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 2

Grundlagen

Die Beschreibungslogik $\mathcal{ALC}$

ALC

Es gibt viele verschiedene Beschreibungslogiken:

- viele mögliche Kompromisse bzgl. Ausdrucksstärke vs. Komplexität des Schlussfolgerns
- verschiedene Anwendungen haben unterschiedliche Anforderungen

Hier zunächst $\mathcal{ALC}$:

- die einfachste BL mit allen Booleschen Konstruktoren
- eingeführt 1991 von Schmidt-Schauß und Smolka
- steht für *Attributive (Concept) Language with Complement*

ALC

Wir fixieren von nun an

- eine abzählbar unendliche Menge von Konzeptnamen
  Diese bezeichnen Klassen / unäre Prädikate
  z.B. Person, Kurs, Universität, Tafel, Student, etc.
- eine abzählbar unendliche Menge von Rollennamen
  Diese bezeichnen Relationen / binäre Prädikate
  z.B. hört, lehrt, istTeilVon, etc.

Wir nehmen die beiden Mengen als disjunkt an und unterscheiden Konzept- und Rollennamen über Groß- / Kleinschreibung.

ALC

Konzepte dienen der Beschreibung einer Klasse von Objekten, sind zusammengesetzt aus

- Konzeptnamen
- Rollennamen
- Konzeptkonstruktoren
- (manchmal auch Rollenkonstruktoren)

Es gibt viele verschiedene Konstruktoren

Verschiedene Konstruktormengen ergeben verschiedene Beschreibungslogiken

ALC: Syntax

Definition 2.1 ($\mathcal{ALC}$-Konzepte).
Die Menge der $\mathcal{ALC}$-*Konzepte* ist induktiv definiert:

- Jeder Konzeptname ist $\mathcal{ALC}$-Konzept
- Wenn C, D $\mathcal{ALC}$-Konzepte, so auch
  - $\neg C$ (Negation)
  - $C \sqcap D$ (Konjunktion)
  - $C \sqcup D$ (Disjunktion)
- Wenn C $\mathcal{ALC}$-Konzept und r Rollenname, so sind
  - $\exists r.C$ (Existenzrestriktion)
  - $\forall r.C$ (Werterestriktion)

  $\mathcal{ALC}$-Konzepte.

T2.1

ALC: Syntax

Verwendete Symbole:

- A, B für Konzeptnamen
- C, D für zusammengesetzte Konzepte
- r, s für Rollennamen

Zwei nützliche Abkürzungen: wir schreiben

- $\top$ für $A \sqcup \neg A$ (Top-Konzept)
- $\bot$ für $A \sqcap \neg A$ (Bottom-Konzept)

wobei A ein beliebiger Konzeptname ist.

ALC: Syntax

Präzedenzregel:

- $\neg$, $\exists$, $\forall$ binden stärker als $\sqcap$ und $\sqcup$

Also zum Beispiel:

$\forall r.(\exists r.A \sqcap B)$ steht für $\forall r.((\exists r.A) \sqcap B)$
und nicht für $\forall r.(\exists r.(A \sqcap B))$

Keine Präzedenz zwischen $\sqcap$ und $\sqcup$: Klammern verwenden!

ALC: Semantik

Definition 2.2 ($\mathcal{ALC}$ Semantik).
Eine *Interpretation* $\mathcal{I}$ ist Paar $(\Delta^\mathcal{I}, \cdot^\mathcal{I})$ mit

- $\Delta^\mathcal{I}$ nicht-leere Menge (*Domäne*)
- $\cdot^\mathcal{I}$ *Interpretationsfunktion* bildet ab:
  - jeden Konzeptnamen A auf Menge $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$
  - jeden Rollennamen r auf Relation $r^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$

T2.2

Abbildung $\cdot^\mathcal{I}$ wird induktiv auf zusammengesetzte Konzepte erweitert:

