

Projekt

Reasoning Services: Tableau-Beweiser für Beschreibungslogiken

Carola Eschenbach, Özgür Özçep
Universität Hamburg, FB Informatik
AB Wissens- und Sprachverarbeitung (WSV)

Wintersemester 2006/2007

Einordnung in den Studienplan

Profil Intelligente Systeme

- Wissensverarbeitung
- Sprachverarbeitung
- (Bildverarbeitung)

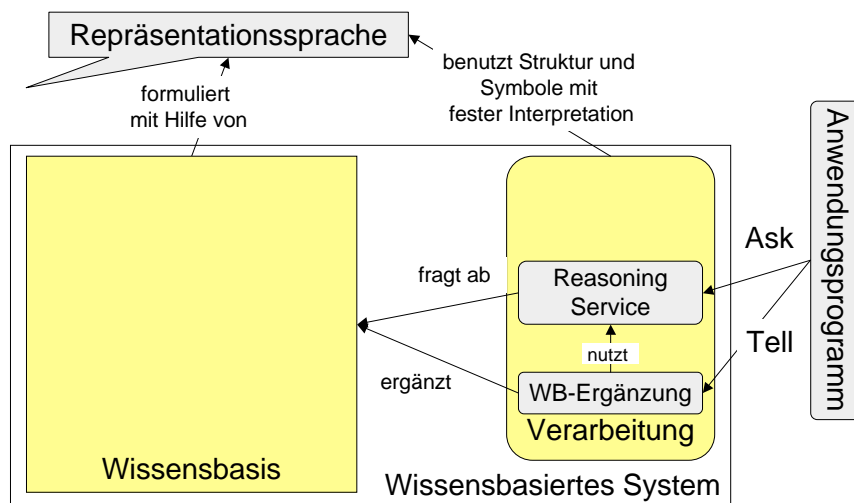
Logik

- Fortsetzung von LOS
- Detailstudie von Logik-Formalismen mit guten Verarbeitungseigenschaften

Aktuelle Anwendungen

- Semantic Web: Semantik-basierte Nutzbarkeit

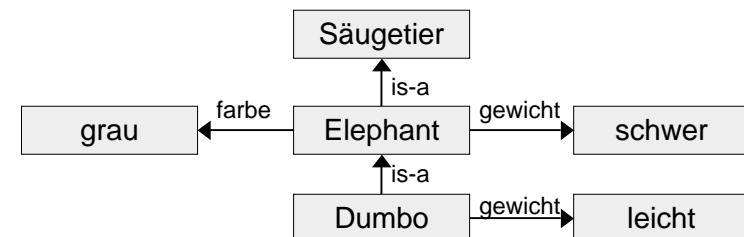
Wissensbasiertes System: Tell/Ask-Interface



Historischer Ursprung von Beschreibungslogiken

Semantische Netze

- Repräsentationen für Konzept-Wissen
- Graphen mit Knoten für Konzepte und Individuen und beschriftete Pfeilen für binäre Relationen
- Definiert über Bilder, keine klare Semantik, keine spezifizierten Verarbeitungsmechanismen



Beschreibungslogik-Ansatz

Im Kontrast zu semantischen Netzen

- Explizite modelltheoretische Semantik
 - der Name ‚Logik‘ will verdient sein
- Unterscheidung von
 - is-a zwischen Konzepten: Subsumption; und
 - is-a zwischen Objekten und Konzepten: Instanziierung
 - Diese beiden Relationen gehören zu den ausgezeichneten Relationen mit fester Interpretation
- Fokussierung von Verarbeitungsmöglichkeiten und Berechenbarkeitsfragen

Beschreibungslogiken

Andere Namen

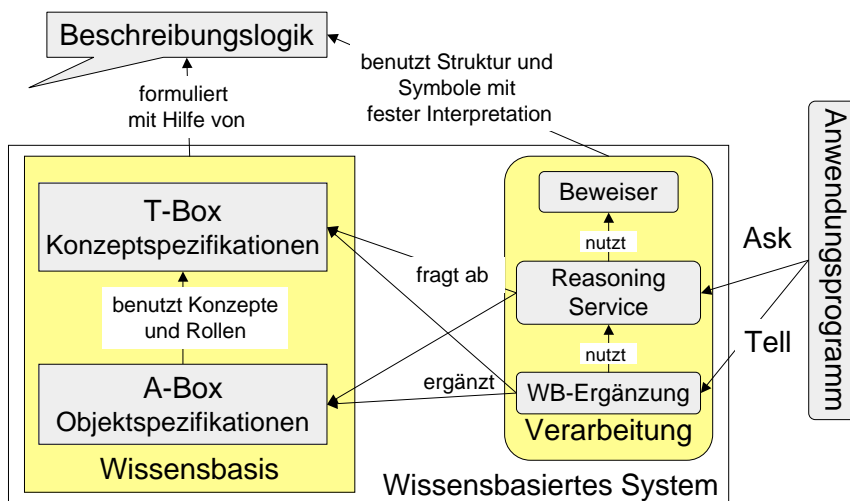
- Terminologische Logiken, KL-ONE-artige Sprachen
- Description logic, terminological logics, taxonomic logics, term subsumption systems, KL-ONE-like systems

Definition eines logischen Systems: Generelles Schema (s. LOS)

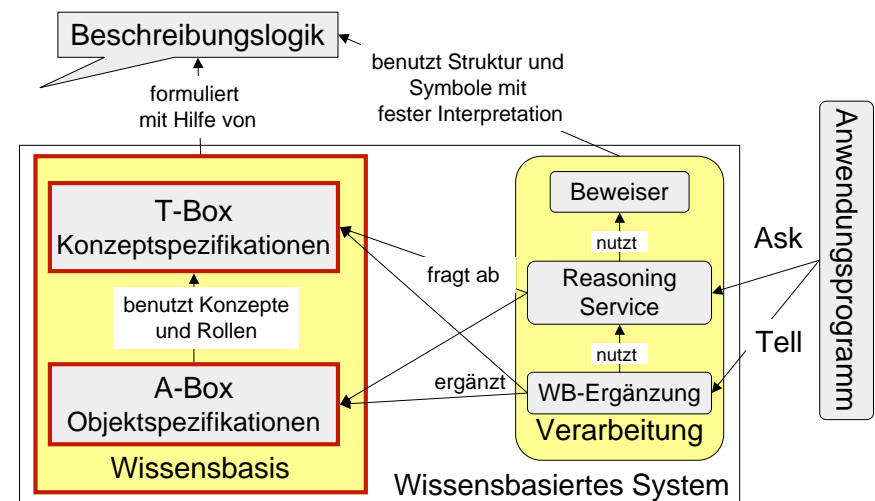
- eine formale Sprache (zur Repräsentation)
- Evaluations- / Interpretationsprinzipien
- semantische Kategorisierungen und Beziehungen
- Ableitungs-, Beweisverfahren

➤ Beschreibungslogiken bilden logische Systeme

Wissensbasiertes System mit Beschreibungslogiken



Beschreibungslogik-Wissensbasen



T-Box: Spezifikationen von Konzepten

Primitive Konzepte / Atomare Beschreibungen

- keine Definition vorhanden
- hinreichende Bedingungen nicht bekannt
- explizite Einordnung in Subsumptionshierarchie

