

Struktur Vorlesung

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

Grundlagen

Die Beschreibungslogik *ALC*

ALC

Es gibt viele verschiedene Beschreibungslogiken:

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

Hier zunächst *ALC*:

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

ALC

Von nun an sei

- \mathbf{N}_C eine unendliche Menge von Konzeptnamen
Diese bezeichnen Klassen / unäre Prädikate
z.B. Person, Kurs, Universität, Tafel, Student, etc.
- \mathbf{N}_R eine unendliche Menge von Rollennamen
Diese bezeichnen Relationen / binäre Prädikate
z.B. hört, lehrt, istTeilVon, etc.

Wir nehmen \mathbf{N}_C und \mathbf{N}_R als disjunkt und abzählbar unendlich an

Beachte Gross- / Kleinschreibung!

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

- Konzeptnamen
- Rollennamen
- Konzeptkonstruktoren (manchmal auch Rollenkonstruktoren)

Es gibt viele verschiedene Konstruktoren

Versch. Konstruktormengen ergeben versch. Beschreibungslogiken

ALC Syntax

Definition 2.1 (*ALC*-Konzepte).

Die Menge der *ALC*-Konzepte ist induktiv definiert:

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

T2.1

ALC Syntax

Verwendete Symbole:

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

Abkürzungen:

- $A \rightarrow B$ für $\neg A \sqcup B$
- $A \leftrightarrow B$ für $(A \rightarrow B) \sqcap (B \rightarrow A)$
- \top für $A \sqcup \neg A$ (Top Konzept)
- \perp für $\neg \top$ (Bottom Konzept)

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 (**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 $A \in \mathbf{N}_C$ auf Menge $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
 - jeden Rollennamen $r \in \mathbf{N}_R$ auf Relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$

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

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

T2.2

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, lassen wir es weg;
- e ist *erreichbar* von d wenn es d_0, \dots, d_n gibt ($n \geq 0$) mit $d_0 = d$, $d_n = e$ und d_{i+1} ist Nachfolger von d_i für alle $i < n$;
- \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 .

Interpretation \mathcal{I} ist *Modell* von Konzept C gdw. $C^{\mathcal{I}} \neq \emptyset$.

Grundlagen

TBoxen und Ontologien

TBoxen

Zur Erinnerung:

TBoxen (terminologische Boxen)

- definieren Konzepte
- setzen diese zueinander in Beziehung

Konzeptdefinition z.B.

$\text{Student} \equiv \text{Mensch} \sqcap \exists \text{hört.Vorlesung}$

Allgemeines Hintergrundwissen / Constraint z.B.

$\text{Student} \sqcap \text{Vorlesungssaal} \sqsubseteq \perp$

TBoxen

Wir führen zwei Arten von TBoxen ein:

- Generelle TBoxen
Ausdrucksstark, aber recht hohe Komplexität beim Schlussfolgern
- Azyklische TBoxen.
Erlauben nur die Definition von Konzepten, keine allgemeinen Constraints
für viele Anwendungen ausreichend, z.B. SNOMED, NCI Thesaurus

Wir setzen TBox = Ontologie.

Folgende Definitionen sind unabhängig von *ALC*.

Generelle TBox - Syntax und Semantik

Definition 2.3 (Generelle TBox—Syntax)

Konzeptinklusion ist Ausdruck $C \sqsubseteq D$, mit C, D Konzepten.

Generelle TBox ist endliche Menge von Konzeptinklusionen.

Wir verwenden $C \equiv D$ als Abkürzung für $C \sqsubseteq D, D \sqsubseteq C$.

T2.3

Definition 2.4 (Generelle 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. $C^{\mathcal{I}} = D^{\mathcal{I}}$

Wir schreiben $\mathcal{I} \models C \sqsubseteq D$, $\mathcal{I} \models C \equiv D$, $\mathcal{I} \models \mathcal{T}$.

T2.3 cont

Von generellen zu azyklischen TBoxen

Oft nur Konzeptnamen auf der linken Seite von Konzeptinklusionen:

Perikardium \sqsubseteq Gewebe \sqcap \exists teilVon.Herz

Perikarditis \equiv Entzündung \sqcap \exists ort.Perikardium

Entzündung \sqsubseteq Krankheit \sqcap \exists wirktAuf.Gewebe

Intuition:

- Definition $A \equiv C$ gibt notwendige und hinreichende Bedingungen dafür, eine Instanz des Konzeptes A zu sein.
- Inklusion $A \sqsubseteq C$ gibt nur notwendige Bedingungen.

