Beschreibungslogik
Kapitel 6: Effiziente Beschreibungslogiken

Sommersemester 2018 Thomas Schneider

AG Theorie der künstlichen Intelligenz (TdKI)

http://tinyurl.com/ss18-bl

Vorlesungsübersicht

Ziel des Kapitels

Für manche Anwendungen ist $\mathcal{ALC}$ zu komplex:

- Auch hoch-optimierte Reasoner können sehr große und komplexe Ontologien oft nicht verarbeiten (oder nur nach intensivem Tuning)
- In der Anfragebeantwortung muss man oft mit sehr großen Datenmengen umgehen und braucht schnelle Antworten (Kapitel 7)

Wir betrachten die Beschreibungslogik $\mathcal{EL}$:

- viel weniger ausdrucksstark als $\mathcal{ALC}$, Basisoperatoren nur $\sqcap$ und $\exists r.C$
- Erfüllbarkeit und Subsumtion in Polyzeit entscheidbar

Kapitel 6: Effiziente Beschreibungslogiken

Kapitel 6: Effiziente Beschreibungslogiken

$\mathcal{EL}$

Definition 6.1 ($\mathcal{EL}$)

Ein **$\mathcal{EL}$-Konzept** ist ein $\mathcal{ALC}$-Konzept, in dem nur die Konstruktoren $\top$, $\sqcap$ und $\exists r.C$ verwendet werden.

$\mathcal{EL}$ ist beliebt für biomedizinische Ontologien
(die sind oft groß und mit hohem Abstraktionsgrad):

$$\begin{aligned} \text{Perikardium} &\sqsubseteq \text{Gewebe} \sqcap \exists \text{teilVon.Herz} \\ \text{Perikarditis} &\equiv \text{Entzündung} \sqcap \exists \text{ort.Perikardium} \\ \text{Entzündung} &\sqsubseteq \text{Krankheit} \sqcap \exists \text{wirktAuf.Gewebe} \end{aligned}$$

SNOMED CT ist in unwesentlicher Erweiterung von $\mathcal{EL}$ formuliert.

$\mathcal{EL}$ ist Grundlage des **EL-Profils von OWL 2.**

Simulation

Intuitiv: $\mathcal{EL}$ ist die „Hälfte von $\mathcal{ALC}$“;
Simulation entspricht der **„Hälfte von Bisimulation“.**

Definition 6.2 (Simulation)

Seien $\mathcal{I}_1$ und $\mathcal{I}_2$ Interpretationen.

Relation $\rho \subseteq \Delta^{\mathcal{I}_1} \times \Delta^{\mathcal{I}_2}$ ist **Simulation** von $\mathcal{I}_1$ nach $\mathcal{I}_2$, wenn gilt:

1. Wenn $d_1 \ \rho \ d_2$, dann gilt für alle Konzeptnamen A:

$$d_1 \in A^{\mathcal{I}_1} \qquad \textbf{impliziert} \qquad d_2 \in A^{\mathcal{I}_2}$$

2. Wenn $d_1 \ \rho \ d_2$ und $(d_1, d_1') \in r^{\mathcal{I}_1}$ für beliebigen Rollennamen r, dann gibt es ein $d_2' \in \Delta^{\mathcal{I}_2}$ mit $d_1' \ \rho \ d_2'$ und $(d_2, d_2') \in r^{\mathcal{I}_2}$.

T 6.1

Beachte:
Im Gegensatz zu Bisimulationen sind **Simulationen gerichtet.**

Simulation

Seien $\mathcal{I}_1$ und $\mathcal{I}_2$ Interpretationen, $d_1 \in \Delta^{\mathcal{I}_1}$, $d_2 \in \Delta^{\mathcal{I}_2}$.

Wir schreiben $(\mathcal{I}_1, d_1) \precsim (\mathcal{I}_2, d_2)$,
wenn es Simulation ρ von $\mathcal{I}_1$ nach $\mathcal{I}_2$ gibt mit $d_1 \; \rho \; d_2$
(wir sagen: d_1 **wird simuliert von** d_2).

Theorem 6.3

Seien $\mathcal{I}_1, \mathcal{I}_2$ Interpretationen, $d_1 \in \Delta^{\mathcal{I}_1}$ und $d_2 \in \Delta^{\mathcal{I}_2}$.

Wenn $(\mathcal{I}_1, d_1) \precsim (\mathcal{I}_2, d_2)$, dann gilt für alle $\mathcal{EL}$-Konzepte C:

$$d_1 \in C^{\mathcal{I}_1} \quad \textbf{impliziert} \quad d_2 \in C^{\mathcal{I}_2}$$

T 6.2

Simulation vs. Bisimulation; Ausdrucksstärke

Intuitiv: Wenn $(\mathcal{I}_1, d_1) \precsim (\mathcal{I}_2, d_2)$ und $(\mathcal{I}_2, d_2) \precsim (\mathcal{I}_1, d_1)$, dann kann $\mathcal{EL}$ nicht zwischen d_1 und d_2 „unterscheiden“.

