Tableaux para chequear satisfactibilidad de un concepto de $\mathcal{ALC}$.

Paula Severi

University of Leicester

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

Resumen

1. Tableaux para chequear satisfactibilidad de un concepto de $\mathcal{ALC}$ (sin base de conocimiento).

2. Correctitud del Algoritmo.

Lógica de descripciones $\mathcal{ALC}$. Sintaxis.

- Nombres. Suponemos $\mathsf{N_C}$, $\mathsf{N_R}$ dos conjuntos infinitos numerables disjuntos.

Elementos en $\mathsf{N_C}$	Elementos en $\mathsf{N_R}$
nombres de conceptos (conceptos o clases atómicas) A, B	nombres de roles (roles o propiedades atómicas) R, S

Conceptos (conceptos compuestos)

Se denotan C, D y se definen inductivamente por medio de la gramática siguiente:

$$C, D := \bot \mid \top \mid A \mid \neg C \mid C \sqcap D \mid C \sqcup D \mid \forall R.C \mid \exists R.C$$

donde $A \in \mathsf{N_C}$ y $R \in \mathsf{N_R}$.

Interpretación.

1. Sea $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ tal que
   1. $\Delta^{\mathcal{I}}$ es un conjunto no vacío, dominio de la interpretación
   2. $\cdot^{\mathcal{I}}$ es una función de interpretación que asigna
      1. a cada nombre A, un conjunto $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ y
      2. cada nombre R una relación binaria $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
2. La función de interpretación se extiende a conceptos compuestos en forma inductiva:

$$
\begin{array}{ll}
\bot^{\mathcal{I}} & = \emptyset \\
\top^{\mathcal{I}} & = \Delta^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}} & = \Delta^{\mathcal{I}} \backslash C^{\mathcal{I}} \\
(C \sqcap D)^{\mathcal{I}} & = C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
(C \sqcup D)^{\mathcal{I}} & = C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
(\forall R.C)^{\mathcal{I}} & = \{x \mid \forall y.(x,y) \in R^{\mathcal{I}} \to \ y \in C^{\mathcal{I}}\} \\
(\exists R.C)^{\mathcal{I}} & = \{x \mid \exists y.(x,y) \in R^{\mathcal{I}} \wedge \ y \in C^{\mathcal{I}}\}
\end{array}
$$

Axiomas sobre conceptos

Sea $\mathcal{I}$ una intepretación de $\mathcal{ALC}$.

Semántica de la inclusión $C \sqsubseteq D$

$\mathcal{I}$ satisface la inclusión $C \sqsubseteq D$ si $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. Notación: $\mathcal{I} \models C \sqsubseteq D$.

Semántica de la Igualdad $C = D$

$C = D$ es una abreviación para $C \sqsubseteq D$ y $D \sqsubseteq C$, i.e.
$\mathcal{I} \models C = D$ si $\mathcal{I} \models C \sqsubseteq D$ y $\mathcal{I} \models D \sqsubseteq C$.

Problemas de Inferencia Básicos (sin Base de Conocimiento)

Este es un caso particular de los problemas ya vistos poniendo $\mathcal{K} = \emptyset$.

- C es satisfactible si existe $\mathcal{I}$ tal que $C^{\mathcal{I}} \neq \emptyset$.
- C está subsumido (incluido) en D si $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ se satisface para todo $\mathcal{I}$. Notación: $\models C \sqsubseteq D$.

Propiedad

1. C es satisfactible si y solo si C está subsumido en $\perp$.
2. C está subsumido en D si $C \sqcap \neg D$ es insatisfactible.

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Ejemplo

¿ El concepto C_0 es satisfactible?

$$C_0 := \exists R.A \sqcap \neg(\exists R.(A \sqcup B))$$

1. Convertimos la fórmula a su forma normal de negación usando las leyes de Morgan

$$\exists R.A \sqcap \forall R.(\neg A \sqcap \neg B)$$

2. Tenemos que encontrar un modelo de $C_0(x)$, i.e. tenemos que encontrar una interpretación tal que $x^{\mathcal{I}} \in C_0{}^{\mathcal{I}}$.

