Using ABoxes to store Data
ABoxes (Assertion Boxes)

Knowledge Base (KB)

**TBox** (terminological box, schema)

Man ≡ Human ⊓ Male
HappyFather ≡ Man ⊓ ∃hasChild
...

**ABox** (assertion box, data)

john : Man
(john, mary) : hasChild
...

Inference System

Interface
Assertion Box (ABox)

Let $\mathcal{L}$ be a description logic. A $\mathcal{L}$-ABox is a finite set $\mathcal{A}$ of assertions of the form

$$C(a), \quad r(a, b),$$

where $C$ is an $\mathcal{L}$-concept, $r$ a role name, and $a, b$ are individual names.

- $C(a)$ says that $a$ is an instance of $C$;
- $r(a, b)$ says that $(a, b)$ is an instance of $r$.

ABoxes generalize database instances in which only ground sentences

$$A(a), \quad r(a, b)$$

with $A$ a concept name and $r$ a role name are allowed. We sometimes call ABoxes that are database instances simple ABoxes.
Semantics for ABoxes (Open World Assumption)

Let \( \mathcal{A} \) be an ABox. By \( \text{Ind}(\mathcal{A}) \) we denote the set of individual names in \( \mathcal{A} \). An interpretation \( \mathcal{I} \) is a model of \( \mathcal{A} \), in symbols \( \mathcal{I} \models \mathcal{A} \), if

- \( \text{Ind}(\mathcal{A}) \subseteq \Delta^\mathcal{I} \);
- If \( C(a) \in \mathcal{A} \), then \( a \in C^\mathcal{I} \);
- If \( r(a, b) \in \mathcal{A} \), then \( (a, b) \in r^\mathcal{I} \).

The set of models of \( \mathcal{A} \) is denoted by \( \text{Mod}(\mathcal{A}) \).

Let \( F(x_1, \ldots, x_n) \) be an FOPL query. Then \( (a_1, \ldots, a_n) \) in \( \text{Ind}(\mathcal{A}) \) is a certain answer to \( F(x_1, \ldots, x_n) \) in \( \mathcal{A} \), in symbols

\[
\mathcal{A} \models F(a_1, \ldots, a_n),
\]

if \( \mathcal{I} \models F(a_1, \ldots, a_n) \) for all \( \mathcal{I} \in \text{Mod}(\mathcal{A}) \).

The set of certain answers to \( F(x_1, \ldots, x_n) \) in \( \mathcal{A} \) is

\[
\text{certanswer}(F(x_1, \ldots, x_n), \mathcal{A}) = \{(a_1, \ldots, a_n) \mid \mathcal{A} \models F(a_1, \ldots, a_n)\}
\]
FOPL Query Answering (Open World Semantics)

• ‘Yes’ is the certain answer to a Boolean query $F$ if $\mathcal{I} \models F$ for all $\mathcal{I} \in \text{Mod}(\mathcal{A})$.

• ‘No’ is the certain answer to a Boolean query $F$ if $\mathcal{I} \not\models F$ for all $\mathcal{I} \in \text{Mod}(\mathcal{A})$

• If neither ‘Yes’ nor ‘No’ is a certain answer, then we say that the certain answer is ‘Don’t know’.
What is the answer to this query?

Consider the ABox \( \mathcal{A} \):

1. friend(john, susan)
2. friend(john, andrea)
3. loves(susan, andrea)
4. loves(andrea, bill)
5. Female(susan)
6. \( \neg \)Female(bill)

Does John have a female friend who is in love with a not female person?

The corresponding Boolean FOPL query is

\[
F = \exists x. (\text{friend}(\text{john}, x) \land \text{Female}(x) \land \exists y. (\text{loves}(x, y) \land \neg \text{Female}(y)))
\]

or, in description logic notation:

\[
\exists \text{friend}. (\text{Female} \sqcap \exists \text{loves}. \neg \text{Female})(\text{john})
\]
Answers: Example

Let

\[ \mathcal{A} = \{ \text{Male}(harry), \text{hasChild}(peter, harry) \} \]

The answer to the query “Are all children of Peter male?”, in symbols

\[ F = \forall x. (\text{hasChild}(peter, x) \rightarrow \text{Male}(x)) , \]

given by \( \mathcal{A} \) is “don’t know”.

In order to prevent this, we could add

- \( \forall \text{hasChild}. \text{Male}(peter) \)
- \( \text{or} (\leq 1 \text{hasChild} \cdot \top)(peter) \)

to the ABox \( \mathcal{A} \).
3-Colorability

A graph $G$ is a pair $(W, E)$ consisting of a set $W$ and a symmetric relation $E$ on $W$.

$G$ is 3-colorable if there exist subsets $\text{blue}$, $\text{red}$, and $\text{green}$ of $W$ such that

- the sets $\text{blue}$, $\text{green}$, and $\text{red}$ are mutually disjoint;
- $\text{blue} \cup \text{red} \cup \text{green} = W$;
- if $(a, b) \in E$, then $a$ and $b$ do not have the same color.

3-colorability of graphs is an NP-complete problem.
3-Colorability as a Query Answering Problem

Assume $G = (W, E)$ is given. Construct the ABox $\mathcal{A}$ by taking a role name $r$ and concept names Blue, Green, and Red and setting

- $r(a, b) \in \mathcal{A}$ for all $a, b \in W$ with $(a, b) \in E$.
- Blue $\sqcup$ Green $\sqcup$ Red$(a) \in \mathcal{A}$ for all $a \in W$.
- $(\text{Blue} \rightarrow \forall r. (\text{Red} \sqcup \text{Green}))(a) \in \mathcal{A}$, for all $a \in W$;
- $(\text{Red} \rightarrow \forall r. (\text{Blue} \sqcup \text{Green}))(a) \in \mathcal{A}$, for all $a \in W$;
- $(\text{Green} \rightarrow \forall r. (\text{Red} \sqcup \text{Blue}))(a) \in \mathcal{A}$, for all $a \in W$.

