Struktur Vorlesung

- 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

Kapitel 7

Effiziente Beschreibungslogiken

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

EL

Definition 7.1

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):

$$\text{Perikardium} \sqsubseteq \text{Gewebe} \sqcap \exists \text{teilVon.Herz}$$

$$\text{Perikarditis} \equiv \text{Entzündung} \sqcap \exists \text{ort.Perikardium}$$

$$\text{Entzündung} \sqsubseteq \text{Krankheit} \sqcap \exists \text{wirktAuf.Gewebe}$$

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

$\mathcal{EL}$ ist Grundlage des OWL EL Profiles von OWL2

Simulation

Intuitiv: $\mathcal{EL}$ ist die "Hälfte von $\mathcal{ALC}$", entspricht der "Hälfte von Bisimulation"

Definition 7.2 (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 $A \in \mathsf{N_C}$

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 T7.1

Beachte: im Ggs. zu Bisimulationen sind Simulationen gerichtet!

Simulation

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

$(\mathcal{I}_1, d_1) \precsim (\mathcal{I}_2, d_2)$: 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 7.3.
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:
$d_1 \in C^{\mathcal{I}_1}$ impliziert $d_2 \in C^{\mathcal{I}_2}$. T7.2

Simulation

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 7.4.

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

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

T7.3

Man kann nun wieder Nicht-Ausdrückbarkeitsresultate zeigen:

Lemma 7.5.

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

T7.3 cont

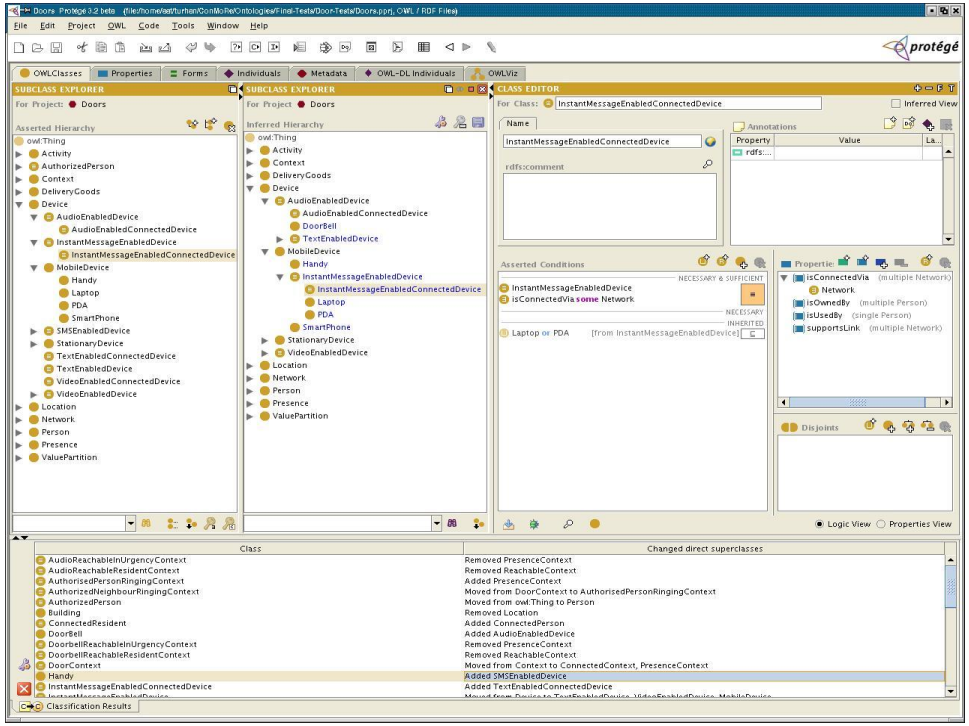
EL

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

Lemma 7.6

Jedes $\mathcal{EL}$-Konzept ist erfüllbar bzgl. jeder TBox. T7.4

Darum konzentrieren wir uns auf Subsumtion



Subsumtion ohne TBox

Subsumtion ohne TBox

Eine Subsumtion $C \sqsubseteq D$ gilt in $\mathcal{EL}$ im Prinzip genau dann,

wenn man D syntaktisch "in C wiederfindet"

