

- Kapitel 1: Einleitung
- Kapitel 2: Grundlagen
- Kapitel 3: Ausdrucksstärke und Modellkonstruktionen
- Kapitel 4: Tableau-Algorithmen
- Kapitel 5: Komplexität
- ➔ Kapitel 6: Effiziente Beschreibungslogiken
- Kapitel 7: ABoxen und Anfragebeantwortung

1

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 \sqcap und $\exists r.C$
- Erfüllbarkeit und Subsumption in Polyzeit entscheidbar

3

Effiziente Beschreibungslogiken

2

EL

Definition 6.1

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

Beliebt für biomedizinische Ontologien (groß, hoher Abstraktionsgrad):

Perikardium \sqsubseteq Gewebe \sqcap \exists teilVon.Herz
Perikarditis \equiv Entzündung \sqcap \exists ort.Perikardium
Entzündung \sqsubseteq Krankheit \sqcap \exists wirktAuf.Gewebe

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

\mathcal{EL} ist Grundlage des EL-Profiles von OWL 2.

4

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

1. $d_1 \rho d_2$ und $d_1 \in A^{\mathcal{I}_1}$ impliziert $d_2 \in A^{\mathcal{I}_2}$, für alle Konzeptnamen A .
2. $d_1 \rho d_2$ und $(d_1, d'_1) \in r^{\mathcal{I}_1}$ impliziert die Existenz eines $d'_2 \in \Delta^{\mathcal{I}_2}$ mit $d'_1 \rho d'_2$ und $(d_2, d'_2) \in r^{\mathcal{I}_2}$, für alle Rollennamen r .

T6.1

Beachte: im Gegensatz zu Bisimulationen sind Simulationen gerichtet!

5

Simulation

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

$(\mathcal{I}_1, d_1) \lesssim (\mathcal{I}_2, d_2)$: es gibt Simulation ρ von \mathcal{I}_1 nach \mathcal{I}_2 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 $(\mathcal{I}_1, d_1) \lesssim (\mathcal{I}_2, d_2)$, dann gilt für alle \mathcal{EL} -Konzepte C :

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

T6.2

6

Simulation

Intuitiv: Wenn $(\mathcal{I}_1, d_1) \lesssim (\mathcal{I}_2, d_2)$ und $(\mathcal{I}_2, d_2) \lesssim (\mathcal{I}_1, d_1)$, 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

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

T6.3

Man kann nun wieder Nicht-Ausdrückbarkeitsresultate zeigen:

Lemma 6.5.

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

T6.3 cont

7

EL

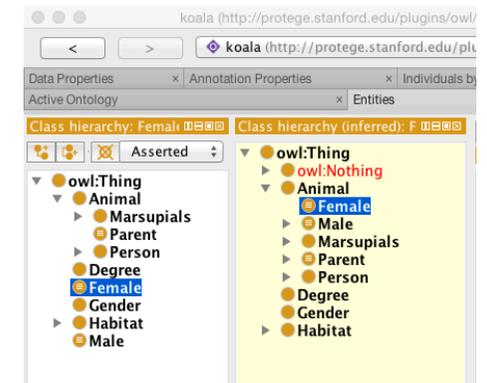
In \mathcal{EL} ist Erfüllbarkeit **kein** interessantes Schlussfolgerungsproblem:

Lemma 6.6

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

T6.4

Darum konzentrieren wir uns auf Subsumtion.



8

Subsumtion ohne TBox

9

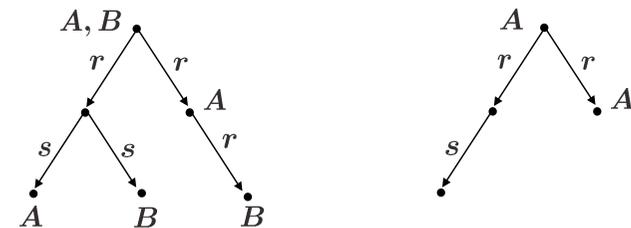
Subsumtion ohne TBox

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

$$\begin{array}{ll} \text{Z.B.: } C = A \sqcap B & D = A \\ \sqcap \exists r. (\exists s. A \sqcap \exists s. B) & \sqcap \exists r. \exists s. \top \\ \sqcap \exists r. (A \sqcap \exists r. B) & \sqcap \exists r. A \end{array}$$

Konzepte dargestellt als Bäume:

„Wiederfinden“ entspricht Simulation von D -Baum in C -Baum (Richtung!)