Achtung:

Bisimulation und wechselseitige Simulation sind **nicht dasselbe:**

Lemma 6.4

Es gibt $(\mathcal{I}_1, d_1)$ und $(\mathcal{I}_2, d_2)$, so dass:

- $(\mathcal{I}_1, d_1) \precsim (\mathcal{I}_2, d_2)$ und $(\mathcal{I}_2, d_2) \precsim (\mathcal{I}_1, d_1)$
- $(\mathcal{I}_1, d_1) \not\sim (\mathcal{I}_2, d_2)$

T 6.3

Man kann nun wieder Nicht-Ausdrückbarkeitsresultate zeigen:

Lemma 6.5

Das $\mathcal{ALC}$-Konzept $\forall r.A$ ist **nicht** in $\mathcal{EL}$ ausdrückbar.

T 6.3 Forts.

Schlussfolgerungsprobleme in $\mathcal{EL}$

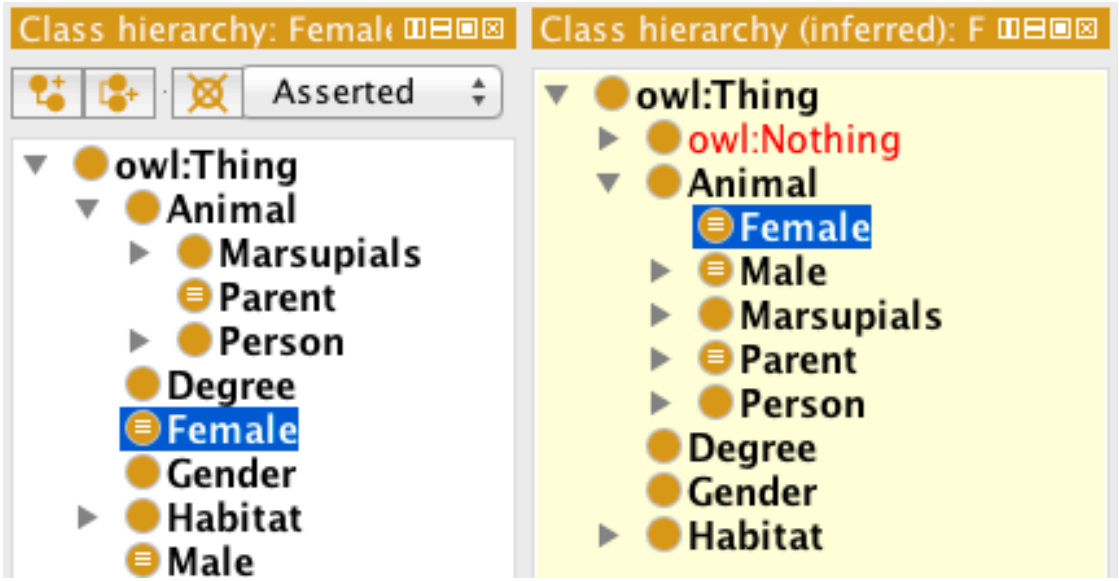
In $\mathcal{EL}$ ist Erfüllbarkeit **kein** interessantes Schlussfolgerungsproblem:

Lemma 6.6

Jedes $\mathcal{EL}$-Konzept ist erfüllbar bzgl. jeder TBox.

T 6.4

Darum konzentrieren wir uns auf **Subsumtion.**



Kapitel 6: Effiziente Beschreibungslogiken

Intuition

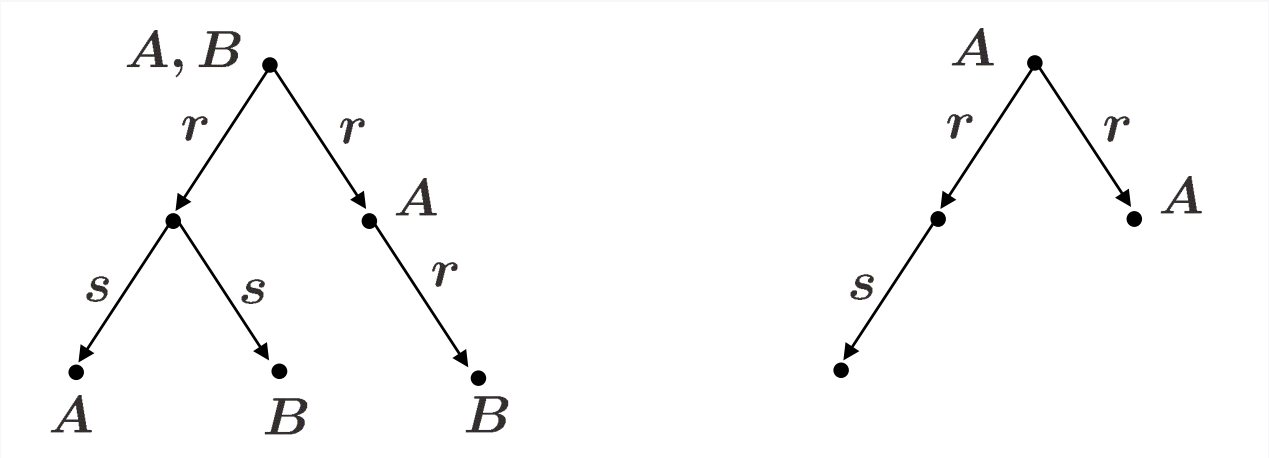
Eine Subsumtion $C \sqsubseteq D$ gilt in $\mathcal{EL}$ im Prinzip genau dann, wenn man **D syntaktisch „in C wiederfindet“.**