- $(\neg C)^\mathcal{I} = \Delta^\mathcal{I} \setminus C^\mathcal{I}$,
- $(C \sqcap D)^\mathcal{I} = C^\mathcal{I} \cap D^\mathcal{I}$, $\quad (C \sqcup D)^\mathcal{I} = C^\mathcal{I} \cup D^\mathcal{I}$,
- $(\exists r.C)^\mathcal{I} = \{d \in \Delta^\mathcal{I} \mid$ es gibt $e \in \Delta^\mathcal{I}$ mit $(d,e) \in r^\mathcal{I}$ und $e \in C^\mathcal{I}\}$,
- $(\forall r.C)^\mathcal{I} = \{d \in \Delta^\mathcal{I} \mid$ für alle $e \in \Delta^\mathcal{I}$, $(d,e) \in r^\mathcal{I}$ impliziert $e \in C^\mathcal{I}\}$

T2.2 cont.

ALC: Semantik

Verwendete Symbole:

- $\mathcal{I}, \mathcal{J}$ für Interpretationen
- d, e für Elemente der Domäne

Für Interpretationen verwenden wir übliche Terminologie für Graphen:

- e ist r-*Nachfolger* von d (in $\mathcal{I}$) wenn $(d,e) \in r^\mathcal{I}$;
- e ist r-*Vorgänger* von d (in $\mathcal{I}$) wenn $(e,d) \in r^\mathcal{I}$;
- wenn r unwichtig, sprechen wir nur von *Nachfolgern* / *Vorgängern*;
- $\mathcal{I}$ ist *endlich* gdw. $\Delta^\mathcal{I}$ endlich ist.

Extension/ Modell

Wir nennen

- $C^\mathcal{I}$ die *Extension* des Konzeptes C;
- jedes $d \in C^\mathcal{I}$ eine *Instanz* des Konzeptes C;
- $r^\mathcal{I}$ die *Extension* der Rolle r.

Beachte:

- $\top^\mathcal{I}$ ist für jede Interpretation $\mathcal{I}$ identisch mit $\Delta^\mathcal{I}$
  $\top$ repräsentiert also immer die Menge *aller* Elemente

  Mensch $\sqcap$ $\exists$hatKind.$\top$ — Menschen, die ein nicht näher spezifiziertes Kind haben

- $\bot^\mathcal{I}$ ist für jede Interpretation $\mathcal{I}$ leer
  Intuitiv repräsentiert $\bot$, dass etwas unmöglich ist, z.B.:

  Mensch $\sqcap$ $\forall$hatKind.$\bot$ — Menschen, die keine Kinder haben

T2.3

Erfüllbarkeit, Subsumtion, Äquivalenz

Folgende Begriffe aus der Aussagenlogik spielen auch für $\mathcal{ALC}$ eine Rolle:

Definition 2.3 (Erfüllbar, subsumiert, äquivalent)
Seien C und D $\mathcal{ALC}$-Konzepte. Dann

- ist C *erfüllbar* wenn es Interpretation $\mathcal{I}$ gibt mit $C^\mathcal{I} \neq \emptyset$
  wir nennen $\mathcal{I}$ dann ein *Modell* von C;
- wird C *von* D *subsumiert* wenn $C^\mathcal{I} \subseteq D^\mathcal{I}$ in allen Interpretationen $\mathcal{I}$ (Notation $C \sqsubseteq D$)
- sind C und D *äquivalent* wenn $C^\mathcal{I} = D^\mathcal{I}$ in allen Interpretationen $\mathcal{I}$ (Notation $C \equiv D$).