Definierte Konzepte / Komplexe Beschreibungen

- Definition verfügbar
- notwendige und hinreichende Bedingungen
- basierend auf
 - (primitiven) Konzepten
 - Relationen / Rollen
 - Konzeptbildungsoperatoren
- implizite Einordnung in Subsumptionshierarchie

Inhalt der T-Box (Terminologie)

Es seien

C und D Konzeptbeschreibungen (atomar oder komplex)

R und S Rollenbeschreibungen (atomar oder komplex)

T-Box-Einträge (T-Box-Axiome)

- Formeln der Form $(C \sqsubseteq D)$ ($\text{Studierender} \sqsubseteq \text{Mensch}$)
 - C ist ein Unterkonzept von D
 - D subsummiert C
- und / oder Formeln der Form $(C \doteq D)$ ($\text{Apfelsine} \doteq \text{Orange}$)
 - C und D sind gleich, C ist durch D definiert
- ggf. auch Formeln der Form $(R \sqsubseteq S)$, $(R \doteq S)$

Weitere Beschränkungen z.B.

- C muss atomar sein, C darf nur einmal links vorkommen, keine Zyklen in der T-Box

T-Box-Restriktionen (Beispiele)

nur Definitionen $A \doteq C$

- keine *Spezialisierungen* wie $A \sqsubseteq C$ erlaubt
- Normalisierung: $A \doteq A' \sqcap C$

Zyklenfreiheit

- Beispiel für eine zyklische Definition: Ein Mensch ist ein Lebewesen mit menschlichen Eltern.
 $\text{Human} \doteq \text{Animal} \sqcap \forall \text{hasParent.Human}$
- Jede azyklische T-Box ist definitiv, d.h. bei einer gegebenen Interpretation der Basissymbole, ist genau eine Interpretation der definierten Konzepte möglich.
- Zyklische T-Boxen können, aber müssen nicht definitiv sein.



Inhalt der A-Box (Assertionen , Weltmodell)

Es seien

C und D Konzeptbeschreibungen (atomar oder komplex)

R und S Rollenbeschreibungen (atomar oder komplex)

a und b Individuenbezeichnungen (atomar oder komplex)

A-Box-Einträge

- Formeln der Form $C(a)$ ($\text{student}(\text{jens_waechter})$)
- $\neg C(a)$ ($\neg \text{student}(\text{carola_eschenbach})$)
- $R(a, b)$ ($\text{studiert}(\text{jens_waechter}, \text{informatik})$)
- $\neg R(a, b)$ ($\text{studiert}(\text{jens_waechter}, \text{mathematik})$)

Erweiterungen z.B.

- (Boole'sche A-Box) Kombinationen von atomaren Formeln mit aussagenlogischen Junktoren

Interpretationen sind Paare $\mathcal{S} = \langle \mathcal{D}, I \rangle$

\mathcal{D} : nicht-leere Menge von Objekten (Domäne, Universum, Diskursbereich)

I : Interpretation der frei verfügbaren Symbole, durch rekursive Definition erweitert auf alle Konzepte, Rollen und Individuenbezeichnungen

- Abbildung von Konzepten auf Teilmengen von \mathcal{D}
- Abbildung von Rollen auf Teilmengen von \mathcal{D}^2
- Abbildung von Individuenbezeichnungen auf ein Objekte in \mathcal{D}

Interpretationen sind Paare $\mathcal{S} = \langle \mathcal{D}, I \rangle$

Fortsetzung von I für Formeln

$I((C \sqsubseteq D)) = \text{wahr}$ genau dann, wenn $I(C) \subseteq I(D)$.

$I((C \doteq D)) = \text{wahr}$ genau dann, wenn $I(C) = I(D)$.

$I(C(a)) = \text{wahr}$ genau dann, wenn $I(a) \in I(C)$

$I(\neg C(a)) = \text{wahr}$ genau dann, wenn $I(a) \notin I(C)$

$I(R(a, b)) = \text{wahr}$ genau dann, wenn $(I(a), I(b)) \in I(R)$

$I(\neg R(a, b)) = \text{wahr}$ genau dann, wenn $(I(a), I(b)) \notin I(R)$

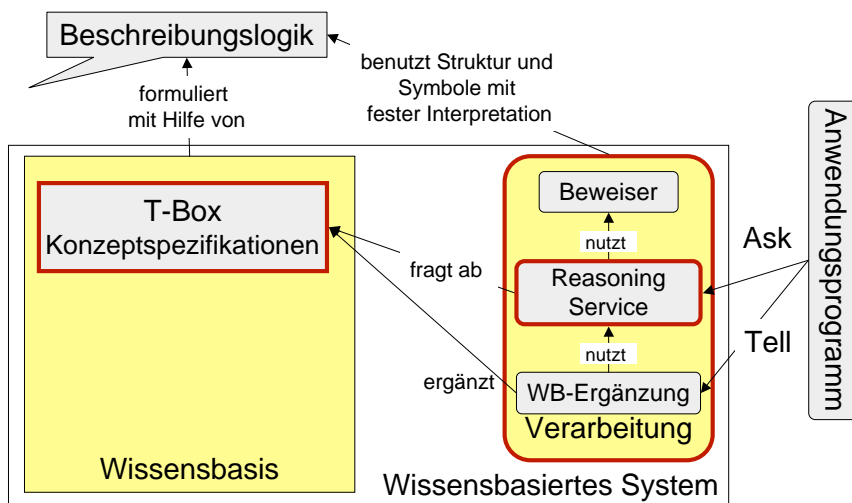
Wir schreiben dann auch

$\mathcal{S} \models (C \sqsubseteq D)$ bzw. $\mathcal{S} \models (C \doteq D)$ bzw. $\mathcal{S} \models C(a) \dots$

und sagen

\mathcal{S} ist ein Modell der Formel / macht die Formel wahr.

Simpler T-Box-Reasoning-Services



T-Box-Inferenzen

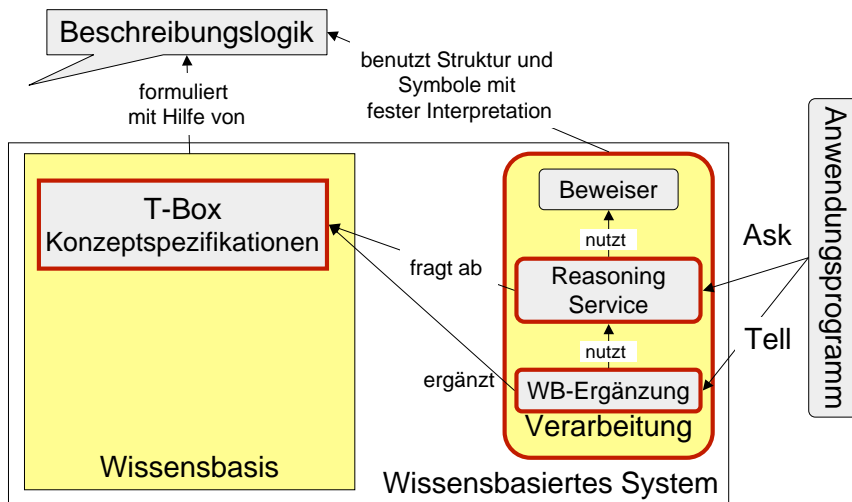
Definition

- Sei \mathcal{T} eine T-Box. Eine Interpretation \mathcal{S} **macht** genau dann \mathcal{T} **wahr**, wenn \mathcal{S} jedes Element von \mathcal{T} wahr macht ($\mathcal{S} \models \mathcal{T}$).
- \mathcal{T} ist genau dann **konsistent**, wenn es eine Interpretation $\mathcal{S} = \langle \mathcal{D}, I \rangle$ mit nicht leerem \mathcal{D} gibt, die \mathcal{T} wahr macht.
- Eine T-Box-Formel F **folgt** genau dann **aus** \mathcal{T} , wenn jede Interpretation \mathcal{S} , die \mathcal{T} wahr macht, auch F wahr macht ($\mathcal{T} \models F$).