Azyklische TBoxen schränken sich auf diesen Fall ein.

Azyklische TBox - Syntax

Definition 2.3 (Azyklische TBox)

- *Konzeptdefinition* ist Ausdruck $A \equiv C$
- *Primitive Konzeptinklusion* ist Ausdruck $A \sqsubseteq C$.

mit A Konzeptname und C Konzept.

Azyklische TBox ist endliche Menge von Konzeptdefinitionen und primitiven Konzeptinklusionen so dass

- linke Seiten eindeutig
- keine Zyklen vorkommen, also keine Ausdrücke

$$A_0 \sim_0 C_0 \quad , \quad \dots \quad , \quad A_{n-1} \sim_{n-1} C_{n-1}, \quad \text{mit } \sim_i \subseteq \{\sqsubseteq, \equiv\}$$

so dass A_i in C_{i+1} vorkommt, für alle $i < n$ (modulo n)

T2.4

Azyklische TBox - Semantik

Da azyklische TBoxen Spezialfall von generellen TBoxen, ist Semantik bereits definiert.

Für azyklische TBox \mathcal{T} :

- Konzeptname A is *definiert* in \mathcal{T} gdw. \mathcal{T} enthält $A \equiv C$
- sonst ist A *primitiv* in \mathcal{T} .

T2.5

Im folgenden verwenden wir meistens generelle TBoxen,
verwenden “TBox” synonym zu “generelle TBox”

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 möglich, aber wenig genug, um effizientes/entscheidbares Schlussfolgern zu erlauben.

Erfüllbarkeit, Subsumtion

Definition 2.6 (Erfüllbar, subsumiert, äquivalent)

Seien C, D \mathcal{ALC} -Konzepte und \mathcal{T} TBox. Dann

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

Diese Begriffe existieren auch ohne Bezug auf TBox (i.e., TBox leer)

T2.6

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 $\mathcal{T} \models C \equiv D$;

Alle drei Probleme auch ohne (mit leerer) TBox.

Untersch. TBox-Formalisten geben untersch. Entscheidungsprobleme.

Motivation

Anwendung:

- Modellierungsfehler finden:
unerfüllbare Konzepte im allg. unerwünscht;
- die Struktur der TBox explizit machen, z.B. für Browsing:
Subsumption (= Is-A Relation) 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 2.7

Seit \mathcal{T} TBox mit mind. einem Modell. Die Relation " \sqsubseteq bzgl. \mathcal{T} " auf der Menge aller \mathcal{ALC} -Konzepte ist

- reflexiv ($\mathcal{T} \models C \sqsubseteq C$),
- 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-Diagramm darstellen, dessen Knoten mit Mengen von Konzepten beschriftet sind.

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

T2.7

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.

Doors Protégé 3.2 beta (file:/home/aa/turhan/GonMoRe/Ontologies/Final-Tests/Door-Tests/Doors.pprj, OWL / RDF Files)

File Edit Project OWL Code Tools Window Help

owl:Thing

Activity

AuthorizedPerson

Context

DeliveryGoods

Device

AudioEnabledDevice

AudioEnabledConnectedDevice

InstantMessageEnabledDevice

InstantMessageEnabledConnectedDevice

MobileDevice

Handy

Laptop

PDA

SmartPhone

SMSEnabledDevice

StationaryDevice

TextEnabledConnectedDevice

TextEnabledDevice

VideoEnabledConnectedDevice

VideoEnabledDevice

Location

Network

Person

Presence

ValuePartition

owl:Thing

Activity

Context

DeliveryGoods

Device

AudioEnabledDevice

AudioEnabledConnectedDevice

DoorBell

TextEnabledDevice

MobileDevice

Handy

InstantMessageEnabledDevice

InstantMessageEnabledConnectedDevice

Laptop

PDA

SmartPhone

StationaryDevice

VideoEnabledDevice

Location

Network

Person

Presence

ValuePartition

InstantMessageEnabledConnectedDevice

Name

InstantMessageEnabledConnectedDevice

Annotations

Property Value La...

rdfs:comment

Asserted Conditions

InstantMessageEnabledDevice NECESSARY & SUFFICIENT

isConnectedVia some Network NECESSARY

Laptop or PDA [from InstantMessageEnabledDevice] INHERITED

Properties

isConnectedVia (multiple Network)