T2.4

In der Aussagenlogik sagt man “D folgt aus C” statt “D subsumiert C”

Erfüllbarkeit, Subsumtion, Äquivalenz

Die üblichen aussagenlogischen Äquivalenzen gelten auch in $\mathcal{ALC}$, z.B.:

$\neg(C \sqcap D) \equiv \neg C \sqcup \neg D$	de Morgansches Gesetz
$\neg(C \sqcup D) \equiv \neg C \sqcap \neg D$	de Morgansches Gesetz
$C \sqcap D \equiv D \sqcap C$	Kommutativität von Konjunktion
$C \sqcup D \equiv D \sqcup C$	und Disjunktion
$C \sqcap (D \sqcap E) \equiv (C \sqcap D) \sqcap E$	Assoziativität von Konjunktion
$C \sqcup (D \sqcup E) \equiv (C \sqcup D) \sqcup E$	und Disjunktion
$C \sqcap \top \equiv C$	Neutrales Element Konjunktion
$C \sqcup \bot \equiv C$	Neutrales Element Disjunktion

Kapitel 2

Grundlagen

TBoxen

TBoxen

Zur Erinnerung:

TBoxen (terminologische Boxen)

- definieren Konzepte
- setzen diese zueinander in Beziehung

Konzeptdefinition z.B.

Student $\equiv$ Mensch $\sqcap$ $\exists$hört.Vorlesung

Allgemeines Hintergrundwissen / Constraint z.B.

Student $\sqcap$ Vorlesungssaal $\sqsubseteq \bot$

TBoxen - Syntax und Semantik

Definition 2.4 (TBox—Syntax)
Konzeptinklusion ist Ausdruck $C \sqsubseteq D$, mit C, D Konzepten.
TBox ist endliche Menge von Konzeptinklusionen.
Wir verwenden $C \equiv D$ als Abkürzung für $C \sqsubseteq D$, $D \sqsubseteq C$. T2.6

Konzeptinklusion $C \sqsubseteq D$ wird gelesen als „C impliziert D“.

Definition 2.5 (TBox—Semantik)
Interpretation $\mathcal{I}$

- *erfüllt* Konzeptinklusion $C \sqsubseteq D$ gdw. $C^\mathcal{I} \subseteq D^\mathcal{I}$;
- ist *Modell* von TBox $\mathcal{T}$ gdw. $\mathcal{I}$ alle Konzeptinklusionen in $\mathcal{T}$ erfüllt.

Beachte: $\mathcal{I}$ erfüllt $C \equiv D$ gdw. $C^\mathcal{I} = D^\mathcal{I}$ T2.6 cont

TBoxen - Semantik

Beachte: in einer konkreten Interpretation $\mathcal{I}$ werden

- *Konzepte* als *Mengen von Elementen* interpretiert
- *TBoxen* entweder erfüllt oder nicht erfüllt, ihnen wird also ein *Wahrheitswert* zugewiesen.

Intuitiv entspricht jede Interpretation einer *möglichen Welt*.

Manche dieser Welten möchten wir *ausschließen*, weil wir sie aufgrund unseres Wissens nicht für möglich halten.

Genau dazu dienen TBoxen:
jede Konzeptinklusion *„eliminiert“ unerwünschte Interpretationen*.

TBoxen – Modellierung

In der Praxis bestehen TBoxen zu einem großen Teil aus

- Konzeptinklusionen $A \sqsubseteq C$, wobei A ein Konzept*name* ist

  Intuitiv ist C *notwendige Bedingung* dafür, eine Instanz von A zu sein

  Zum Beispiel in SNOMED:

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

- Konzeptdefinitionen $A \equiv C$, wobei A ein Konzept*name* ist

  Intuitiv ist C *notwendige und hinreichende Bedingung* dafür, eine Instanz von A zu sein

  Zum Beispiel in SNOMED:

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

Hinreichende Bedingungen lassen sich für viele Konzepte nicht leicht finden.

TBoxen – Modellierung

Modellierungsmuster, die nicht in dieses Schema passen z.B.:

- Disjunktheitsconstraints $C \sqcap D \sqsubseteq \bot$

  Kein Objekt kann sowohl zu Konzept C als auch zu Konzept D gehören

  Zum Beispiel in SNOMED:

  $$\mathsf{Befund} \sqcap \mathsf{Körperstruktur} \sqsubseteq \bot$$

- Komplexe Zusammenhänge zwischen mehreren Konzepten

  Zum Beispiel:

  $$\mathsf{Professor} \sqcap \exists \mathsf{hat.Lehrdeputat} \sqsubseteq \exists \mathsf{hält.Vorlesung}$$