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \neg(\exists R.(A \sqcup B)$

$x \qquad \mathcal{L}(x) = \{\exists R.A \sqcap \forall R.(\neg A \sqcap \neg B),$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \neg(\exists R.(A \sqcup B)$

x $\quad\quad \mathcal{L}(x) = \{\exists R.A \sqcap \forall R.(\neg A \sqcap \neg B), \exists R.A, \forall R.(\neg A \sqcap \neg B)\}$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \neg(\exists R.(A \sqcup B)$

$$x \qquad \mathcal{L}(x) = \{\exists R.A \sqcap \forall R.(\neg A \sqcap \neg B), \exists R.A, \forall R.(\neg A \sqcap \neg B)\}$$

$$x \xrightarrow{R} y$$

$$y \qquad \mathcal{L}(y) = \{A\}$$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \neg(\exists R.(A \sqcup B)$

$$
\begin{array}{ll}
x & \mathcal{L}(x) = \{\exists R.A \sqcap \forall R.(\neg A \sqcap \neg B), \exists R.A, \forall R.(\neg A \sqcap \neg B)\} \\
\Big\downarrow R & \\
y & \mathcal{L}(y) = \{A, \neg A \sqcap \neg B, \}
\end{array}
$$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \neg(\exists R.(A \sqcup B)$

$$x \qquad \mathcal{L}(x) = \{\exists R.A \sqcap \forall R.(\neg A \sqcap \neg B), \exists R.A, \forall R.(\neg A \sqcap \neg B)\}$$

$$\Big\downarrow R$$

$$y \qquad \mathcal{L}(y) = \{A, \neg A \sqcap \neg B, \neg A, \neg B\}$$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \neg(\exists R.(A \sqcup B))$

$$
\begin{array}{ll}
x & \mathcal{L}(x) = \{\exists R.A \sqcap \forall R.(\neg A \sqcap \neg B), \exists R.A, \forall R.(\neg A \sqcap \neg B)\} \\
\big\downarrow R & \\
y & \mathcal{L}(y) = \{A, \neg A \sqcap \neg B, \neg A, \neg B\}
\end{array}
$$

Grafo tiene una contradicción! El concepto $C_0 := \exists R.A \sqcap \forall R.(\neg A \sqcap \neg B)$ es insatisfactible.

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Ejemplo

¿ El concepto C_0 es satisfactible?

$$C_0 := \exists R.A \sqcap \neg(\forall R.B)$$

1. Convertimos la fórmula a su forma normal de negación usando las leyes de Morgan
$$\exists R.A \sqcap \exists R.\neg B$$
2. Tenemos que encontrar un modelo de $C_0(x)$, i.e. tenemos que encontrar una interpretación tal que $x^{\mathcal{I}} \in C_0{}^{\mathcal{I}}$.

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \neg(\forall R.B)$

$x \qquad \mathcal{L}(x) = \{\exists R.A \sqcap \exists R.\neg B\}$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \neg(\forall R.B)$

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.\neg B\}$$

$$\begin{tikzcd} x \arrow[d, "R"'] \\ y \end{tikzcd}$$

$$\mathcal{L}(y) = \{A\}$$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \neg(\forall R.B)$

$$\begin{tikzcd}
x \arrow[d, "R"'] \arrow[drr, "R"] & & \\
y & & z
\end{tikzcd}$$

$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.\neg B\}$

$\mathcal{L}(y) = \{A\}$

$\mathcal{L}(z) = \{\neg B\}$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \neg(\forall R.B)$

$$
\begin{tikzcd}
x \arrow[d, "R"'] \arrow[drr, "R"] & & \\
y & & z
\end{tikzcd}
$$

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.\neg B\}$$

$$\mathcal{L}(y) = \{A\} \qquad \mathcal{L}(z) = \{\neg B\}$$

Model canónico.

$$\Delta^{\mathcal{I}} = \{x, y, z\} \quad R^{\mathcal{I}} = \{(x, y), (x, z)\}$$

El concepto $C_0 := \exists R.A \sqcap \exists R.\neg B$ es satisfactible.

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Ejemplo

¿ El concepto C_0 es satisfactible?

$$C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$$

1. Convertimos la fórmula a su forma normal de negación usando las leyes de Morgan

$$\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.\neg B)$$

2. Tenemos que encontrar un modelo de $C_0(x)$, i.e. tenemos que encontrar una interpretación tal que $x^{\mathcal{I}} \in C_0{}^{\mathcal{I}}$.