z. B.:

$$C = A \sqcap B \sqcap \exists r.(\exists s.A \sqcap \exists s.B) \sqcap \exists r.(A \sqcap \exists r.B)$$

$$D = A \sqcap \exists r.\exists s.\top \sqcap \exists r.A$$

Konzepte dargestellt als Bäume: „Wiederfinden“ entspricht **Simulation** von D-Baum in C-Baum **(Richtung!)**



Kanonisches Modell

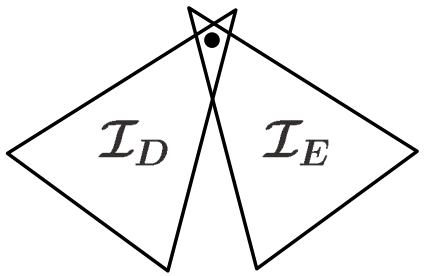
Definition 6.7

Wir ordnen induktiv jedem $\mathcal{EL}$-Konzept C eine Interpretation $\mathcal{I}_C$ zu:

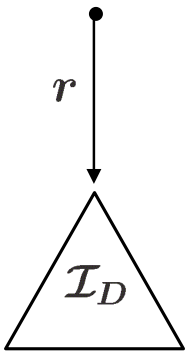
$C = \top$ Modell $\mathcal{I}_C$:

$C = A$ Modell $\mathcal{I}_C$:

$C = D \sqcap E$ Modell $\mathcal{I}_C$:



$C = \exists r.D$ Modell $\mathcal{I}_C$:



Kanonisches Modell

Wir nennen $\mathcal{I}_C$ **kanonisches Modell**, bezeichnen Wurzel stets mit d_W .

Man sieht leicht, dass $\mathcal{I}_C$ wirklich ein **Modell** ist:

Lemma 6.8

Für alle $\mathcal{EL}$-Konzepte C gilt:

Die Interpretation $\mathcal{I}_C$ ist Modell von C mit $d_W \in C^{\mathcal{I}_C}$.

T 6.5

Die zentrale Eigenschaft kanonischer Modelle:

Lemma 6.9

Für alle $\mathcal{EL}$-Konzepte C, Interpretationen $\mathcal{I}$ und $e \in \Delta^{\mathcal{I}}$ gilt:

$$e \in C^{\mathcal{I}} \quad \textbf{gdw.} \quad (\mathcal{I}_C, d_W) \precsim (\mathcal{I}, e)$$

T 6.6

Charakterisierung von Subsumtion

Wir zeigen nun, dass $C \sqsubseteq D$ gdw. man $\mathcal{I}_D$ in $\mathcal{I}_C$ "wiederfindet":

Lemma 6.10

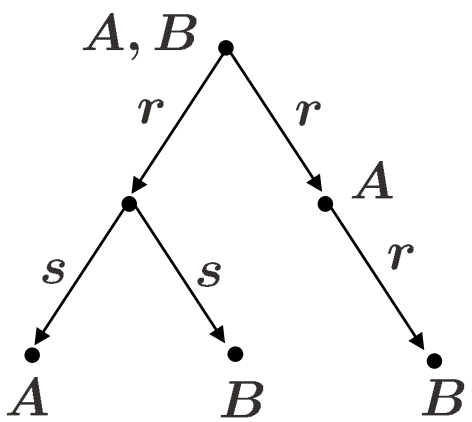
Für alle $\mathcal{EL}$-Konzepte C, D gilt:

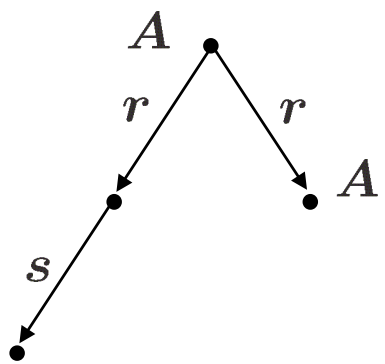
$$C \sqsubseteq D \quad \textbf{gdw.} \quad (\mathcal{I}_D, d_W) \precsim (\mathcal{I}_C, d_W)$$

T 6.7

z. B.: $C = A \sqcap B \sqcap \exists r.(\exists s.A \sqcap \exists s.B) \sqcap \exists r.(A \sqcap \exists r.B)$

$D = A \sqcap \exists r.\exists s.\top \sqcap \exists r.A$





Entscheidungsverfahren für Subsumtion

Nun folgt:

Theorem 6.11

Subsumtion in $\mathcal{EL}$ kann in polynomieller Zeit entschieden werden.