Inferenzdienst

- Konsistenz einer T-Box
- Folgt F aus \mathcal{T} ?
- Sind zwei T-Boxen äquivalent?

T-Box-Aufbau



Inferenzdienste für einen Taxonomieaufbau

Einfügen von T-Box-Axiomen

- T-Box-Axiome spezifizieren eine nicht-strikte Ordnung (Subsumption, \sqsubseteq).
- \supseteq steht für den symmetrischen Anteil von \sqsubseteq
- Die Struktur der T-Box kann den Zugriff auf diese Ordnung unterstützen.
 - Beispiel: Graphenstruktur (gerichtet, azyklisch), wobei
 - jeder Knoten einer Menge äquivalenter Konzeptbezeichnungen entspricht und
 - Kanten zu den direkten Vorgängern und Nachfolgern in der Subsumptionsordnung bestehen.
- Beim Einfügen eines Konzeptes müssen
 - die **direkten Vorgänger und Nachfolger in der Subsumptionsordnung** gefunden werden und
 - der Graph angepasst werden.

Ergänzung der Wissensbasis: Taxonomien und Klassifikation

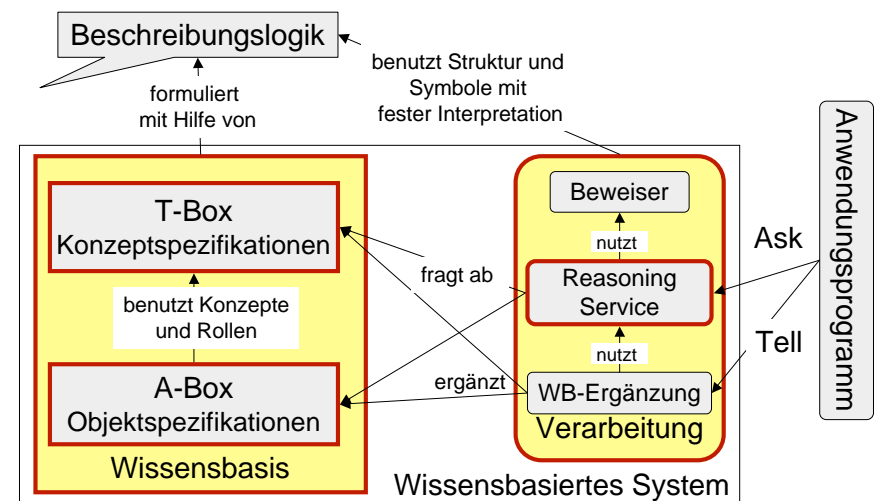
Taxonomie

- Subsumptionsstruktur (partielle Ordnung) von atomaren Konzepten, Gespeichert als zyklener Graph
- Verbindungen nur zu den direkten Nachbarn

Klassifikation

- Bestimmung der Position eines neuen Konzeptes
 - 'unterhalb' aller subsumierenden Knoten
 - 'oberhalb' aller subsumierten Knoten
- Rahmenannahme
 - Der Graph wird sukzessive aufgebaut, beginnend mit dem einzigen Knoten für \top
 - Die relative Position der Knoten wird nur durch das Einfügen **neuer** Konzepte verändert. (keine Zyklen)

A-Box-Reasoning-Services

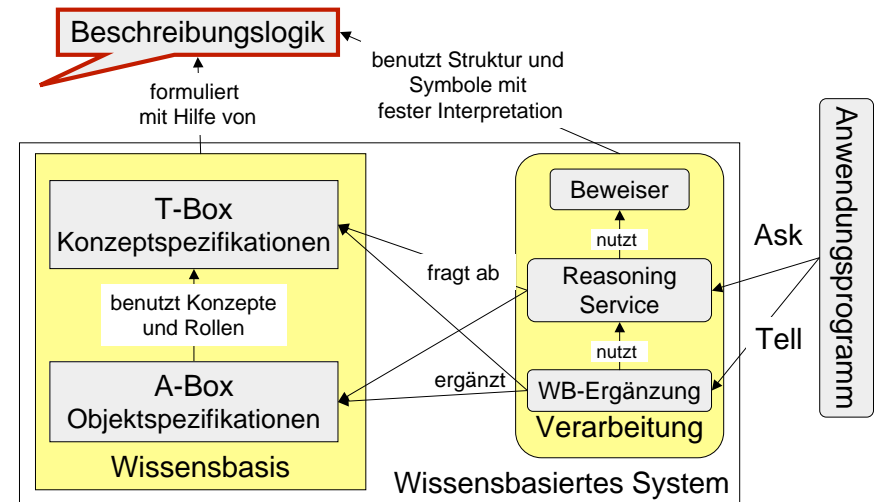


Auswertungsaufgaben für die Wissensbasis

A-Box (basierend auf einer T-Box)

- Erfüllbarkeit / Konsistenz (gibt es ein Modell)
- Klassifikation: Unter welche Konzepte fällt ein Objekt?
- Welche Objekte fallen alle unter ein Konzept?

Beschreibungslogiken für Wissensbasierte Systeme



Beschreibungslogiken: Varianten

Unterschiede zwischen verschiedenen Beschreibungslogiken

- Auswahl von Konzeptbildungsoperatoren
- Auswahl von Rollenbildungsoperatoren
- Konzeptbildungsoperatoren auf Basis von Konstanten

Auswirkungen

- Ausdrucksmächtigkeit und Verarbeitbarkeit
- Manchmal keine: Einschränkung von Formulierungsvarianten desselben Inhaltes
- Reine Konzeptsysteme vs. Konzeptsysteme + Weltausschnitt

Konzeptbildungsoperatoren (Concept constructors)

Standardnotation der Konstruktoren

- \top : das universelle Konzept
- \perp : das leere Konzept
- $C \sqcap D$: Durchschnitt, Bedingungskonjunktion
- $C \sqcup D$: Vereinigung, Bedingungsdisjunktion
- $\neg A$: Komplement, Negation nur atomarer Konzepte
- $\neg C$: Komplement, Negation nur beliebige Konzepte
- $\forall R.C$: Werterestriktion
- $\exists R.T$: eingeschränkte Existenzrestriktion
- $\exists R.D$: freie Existenzrestriktion
- $(\geq n R)$: Anzahlrestriktion mindestens
- $(\leq n R)$: Anzahlrestriktion höchstens