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.\neg B),$$

$$\}$$

x

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.\neg B),$$
$$\exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.\neg B)$$
$$\}$$

x

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$

$$
\begin{aligned}
\mathcal{L}(x) = \{&\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.\neg B),\\
&\exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.\neg B)\\
&\}
\end{aligned}
$$

$$
\begin{tikzcd}
 & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\
y & & z
\end{tikzcd}
$$

$\mathcal{L}(y) = \{A\}$ $\mathcal{L}(z) = \{B\}$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$

$$
\begin{aligned}
\mathcal{L}(x) = \{&\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.\neg B),\\
&\exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.\neg B)\\
&\forall R.\neg A\}
\end{aligned}
$$

$$
\begin{tikzcd}
 & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\
y & & z
\end{tikzcd}
$$

$\mathcal{L}(y) = \{A\}$ $\qquad\qquad$ $\mathcal{L}(z) = \{B\}$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.\neg B),$$

$$\exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.\neg B)$$

$$\forall R.\neg A\}$$

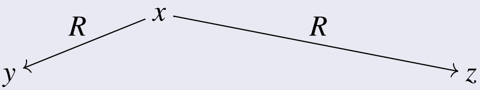
$$\begin{tikzcd}
 & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\
y & & z
\end{tikzcd}$$

$$\mathcal{L}(y) = \{A, \neg A\} \qquad \mathcal{L}(z) = \{B, \neg A\}$$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.\neg B),$$
$$\exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.\neg B)$$
$$\forall R.\neg A\}$$



$\mathcal{L}(y) = \{A, \neg A\}$ $\mathcal{L}(z) = \{B, \neg A\}$

Grafo tiene inconsistencia! Hacemos backtracking y probamos con $\forall R.\neg B$.

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$ (backtracking)

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.\neg B),$$
$$\exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.\neg B)$$
$$\}$$

$$\begin{tikzcd}
 & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\
y & & z
\end{tikzcd}$$

$$\mathcal{L}(y) = \{A\} \qquad \mathcal{L}(z) = \{B\}$$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$ (backtracking)

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.\neg B),$$
$$\exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.\neg B)$$
$$\forall R.\neg B\}$$

$$\begin{tikzcd} & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\ y & & z \end{tikzcd}$$

$\mathcal{L}(y) = \{A\}$ $\qquad$ $\mathcal{L}(z) = \{B\}$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$ (backtracking)

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.\neg B),$$
$$\exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.\neg B)$$
$$\forall R.\neg B\}$$

$$\begin{tikzcd} & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\ y & & z \end{tikzcd}$$

$\mathcal{L}(y) = \{A, \neg B\}$ $\qquad$ $\mathcal{L}(z) = \{B, \neg B\}$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$ (backtracking)

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.\neg B),$$
$$\exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.\neg B)$$
$$\forall R.\neg B\}$$

$$y \xleftarrow{R} x \xrightarrow{R} z$$

$\mathcal{L}(y) = \{A, \neg B\}$

$\mathcal{L}(z) = \{B, \neg B\}$

Este grafo también tiene inconsistencia! Las dos ramas del OR nos dieron grafos inconsistentes. El concepto $C_0 = \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$ es insatisfactible

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Ejemplo

¿ El concepto C_0 siguiente es satisfactible?

$$C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.\neg B)$$

1. Convertimos la fórmula a su forma normal de negación usando las leyes de Morgan

$$\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.B)$$

2. Tenemos que tratar de encontrar una interpretación tal que $x^{\mathcal{I}} \in C^{\mathcal{I}}$.

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.B),$$

$$\}$$

x

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.B),$$

$$\exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.B)$$

$$\}$$

x

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.B),$$
$$\exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.B)$$
$$\}$$

$$\begin{tikzcd}
 & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\
y & & z
\end{tikzcd}$$

$\mathcal{L}(y) = \{A\}$ $\qquad\qquad$ $\mathcal{L}(z) = \{B\}$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$

$$\begin{aligned}
\mathcal{L}(x) = \{&\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.B),\\
&\exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.B)\\
&\forall R.\neg A\}
\end{aligned}$$

$$\begin{tikzcd}
 & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\
y & & z
\end{tikzcd}$$

$\mathcal{L}(y) = \{A\}$ $\qquad\qquad$ $\mathcal{L}(z) = \{B\}$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$

$$\begin{aligned} \mathcal{L}(x) = \{ & \exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.B), \\ & \exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.B) \\ & \forall R.\neg A \} \end{aligned}$$

$$\begin{tikzcd} & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\ y & & z \end{tikzcd}$$