Erfüllbarkeit, Subsumtion, Äquivalenz

Wir erweitern unsere grundlegenden Begriffe auf TBoxen:

Definition 2.6 (Erfüllbar, subsumiert, äquivalent bezüglich einer TBox)
Seien C, D $\mathcal{ALC}$-Konzepte und $\mathcal{T}$ TBox. Dann

- ist C *erfüllbar bzgl.* $\mathcal{T}$ gdw. $\mathcal{T}$ Modell $\mathcal{I}$ hat mit $C^{\mathcal{I}} \neq \emptyset$
- wird C *von* D *subsumiert bzgl.* $\mathcal{T}$ gdw $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ in allen Modellen $\mathcal{I}$ von $\mathcal{T}$ (Notation $\mathcal{T} \models C \sqsubseteq D$)
- sind C und D *äquivalent bzgl.* $\mathcal{T}$ gdw $C^{\mathcal{I}} = D^{\mathcal{I}}$ in allen Modellen $\mathcal{I}$ von $\mathcal{T}$ (Notation $\mathcal{T} \models C \equiv D$).

T2.7

Kleine Übung für Zwischendurch

Aufgabe 1

$$\mathcal{T} = \{ \quad A \sqsubseteq \exists r.B \sqcap \exists s.B, \quad \exists r.B \sqsubseteq \forall r.A \quad \}$$

Ist das Konzept A erfüllbar bzgl. der TBox $\mathcal{T}$?

Aufgabe 2

$$\mathcal{T} = \{ \exists r.\top \sqsubseteq A_1, \quad \forall r.B \sqsubseteq A_2 \}$$

Gilt $\mathcal{T} \models \top \sqsubseteq A_1 \sqcup A_2$?

Erfüllbarkeit zeigen: Modell angeben.
Unerfüllbarkeit zeigen: semantisch argumentieren.

Subsumtion zeigen: semantisch argumentieren.
Nicht-Subsumtion zeigen: Gegenmodell angeben.

Monotonie

Das Erweitern einer TBox um zusätzliche Konzeptinklusionen wirkt sich wie folgt auf Erfüllbarkeit und Subsumption aus:

Lemma 2.7 Seien $\mathcal{T}_1$ und $\mathcal{T}_2$ TBoxen mit $\mathcal{T}_1 \subseteq \mathcal{T}_2$. Dann gilt:

1. Wenn C erfüllbar bzgl. $\mathcal{T}_2$, dann ist C erfüllbar bzgl. $\mathcal{T}_1$;
2. Wenn $\mathcal{T}_1 \models C \sqsubseteq D$, dann $\mathcal{T}_2 \models C \sqsubseteq D$.

Die umgekehrten Aussagen sind im Allgemeinen nicht richtig. T2.8

Eine Logik, die diese Eigenschaften erfüllt, nennt man *monoton*
(das Hinzunehmen von Formeln kann nur zu *zusätzlichen* Konsequenzen, führen, aber nicht dazu, dass Konsequenzen ungültig werden)

Kapitel 2

Grundlagen

Schlussfolgerungsprobleme

Schlussfolgerungsprobleme

Zur Erinnerung: Schlussfolgern ...

- ist der wichtigste Bestandteil eines intelligenten Systems
- dient dazu, aus explizit gegebenem Wissen neues Wissen abzuleiten, das vorher nur implizit vorhanden war

Wichtigstes Designkriterium für Beschreibungslogiken:

So viel Ausdrucksstärke wie nötig, aber nicht mehr, um möglichst effizientes Schlussfolgern zu erlauben.

Schlussfolgerungsprobleme

Erfüllbarkeitsproblem:

gegeben C und $\mathcal{T}$, entscheide ob C erfüllbar bzgl. $\mathcal{T}$;

Subsumtionsproblem:

gegeben C, D und $\mathcal{T}$, entscheide ob $\mathcal{T} \models C \sqsubseteq D$;

Äquivalenzproblem:

gegeben C, D und $\mathcal{T}$, entscheide ob $\mathcal{T} \models C \equiv D$;

Diese Entscheidungsprobleme kann man auch mit leerer TBox $\mathcal{T} = \emptyset$ betrachten.

