Tableaux para chequear consistencia de un concepto en $\mathcal{ALC}$ con respecto a una Tbox simple

Paula Severi

University of Leicester

Curso Noviembre-Diciembre 2018. Facultad de Ingeniería.
Universidad de la República, Montevideo, Uruguay.

Resumen

1. Definiciones. Tbox primitiva y Tbox simple.
2. Tableaux para chequear satisfactibilidad de un concepto de $\mathcal{ALC}$ con respecto a una Tbox simple.
3. Algoritmo Pspace.

Definiciones

Axiomas que son definiciones:

$$\begin{aligned} \texttt{Persona} &\equiv \texttt{Mujer} \sqcup \texttt{Hombre} \\ \texttt{Padre} &\equiv \texttt{Persona} \sqcap \exists \texttt{tieneHijo}.\texttt{Persona} \\ \texttt{Abuelo} &\equiv \texttt{Persona} \sqcap \exists \texttt{tieneHijo}.\texttt{Padre} \end{aligned}$$

Tbox definitorial: nos podemos deshacer de los símbolos que estan en la izquierda.

Reemplazo de definiciones. Método ineficiente ya que es exponencial.

$$\begin{aligned} A_0 &\equiv \forall R.A_1 \sqcap \forall S.A_1 \\ &\vdots \\ A_{n-1} &\equiv \forall R.A_n \sqcap \forall S.A_n \end{aligned}$$

Tbox primitiva

$\mathcal{T}$ es primitiva si se cumple

1. Todos los axiomas son de la forma $A \equiv C$ donde A es un nombre atómico.
2. Los nombres de concepto ocurren a lo sumo una vez en el lado izquierdo de los axiomas.
3. $\mathcal{T}$ es acíclica, i.e. no hay ningún A que se use a si mismo.

A *usa directamente* a B si $A \equiv C$ y B ocurre en C.

usar es la clausura transitiva de *usar directamente*.

Ejemplo de Tbox primitiva

$$\begin{aligned} \texttt{Persona} &\equiv \texttt{Mujer} \sqcup \texttt{Hombre} \\ \texttt{Padre} &\equiv \texttt{Persona} \sqcap \exists \texttt{tieneHijo}.\texttt{Persona} \\ \texttt{Abuelo} &\equiv \texttt{Persona} \sqcap \exists \texttt{tieneHijo}.\texttt{Padre} \end{aligned}$$

Es primitiva.

Ejemplo de Tbox que no es primitiva

$$\begin{aligned} \texttt{Persona} &\equiv \texttt{Mujer} \sqcup \texttt{Hombre} \\ \texttt{Padre} &\equiv \texttt{Persona} \sqcap \exists \texttt{tieneHijo}.\texttt{Padre} \\ \texttt{Abuelo} &\equiv \texttt{Persona} \sqcap \exists \texttt{tieneHijo}.\texttt{Padre} \end{aligned}$$

El segundo axioma tiene un ciclo.

Ejemplo de Tbox que no es primitiva

$$\begin{aligned} \texttt{Parent} &\equiv \texttt{Mother} \sqcup \texttt{Father} \\ \texttt{Parent} &\equiv \texttt{Person} \sqcap \exists \texttt{son}.(\texttt{Mother} \sqcup \texttt{Father}) \end{aligned}$$

Parent tiene dos definiciones.

Ejemplo de Tbox que no es primitiva

$$\begin{aligned} \texttt{Parent} &\equiv \exists \texttt{son}.\texttt{HumanParent} \\ \texttt{HumanParent} &\equiv \texttt{Person} \sqcap \texttt{Parent} \end{aligned}$$

`Parent` usa a `HumanParent`
`HumanParent` usa a `Parent`

Tenemos un ciclo.

Tbox simple

$\mathcal{T}$ es simple si se cumple

1. El lado izquierdo de los axiomas son nombres atómicos, i.e. los axiomas son de la forma $A \sqsubseteq C$ o $A \equiv C$.
2. Los nombres de concepto ocurren a lo sumo una vez en el lado izquierdo de los axiomas.
3. $\mathcal{T}$ es acíclica, i.e. no hay ningún A que se use a si mismo.

A usa directamente a *B* si $A \sqsubseteq C$ o $A \equiv C$ y *B* ocurre en *C*.

usar es la clausura transitiva de *usar directamente*.

Una Tbox primitiva es simple.

Ejemplo de Tbox simple

$$
\begin{aligned}
\texttt{Persona} &\equiv \texttt{Mujer} \sqcup \texttt{Hombre} \\
\texttt{Padre} &\equiv \texttt{Persona} \sqcap \exists \texttt{tieneHijo}.\texttt{Persona} \\
\texttt{Abuelo} &\equiv \texttt{Persona} \sqcap \exists \texttt{tieneHijo}.\texttt{Padre} \\
\texttt{Mujer} &\sqsubseteq \texttt{Animal} \\
\texttt{Hombre} &\sqsubseteq \texttt{Animal}
\end{aligned}
$$

Es simple.