T 6.8

Beweis. (Recht einfacher) Algorithmus:

(1) Konstruiere $\mathcal{I}_C$ und $\mathcal{I}_D$ in polynomieller Zeit.

(2) Überprüfe in polynomieller Zeit, ob $(\mathcal{I}_D, d_W) \precsim (\mathcal{I}_C, d_W)$:

- **(a)** Berechne **maximale Simulation** ρ von $\mathcal{I}_D$ nach $\mathcal{I}_C$ (siehe Übungsblatt 5)
- **(b)** Teste, ob $(d_W, d_W) \in \rho$. □

Kapitel 6: Effiziente Beschreibungslogiken

Intuition

Wir verwenden ein so genanntes **konsequenzbasiertes** Verfahren.

Grundidee:

- Mit Hilfe von **Regeln** werden zur TBox nach und nach neue Konzeptinklusionen hinzugefügt.
- Am Ende muss man dann nur noch nachschauen, ob die gewünschte Subsumtion in der TBox **explizit enthalten** ist.

In der Praxis haben sich derartige Verfahren als **äußerst effizient** herausgestellt.

Sie sind verwandt mit **Sequenzenkalkülen** aus der klassischen Logik.

Vereinfachende Annahme 1

Betrachten **o. B. d. A.** nur Subsumtion von Konzept**namen** bzgl. TBoxen.

Seien $\mathcal{T}$ $\mathcal{EL}$-TBox und C, D $\mathcal{EL}$-Konzepte (beliebig).

Um $\mathcal{T} \models C \sqsubseteq D$ zu entscheiden:

- Nimm zwei neue Konzept**namen** A_C, A_D .
- Füge zu $\mathcal{T}$ hinzu: $A_C \sqsubseteq C$ und $D \sqsubseteq A_D \rightsquigarrow$ TBox $\mathcal{T}'$.
- Entscheide, ob $\mathcal{T}' \models A_C \sqsubseteq A_D$ gilt.

Korrektheit dieses Verfahrens:

Lemma 6.12

$\mathcal{T} \models C \sqsubseteq D$ gdw. $\mathcal{T}' \models A_C \sqsubseteq A_D$ **(Übung)**

Lemma 6.12 liefert **Polyzeit-Reduktion**

von Subsumtion zwischen beliebigen Konzepten bzgl. TBoxen
auf Subsumtion zwischen Konzeptnamen bzgl. TBoxen.

Annahme 2: Normalform

Betrachten **o. B. d. A.** nur TBoxen in folgender Normalform:

Definition 6.13

Eine TBox ist in **Normalform (NF)**, wenn sie nur Inklusionen der Form

$$A_1 \sqcap \cdots \sqcap A_n \sqsubseteq A \qquad A \sqsubseteq \exists r.A_1 \qquad \exists r.A \sqsubseteq A_1$$

enthält, wobei $A, A_1, \ldots, A_n$ Konzept**namen** oder $\top$ sind.

Lemma 6.14

Jede $\mathcal{EL}$-TBox $\mathcal{T}$ kann in polynomieller Zeit in eine TBox $\mathcal{T}'$ in NF gewandelt werden, so dass für alle Konzeptnamen A, B in $\mathcal{T}$ gilt:

$$\mathcal{T} \models A \sqsubseteq B \quad \text{gdw.} \quad \mathcal{T}' \models A \sqsubseteq B \qquad (*)$$

Wenn $(*)$ gilt, sagen wir: $\mathcal{T}'$ ist **konservative Erweiterung** von $\mathcal{T}$.

Um die NF herzustellen, wenden wir **Normalisierungsregeln** an. **T 6.9**

Normalisierung

NF1 $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq E \rightsquigarrow$
$C_i \sqsubseteq A_{C_i}$, $C_1 \sqcap \cdots \sqcap C_{i-1} \sqcap A_{C_i} \sqcap C_{i+1} \sqcap \cdots \sqcap C_n \sqsubseteq E$
wenn C_i Existenzrestriktion

NF2 $\exists r.C \sqsubseteq E \rightsquigarrow C \sqsubseteq A_C,\ \exists r.A_C \sqsubseteq E$

NF3 $C \sqsubseteq \exists r.D \rightsquigarrow C \sqsubseteq A_C,\ A_C \sqsubseteq \exists r.D$