Motivation

Anwendung:

- Modellierungsfehler finden:
  unerfüllbare Konzepte sind im Allgemeinen unerwünscht;
- die Struktur der TBox explizit machen, z. B. für Browsing:
  Subsumtion (= Is-A-Relation) ist Haupt-Strukturierungsmittel für TBoxen
  $C \sqsubseteq D$ kann gelesen werden als „D ist genereller als C"
- Redundanzen finden:
  zwei äquivalente Konzepte sind u. U. unerwünscht.

Subsumtion als Ordnungsrelation

Lemma 2.8
Für jede TBox $\mathcal{T}$ ist die Relation „$\sqsubseteq$ bzgl. $\mathcal{T}$"

- reflexiv ($\mathcal{T} \models C \sqsubseteq C$) und
- transitiv ($\mathcal{T} \models C \sqsubseteq D$ und $\mathcal{T} \models D \sqsubseteq E$ impliziert $\mathcal{T} \models C \sqsubseteq E$).

Bis auf fehlende Antisymmetrie ist $\sqsubseteq$ also partielle Ordnung.

Man kann $\sqsubseteq$ als Hasse-Diagram darstellen, dessen Knoten mit Mengen von Konzepten beschriftet sind.

Normalerweise Einschränkung auf die in $\mathcal{T}$ verwendeten Konzeptnamen

T2.9

Klassifikation

Ein weiteres Schlussfolgerungsproblem:

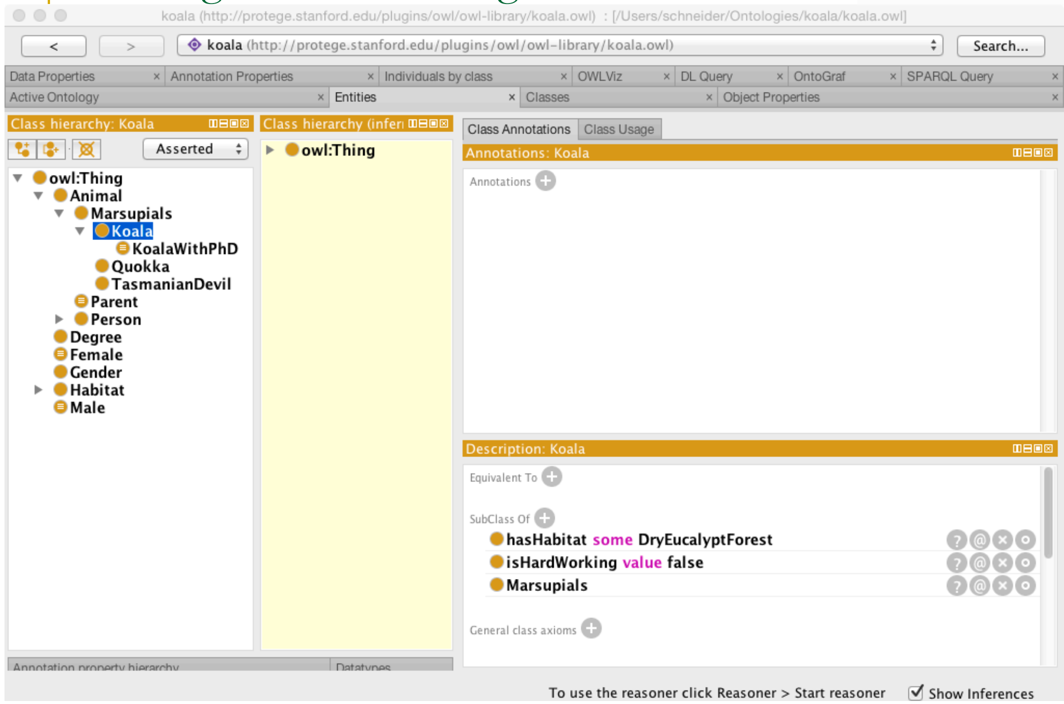
- *Klassifikation*: gegeben $\mathcal{T}$, berechne das Hasse-Diagramm für $\sqsubseteq$ bzgl. $\mathcal{T}$, eingeschränkt auf Konzeptnamen in $\mathcal{T}$.