$\mathcal{L}(y) = \{A, \neg A\}$ $\qquad$ $\mathcal{L}(z) = \{B, \neg A\}$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.B)$

$$
\begin{aligned}
\mathcal{L}(x) = \{&\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.B),\\
&\exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.B)\\
&\forall R.\neg A\}
\end{aligned}
$$

$$
\begin{tikzcd}
y & x \arrow[l, "R"'] \arrow[r, "R"] & z
\end{tikzcd}
$$

$\mathcal{L}(y) = \{A, \neg A\}$ $\qquad$ $\mathcal{L}(z) = \{B, \neg A\}$

Grafo tiene inconsistencia! Hacemos backtracking y probamos con $\forall R.B$.

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.\neg B)$ (backtracking)

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.B), \exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.B) \}$$

$$\begin{tikzcd}
 & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\
y & & z
\end{tikzcd}$$

$\mathcal{L}(y) = \{A\}$ $\qquad$ $\mathcal{L}(z) = \{B\}$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.\neg B)$ (backtracking)

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.B),$$
$$\exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.B)$$
$$\forall R.B\}$$

$$\begin{tikzcd}
 & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\
y & & z
\end{tikzcd}$$

$\mathcal{L}(y) = \{A\}$ $\qquad\qquad$ $\mathcal{L}(z) = \{B\}$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.\neg B)$ (backtracking)

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.B),\ \exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.B)\ \forall R.B\}$$

$$\begin{tikzcd}
 & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\
y & & z
\end{tikzcd}$$

$$\mathcal{L}(y) = \{A, B\} \qquad \mathcal{L}(z) = \{B\}$$

Algoritmo de Tableaux para chequear satisfactibilidad de un concept en $\mathcal{ALC}$

Construcción del grafo para $C_0 := \exists R.A \sqcap \exists R.B \sqcap \neg(\exists R.A \sqcap \exists R.\neg B)$ (backtracking)

$$\mathcal{L}(x) = \{\exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.B),$$
$$\exists R.A, \exists R.B, (\forall R.\neg A \sqcup \forall R.B)$$
$$\forall R.B\}$$

$$\begin{tikzcd} & x \arrow[dl, "R"'] \arrow[dr, "R"] & \\ y & & z \end{tikzcd}$$

$\mathcal{L}(y) = \{A, B\}$ $\quad$ $\mathcal{L}(z) = \{B\}$

El modelo canónico para $C_0 = \exists R.A \sqcap \exists R.B \sqcap (\forall R.\neg A \sqcup \forall R.B)$ es:

$$\Delta^{\mathcal{I}} = \{x, y, z\} \quad A^{\mathcal{I}} = \{y\} \quad B^{\mathcal{I}} = \{y, z\} \quad R^{\mathcal{I}} = \{(x, y), (x, z)\}$$

Forma Normal de Negación

Por conveniencia transformamos la base de conocimiento a forma normal de negación.

Forma Normal de Negación (FNN)

Un concepto está en FNN es la negación ocurre únicamente delante de conceptos átomicos.

Ejemplo:

- $\texttt{Abuela} \sqcap \neg\texttt{Madre}$ está en FFN.
- $\neg(\neg\texttt{Abuela} \sqcup \texttt{Madre})$ no está en FFN .

Conversión a FNN

$$
\begin{aligned}
\mathsf{FNN}(A) &= A\\
\mathsf{FNN}(C \sqcup D) &= \mathsf{FNN}(C) \sqcup \mathsf{FNN}(D)\\
\mathsf{FNN}(C \sqcap D) &= \mathsf{FNN}(C) \sqcap \mathsf{FNN}(D)\\
\mathsf{FNN}(\forall R.C) &= \forall R.\mathsf{FNN}(C)\\
\mathsf{FNN}(\exists R.C) &= \exists R.\mathsf{FNN}(C)\\
\mathsf{FNN}(\neg A) &= \neg A\\
\mathsf{FNN}(\neg\neg C) &= \mathsf{FNN}(C)\\
\mathsf{FNN}(\neg(C \sqcup D)) &= \mathsf{FNN}(\neg C) \sqcap \mathsf{FNN}(\neg D)\\
\mathsf{FNN}(\neg(C \sqcap D)) &= \mathsf{FNN}(\neg C) \sqcup \mathsf{FNN}(\neg D)\\
\mathsf{FNN}(\neg(\forall R.C)) &= \exists R.\mathsf{FNN}(\neg C)\\
\mathsf{FNN}(\neg(\exists R.C)) &= \forall R.\mathsf{FNN}(\neg C)
\end{aligned}
$$

Grafo

- un conjunto de nodos etiquetados cuyas etiquetas son nombres de individuos o nombres de variables,
- arcos dirigidos entre nodos,
- dado un nodo con etiqueta x, $\mathcal{L}(x)$ es un conjunto de conceptos,
- por cada par de nodos x, y, se tiene que $\mathcal{L}(x, y)$ es un conjunto de nombres de roles.