Reglas de expansión para chequear satisfactibilidad de un concepto en $\mathcal{ALC}$ con respecto a una Tbox simple

Regla $\sqcap$:
Si $C \sqcap D \in \mathcal{L}(x)$ y $\{C, D\} \nsubseteq \mathcal{L}(x)$ entonces agregamos C y D a $\mathcal{L}(x)$.
Regla $\sqcup$:
Si $C \sqcup D \in \mathcal{L}(x)$ y $\{C, D\} \cap \mathcal{L}(x) = \emptyset$ entonces agregamos C o D a $\mathcal{L}(x)$.
Regla $\exists$: Si $\exists R.C \in \mathcal{L}(x)$ y no existe y tal que $R \in \mathcal{L}(x, y)$ y $C \in \mathcal{L}(y)$

1. creamos un nodo nuevo y
2. $\mathcal{L}(x, y) = \{R\}$
3. $\mathcal{L}(y) = \{C\}$

Regla $\forall$:
Si $\forall R.C \in \mathcal{L}(x)$ y existe y tal que $R \in \mathcal{L}(x, y)$ y $C \notin \mathcal{L}(y)$ entonces agregamos C a $\mathcal{L}(y)$.

Reglas de expansión para chequear satisfactibilidad de un concepto en $\mathcal{ALC}$ con respecto a una Tbox simple

A las reglas de expansión que teniamos agregamos las siguientes reglas:

Reglas del reemplazo perezoso (lazy unfolding)

Regla Tbox simple I:
Si $A \equiv C$ o $A \sqsubseteq C$ está en la Tbox, $A \in \mathcal{L}(x)$ and $C \notin \mathcal{L}(x)$ entonces agregamos C a $\mathcal{L}(x)$.

Regla Tbox simple II:
Si $A \equiv C$ está en la Tbox, $\neg A \in \mathcal{L}(x)$ y $\neg C \notin \mathcal{L}(x)$ entonces agregamos $\neg C$ a $\mathcal{L}(x)$.

Chequeo de satisfactibilidad de un concepto en $\mathcal{ALC}$ con respecto a una Tbox simple

Condición de terminación del algoritmo

Sea $\mathcal{T}$ una Tbox simple. El algoritmo termina cuando se llega a

- a un grafo $\mathcal{L}$ completo y sin contradicciones. En este caso, el concepto C es satisfactible con respecto a $\mathcal{T}$.
- cuando todas las posibilidades nos dan un grafo con contradicciones. En este caso, el concepto C es insatisfactible con respecto a $\mathcal{T}$.

Ejemplo

$$C_0 \equiv \begin{array}{l} \texttt{Abuelo} \sqcap \\ \forall \texttt{tieneHijo}.\neg\texttt{Mujer} \sqcap \\ \forall \texttt{tieneHijo}.\forall \texttt{tieneHijo}.\neg\texttt{Hombre} \end{array}$$

es satisfactible con respecto a

$$\begin{array}{rcl} \texttt{Persona} & \equiv & \texttt{Mujer} \sqcup \texttt{Hombre} \\ \texttt{Padre} & \equiv & \texttt{Persona} \sqcap \exists \texttt{tieneHijo}.\texttt{Persona} \\ \texttt{Abuelo} & \equiv & \texttt{Persona} \sqcap \exists \texttt{tieneHijo}.\texttt{Padre} \\ \texttt{Mujer} & \sqsubseteq & \texttt{Animal} \\ \texttt{Hombre} & \sqsubseteq & \texttt{Animal} \end{array}$$

Ejemplo

$$C_0 := \texttt{Abuelo} \sqcap \forall \texttt{tieneHijo}.\forall \texttt{tieneHijo}.(\neg \texttt{Hombre} \sqcap \neg \texttt{Mujer})$$

es insatisfactible con respecto a

$$\begin{aligned} \texttt{Persona} &\equiv \texttt{Mujer} \sqcup \texttt{Hombre} \\ \texttt{Padre} &\equiv \texttt{Persona} \sqcap \exists \texttt{tieneHijo}.\texttt{Persona} \\ \texttt{Abuelo} &\equiv \texttt{Persona} \sqcap \exists \texttt{tieneHijo}.\texttt{Padre} \\ \texttt{Mujer} &\sqsubseteq \texttt{Animal} \\ \texttt{Hombre} &\sqsubseteq \texttt{Animal} \end{aligned}$$

Satisfactibilidad de conceptos relativo a una Tbox simple

Algoritmo de complejidad Pspace

Modificamos el algoritmo para que la complejidad sea Pspace:

1. Aplicamos primero las reglas del $\sqcap$, $\sqcup$ y de lazy unfolding hasta que no se puedan aplicar mas.
2. Creamos un sucesor por cada existencial uno después del otro reusando espacio.
3. Cuando creamos un sucesor aplicamos las reglas del $\forall$ y de lazy unfolding hasta que no se puedan aplicar mas.

Bibliografía

- Schmidt-Schauß, Smolka. *Attribute Concept Descriptions with Complements.* Artifical Intelligence 1991.

- Capítulo 2 del *Description Logic Handbook: Theory, Implementation, and Applications.* 2003.

- Tesis de doctorado de Stephan Tobies.