NF4 $A \sqsubseteq \exists r.C \rightsquigarrow A \sqsubseteq \exists r.A_C,\ A_C \sqsubseteq C$

NF5 $A \sqsubseteq C_1 \sqcap C_2 \rightsquigarrow A \sqsubseteq C_1,\ A \sqsubseteq C_2$

} wenn C weder Konzeptname noch $\top$

A ist Konzept**name**; C_i, D, E sind **beliebige** Konzepte; $A_{C_{(i)}}$ sind **neue** Konzept**namen.**

Lemma 6.15

Jede $\mathcal{EL}$-TBox $\mathcal{T}$ kann durch **linear viele** Regelanwendungen in TBox in NF transformiert werden, die konservative Erweiterung von $\mathcal{T}$ ist.

T 6.10

Subsumtion mit TBox

Der Algorithmus beginnt mit der ursprünglichen TBox und wendet dann erschöpfend folgende Regeln an.

R1 $\dfrac{}{A \sqsubseteq A}$ wenn A in $\mathcal{T}$ vorkommt

R2 $\dfrac{}{A \sqsubseteq \top}$ wenn A in $\mathcal{T}$ vorkommt

R3 $$\frac{A \sqsubseteq A_1 \quad \cdots \quad A \sqsubseteq A_n \qquad A_1 \sqcap \cdots \sqcap A_n \sqsubseteq B}{A \sqsubseteq B}$$

R4 $$\frac{A \sqsubseteq \exists r.A_1 \qquad A_1 \sqsubseteq B_1 \qquad \exists r.B_1 \sqsubseteq B}{A \sqsubseteq B}$$

T 6.11

Beachte: $A, A_1, \ldots, A_n, B$ sind Konzept**namen oder** $\top$.

Saturierung

Für eine $\mathcal{EL}$-TBox $\mathcal{T}$ sei $\mathcal{T}^*$ das Ergebnis erschöpfender Regelanwendung. Wir nennen $\mathcal{T}^*$ die **Saturierung** von $\mathcal{T}$.

$\mathcal{T}^*$ macht alle Subsumtionen zwischen Konzeptnamen explizit:

Theorem 6.16

Für alle Konzeptnamen A, B in $\mathcal{T}$ gilt:

$$\mathcal{T} \models A \sqsubseteq B \quad \textbf{gdw.} \quad A \sqsubseteq B \in \mathcal{T}^*$$

Der Algorithmus **klassifiziert** also die Konzeptnamen **vollständig;** berechnet nicht nur eine einzelne Subsumtion.

Vor dem Beweis des Theorems: Terminierung.

Terminierung des Algorithmus

Theorem 6.17

Die Konstruktion von $\mathcal{T}^*$ **terminiert** nach $\mathcal{O}(|\mathcal{T}|^2)$ vielen Regelanwendungen.

Beweis.

Jede Regelanwendung erzeugt neue Konzeptinklusion $A \sqsubseteq B$, mit A, B Konzeptnamen aus der ursprünglichen TBox $\mathcal{T}$ (oder $\top$).

Es gibt nur $\mathcal{O}(|\mathcal{T}|^2)$ viele solche Inklusionen. □

Korrektheit des Algorithmus

Wir beweisen nun Theorem 6.16. Korrektheit:

Lemma 6.18

Für alle Konzeptnamen A, B in $\mathcal{T}$ gilt:

$$A \sqsubseteq B \in \mathcal{T}^* \quad \textbf{impliziert} \quad \mathcal{T} \models A \sqsubseteq B$$

Beweis.

Sei $\mathcal{T} = \mathcal{T}_0, \mathcal{T}_1, \ldots, \mathcal{T}_n = \mathcal{T}^*$ die durch Regelanwendung erzeugte Folge von TBoxen. Es genügt zu zeigen:

Beh.: Für alle $i < n$ gilt: $\mathcal{T}_i \models \mathcal{T}_{i+1}$
(d. h.: $\mathcal{T}_i \models C \sqsubseteq D$ für alle $C \sqsubseteq D \in \mathcal{T}_{i+1}$)

T 6.12

Daraus folgt direkt $\mathcal{T} \models \mathcal{T}^*$. □

Vollständigkeit

Konstruieren ein einziges Modell, das **alle nicht** aus $\mathcal{T}$ folgenden Subsumtionen zwischen Konzeptnamen **gleichzeitig** falsch macht.

Existenz solcher **kanonischer Modelle** ist **zentrale Eigenschaft** von $\mathcal{EL}$.