Interpretation der Konstruktoren

Interpretationen sind Paare $\mathcal{I} = \langle \mathcal{D}, I \rangle$

$$I(\top) = \mathcal{D}$$

$$I(\perp) = \emptyset$$

$$I(C \sqcap D) = I(C) \cap I(D)$$

$$I(C \sqcup D) = I(C) \cup I(D)$$

$$I(\neg C) = \mathcal{D} \setminus I(C)$$

$$I(\forall R.C) = \{a \in \mathcal{D} \mid \forall b [(a, b) \in I(R) \Rightarrow b \in I(C)]\}$$

$$I(\exists R.\top) = \{a \in \mathcal{D} \mid \exists b [(a, b) \in I(R)]\}$$

$$I(\exists R.D) = \{a \in \mathcal{D} \mid \exists b [(a, b) \in I(R) \wedge b \in I(D)]\}$$

$$I(\geq n R) = \{a \in \mathcal{D} \mid |\{b \in \mathcal{D} \mid (a, b) \in I(R)\}| \geq n\}$$

$$I(\leq n R) = \{a \in \mathcal{D} \mid |\{b \in \mathcal{D} \mid (a, b) \in I(R)\}| \leq n\}$$

Rollenkonstruktoren mit Interpretation

$$I(U) = \mathcal{D} \times \mathcal{D}$$

$$I(\text{Id}) = \{(a, a) \mid a \in \mathcal{D}\}$$

$$I(R \sqcap S) = I(R) \cap I(S)$$

$$I(R \sqcup S) = I(R) \cup I(S)$$

$$I(\neg R) = \mathcal{D} \times \mathcal{D} \setminus I(R)$$

$$I(R^-) = \{(a, b) \in \mathcal{D} \times \mathcal{D} \mid (b, a) \in I(R)\}$$

$$I(R \circ S) = \{(a, b) \in \mathcal{D} \times \mathcal{D} \mid \exists c [(a, c) \in I(R) \wedge (c, b) \in I(S)]\}$$

$$I(R^+) = \text{Transitive Hülle von } I(R)$$

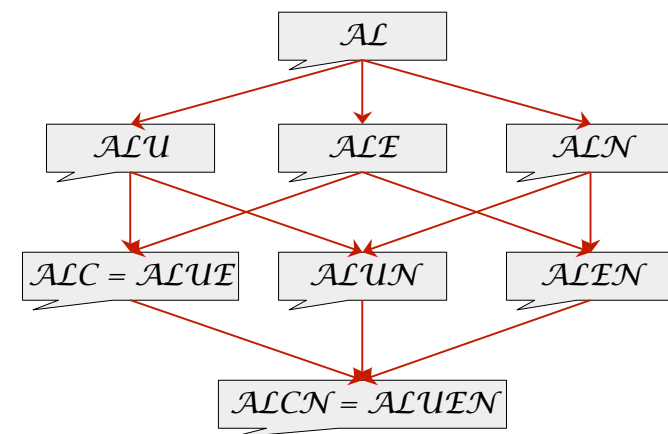
$$I(R^*) = \text{Reflexive, transitive Hülle von } I(R)$$

$$I(R|_C) = I(R) \cap \mathcal{D} \times I(C)$$

Die Basissprache \mathcal{AL} und Erweiterungen

	\mathcal{AL}	\mathcal{ALU}	\mathcal{ALF}	\mathcal{ALN}	\mathcal{ALC}
\top	+	+	+	+	+
\perp	+	+	+	+	+
$C \sqcap D$	+	+	+	+	+
$C \sqcup D$	-	+	-	-	-
$\neg A$	+	+	+	+	+
$\neg C$	-	-	-	-	+
$\forall R.C$	+	+	+	+	+
$\exists R.\top$	+	+	+	+	+
$\exists R.D$	-	-	+	-	-
$(\geq n R)$	-	-	-	+	-
$(\leq n R)$	-	-	-	+	-

\mathcal{AL} -Sprachfamilie (Ausdrucksmächtigkeit)



Einordnung bezüglich anderer Logiken

Beschreibungslogiken sind

- ausdruckschwächer als Prädikatenlogik

ALU und Erweiterungen sind

- ausdrucksreicher als Aussagenlogik

Entscheidbarkeit

- Prädikatenlogik ist semi-entscheidbar
- Aussagenlogik ist entscheidbar
- Bei Beschreibungslogiken gibt es entscheidbare und semi-entscheidbare Varianten
 - gesucht wird nach entscheidbaren Varianten mit geringer Komplexität

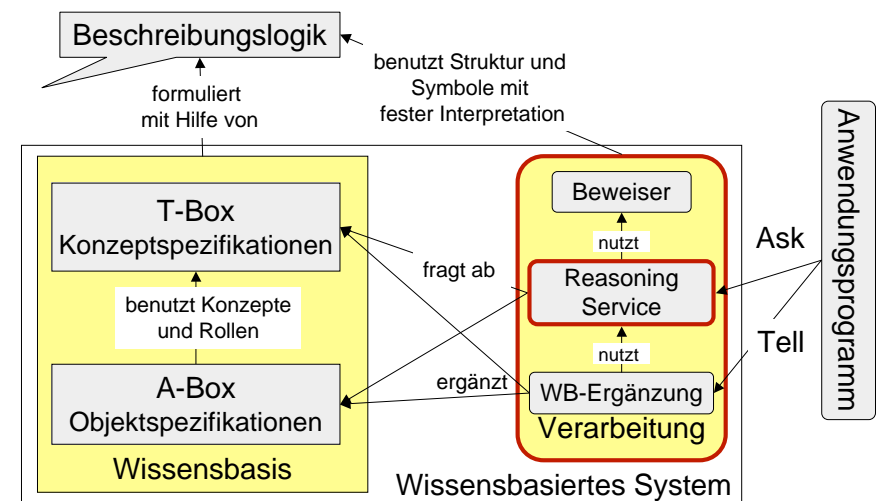
Hintergrundmaterialien

- Baader, Franz (2003). Appendix 1. Description logic terminology. In F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi & P. Patel-Schneider (eds.) *The Description Logic Handbook. Theory, Implementation and Application* (pp. 485–495). Cambridge UP: Cambridge, NY. (dlhb-appendix)
- Nardi, Daniele & Ronald J. Brachman (2003). An introduction to description logics. In F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi & P. Patel-Schneider (eds.) *The Description Logic Handbook. Theory, Implementation and Application* (pp. 1–40). Cambridge UP: Cambridge, NY. (dlhb-01)