Define query $F$ by setting

$$F = \exists x((\text{Blue}(x) \land \text{Red}(x)) \lor (\text{Blue}(x) \land \text{Green}(x)) \lor (\text{Red}(x) \land \text{Green}(x)))$$

Then $G$ is not 3-colorable if, and only if, the certain answer to $F$ in $\mathcal{A}$ is ‘Yes’. Thus, query answering is coNP-hard (the complement of NP) in data complexity!
Using the $\mathcal{ALC}$ Tableau to Answer Queries

Consider an $\mathcal{ALC}$ ABox $\mathcal{A}$ and a query of the form $C(x)$, where $C$ is an $\mathcal{ALC}$ concept. Assume $a \in \text{Ind}(\mathcal{A})$ is given. We want to know whether

$$a \in \text{certanswer}(C(x), \mathcal{A}),$$

in other words, we want to know whether $a \in C^\mathcal{I}$ for all interpretations $\mathcal{I} \in \text{Mod}(\mathcal{A})$.

We can reformulate this problem as follows: Let $\mathcal{A}' = \mathcal{A} \cup \{\neg C(a)\}$. Then $a \in \text{certanswer}(C(x), \mathcal{A})$ if there does not exist any model of $\mathcal{A}'$. 
Tableau Algorithm Deciding whether $\mathcal{A}$ has a Model

Consider $\mathcal{ALC}$ ABox $\mathcal{A}$. We may assume that each concept $D$ in $\mathcal{A}$ is in negation normal form and obtain the constraint system $\mathcal{A}^*$ as the set of constraints

- $a : C$ for all $C(a) \in \mathcal{A}$;
- $(a, b) : r$ for all $r(a, b) \in \mathcal{A}$.

Then $\mathcal{A}$ has a model if, and only if, starting from $\mathcal{A}^*$ there is a sequence of completion rule applications that terminates with a set of constraints containing no clash.
Example

Consider again the ABox $\mathcal{A}$:

1. $\text{friend}(\text{john}, \text{susan})$
2. $\text{friend}(\text{john}, \text{andrea})$
3. $\text{loves}(\text{susan}, \text{andrea})$
4. $\text{loves}(\text{andrea}, \text{bill})$
5. $\text{Female}(\text{susan})$
6. $\neg \text{Female}(\text{bill})$

Does John have a female friend who is in love with a not female person?

Thus, we want to know whether ‘Yes’ is the certain answer to the query:

$$\exists \text{friend}. (\text{Female} \sqcap \exists \text{loves}. \neg \text{Female})(\text{john})$$
To this end we check whether
\[ \mathcal{A} \cup \{\neg \exists \text{friend}. (\text{Female} \land \exists \text{loves.} \neg \text{Female})(\text{john})\} \]
has a model. If not, then ‘Yes’ is indeed the certain answer to

\[ \exists \text{friend}. (\text{Female} \land \exists \text{loves.} \neg \text{Female})(\text{john}) \]

Transformation into negation normal form gives:

\[ \forall \text{friend}. (\neg \text{Female} \lor \forall \text{loves.} \text{Female})(\text{john}) \]
Thus, we apply the tableau to the constraint system

\[ A^* \cup \{\text{john : } \forall \text{friend}.(\neg \text{Female} \sqcup \forall \text{loves.Female})\} \]

given by

1. (john, susan) : friend
2. (john, andrea) : friend
3. (susan, andrea) : loves
4. (andrea, bill) : loves
5. susan : Female
6. bill : \neg Female
7. john : \forall \text{friend}.(\neg \text{Female} \sqcup \forall \text{loves.Female})
Example

Two applications of the rule $\rightarrow_\forall$ give the additional constraints:

$$\text{susan} : (\neg \text{Female} \sqcup \forall \text{loves.Female})$$

and

$$\text{andrea} : (\neg \text{Female} \sqcup \forall \text{loves.Female})$$

We now apply the rule $\rightarrow_\forall$ to the first constraint:

- Adding the constraint susan : $\neg \text{Female}$ results in a clash since we have already susan : $\text{Female} \in \mathcal{A}^*$. 

- Thus we add the constraint susan : $\forall \text{loves.Female}$ to the constraint system.
Example

We now apply $\rightarrow_\forall$ to

\[ \text{susan} : \forall \text{loves. Female}, \quad (\text{susan}, \text{andrea}) : \text{loves} \]

and add

\[ \text{andrea} : \text{Female} \]

to the constraint system. We apply $\rightarrow_\forall$ to

\[ \text{andrea} : (\neg \text{Female} \sqcup \forall \text{loves. Female}) \]

- Adding andrea : $\neg$Female to the constraint systems results in a clash since andrea : Female is in the constraint system.

- Thus we add the constraint andrea : $\forall$loves.Female to the constraint system.
Example

Now we apply $\rightarrow_\forall$ to

$$\forall \text{loves.Female, } (\text{andrea, bill}) : \text{loves}$$

and add

$$\text{bill : Female}$$

to the constraint system. But this results in a clash since $\text{bill : } \neg \text{Female}$ is already in the constraint system.

It follows that every sequence of completion rule application results in a clash.