(b) \\
\{\ldots, (\neg C')(b)\} \text{ keep: no contradiction}
\end{array}\right\} \text{ by } \sqcup\text{-rule}\]

Similarly, \((\exists R.C')(a)\) leads to \{\ldots, R(a, c), C'(c), \ldots, (\neg C)(c)\}.

Model (counter-example):

\[\llbracket R \rrbracket = \{([a], [b]), ([a], [c])\}\]

\[\llbracket C \rrbracket = \{[b]\} \neq \llbracket C' \rrbracket = \{[c]\}\]

e.g. \([a] = [b] = 0, [c] = 1\)