Definition 6.19

Die *kanonische Interpretation* $\mathcal{I}$ ist wie folgt definiert:

$$\Delta^{\mathcal{I}} = \{d_A \mid A \text{ Konzeptname in } \mathcal{T}^*\} \cup \{d_\top\}$$
$$A^{\mathcal{I}} = \{d_B \mid B \sqsubseteq A \in \mathcal{T}^*\}$$
$$r^{\mathcal{I}} = \{(d_A, d_B) \mid A \sqsubseteq A' \in \mathcal{T}^* \text{ und } A' \sqsubseteq \exists r.B \in \mathcal{T}^*, A' \text{ Konzeptname}\}$$

T 6.13

Lemma 6.20

Die kanonische Interpretation $\mathcal{I}$ ist ein Model von $\mathcal{T}^*$. T 6.14

Vollständigkeit

Jetzt zeigt man leicht:

Lemma 6.21

Für alle Konzeptnamen A, B in $\mathcal{T}$ gilt:

$$\mathcal{T} \models A \sqsubseteq B \quad \textbf{impliziert} \quad A \sqsubseteq B \in \mathcal{T}^*$$

Beweis. Angenommen $A \sqsubseteq B \notin \mathcal{T}^*$. $(*)$

Betrachte Element d_A der kanonischen Interpretation $\mathcal{I}$.

Wegen R1 ist $A \sqsubseteq A \in \mathcal{T}^*$, also $d_A \in A^{\mathcal{I}}$.

Def. von $\mathcal{I}$ und $(*)$ liefert $d_A \notin B^{\mathcal{I}}$.

Da $\mathcal{I}$ Modell von $\mathcal{T}^*$ (Lemma 6.20) und damit von $\mathcal{T}$ ist, folgt $\mathcal{T} \not\models A \sqsubseteq B$. □

Kapitel 6: Effiziente Beschreibungslogiken

„Gute“ und „schlechte“ Erweiterungen

Der Algorithmus kann angepasst werden an $\mathcal{EL}$ erweitert mit:

- $\bot$
- „Range Restrictions“ $\top \sqsubseteq \forall r.C$ und
  „Domain Restrictions“ $\top \sqsubseteq \forall r^{-}.C$

  z. B. $\top \sqsubseteq \forall\, \mathsf{hatKind}.\mathsf{Mensch}$
  $\top \sqsubseteq \forall\, \mathsf{hatKind}^{-}.\mathsf{Elternteil}$
- Rolleninklusionen mit Verkettung: $r_1 \circ \cdots \circ r_n \sqsubseteq r$

  z. B. $\mathsf{hatGeschwister} \circ \mathsf{hatTochter} \sqsubseteq \mathsf{hatNichte}$
- $\cdots$

Dies (und mehr) ist im EL-Profil von OWL 2 realisiert.

„Gute“ und „schlechte“ Erweiterungen

- **Viele andere Erweiterungen sind jedoch ExpTime-vollständig (wie $\mathcal{ALC}$).**

Wir betrachten exemplarisch

- $\mathcal{ELU}$, die Erweiterung von $\mathcal{EL}$ mit $\sqcup$
- $\mathcal{EL}_\forall$, die Erweiterung von $\mathcal{EL}$ mit $\forall r.C$
- $\mathcal{EL}^{\geqslant 2}$, die Erweiterung von $\mathcal{EL}$ mit $\geqslant 2\, r.\top$

$\mathcal{EL}$ mit Disjunktion und $\bot$

Theorem 6.22

Erfüllbarkeit in $\mathcal{ELU}_\bot$ bzgl. TBoxen ist ExpTime-vollständig.

Beweis.

Reduktion von **Erfüllbarkeit** von Konzeptnamen A bzgl. $\mathcal{ALC}$-TBox $\mathcal{T}$

Schritt 1. Ersetze in $\mathcal{T}$ alle Werte- durch Existenzrestriktionen:

$$\forall r.C \qquad \text{wird} \qquad \neg\exists r.\neg C$$

Schritt 2. Modifiziere $\mathcal{T}$ so, dass $\neg$ nur vor Konzeptnamen auftritt:

z. B. $A \sqsubseteq \exists s.(B' \sqcup \neg\exists r.B)$ wird $A \sqsubseteq \exists s.(B' \sqcup \neg X)$, $X \equiv \exists r.B$ (X neuer Konzeptname)

$\mathcal{EL}$ mit Disjunktion und $\bot$

Schritt 3. Entferne Negation vollständig aus $\mathcal{T}$:

- Ersetze jedes $\neg X$ durch neuen Konzeptnamen $\overline{X}$
- Erzwinge korrektes Verhalten der $\overline{X}$:

$$\top \sqsubseteq X \sqcup \overline{X}$$
$$X \sqcap \overline{X} \sqsubseteq \bot$$

Die resultierende $\mathcal{ELU}_\bot$-TBox sei $\mathcal{T}'$.

Lemma 6.23

Für alle Konzeptnamen A gilt:

A erfüllbar bzgl. $\mathcal{T}$ **gdw.** A erfüllbar bzgl. $\mathcal{T}'$

T 6.15

$\mathcal{EL}$ mit Disjunktion

Theorem 6.24

Subsumtion in $\mathcal{ELU}$ bzgl. TBoxen ist ExpTime-vollständig.