Berechnungsproblem, kein Entscheidungsproblem.

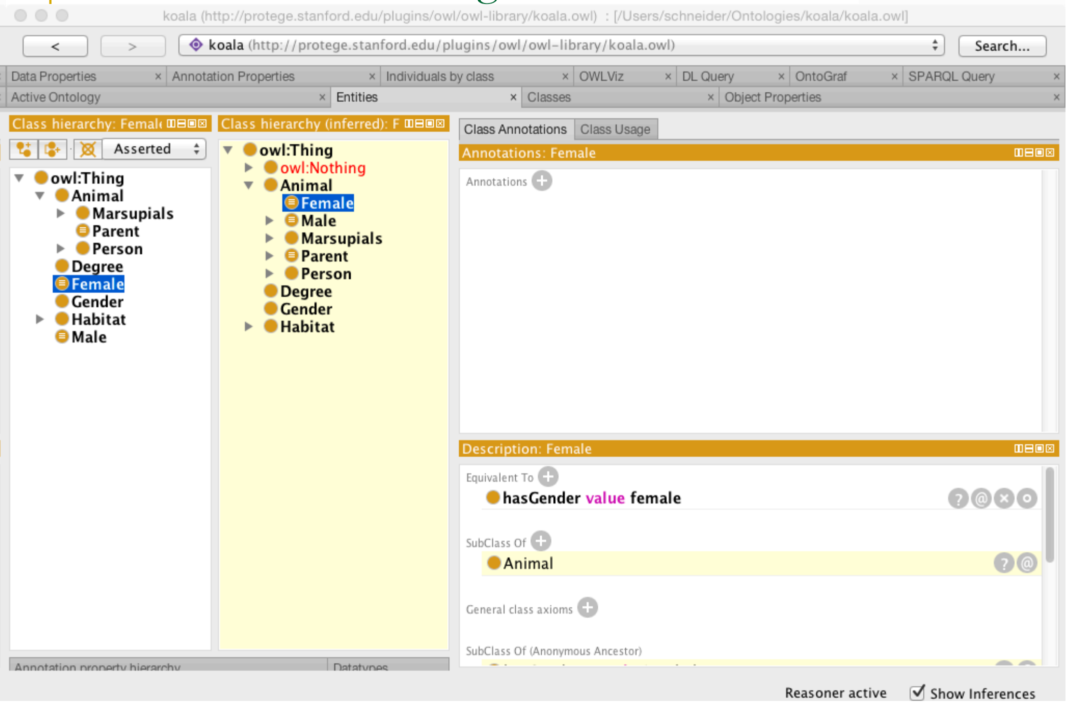
In der Praxis:

- berechenbar durch n^2 Subsumtionsberechnungen;
- zahlreiche Optimierungen verfügbar.

Ontologie-Editor Protégé



Klassifikation in Protégé



Reduktionen

Erfüllbarkeit, Subsumtion, Äquivalenz wechselseitig polynomiell reduzierbar:

Lemma 2.9.

