Using ABoxes to store Data

ABoxes (Assertion Boxes)

Knowledge Base (KB)

TBox (terminological box, schema)

$$\text{Man} \equiv \text{Human} \sqcap \text{Male}$$
$$\text{HappyFather} \equiv \text{Man} \sqcap \exists \text{hasChild}$$
...

ABox (assertion box, data)

$$\text{john} : \text{Man}$$
$$(\text{john}, \text{mary}) : \text{hasChild}$$
...

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; C可以是complex concept
- $r(a,b)$ says that (a,b) is an instance of r. 但Database中不可以.

ABoxes generalize database instances in which only ground sentences

不允许逻辑符号出现

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

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 **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

- **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 **Mod**$(\mathcal{A})$.

Let $F(x_1, \ldots, x_n)$ be an FOPL query. Then $(a_1, \ldots, a_n)$ in **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 $\mathcal{I} \in$ **Mod**$(\mathcal{A})$. 所有A的model

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

$$\textbf{certanswer}(F(x_1, \ldots, x_n), \mathcal{A}) = \{(a_1, \ldots, a_n) \mid \mathcal{A} \models F(a_1, \ldots, a_n)\}$$

FOPL Query Answering (Open World Semantics)

- 'Yes' is the certain answer to a Boolean query $\boldsymbol{F}$ if $\mathcal{I} \models \boldsymbol{F}$ for all $\mathcal{I} \in \mathbf{Mod}(\mathcal{A})$.
- 'No' is the certain answer to a Boolean query $\boldsymbol{F}$ if $\mathcal{I} \not\models \boldsymbol{F}$ for all $\mathcal{I} \in \mathbf{Mod}(\mathcal{A})$
- 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. ¬Female(bill)

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

The corresponding Boolean FOPL query is

$$F = \exists x.(\text{friend}(\text{john}, x) \wedge \text{Female}(x) \wedge \exists y.(\text{loves}(x, y) \wedge \neg\text{Female}(y)))$$

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

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

given by $\mathcal{A}$ is "don't know".

In order to prevent this, we could add

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

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 **blue**, **red**, and **green** of W such that

- the sets **blue**, **green**, and **red** are mutually disjoint;
- $\textbf{blue} \cup \textbf{red} \cup \textbf{green} = W$;
- 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 $(a, b) \in E$.
- $\textbf{Blue} \sqcup \textbf{Green} \sqcup \textbf{Red}(a) \in \mathcal{A}$ for all $a \in W$.
- $(\textbf{Blue} \rightarrow \forall r.(\textbf{Red} \sqcup \textbf{Green}))(a) \in \mathcal{A}$, for all $a \in W$;
- $(\textbf{Red} \rightarrow \forall r.(\textbf{Blue} \sqcup \textbf{Green}))(a) \in \mathcal{A}$, for all $a \in W$;
- $(\textbf{Green} \rightarrow \forall r.(\textbf{Red} \sqcup \textbf{Blue}))(a) \in \mathcal{A}$, for all $a \in W$.

Define query F by setting

$$F = \exists x((\textbf{Blue}(x) \wedge \textbf{Red}(x)) \vee (\textbf{Blue}(x) \wedge \textbf{Green}(x)) \vee (\textbf{Red}(x) \wedge \textbf{Green}(x))$$

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 $a \in \mathbf{Ind}(\mathcal{A})$ is given. We want to know whether

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

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

We can reformulate this problem as follows: Let $\mathcal{A}' = \mathcal{A} \cup \{\neg C(a)\}$. Then $a \in \mathbf{certanswer}(C(x), \mathcal{A})$ 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

把A中元素按入转化

- $a : C$ 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.

如 若 $A = \{Male(jack)\}$

则 $A^* = \{jack : Male\}$

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:

$$\exists\text{friend}.(\text{Female} \sqcap \exists\text{loves}.\neg\text{Female})(\text{john})$$

Example

To this end we check whether

$$\mathcal{A} \cup \{\neg\exists \text{friend}.(\text{Female} \sqcap \exists \text{loves}.\neg\text{Female})(\text{john})\}$$

has a model. If not, then 'Yes' is indeed the certain answer to

$$\exists \text{friend}.(\text{Female} \sqcap \exists \text{loves}.\neg\text{Female})(\text{john})$$

Transformation into negation normal form gives:

$$\forall \text{friend}.(\neg\text{Female} \sqcup \forall \text{loves}.\text{Female})(\text{john})$$

Example

Thus, we apply the tableau to the constraint system

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

given by

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

Example

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

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

and

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

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

- Adding the constraint susan : ¬Female results in a clash since we have already susan : Female $\in \mathcal{A}^*$.
- Thus we add the constraint susan : ∀loves.Female to the constraint system.

Example

We now apply $\rightarrow_{\forall}$ to

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

and add

$$\text{andrea} : \text{Female}$$

to the constraint system. We apply $\rightarrow_{\sqcup}$ to

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

- Adding andrea : ¬Female to the constraint systems results in a clash since andrea : Female is in the constraint system.
- Thus we add the constraint andrea : ∀loves.Female to the constraint system.

Example

Now we apply $\rightarrow_\forall$ to

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

and add

$$\text{bill} : \text{Female}$$

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

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