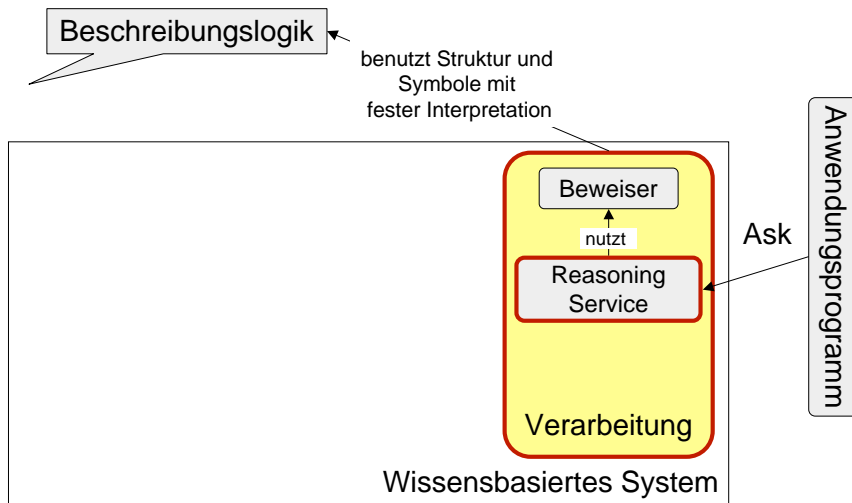
Gemeinsam zu lesen und diskutieren

- Baader, Franz & Werner Nutt (2003). Basic description logics. F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi & P. Patel-Schneider (eds.) *The Description Logic Handbook. Theory, Implementation and Application* (pp. 43–95). Cambridge UP: Cambridge, NY. (dlhb-02)

Beschreibungslogik-Reasoning-Services



Wissensbasiertes System ohne Wissensbasis



Konzept-Inferenzen bzgl. einer T-Box

Definition

- Sei \mathcal{T} eine T-Box und C ein Konzept.
- C ist genau dann **konsistent**, wenn es eine Interpretation gibt, die C auf eine nicht leere Menge abbildet.
- C ist genau dann **konsistent bzgl. \mathcal{T}** , wenn es eine Interpretation gibt, die \mathcal{T} wahr macht und C auf eine nicht leere Menge abbildet.

Inferenzdienst

- Konsistenz eines (komplexen) Konzeptes
 - Konsistenz eines Konzeptes bzgl. einer T-Box
- => „klassische Inferenzdienste“

Reasoning Services: Klassisch

„Klassische“ Inferenzdienste für DL-Logiken:

- Subsumption von Konzepten
- Äquivalenz von Konzepten (oder ganzen T-Boxen)
- Konsistenz von Konzepten / von T-Boxen / von A-Boxen
- Anfragen
 - Ist Objekt x Instanz eines Konzeptes C
 - Stehen zwei Objekte in einer Rollenbeziehung
 - Komplexere Anfragen, z.B. konjunktive Anfragen

Reasoning Services: Nicht-klassisch

„Nicht-klassische“ Inferenzdienste

- Berechnung des **gemeinsamen spezifischsten Oberkonzeptes**
- Berechnung des **spezifischsten Konzeptes** (für ein Objekt)
- **Matching**: Gibt es Substitution s für Konzeptvariablen im Pattern D , so dass $C \equiv s(D)$?
- **Rewriting**: Umschreiben einer Konzeptbeschreibung (ohne definierte Konzepte) zu einer „besseren“ mit definierten Konzepten
- **Approximation**: Approximieren (i.e. Aufrechterhalten möglichst vieler Subsumptionsbeziehungen) eines Konzeptes in einer Beschreibungslogik $DL1$ durch ein Konzept in einer schwächeren Beschreibungslogik $DL2$

Projekt-Aufgabe: Reasoning Services

Welche Reasoning-Services sollen angeboten werden?

- Wie können die Services aufeinander abgebildet werden?
- oder: Wie können die Services auf durch den Beweiser zu lösende Aufgaben abgebildet werden?

Schnittstellen

- Wie ist das Interface zum Angebot der Service zu gestalten?
 - Eingabe-Maske
 - API

Anwendungsperspektive: Ontologien für das Semantic Web

OWL, RDF(S)

- Horrocks, Ian, Peter F. Patel-Schneider & Frank van Harmelen (2003). From SHIQ and RDF to OWL: The making of a web ontology language. *Journal of Web Semantics* 1. 7–26.

Ontologien

- Rector, A. & I. Horrocks (1997). Experience building a large, re-usable medical ontology using a description logic with transitivity and concept inclusions. In Proceedings of the Workshop on Ontological Engineering, AAAI Spring Symposium (AAAI'97). AAAI Press: Menlo Park, California.

Semantic Web: Ausgangspunkt

Lit: T. Berners-Lee, J. Hendler, and O. Lassila: The semantic web. Scientific American, 2001. (<http://www.scientificamerican.com/>)

Syntaktisches WWW

- Sammelsurium an verlinkten Ressourcen
- Semantische Information bisher nur von Menschen erfassbar
 - Bsp.: Link mit Aufschrift „curriculum vitae“
 - Bsp.: Web-Page einer Universität: Nicht nur Seite mit Schlüsselwörtern wie „Geschäftszeit, Universität, Studienplan, Student“; sondern: Das Geschäftszimmer hat Montags, Donnerstags von 16-19 Uhr geöffnet und Frau so-und-so ist dort zu erreichen...
- Maschinen können Information nicht interpretieren / verstehen, daher auch komplexere Aufgaben, die das Erfassen der Bedeutung erfordern, nicht ausführen

Semantic Web: Vision

Erweiterung des WWW um semantische Information

- Maschinen werden zumindest in naher Zukunft nicht wirklich natürliche Sprache verstehen lernen können, aber ...
- ... sie können uns einen Teil der Verstehensleistung abnehmen, wenn wir ihnen das nötige Wissen durch semantische Annotation zur Verfügung stellen.
- Semantic Web = WWW (von heute) + semantische Annotation + KI

Bsp.-Anwendung

- Beauftrage einen Software-Agenten, möglichst günstig in einem Pizza-Onlineshop eine Pizza zu kaufen, die einen knusprigen Boden hat und als Belag entweder eine beliebige Käsesorte und Tomaten oder Parmesan und Anchovis, aber auf keinen Fall Salami.

Semantic Web: Beispiel

- Finde einen geeigneten Online-Shop und vollführe die Transaktion durch Angabe der vom Onlineshop geforderten Informationen
- Was muss der Agent „wissen“, „kennen“, „verstehen“?
 - Finden eines geeigneten Onlineshops
 - Was ist eine Pizza? Was ist ein Pizzaboden? Mozzarella ist eine Käsesorte!? Pizza mit Salami bedeutet Pizza mit Fleisch ...
 - Was heißt möglichst günstig?
- Wie ist die Gesamtarchitektur zu gestalten?
- Wie erreicht man das nötige „Verstehen von Inhalten“ im Netz?
 - Nutze Ontologien (im informatischen Sinne), um die Bedeutung der semantischen Annotationen zu spezifizieren

Ontologien: Erläuterung

Informatischer Ontologiebegriff (sprachabhängig)

- Formale Beschreibung eines Anwendungsbereichs zum Zwecke des Gebrauchs durch verschiedene Anwendungen - verfasst in einer Sprache, die zum (automatischen) Schließen benutzt werden kann

Komponenten von Ontologien

- Terme zur Bezeichnung der relevanten Objekte, Konzepte in einer Domäne und ihrer Beziehungen
- Beschreibung der Bedeutung der Terme in der einen oder anderen Form
- Eventuell Mittel zur Beschreibung von Hintergrund-/Weltwissen