1. Subsumtion polynomiell reduzierbar auf (Un)erfüllbarkeit:
   $\mathcal{T} \models C \sqsubseteq D$ gdw. $C \sqcap \neg D$ unerfüllbar bzgl. $\mathcal{T}$
2. Erfüllbarkeit polynomiell reduzierbar auf (Nicht-)Äquivalenz:
   C erfüllbar bzgl. $\mathcal{T}$ gdw. $\mathcal{T} \not\models C \equiv \bot$
3. Äquivalenz polynomiell reduzierbar auf Subsumtion:
   $\mathcal{T} \models C \equiv D$ gdw. $\mathcal{T} \models \top \sqsubseteq (C \sqcap D) \sqcup (\neg C \sqcap \neg D)$ T2.10

Algorithmus für eines der Probleme kann also auch für die beiden anderen verwendet werden; alle drei Probleme haben dieselbe Komplexität (in $\mathcal{ALC}$)

Im Folgenden konzentrieren wir uns meist auf Erfüllbarkeit

Kapitel 2

Grundlagen

Erweiterungen von $\mathcal{ALC}$

Erweiterungen von ALC

Wir betrachten exemplarisch zwei Erweiterungen von $\mathcal{ALC}$:

- $\mathcal{ALCI}$: $\mathcal{ALC}$ mit inversen Rollen (Rollenkonstruktor)
- $\mathcal{ALCQ}$: $\mathcal{ALC}$ mit Zahlenrestriktionen (Konzeptkonstruktor)

Beide Erweiterungen sind in OWL realisiert.

Namenschema:

- ein Buchstabe pro Erweiterung
- kann kombiniert werden, z.B. $\mathcal{ALCQI}$

Inverse Rollen

Häufig möchte man über Rollen „in beiden Richtungen“ reden:

- SNOMED: hatTeil / istTeilVon (z.B. in der Anatomie)
- Universitätsbeispiel: hört / wirdGehörtVon ,
  gibt / wirdGegebenVon

Verwendet man diese Rollen einfach als Namen in $\mathcal{ALC}$,
gibt es unintuitive Konsequenzen.

T2.11

Inverse Rollen

Definition 2.10. (Inverse Rollen)
Für jeden Rollennamen r ist r^- die *inverse Rolle* zu r. Wir definieren

$$(r^-)^\mathcal{I} = \{(e, d) \mid (d, e) \in r^\mathcal{I}\}.$$

$\mathcal{ALCI}$: $\mathcal{ALC}$ erweitert um die Möglichkeit, inverse Rollen in Existenz- und Werterestriktionen zu benutzen.

T2.11 cont.

Zahlenrestriktionen

Häufig möchte man Rollennachfolger „zählen“ können:

- SNOMED: Eine Hand ist ein Organ mit fünf Teilen, die ein Finger sind.
- Universitätsbeispiel: in jedem Semester werden mindestens 2
  Wahlpflichtmodule angeboten

Wir werden sehen:
In $\mathcal{ALC}$ ist Zählen nicht ohne weiteres möglich

Zahlenrestriktionen

Definition 2.11. (Zahlenrestriktion)
Für jede natürliche Zahl n, jeden Rollennamen r und jedes Konzept C:

- $(\leqslant n\ r\ C)$ (Höchstens-Restriktion);
- $(\geqslant n\ r\ C)$ (Mindestens-Restriktion);

Die Semantik ist

- $(\leqslant n\ r\ C)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \#\{e \mid (d, e) \in r^{\mathcal{I}} \wedge e \in C^{\mathcal{I}}\} \leq n\}$
- $(\geqslant n\ r\ C)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \#\{e \mid (d, e) \in r^{\mathcal{I}} \wedge e \in C^{\mathcal{I}}\} \geq n\}$

Beachte:

T2.12

- $\exists r.C$ ist äquivalent zu $(\geqslant 1\ r\ C)$
- $\forall r.C$ ist äquivalent zu $(\leqslant 0\ r\ \neg C)$

$\mathcal{ALCQ}$: $\mathcal{ALC}$ erweitert mit Zahlenrestriktionen.

Weitere Erweiterungen

Es gibt noch viel mehr Erweiterungen:

- Spezielle Rolleninterpretationen (transitiv, symmetrisch, reflexiv etc.)
- Konzepte, die nur eine einzige Instanz haben können (Nominale)
- Inklusionen zwischen Rollen oder Rollenketten
- Numerische Werte
- Fixpunktoperatoren
- temporale Operatoren
- ...

Viele davon sind in OWL realisiert.