Notación. $\mathcal{L}$ denota el grafo junto con los conjuntos de conceptos asociados a los nodos y los conjuntos de roles asociados a los arcos.

Algoritmo de Tableaux para chequear satisfactibilidad de un concepto en $\mathcal{ALC}$

Inicialización

Dado un concepto C_0 en FNN, el grafo inicial para C_0 se define:

1. creamos un único nodo x_0 tal que $\mathcal{L}_0(x_0) = \{C_0\}$.

Después de la inicialización, el algoritmo procede a aplicar las reglas de expansión en forma no determinística.

Reglas de expansión para chequear satisfactibilidad de un concepto en $\mathcal{ALC}$

Regla $\sqcap$: Si $C \sqcap D \in \mathcal{L}(x)$ y $\{C, D\} \not\subseteq \mathcal{L}(x)$ 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 $C \in \mathcal{L}(y)$ entonces
1. Agregamos un nodo con etiqueta y (donde y es un nombre nuevo),
2. ponemos $\mathcal{L}(x, y) = \{R\}$,
3. ponemos $\mathcal{L}(y) = \{C\}$.

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 $\mathcal{L}(y)$

No Determinismo en el Algoritmo de tableaux para $\mathcal{ALC}$

No determinismo:

1. Causado por que regla aplicar en el próximo paso. El orden de las reglas se puede intercambiar (aunque sea mas ineficiente una opción que otra).

2. Causado por $\mathcal{L}(x) \leftarrow C$ o $\mathcal{L}(x) \leftarrow D$ en la regla de disjunción. La elección de C puede dar una contradicción. Hay que hacer *backtracking* y tratar el caso de D.

Tableaux para chequear satisfactibilidad de un concepto

Aplicamos las reglas de expansión a nuestro grafo original $\mathcal{L}_0(x_0) = \{C_0\}$.

$$
\begin{tikzcd}
 & & \mathcal{L}_0 \arrow[d, "\rho \neq \sqcup"] & & \\
 & & \mathcal{L}_1 \arrow[dl, "\sqcup"'] \arrow[dr, "\sqcup"] & & \\
 & \mathcal{L}_{21} \arrow[dl, "\sqcup"'] \arrow[d, "\sqcup"] & & \mathcal{L}_{22} \arrow[d, "\rho \neq \sqcup"] & \\
\mathcal{L}_{31} & \mathcal{L}_{32} & & \mathcal{L}_{33} &
\end{tikzcd}
$$

Si todas las posibilidades nos dan grafos $\mathcal{L}_{31}$, $\mathcal{L}_{32}$, $\mathcal{L}_{33}$ que tienen contradicciones, entonces la base original es inconsistente.

Notación. Sea $\rho \in \{\sqcap, \sqcup, \exists, \forall\}$ una de las reglas del algoritmo. $\mathcal{L} \rightarrow_\rho \mathcal{L}'$ si $\mathcal{L}'$ se obtiene de $\mathcal{L}$ luego de aplicar la regla ρ.

¿ Cuando paramos de aplicar las reglas?

- $\mathcal{L}$ tiene una contradicción cuando existen x y A tal que $\{A, \neg A\} \subseteq \mathcal{L}(x)$.
- $\mathcal{L}$ está completo cuando no se le pueden aplicar mas reglas.