Ontologien: OWL

Webontologiesprache: OWL

- Entstanden aus DAML-OIL (das Produkt aus dem in Europa entwickelten OIL und in Amerika entwickelten DAML-ONT ist)
- Als Standard von W3C vorgegeben
- Logikbasiert
- OWL-Lite \subset OWL-DL \subset OWL-Full
- Syntax im RDF- (Resource Description Framework) bzw. XML-Format
 - Kann als Ontologiebeschreibungssprache aufgefasst werden
 - Graphenbasiert; Format: (Subjekt Prädikat Objekt)
- Begriffe (OWL \leftrightarrow DL)
 - Class \leftrightarrow concept
 - Property \leftrightarrow role
 - Individual \leftrightarrow individual

OWL: Beispiel 1

1. Beispiel aus pizza.owl

In XML-Syntax:

```
<owl:Class rdf:ID="CheesyVegetableTopping">
  <rdfs:subClassOf rdf:resource="#VegetableTopping"/>
</rdfs:subClassOf>
  <owl:Class rdf:about="#CheeseTopping"/>
</rdfs:subClassOf>
</owl:Class>
```

In DL-Syntax:

```
CheesyVegetableTopping  $\sqsubseteq$  VegetableTopping,
CheesyVegetableTopping  $\sqsubseteq$  CheeseTopping
```

OWL: Beispiel 2

2. Beispiel aus pizza.owl (XML-Syntax)

```
<owl:Class rdf:ID="NonVegetarianPizza">
  <owl:disjointWith> <owl:Class rdf:ID="VegetarianPizza"/></owl:disjointWith>
  <owl:equivalentClass>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Class>
          <owl:complementOf>
            <owl:Class rdf:about="#VegetarianPizza"/>
          </owl:complementOf>
        </owl:Class>
        <owl:Class rdf:about="#Pizza"/>
      </owl:intersectionOf>
    </owl:Class>
  </owl:equivalentClass>
</owl:Class>
```

OWL: Beispiel 2

2. Beispiel aus pizza.owl (DL-Syntax)

$\text{NonVegetarianPizza} \sqsubseteq \neg \text{VegetarianPizza}$,
 $\text{NonVegetarianPizza} \equiv \neg \text{VegetarianPizza} \sqcap \text{Pizza}$

OWL: Beispiel 3

3. Beispiel aus pizza.owl (XML-Syntax)

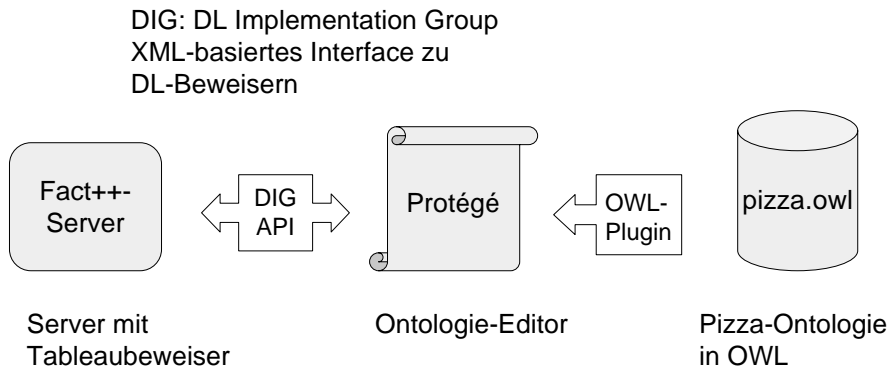
```
<owl:Class rdf:ID="SpicyPizza">
  <owl:equivalentClass>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#Pizza"/>
        <owl:Restriction>
          <owl:someValuesFrom rdf:resource="#SpicyTopping"/>
          <owl:onProperty>
            <owl:ObjectProperty rdf:about="#hasTopping"/>
          </owl:onProperty>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </owl:equivalentClass>
</owl:Class>
```

OWL: Beispiel 3

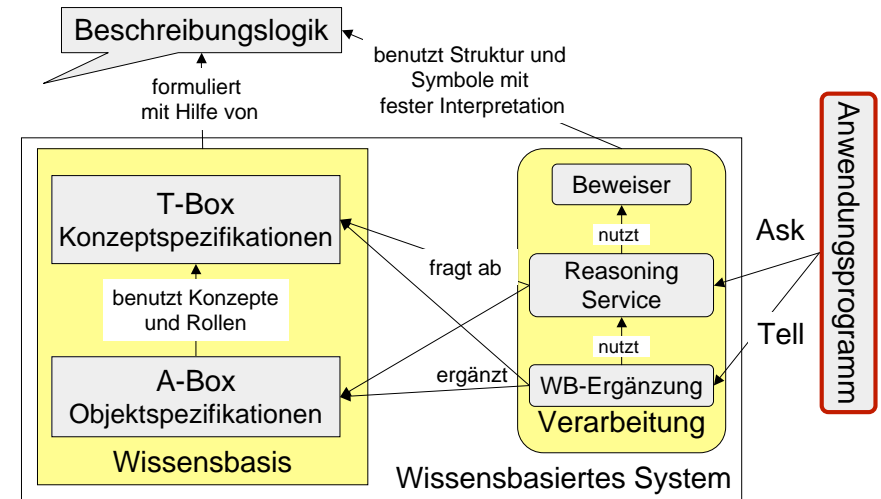
3. Beispiel aus pizza.owl (DL-Syntax)

$\text{SpicyPizza} \equiv \text{Pizza} \sqcap \exists \text{hasTopping} \text{SpicyTopping}$

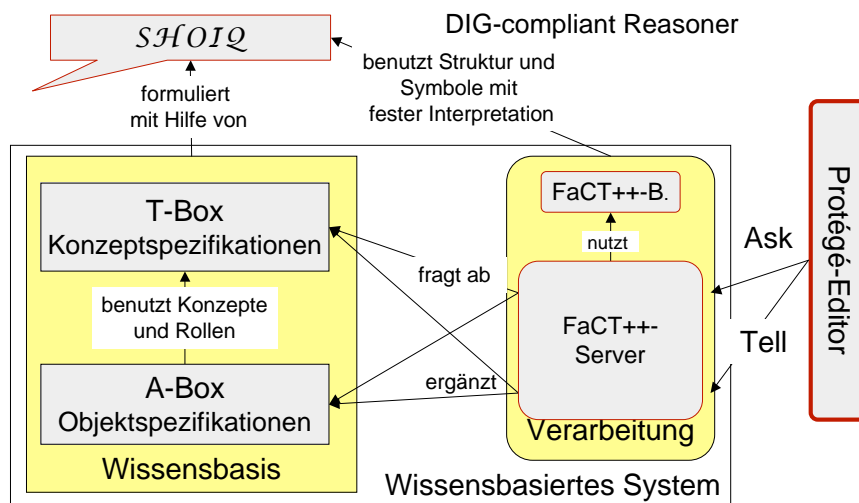
Demo: Pizza-Ontologie



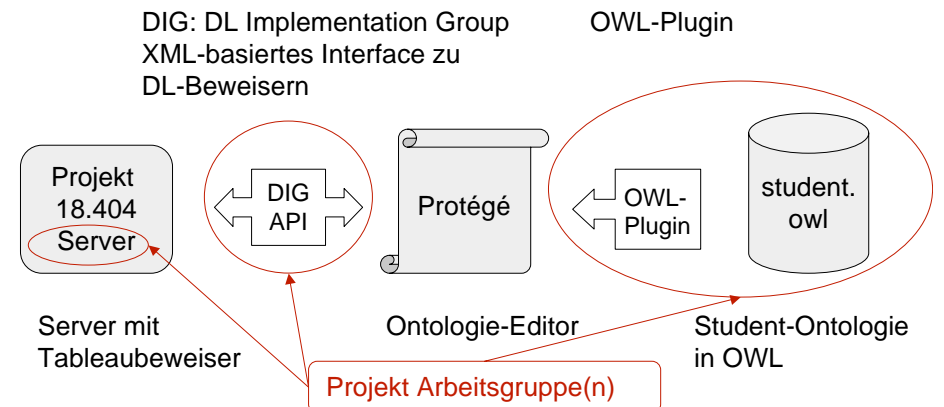
Zur Erinnerung: Beschreibungslogiken für Wissensbasierte Systeme