10

Subsumtion ohne TBox

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 : • A
- $C = D \sqcap E$ Modell \mathcal{I}_C :
- $C = \exists r. D$ Modell \mathcal{I}_C :

11

Subsumtion ohne TBox

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

Man sieht leicht, dass kanonische Modelle wirklich *Modelle* sind:

Lemma 6.8.

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

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

T6.5

Die zentrale Eigenschaft kanonischer Modelle:

Lemma 6.9.

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

$$e \in C^{\mathcal{I}} \text{ gdw. } (\mathcal{I}_C, d_W) \preceq (\mathcal{I}, e)$$

T6.6

12

Subsumtion ohne TBox

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

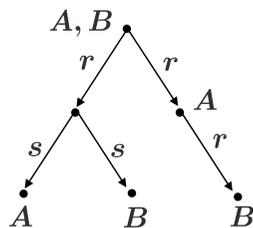
Lemma 6.10.

Für alle \mathcal{EL} -Konzepte C, D gilt: $C \sqsubseteq D$ gdw. $(\mathcal{I}_D, d_W) \lesssim (\mathcal{I}_C, d_W)$ **T6.7**

$$C = A \sqcap B$$

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

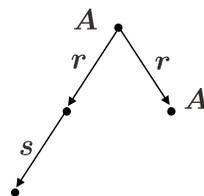
$$\sqcap \exists r. (A \sqcap \exists r. B)$$



$$D = A$$

$$\sqcap \exists r. \exists s. \top$$

$$\sqcap \exists r. A$$



13

Subsumtion ohne TBox

Folgendes Theorem formuliert den (recht einfachen) Algorithmus:

Theorem 6.11.

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

- Konstruiere \mathcal{I}_C und \mathcal{I}_D in polynomieller Zeit.
- Überprüfe in polynomieller Zeit, ob $(\mathcal{I}_D, d_W) \lesssim (\mathcal{I}_C, d_W)$.

T6.8

14

Kapitel 6

Subsumtion mit TBox

Subsumtion mit TBox

Wir verwenden ein sogenanntes 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.

15

16

Subsumtion mit TBox

Wir konzentrieren uns auf Subsumtion von Konzeptnamen bzgl. TBoxen.

Lemma 6.12.

Seien C, D zwei beliebige \mathcal{EL} -Konzepte und \mathcal{T} eine \mathcal{EL} -TBox.

Sei weiterhin $\mathcal{T}' = \mathcal{T} \cup \{A_C \sqsubseteq C, D \sqsubseteq A_D\}$, mit Konzeptnamen A_C, A_D , die nicht in C, D, \mathcal{T} vorkommen.

Dann gilt:

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

Dies liefert Polyzeitreduktion von Subsumtion zwischen beliebigen Konzepten auf Subsumtion zwischen Konzeptnamen (beides bzgl. TBoxen).

17

Normalform

Wir nehmen weiterhin an, dass TBoxen nur Inklusionen folgender Form enthalten:

$$A_1 \sqcap \dots \sqcap A_n \sqsubseteq A \quad A \sqsubseteq \exists r.A_1 \quad \exists r.A \sqsubseteq A_1$$

wobei A, A_1, \dots, A_n Konzeptnamen oder \top

Eine derartige TBox ist *in Normalform*.

Lemma 6.13

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

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

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

Um die Normalform herzustellen, wenden wir Normalisierungsregeln an. **T6.9**

18

Normalform

$$\text{NF1: } C_1 \sqcap \dots \sqcap C_n \sqsubseteq E \quad \rightsquigarrow \quad C_i \sqsubseteq A_{C_i}, \quad C_1 \sqcap \dots \sqcap C_{i-1} \sqcap A_{C_i} \sqcap C_{i+1} \sqcap \dots \sqcap C_n \sqsubseteq E$$

wenn C_i Existenzrestriktion

$$\left. \begin{array}{l} \text{NF2: } \exists r.C \sqsubseteq E \quad \rightsquigarrow \quad C \sqsubseteq A_C, \quad \exists r.A_C \sqsubseteq E \\ \text{NF3: } C \sqsubseteq \exists r.D \quad \rightsquigarrow \quad C \sqsubseteq A_C, \quad A_C \sqsubseteq \exists r.D \\ \text{NF4: } A \sqsubseteq \exists r.C \quad \rightsquigarrow \quad A \sqsubseteq \exists r.A_C, \quad A_C \sqsubseteq C \\ \text{NF5: } A \sqsubseteq C_1 \sqcap C_2 \quad \rightsquigarrow \quad A \sqsubseteq C_1, \quad A \sqsubseteq C_2 \end{array} \right\} \begin{array}{l} \text{wenn } C \\ \text{weder Konzeptname} \\ \text{noch } \top \end{array}$$

Lemma 6.14.