Condición de terminación del algoritmo

El algoritmo termina cuando se llega a

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

Pseudocódigo

consistente($\mathcal{L}$) =
Si $\mathcal{L}$ contiene una contradicción entonces falso
Sino Si $\mathcal{L}$ está completo entonces verdadero
Sino Si $C \sqcap D \in \mathcal{L}(x)$ y $\{C, D\} \not\subseteq \mathcal{L}(x)$ entonces
  consistente($\mathcal{L}'$)
   donde $\mathcal{L}'$ es el resultado de agregar C y D a $\mathcal{L}(x)$
Si $C \sqcup D \in \mathcal{L}(x)$ y $\{C, D\} \cap \mathcal{L}(x) = \emptyset$ entonces
  consistente($\mathcal{L}_1$) or consistente($\mathcal{L}_2$)
   donde $\mathcal{L}_1$ y $\mathcal{L}_2$ son el resultado de agregar respectivamente C y D a $\mathcal{L}(x)$.
... etc.

Notación

$|C|$ es el número de símbolos de C

$\mathsf{subconcepts}(C)$ es el conjunto de subconceptos de C

Subconcepto. Concepto que forma parte de la sintaxis de otro concepto.

Propiedades

- $\mathcal{L}$ es un árbol cuya raiz es x_0.
- $\mathcal{L}(x) \subseteq \mathsf{subconcepts}(C_0)$ para todo nodo x.

Terminación del algoritmo de Tableaux

Lema

1. Para todo nodo x, la cardinalidad de $\mathcal{L}(x)$ esta acotado por $|C|$
2. El largo de un camino de $\mathcal{L}$ esta acotado por $|C|$
3. El grado de salido del grafo $\mathcal{L}$ tambien está acotado por $|C|$.

Terminación del algoritmo de Tableaux

Lema

1. Para todo nodo x, la cardinalidad de $\mathcal{L}(x)$ esta acotado por $|C|$
2. El largo de un camino de $\mathcal{L}$ esta acotado por $|C|$
3. El grado de salido del grafo $\mathcal{L}$ tambien está acotado por $|C|$.

Demostración

1. $\mathcal{L}(x) \subseteq \mathsf{subconcepts}(C_0)$
2. $l(x)$ maximo número de restricciones existenciales o universales anidadas. $l(x) \leq |C|$ y $l(x) < l(y)$ si existe un arco de x a y.
3. Se genera un sucesor por cada $\exists R.C$ que está en $\mathcal{L}(x)$. Se deduce de la parte 1).

Terminación

Terminación

Cualquier secuencia de aplicaciones de las reglas de expansión que empieza en $\mathcal{L}(x_0) = \{C_0\}$ terminan después de un número finito de pasos.

Terminación

Terminación

Cualquier secuencia de aplicaciones de las reglas de expansión que empieza en $\mathcal{L}(x_0) = \{C_0\}$ terminan después de un número finito de pasos.

El tamaño de $\mathcal{L}$ es la suma de las cardinalidades de $\mathcal{L}(x)$ para todo node x del grafo (árbol) $\mathcal{L}$.

Demostración

Por el Lema, sabemos que el tamaño de todos los árboles en cualquier secuencia

$$\mathcal{L}_0 \to \mathcal{L}_1 \to \mathcal{L}_2 \to \dots$$

estan acotados.

Terminación

Terminación

Cualquier secuencia de aplicaciones de las reglas de expansión que empieza en $\mathcal{L}(x_0) = \{C_0\}$ terminan después de un número finito de pasos.

El tamaño de $\mathcal{L}$ es la suma de las cardinalidades de $\mathcal{L}(x)$ para todo node x del grafo (árbol) $\mathcal{L}$.

Demostración

Por el Lema, sabemos que el tamaño de todos los árboles en cualquier secuencia

$$\mathcal{L}_0 \to \mathcal{L}_1 \to \mathcal{L}_2 \to \dots$$

estan acotados. Cada regla de expansión agrega un concepto a $\mathcal{L}(x)$ para algun x o agrega un nodo al árbol. Esto quiere decir que el tamaño del grafo aumenta (nunca queda igual). La secuencia tiene que ser finita.

Correctitud del algoritmo de Tableaux

Modelo de un grafo

$\mathcal{I} \models \mathcal{L}$ si $x^{\mathcal{I}} \in C^{\mathcal{I}}$ para todo $C \in \mathcal{L}(x)$ y todo x del grafo $\mathcal{L}$.

Preservación de consistencia

1. Sea $\mathcal{L} \rightarrow_{\rho} \mathcal{L}'$ y $\rho \neq \sqcup$. Entonces, $\mathcal{I} \models \mathcal{L}$ si y solo si $\mathcal{I} \models \mathcal{L}'$.
2. Sea $\mathcal{L} \rightarrow_{\sqcup} \mathcal{L}_1$ y $\mathcal{L} \rightarrow_{\sqcup} \mathcal{L}_2$. Entonces, $\mathcal{I} \models \mathcal{L}$ si y solo si $\mathcal{I} \models \mathcal{L}_1$ o $\mathcal{I} \models \mathcal{L}_2$.