Demo: Protégé + FaCT++



Perspektive



Protégé

Protégés Architektur ist framebasiert

- Nutzung des OWL-Plugins, um OWL-Ontologien zu editieren
- Darstellung der Klassenhierarchie, der Eigenschaftshierarchie und der Individuen

Das in der Ontologie enthaltene Wissen ist in verschiedenen Bedingungen enthalten

- notwendige und hinreichende Bedingungen (für Klassen)
- Angabe der disjunkten Klassen (disjoint-Panel)
- Eigenschaften von Rollen
 - Domäne und Wertebereich
 - Transitivität, Symmetrie etc.

Betrachtete Reasoning Services

Konzeptkonsistenz

Taxonomiebeziehungen (implizite Subsumptionsbeziehungen)

Berechnung von Konzepten, unter denen ein Individuum fällt



Beispiel Konzeptinkonsistenz

Bsp 1 : CheeseVegetableTopping ist inkonsistent

- Notwendige Bedingungen:
 - $\text{CheeseVegetableTopping} \sqsubseteq \text{CheeseTopping}$
 - $\text{CheeseVegetableTopping} \sqsubseteq \text{VegetableTopping}$
- Disjunkte Klassen
 - $\text{CheeseTopping} \sqsubseteq \neg \text{VegetableTopping}$

Bsp 2: IceCream ist inkonsistent

- hasTopping hat als Domäne das Konzept Pizza, welches disjunkt zu IceCream ist.

Beispiel Taxonomiebeziehungen

Bsp 1 : American \sqsubseteq CheesePizza

- Hinreichende (und notwendige) Bedingungen für CheesePizza werden durch American Pizzas erfüllt (da diese MozzarellaTopping haben, was eine CheeseTopping ist)
- Reasoner-Log: „Added Superclass CheesePizza“

Bsp 2: FruitTopping \sqsubseteq VegetarianTopping

- Hinreichende (und notwendige) Bedingungen für VegetarianTopping werden durch Obstbelege erfüllt. Neuer direkter Vorgänger ist nicht mehr PizzaTopping, sondern VegetarianTopping.
- Reasoner-Log: „Moved from PizzaTopping to VegetarianTopping“

Reasoning Services beruhen auf Beweisern

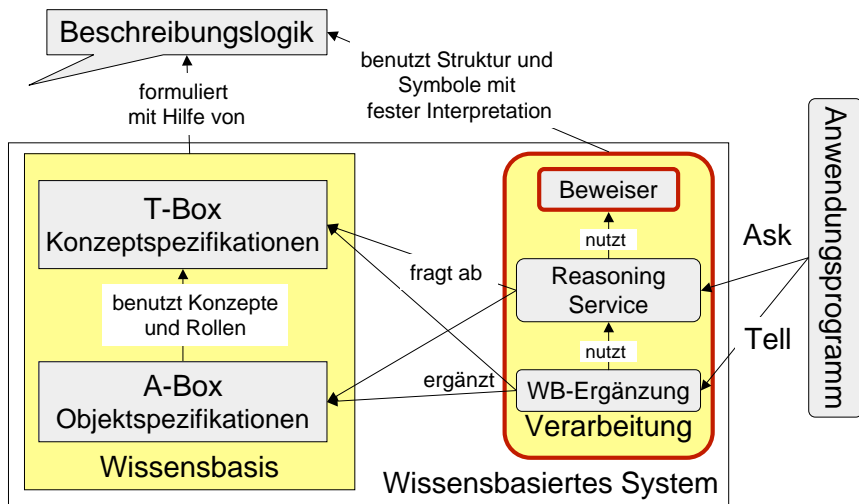
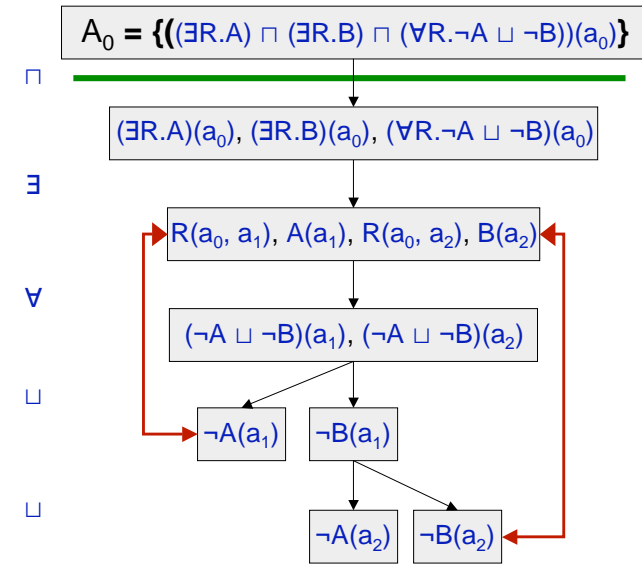
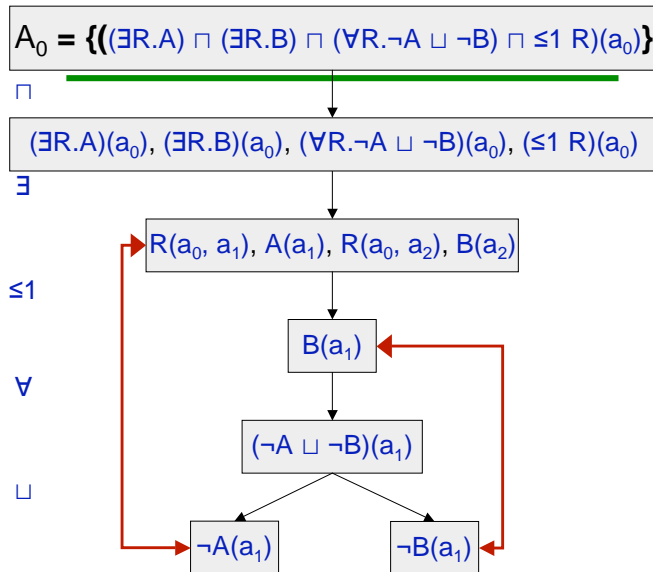


Tableau-Verfahren für Beschreibungslogiken

Tableau-Verfahren

- stellen (Un)Erfüllbarkeit fest
 - hier: (un)Erfüllbarkeit von Beschreibungen
- konstruieren Modelle
 - hier: konsistente A-Box
- bilden (in der AL) disjunktive Normalformen
 - hier: alternative Modelle → Model-Checking