Z.B.: $C = A \sqcap B$

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

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

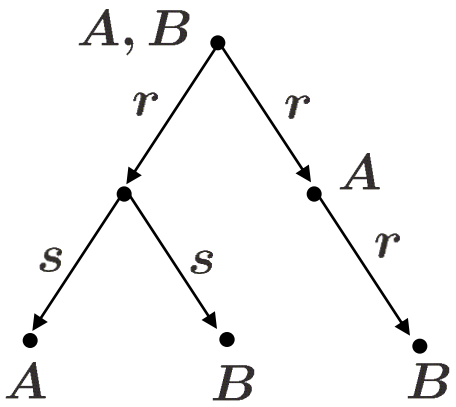
$D = A$

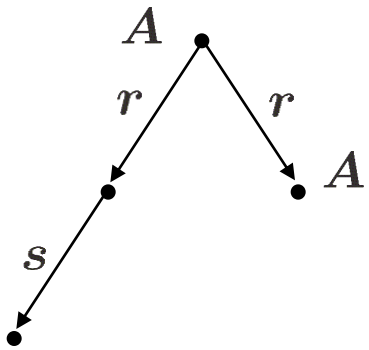
$\sqcap \exists r.\exists s.\top$

$\sqcap \exists r.A$

Konzepte dargestellt als Bäume:

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





Subsumtion ohne TBox

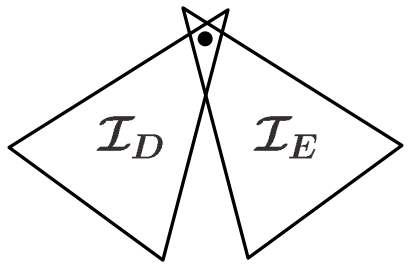
Definition 7.7.

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

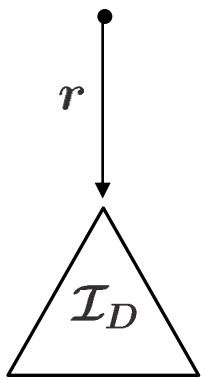
- $C = \top$ Modell $\mathcal{I}_C$:

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

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



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



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 7.8.
Für alle $\mathcal{EL}$-Konzepte C gilt:
Die Interpretation $\mathcal{I}_C$ ist Modell von C mit $d_W \in C^{\mathcal{I}_C}$. T7.5

Die zentrale Eigenschaft kanonischer Modelle:

Lemma 7.9.
Für alle $\mathcal{EL}$-Konzepte C, Interpretation $\mathcal{I}$ und $e \in \Delta^{\mathcal{I}}$ gilt:
$e \in C^{\mathcal{I}}$ gdw. $(\mathcal{I}_C, d_W) \precsim (\mathcal{I}, e)$. T7.6

Subsumtion ohne TBox

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

Lemma 7.10.
Für alle $\mathcal{EL}$-Konzepte C, D gilt: $C \sqsubseteq D$ gdw. $(\mathcal{I}_D, d_W) \precsim (\mathcal{I}_C, d_W)$.

T7.7

Folgendes Theorem formuliert den (recht einfachen) Algorithmus:

Theorem 7.11.
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 $(\mathcal{I}_D, d_W) \precsim (\mathcal{I}_C, d_W)$

T7.8

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 Subsumption 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.

Subsumtion mit TBox

Zwei vereinfachende Annahmen:

- Algorithmus entscheidet Subsumtion zwischen Konzept*namen aus* $\mathcal{T}$:

  O.B.d.A. denn $\mathcal{T} \models C \sqsubseteq D$ gdw. $\mathcal{T}' \models A_C \sqsubseteq A_D$

  mit $\mathcal{T}' = \mathcal{T} \cup \{A_C \sqsubseteq C, D \sqsubseteq A_D\}$

- $\mathcal{T}$ enthält nur Inklusionen der Form

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

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

T7.9

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

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

T7.10

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 Subsumptionen zwischen Konzeptnamen explizit:

Theorem 7.12.
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 Subsumption.

Theorem 7.13.
Die Konstruktion von $\mathcal{T}^*$ terminiert nach $|\mathcal{T}|^2$ vielen Regelanwendungen.

T7.11

Subsumtion mit TBox

Wir beweisen nun Theorem 7.12.