Network

isOwnedBy (multiple Person)

isUsedBy (single Person)

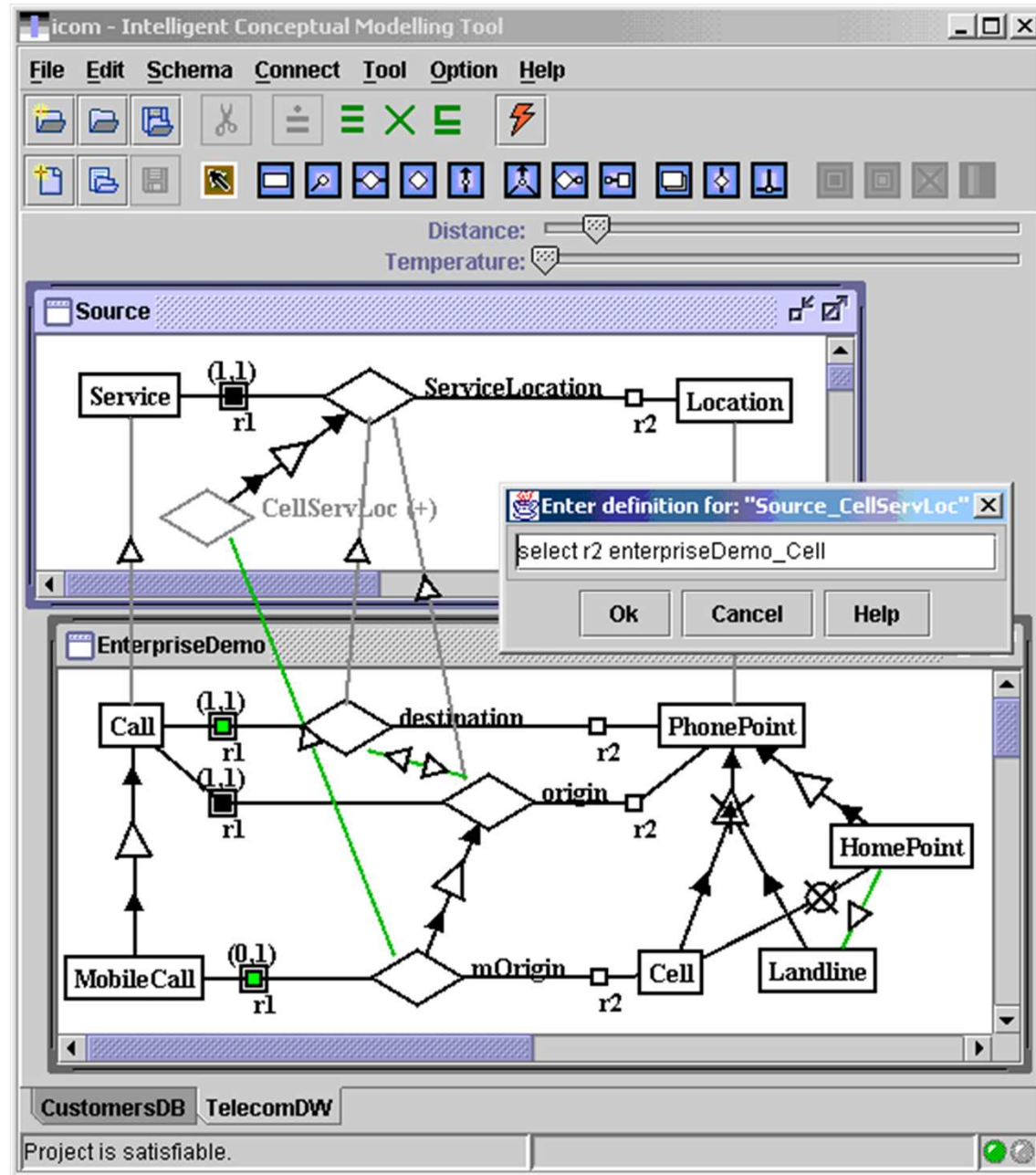
supportsLink (multiple Network)