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

T6.10

19

Subsumtion mit TBox

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

$$\text{R1: } \frac{}{A \sqsubseteq A} \quad (\text{wenn } A \text{ in } \mathcal{T} \text{ vorkommt}) \quad \text{R2: } \frac{}{A \sqsubseteq \top} \quad (\text{wenn } A \text{ in } \mathcal{T} \text{ vorkommt})$$

$$\text{R3: } \frac{A \sqsubseteq A_1 \quad \dots \quad A \sqsubseteq A_n \quad A_1 \sqcap \dots \sqcap A_n \sqsubseteq B}{A \sqsubseteq B}$$

$$\text{R4: } \frac{A \sqsubseteq \exists r.A_1 \quad A_1 \sqsubseteq B_1 \quad \exists r.B_1 \sqsubseteq B}{A \sqsubseteq B}$$

T6.11

20

Subsumtion mit TBox

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 6.15.

Für alle Konzeptnamen A, B in \mathcal{T} gilt: $\mathcal{T} \models A \sqsubseteq B$ gdw. $A \sqsubseteq B \in \mathcal{T}^*$

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

Theorem 6.16.

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

T6.12

21

Subsumtion mit TBox

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

$$\Delta^{\mathcal{I}} = \{d_A \mid A \text{ Konzeptname in } \mathcal{T}^*\} \cup \{d_{\top}\}$$

$$A^{\mathcal{I}} = \{d_B \mid B \sqsubseteq A \in \mathcal{T}^*\}$$

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

T6.14

Lemma 6.18.

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

T6.15

Lemma 6.19. [Vollständigkeit]

Für alle Konzeptnamen A, B gilt: $\mathcal{T} \models A \sqsubseteq B$ impliziert $A \sqsubseteq B \in \mathcal{T}^*$

T6.16

23

Subsumtion mit TBox

Wir beweisen nun Theorem 6.15.

Lemma 6.17. [Korrektheit]

Für alle Konzeptnamen A, B gilt: $A \sqsubseteq B \in \mathcal{T}^*$ impliziert $\mathcal{T} \models A \sqsubseteq B$

T6.13

Interessanter ist der Beweis der Vollständigkeit.

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

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

22

Kapitel 6

Erweiterungen von EL

24

Erweiterungen von EL

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

- \perp
- „Range Restrictions“ $\top \sqsubseteq \forall r.C$ und „Domain Restrictions“ $\top \sqsubseteq \forall r.\neg.C$
- Rolleninklusionen mit Verkettung: $r_1 \circ \dots \circ r_n \sqsubseteq r$
- ...

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

Viele andere Erweiterungen lassen die Komplexität zurück auf EXP TIME springen:

Wir betrachten exemplarisch

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

25

EL mit Disjunktion

Theorem 6.20

Erfüllbarkeit in \mathcal{ELU}_{\perp} bzgl. TBoxen ist EXP TIME-vollständig.

Beweis: Reduktion von Erfüllbarkeit von Konzeptname A bzgl. \mathcal{ALC} -TBox \mathcal{T}

Schritt 1: Ersetze Werterestrictionen in \mathcal{T} durch Existenzrestrictionen:

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

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

$$\begin{aligned} \text{z. B. } A \sqsubseteq \exists s.(B' \sqcup \neg \exists r.B) \quad \text{wird} \quad A \sqsubseteq \exists s.(B' \sqcup \neg X) \\ X \equiv \exists r.B \\ (\text{X neuer Konzeptname}) \end{aligned}$$

26

EL mit Disjunktion

Theorem 6.20

Erfüllbarkeit in \mathcal{ELU}_{\perp} bzgl. TBoxen ist EXP TIME-vollständig.