Beweis.

Reduktion von **Erfüllbarkeit** von Konzeptnamen A bzgl. $\mathcal{ELU}_\bot$-TBox $\mathcal{T}$

Konstruiere $\mathcal{ELU}$-TBox $\mathcal{T}'$:

- nimm o. B. d. A. an, dass $\bot$ nur in der Form $C \sqsubseteq \bot$ vorkommt (jedes $\mathcal{ELU}_\bot$-Konzept ist äquivalent zu $\mathcal{ELU}$-Konzept oder $\bot$):
  - Wende erschöpfend folgende offensichtliche Äquivalenzen an:

    $$C \sqcap \bot \equiv \bot \qquad C \sqcup \bot \equiv C \qquad \exists r.\bot \equiv \bot$$

  - Dann sind alle Axiome in der Form

    $$C \sqsubseteq D \qquad \bot \sqsubseteq D \qquad C \sqsubseteq \bot \qquad \bot \sqsubseteq \bot$$

    mit C, D $\mathcal{ELU}$-Konzepte.

    Fälle $\bot \sqsubseteq D$ und $\bot \sqsubseteq \bot$ sind Tautologien $\rightsquigarrow$ löschen.

    Nun tritt $\bot$ nur noch in Form $C \sqsubseteq \bot$ auf.

$\mathcal{EL}$ mit Disjunktion

Theorem 6.24

Subsumtion in $\mathcal{ELU}$ bzgl. TBoxen ist ExpTime-vollständig.

Beweis.

Reduktion von **Erfüllbarkeit** von Konzeptnamen A bzgl. $\mathcal{ELU}_\bot$-TBox $\mathcal{T}$

Konstruiere $\mathcal{ELU}$-TBox $\mathcal{T}'$:

- nimm o. B. d. A. an, dass $\bot$ nur in der Form $C \sqsubseteq \bot$ vorkommt (jedes $\mathcal{ELU}_\bot$-Konzept ist äquivalent zu $\mathcal{ELU}$-Konzept oder $\bot$)
- ersetze $\bot$ durch neuen Konzeptnamen L
- füge hinzu:

$$\exists r.L \sqsubseteq L \quad \text{für alle Rollennamen } r \text{ in } \mathcal{T}$$

Lemma 6.25

A **un**erfüllbar bzgl. $\mathcal{T}$ **gdw.** $\mathcal{T}' \models A \sqsubseteq L$. **T 6.16**

$\mathcal{EL}$ mit Werterestriktionen

$\mathcal{EL}^\forall$ ist $\mathcal{EL}$ erweitert um $\forall r.C$.

Theorem 6.26

In $\mathcal{EL}^\forall$ ist Subsumtion bzgl. TBoxen ExpTime-vollständig.

Beweis:

Reduktion von **Subsumtion** zwischen Konzeptnamen bzgl. $\mathcal{ELU}$-TBox $\mathcal{T}$

Können annehmen, dass $\sqcup$ nur in den folgenden Formen vorkommt:

- $A_1 \sqcup A_2 \sqsubseteq A \quad \leadsto$ Ersetze durch $A_1 \sqsubseteq A,\ A_2 \sqsubseteq A$
- $A \sqsubseteq B_1 \sqcup B_2 \quad \leadsto$ Ersetze durch $\begin{array}{l} A \sqcap \exists r.\top \sqsubseteq B_1 \\ A \sqcap \forall r.X \sqsubseteq B_2 \end{array}$ r, X neu

Die resultierende $\mathcal{EL}^\forall$-TBox sei $\mathcal{T}'$.

Lemma 6.27

$\mathcal{T} \models A \sqsubseteq B$ **gdw.** $\mathcal{T}' \models A \sqsubseteq B$. **T 6.17**

$\mathcal{EL}$ mit Zahlenrestriktionen

$\mathcal{EL}^{\geqslant 2}$ ist $\mathcal{EL}$ erweitert um $\geqslant 2\, r.\top$.

Theorem 6.28

In $\mathcal{EL}^{\geqslant 2}$ ist Subsumtion bzgl. TBoxen ExpTime-vollständig.

Beweis: Wieder Reduktion von $\mathcal{ELU}$-Subsumtion

Können annehmen, dass $\sqcup$ nur in den folgenden Formen vorkommt:

- $A_1 \sqcup A_2 \sqsubseteq A \quad \rightsquigarrow$ Ersetze durch $A_1 \sqsubseteq A,\ A_2 \sqsubseteq A$
- $A \sqsubseteq B_1 \sqcup B_2 \quad \rightsquigarrow$ Ersetze durch $A \sqsubseteq \exists r.X \sqcap \exists r.Y$, $A \sqcap \exists r.(X \sqcap Y) \sqsubseteq B_1$, $(r, X, Y \text{ neu})$ $A \sqcap \geqslant 2\, r.\top \sqsubseteq B_2$

Die resultierende $\mathcal{EL}^{\forall}$-TBox sei $\mathcal{T}'$.