Adecuación

Si C_0 es satisfactible entonces las reglas de expansión se pueden aplicar a $\mathcal{L}_0(x) = \{C_0\}$ terminando en un grafo que es completo y sin contradicciones.

Demostración

Sea $\mathcal{I}$ tal que $C^{\mathcal{I}} \neq \emptyset$. Entonces $\mathcal{I} \models \mathcal{L}_0$ donde $\mathcal{L}_0$ es el grafo inicial del algoritmo de Tableaux. Supongamos que de $\mathcal{L}_0$ despues de aplicar un numero finito de reglas de expansión, obtenemos un grafo $\mathcal{L}_n$ con contradicciones. Usando la preservación de consistencia, tendriamos que $\mathcal{I} \models \mathcal{L}_n$. Esto es una contradicción, ya que un grafo con una contradicción no puede tener un modelo.

Correctitud del Algoritmo

Completitud

Si las reglas de expansión aplicadas a $\mathcal{L}_0(x) = \{C_0\}$ terminan en un grafo que es completo y sin contradicciones entonces C_0 es satisfactible.

Interpretación canónica

La interpretación canónica $\mathcal{I}_c$ inducida por $\mathcal{L}$ se define:

- $\Delta^{\mathcal{I}_c} = \{x \mid x \text{ es un nodo de } \mathcal{L}\}$
- $A^{\mathcal{I}_c} = \{x \mid A \in \mathcal{L}(x)\}$
- $R^{\mathcal{I}_c} = \{(x, y) \mid \text{existe un arco de } x \text{ a } y \text{ etiquetado con } R\}$

Demostración de completitud

Probamos por inducción en C usando el hecho de que $\mathcal{L}$ es completo: $x \in C^{\mathcal{I}_c}$ si y solo si $C \in \mathcal{L}(x)$ para todo nodo x.

Complejidad

El algoritmo de tableaux para $\mathcal{ALC}$ puede necesitar tiempo y espacio exponencial.

Conceptos C_n definidos por inducción

$$\begin{aligned} C_1 &= \exists R.A \sqcap \exists R.B \\ C_{n+1} &= \exists R.A \sqcap \exists R.B \sqcap \forall R.C_n \end{aligned}$$

El grafo de C_n tiene $2^{n+1} - 1$ nodos.

Modificación del Algoritmo de Tableaux para obtener complejidad PSpace

En vez de guardar todo el grafo, solamente guardamos un camino.

Algoritmo PSpace- Método de las trazas

1. Aplicamos primero las reglas del $\sqcap$ y $\sqcup$ hasta que no se puedan aplicar mas.
2. Creamos un sucesor por cada existencial uno después del otro reusando espacio (se guarda una rama o traza del árbol).
3. Cuando creamos un sucesor aplicamos las reglas del $\forall$ hasta que no se pueda aplicar mas.

Ejemplo

Probar que

$$C_0 = \exists R.\exists S.A \sqcap \exists R.B \sqcap \forall R.\forall S.B$$

es satisfactible usando Tableaux en espacio polinomial.

Método de las trazas funciona para $\mathcal{ALC}$

Funciona porque las etiquetas de los nodos no cambian al expandir los sucesores.

Ejemplo con roles inversos

Método de las trazas no funciona con roles inversos

Ejemplo

$$C_0 = \exists R.A \sqcap \exists S.B \sqcap \forall S^{-}.\forall R.\neg A$$

La contradicción no se encuentra si borramos la rama del $\exists R.A$ para ahorrar espacio.

Algoritmo de Tableaux tiene complejidad Pspace

Método de las trazas tiene complejidad Pspace

1. Una contradicción para x en $\mathcal{L}(x)$ se genera antes de generar cualquier sucesor y de x y luego de generar y, no va a ocurrir (esa parte del grafo, no cambia).
2. una vez que sabemos que el subgrafo comenzando en y no tiene contradicciones se puede borrar.

Bibliografía

- Schmidt-Schauß, Smolka. *Attribute Concept Descriptions with Complements.* Artifical Intelligence 1991.
- Capítulo 2 del *Description Logic Handbook: Theory, Implementation, and Applications.* 2003.