Lemma 7.14. [Korrektheit]
Für alle Konzeptnamen $\boldsymbol{A}, \boldsymbol{B}$ gilt: $\boldsymbol{A} \sqsubseteq \boldsymbol{B} \in \boldsymbol{\mathcal{T}}^*$ impliziert $\boldsymbol{\mathcal{T}} \models \boldsymbol{A} \sqsubseteq \boldsymbol{B}$.

T7.12

Interessanter ist der Beweis der Vollständigkeit.

Hier konstruieren wir ein einziges Modell, dass *alle* nicht aus $\boldsymbol{\mathcal{T}}$ folgenden Subsumptionen zwischen Konzeptnamen gleichzeitig falsch macht.

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

Subsumtion mit TBox

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

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

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

$$r^{\mathcal{I}} = \{(A, B) \mid B \sqsubseteq \exists r.A \in \mathcal{T}^*\} \qquad \text{T7.13}$$

Lemma 7.15.
Die kanonische Interpretation $\mathcal{I}$ ist ein Model von $\mathcal{T}^*$. T7.14

Lemma 7.16. [Vollständigkeit]
Für alle Konzeptnamen A, B gilt:$\mathcal{T} \models A \sqsubseteq B$ impliziert $A \sqsubseteq B \in \mathcal{T}^*$.
T7.15

Erweiterungen von EL

Erweiterungen von EL

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

- $\bot$
- Range Restrictions $\top \sqsubseteq \forall r.C$ und Domain Restrictions $\top \sqsubseteq \forall r^-.C$
- allgemeine Rolleninklusionen $r_1 \circ \cdots \circ r_n \sqsubseteq r$
- $\cdots$

Dies (und mehr) ist im OWL EL Profil von OWL2 realisiert.

Viele andere Erweiterungen lassen die Komplexität zurück auf ExpTime 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 $(\geq 2\, r\, \top)$

Erweiterungen von EL

Theorem 7.17.

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

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

Schritt 1: Ersetze Werterestriktionen in $\mathcal{T}$ durch Existenzrestriktionen:

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

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

$$A \sqsubseteq \exists s.(B' \sqcup \neg\exists r.B) \qquad \text{wird} \qquad A \sqsubseteq \exists s.(B' \sqcup \neg X)$$

$$X \equiv \exists r.B$$

(X neuer Konzeptname)

Erweiterungen von EL

Theorem 7.17.

Erfüllbarkeit in $\mathcal{ELU}_\bot$ bzgl. TBoxen ist ExpTime-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 $\overline{X}$, $\overline{X}$ neuer Konzeptname
- Erzwinge korrektes Verhalten von $\overline{X}$:

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

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

T7.16

Lemma 7.18

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

Erweiterungen von EL

Theorem 7.19.

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

Beweis: Reduktion von Erfüllbarkeit von Konzeptname 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:

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

Lemma 7.20

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

T7.17

Erweiterungen von EL

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

Theorem 7.21. 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, das Disjunktion nur in den folgenden Formen vorkommt:

$A_1 \sqcup A_2 \sqsubseteq A$ und $A \sqsubseteq B_1 \sqcup B_2$

$= A_1 \sqsubseteq A,\ A_2 \sqsubseteq A$

ersetze in $\mathcal{T}'$ durch

$A \sqcap \exists r.\top \sqsubseteq B_1$

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

r, X neu

Lemma 7.22

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

T7.18

Erweiterungen von EL

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

Theorem 7.23. In $\mathcal{EL}^{\geq 2}$ ist Subsumtion bzgl. TBoxen ExpTime-vollständig.

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

Wir können annehmen, das 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

$$A \sqsubseteq \exists r.X \sqcap \exists r.Y$$
$$A \sqcap \exists r.(X \sqcap Y) \sqsubseteq B_1 \qquad r, X, Y \text{ neu}$$
$$A \sqcap (\geq 2\, r) \sqsubseteq B_2$$

Lemma 7.24

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

Erweiterungen von EL

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

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

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

$$\top \models \exists r.\top \sqcup \forall r.X, \text{ aber } \top \not\models \exists r.\top \text{ und } \top \not\models \forall r.X$$

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.

Diskussion

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
  ($> \mathbf{400.000}$ Konzepte, wird in wenigen Sekunden klassifiziert)
- Die Beantwortung konjunktiver Anfragen ist NP-vollständig,...
- ...kann aber skalierbar mit normalen relationalen Datenbanksystemen implementiert werden