Lemma 6.29

$\mathcal{T} \models A \sqsubseteq B$ **gdw.** $\mathcal{T}' \models A \sqsubseteq B$. **(Übung)**

Konvexität

Eine Erweiterung von $\mathcal{EL}$ ist **konvex**,
wenn für alle TBoxen $\mathcal{T}$ und Konzepte C, D_1, D_2 gilt:

$\mathcal{T} \models C \sqsubseteq D_1 \sqcup D_2$ impliziert $\mathcal{T} \models C \sqsubseteq D_1$ oder $\mathcal{T} \models C \sqsubseteq D_2$

$\mathcal{EL}^\forall$ ist nicht konvex:

$\mathcal{T} \models \top \sqsubseteq \exists r.\top \sqcup \forall r.X$ aber $\mathcal{T} \not\models \top \sqsubseteq \exists r.\top$
und $\mathcal{T} \not\models \top \sqsubseteq \forall r.X$

Unsere Beweise zeigen im Prinzip:
jede **nicht-konvexe** Erweiterung von $\mathcal{EL}$ ist **ExpTime-hart.**

Aber auch konvexe Erweiterungen sind nicht zwangsläufig in P:
Z. B. ist $\mathcal{ELI}$ ($\mathcal{EL}$ „plus“ $\exists r^-.C$) konvex, aber ExpTime-vollst.

Für konvexe Erweiterungen gibt es oft effiziente konsequenzbasierte Algorithmen.

Zusammenfassung für $\mathcal{EL}$

Die $\mathcal{EL}$-Familie von BLen:

- Erlaubt Schlussfolgern in polynomieller Zeit
- Es gibt viele Reasoner wie ELK, CEL, SNOROCKET
- Skaliert auch auf große Terminologien wie SNOMED CT
  ($>$ 400.000 Konzepte, wird in wenigen Sekunden klassifiziert)
- Stellt viele Operatoren zur Verfügung,
  hat aber eingeschränktes Ausdrucksvermögen
  (z. B. kann keine Disjunktion ausgedrückt werden – Konvexität!)

Literatur für dieses Kapitel (Basis)

Franz Baader, Ian Horrocks, Carsten Lutz, Uli Sattler.

An Introduction to Description Logic.

Cambridge University Press, 2017.

Kapitel 6: Reasoning in the $\mathcal{EL}$ Family of DLs

In SUUB verfügbar: `https://tinyurl.com/suub-intro-dl-ebook`
`https://tinyurl.com/suub-intro-dl`

Literatur für dieses Kapitel (weiterführend 1)

Franz Baader, Sebastian Brandt, Carsten Lutz.

Pushing the $\mathcal{EL}$ Envelope.

IJCAI 2005: 364–369.

`http://ijcai.org/Proceedings/05/Papers/0372.pdf`

Untersucht systematisch Erweiterungen von $\mathcal{EL}$, die in Polyzeit bleiben oder höhere Komplexität verursachen.

Technischer Report mit Beweisdetails: `http://lat.inf.tu-dresden.de/research/reports/2005/BaaderBrandtLutz-LTCS-05-01.ps.gz`

Franz Baader, Sebastian Brandt, Carsten Lutz.

Pushing the $\mathcal{EL}$ Envelope Further.

OWLED (Spring) 2008.

`http://ceur-ws.org/Vol-496/owled2008dc_paper_3.pdf`

Führt die vorangegangene Arbeit fort.

Literatur für dieses Kapitel (weiterführend 2)

Boontawee Suntisrivaraporn.

Optimization and Implementation of Subsumption Algorithms for the Description Logic $\mathcal{EL}$ with Cyclic TBoxes and General Concept Inclusion Axioms.

Masterarbeit, TU Dresden, 2005.

`https://lat.inf.tu-dresden.de/research/mas/Sun-Mas-05.pdf`

Erklärt anschaulich Normalisierung und Klassifikation für $\mathcal{EL}$ (ohne Erweiterungen). Allerdings weichen die Regeln etwas von denen aus der Vorlesung ab; „unsere“ sind besser optimiert.

Links für dieses Kapitel

University of Manchester

OWL API

`http://owlcs.github.io/owlapi/`
`https://github.com/owlcs/owlapi/wiki` (Wiki + Doku.)

Die API der Wahl, um mit Ontologien zu arbeiten.

University of Manchester

List of Reasoners

`http://owl.cs.manchester.ac.uk/tools/list-of-reasoners/`

Eine aktuelle Übersicht von DL-Reasonern
(die speziell für $\mathcal{EL}$ entwickelten haben meist ein „EL“ im Namen)