Dis joints

Logic View Properties View

Class	Changed direct superclasses
AudioReachableInUrgencyContext	Removed PresenceContext
AudioReachableResidentContext	Removed ReachableContext
AuthorisedPersonRingingContext	Added PresenceContext
AuthorizedNeighbourRingingContext	Moved from DoorContext to AuthorisedPersonRingingContext
AuthorizedPerson	Moved from owl:Thing to Person
Building	Removed Location
ConnectedResident	Added ConnectedPerson
DoorBell	Added AudioEnabledDevice
DoorbellReachableInUrgencyContext	Removed PresenceContext
DoorbellReachableResidentContext	Removed ReachableContext
DoorContext	Moved from Context to ConnectedContext, PresenceContext
Handy	Added SMSEnabledDevice
InstantMessageEnabledConnectedDevice	Added TextEnabledConnectedDevice
InstantMessageEnabledDevice	Moved from Device to TextEnabledDevice, VideoEnabledDevice, MobileDevice

Classification Results



Reduktionen

Erfüllbarkeit, Subsumtion, Äquivalenz wechselseitig polynomiall reduzierbar:

Lemma 2.8.

- $\mathcal{T} \models C \sqsubseteq D$ gdw. $C \sqcap \neg D$ unerfüllbar bzgl. \mathcal{T}
- C erfüllbar bzgl. \mathcal{T} gdw. $\mathcal{T} \not\models C \equiv \perp$
- $\mathcal{T} \models C \equiv D$ gdw. $\mathcal{T} \models \top \sqsubseteq C \leftrightarrow D$

T2.8

Natürlich auch korrekt wenn $\mathcal{T} = \emptyset$, erhält Azyklizität von \mathcal{T} .

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

Im folgenden konzentrieren wir uns meist auf Erfüllbarkeit

Reduktionen

Wir wollen zeigen:

Theorem 2.9

Erfüllbarkeit bzgl. azyklischer TBoxen kann reduziert werden auf Erfüllbarkeit ohne TBoxen.

Also kann Algorithmus, der Konzepterfüllbarkeit ohne TBoxen entscheidet auch in Gegenwart azyklischer TBoxen verwendet werden (u.U. aber höhere Komplexität!)

Mit Lemma 2.8 folgt dasselbe für Subsumtion und Äquivalenz.

Wir gehen in drei Schritten vor.

Beweis von Theorem 2.9

Schritt 1:

Ersetzen von primitiven Konzeptinklusionen durch Konzeptdefinitionen

Definitoriale TBox

Definition 2.10 (Definitoriale TBox)

Eine azyklische TBox heißt *definitoriale* wenn sie nur Konzeptdefinitionen aber keine Inklusionen enthält.

Sei \mathcal{T} azyklische TBox.

Definitoriale TBox \mathcal{T}' : ersetze jede Inklusion $A \sqsubseteq C \in \mathcal{T}$ durch

$$A \equiv X_A \sqcap C, \text{ mit } X_A \text{ neuer Konzeptname}$$

Lemma 2.11

Für alle \mathcal{ALC} -Konzepte C , die nicht die neuen Konzeptnamen verwenden, gilt:

$$C \text{ erfüllbar bzgl. } \mathcal{T} \text{ gdw. } C \text{ erfüllbar bzgl. } \mathcal{T}' \quad \text{T2.9}$$

Beweis von Theorem 2.9

Schritt 2:

Eliminieren von definierten Konzeptnamen auf den rechten Seiten von Konzeptdefinitionen

Azyklische TBox - Expandieren

Sei \mathcal{T} definatorische TBox.

Definiere Folge von definatorischen TBoxen $\mathcal{T}_0, \mathcal{T}_1, \dots$ wie folgt:

- $\mathcal{T}_0 = \mathcal{T}$;
- \mathcal{T}_{i+1} entsteht aus \mathcal{T}_i wie folgt:
wähle $A \equiv C \in \mathcal{T}_i$, so dass C definierten Konzeptnamen B enthält
ersetze alle Vorkommen von B in C durch D wenn $B \equiv D \in \mathcal{T}_i$

Da \mathcal{T} azyklisch, gibt es schliessliche kein solches $A \doteq C \in \mathcal{T}_i$ mehr.

Dieses \mathcal{T}_i ist die *Expansion* von \mathcal{T} .

Beachte: Endergebnis ist unabhängig von den konkreten Auswahlen

T2.10

Azyklische TBox - Expandieren

Zwei TBoxen sind *äquivalent* gdw. sie dieselben Modelle haben.