Beweis: Reduktion von Erfüllbarkeit von Konzeptname A bzgl. \mathcal{ALC} -TBox \mathcal{T}

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

- Ersetze jedes $\neg X$ durch \bar{X} , \bar{X} neuer Konzeptname
- Erzwingte korrektes Verhalten von \bar{X} :

$$\begin{aligned} \top \sqsubseteq X \sqcup \bar{X} \\ X \cap \bar{X} \sqsubseteq \perp \end{aligned}$$

\mathcal{T}' sei die resultierende \mathcal{ELU}_{\perp} -TBox.

Lemma 6.21

T6.17

Für alle Konzeptnamen A gilt: A erfüllbar bzgl. \mathcal{T} gdw. A erfüllbar bzgl. \mathcal{T}'

27

EL mit Disjunktion

Theorem 6.22

Subsumtion in \mathcal{ELU} bzgl. TBoxen ist EXP TIME-vollständig.

Beweis: Reduktion von Erfüllbarkeit von Konzeptname A bzgl. \mathcal{ELU}_{\perp} -TBox \mathcal{T}

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

- nimm o. B. d. A. an, dass \perp nur in der Form $C \sqsubseteq \perp$ vorkommt (jedes \mathcal{ELU}_{\perp} -Konzept ist äquivalent zu \mathcal{ELU} -Konzept oder \perp) **T6.18**
- ersetze \perp durch neuen Konzeptnamen L
- füge hinzu:

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

Lemma 6.23

A unerfüllbar bzgl. \mathcal{T} gdw. $\mathcal{T}' \models A \sqsubseteq L$.

T6.19

28

EL mit Wertrestriktionen

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

Theorem 6.24 In \mathcal{EL}^\forall ist Subsumtion bzgl. TBoxen EXPTIME-vollständig.

Beweis: Reduktion von Subsumtion zwischen Konzeptnamen bzgl. \mathcal{ELU} -TBox \mathcal{T}

Wir können annehmen, dass Disjunktion nur in den folgenden Formen vorkommt:

$$\underbrace{A_1 \sqcup A_2 \sqsubseteq A}_{= A_1 \sqsubseteq A, A_2 \sqsubseteq A} \quad \text{und} \quad A \sqsubseteq B_1 \sqcup B_2$$

ersetze in \mathcal{T}' durch

$$A \sqcap \exists r.T \sqsubseteq B_1 \quad r, X \text{ neu}$$

$$A \sqcap \forall r.X \sqsubseteq B_2$$

Lemma 6.25

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

T6.20

29

Konvexität

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

$$\mathcal{T} \models C \sqsubseteq D_1 \sqcup D_2 \quad \text{impliziert} \quad \mathcal{T} \models C \sqsubseteq D_i \quad \text{für ein } i \in \{1, 2\}$$

$\mathcal{EL} + \forall r.C$ ist *nicht konvex*:

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

Unsere Beweise zeigen im Prinzip:

jede nicht-konvexe Erweiterung von \mathcal{EL} ist EXPTIME-hart.

Aber auch konvexe Erweiterungen sind leider nicht zwangsläufig in PTIME:

Zum Beispiel ist \mathcal{ELI} (\mathcal{EL} erweitert mit $\exists r^-.C$) konvex,
aber EXPTIME-vollständig.

Für konvexe Erweiterungen gibt es oft effiziente konsequenzbasierte Algorithmen.

31

EL mit Zahlenrestriktionen

$\mathcal{EL}^{\geq 2}$ ist \mathcal{EL} erweitert um $(\geq 2 r T)$

Theorem 6.26 In $\mathcal{EL}^{\geq 2}$ ist Subsumtion bzgl. TBoxen EXPTIME-vollständig.

Beweis: Reduktion von Subsumtion zwischen Konzeptnamen bzgl. \mathcal{ELU} -TBox \mathcal{T}

Wir können annehmen, dass Disjunktion nur in den folgenden Formen vorkommt:

$$\underbrace{A_1 \sqcup A_2 \sqsubseteq A}_{= A_1 \sqsubseteq A, A_2 \sqsubseteq A} \quad \text{und} \quad A \sqsubseteq B_1 \sqcup B_2$$

ersetze in \mathcal{T}' durch

$$A \sqsubseteq \exists r.X \sqcap \exists r.Y \quad r, X, Y \text{ neu}$$

$$A \sqcap \exists r.(X \sqcap Y) \sqsubseteq B_1$$

$$A \sqcap (\geq 2 r) \sqsubseteq B_2$$

Lemma 6.27

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

30

Zusammenfassung für 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!)

32