Lemma 2.13

Jede definitorische TBox ist äquivalent zu ihrer Expansion.

T2.11

Bemerkungen:

Intuitiv zeigt Lemma 2.13, dass primitive Konzeptdefinitionen nicht mehr sind als Makros

Azyklische TBox - Expandieren

Expansion kann zu einem exponentiellen Aufblasen der TBox führen:

$$\begin{aligned} \text{Expandieren von } \quad A_0 &\equiv \exists r.A_1 \sqcap \exists s.A_1 \\ &A_1 \equiv \exists r.A_2 \sqcap \exists s.A_2 \\ &\dots \\ &A_{n-1} \equiv \exists r.A_n \sqcap \exists s.A_n \end{aligned}$$

gibt TBox der Grösse $> 2^n$.

T2.12

Es gibt keine äquivalente kleinere TBox ohne definierte Konzepte auf den rechten Seiten.

(Größe einer TBox: Anzahl Symbole, mit denen sie aufgeschrieben wird
Konzeptnamen und Rollennamen zählen 1)

Beweis von Theorem 2.9

Schritt 3:

Expandieren der definierten Konzeptnamen im Eingabekonzept.

Zunächst etwas Vorbereitung.

Primitive Interpretation

Definition 2.14 (Primitive Interpretation)

Primitive Interpretation für azyklische TBox \mathcal{T} ist Interpretation, die nur primitive Konzeptnamen und Rollennamen interpretiert, aber keine definierten Konzeptnamen aus \mathcal{T} .

Lemma 2.15

Sei \mathcal{T} definatorische TBox. Jede primitive Interpretation kann in eindeutiger Weise zu Modell von \mathcal{T} erweitert werden.

T2.13

Beweis von Theorem 2.9

Theorem 2.9

Erfüllbarkeit bzgl. azyklischer TBoxen kann reduziert werden auf
Erfüllbarkeit ohne TBoxen.

T2.14

Grundlagen

Erweiterungen von *ALC*

Erweiterungen von ALC

Wir betrachten exemplarisch zwei Erweiterungen von *ALC*:

- *ALCI*: *ALC* mit inversen Rollen (Rollenkonstruktor)
- *ALCQ*: *ALC* mit Zahlenrestriktionen (Konzeptkonstruktor)

Beide Erweiterungen sind in OWL realisiert.

Namenschema:

- ein Buchstabe pro Erweiterung
- kann kombiniert werden, z.B. *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 *ACC*,
gibt es unintuitive Konsequenzen.

T2.15

Inverse Rollen

Definition 2.16. (Inverse Rollen)

Für jedes $r \in \mathbf{N}_R$ ist r^- die *inverse Rolle* zu r . Wir definieren

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

Eine *Rolle* ist entweder ein Rollenname oder ein inverse Rolle.

ALCI: ***ALC*** erweitert um die Möglichkeit, inverse Rollen in Existenz- und Wertrestriktionen zu benutzen.

T2.15 cont.

Schachtelung von Inversen i. allg. nicht sinnvoll da $((r^-)^-)^{\mathcal{I}} = r^{\mathcal{I}}$.

Daher setzen wir $(r^-)^- = r$.

Zahlenrestriktionen

Häufig möchte man Rollennachfolger “zählen” können:

- SNOMED: Eine Hand hat fünf Teile, die ein Finger sind.
- Universitätsbeispiel: wie viele Vorlesungen gibt ein Dozent?

In *ALC* ist zählen nicht ohne weiteres möglich

T2.16

Zahlenrestriktionen

Definition 2.17. (Zahlenrestriktion)

Für jede natürliche Zahl n , jeden Rollennamen r und jedes Konzept C definiere Konzepte wie folgt:

- $(\leq n r C)$ (Höchstens-Restriktion);
- $(\geq n r C)$ (Mindestens-Restriktion);

Die Semantik ist

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

Beachte:

T2.16 cont.

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

Zahlenrestriktionen

ALCQ: *ALC* erweitert mit Zahlenrestriktionen.

ALCQI: *ALC* erweitert mit Zahlenrestriktionen und inversen Rollen.

inverse Rollen auch in Zahlenrestriktionen, z.B.

Vorlesung \sqsubseteq (≥ 3 hört⁻ Student)

Weitere Erweiterungen

Es gibt noch viel mehr Erweiterungen:

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

Viele davon sind in OWL realisiert.