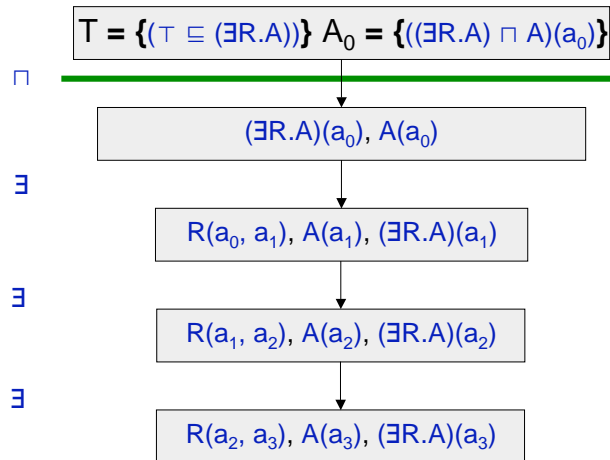


Tableau-Verfahren

- Fitting, Melvin (1999). Introduction. In M. D'Agostino, D.M. Gabbay, R. Hähnle & J. Posegga (eds.) *Handbook of Tableau Methods* (pp. 1–43). Kluwer Academic Publishers: Dordrecht.
- Sattler, Ulrike (1996). A Concept Language Extended with Different Kinds of Transitive Roles. In Görz, G. and Hölldobler, S. (eds.) *20. Deutsche Jahrestagung für Künstliche Intelligenz* (pp. 333–345). Springer Verlag.
- Baader, F. & U. Sattler (2001). An overview of tableau algorithms for description logics. *Studia Logica* 69. 5–40.
- Buchheit, Martin, Francesco M. Donini & Andrea Schaerf (1993). Decidable reasoning in terminological knowledge representation systems. *Journal of Artificial Intelligence Research* 1. 109–138.

Eigenschaften von Modellen

Eigenschaften (vielen) Beschreibungslogiken

- endliche-Modell-Eigenschaft (*finite-model property*)
 - Eine Formel/ein Konzept ist genau dann erfüllbar, wenn es ein endliches Modell hat.
- Baum-Modell-Eigenschaft (*tree-model property*)
 - Eine Formel/ein Konzept ist genau dann erfüllbar, wenn es ein Modell mit Baum-Struktur hat.
- Gegenbeispiel aus der Prädikatenlogik:

$$R(a, b) \wedge \forall x [\neg R(x, x)] \wedge \forall x, y [R(x, y) \Rightarrow \exists z [R(x, z) \wedge R(z, y)]]$$
 - hat nur unendliche Modelle, die keine Baumform haben

Projekt-Aufgabe: DL-Tableau-Beweiser

Umstellung der externen und internen Darstellung Sprache

- Detailliertere Signatur

Anpassung der Tableau-Regeln

- an den neuen Sprachumfang

Blockierung der Expansion

- wenn kein Abschluss ermöglicht werden kann
- (Schleifenerkennung)

Testen

Stufenplan

- Reihung der aufzunehmenden Konstruktoren

Optimierung: Einfache Beispiele

Tableaubeweiser

- die korrekt sind, können trotzdem sehr ineffizient sein.

Verzweigung bei Disjunktion

- Zweig $\theta, A \vee B$ führt zu den Zweigen θ, A und θ, B .
- In beiden Zweigen muss u.U. der Fall, dass A und B beide wahr sind, weiter untersucht werden.
- Abhilfe: Semantic branching: Zweig $\theta, A \vee B$ führt zu den Zweigen θ, A und $\theta, \neg A, B$.

Wahl einer Disjunktion

- Zweig $\neg A, A \vee B, C \vee \neg B$ führt zu den Zweigen $\neg A, A \vee B, C$ und $\neg A, A \vee B, \neg B$ und weiter zu $\neg A, A, C$ \otimes , $\neg A, B, C$ und $\neg A, A, \neg B$ \otimes , $\neg A, B, \neg B$ \otimes oder
- Zweig $\neg A, A \vee B, C \vee \neg B$ führt zu den Zweigen $\neg A, A, C \vee \neg B$ \otimes und $\neg A, B, C \vee \neg B$ und weiter zu $\neg A, B, \neg B$ \otimes und $\neg A, B, C$

Optimierung: Komplexere Fälle

Tableaubeweiser

- die korrekt sind, können trotzdem sehr ineffizient sein.

Dependency-directed Backtracking

- Ist ein Zweig abgeschlossen, so kann es sein, dass sich viele alternative Zweige auf genau dieselbe Weise abschließen lassen, da die Disjunktionen an den Verzweigungspunkten irrelevant waren.

Kodierung von Teilformeln

- Prüfung auf atomaren Abschluss kann sehr ineffizient sein, weil jede Formel bis zu den atomaren Formeln zerlegt werden muss.
- Prüfung auf beliebigen Abschluss kann sehr ineffizient sein, da in jedem Schritt alle Formeln geprüft werden müssen und oberflächliche Kodierungsunterschiede die Erkennung des Abschlusses erschweren.
- Lösung: geeignete Kodierung der Formeln

Verarbeitung: Systeme

Grundideen

- Baader, Franz, Bernhard Hollunder, Bernhard Nebel, Hans-Jürgen Profitlich & Enrico Franconi (1992). An empirical analysis of optimization techniques for terminological representation systems or: Making KRIS get a move on. In B. Nebel, W. Swartout & C. Rich (eds.) *Principles of Knowledge Representation and Reasoning: Proceedings of the 3rd International Conference* (pp. 270–281). Morgan Kaufmann: San Mateo.

Verfeinerungen

- Horrocks, Ian R. (1998). Using an Expressive Description Logic: FaCT or Fiction?. In Anthony G. Cohn, Lenhart Schubert & Stuart C. Shapiro (eds.) *KR'98: Principles of Knowledge Representation and Reasoning* (pp. 636–645). Morgan Kaufmann: San Francisco, California. citeseer.ist.psu.edu/horrocks98using.html
- Horrocks, I. (1998). The FaCT system. In H. de Swart (ed.) *Automated Reasoning with Analytic Tableaux and Related Methods: International Conference Tableaux'98* (pp. 307–312). Springer-Verlag.
- Horrocks, Ian & Peter Patel-Schneider (1999). *Optimizing description logic subsumption*. *Journal of Logic and Computation* 9. 267–293.

Organisatorisches

Projekte

- im Umfang von 6 SWS dienen der **Bearbeitung größerer theoretischer, konstruktiver oder experimenteller Aufgaben** ... Wegen des besonderen **Arbeitsaufwands** wird den Studierenden empfohlen, nicht mehr als ein Projekt zur gleichen Zeit zu belegen.
- Projekte werden **in Gruppen durchgeführt** und ermöglichen das Erlernen von Gruppenarbeit. Die gesamte Projektgruppe arbeitet auf **ein gemeinsames Ergebnis** hin. Dabei kann eine Projektgruppe in **Kleingruppen von 2-3 Studierenden** gegliedert werden.

Eine Teilnahmebestätigung

- erhält, wer **kontinuierlich an den Projektterminen teilnimmt** und sich in die verschiedenen **Aufgaben der Gruppenarbeit** einbringt. Zur Projektarbeit gehört auch die Dokumentation (Protokolle oder Programm-Dokumentation) sowie die Präsentation der Ergebnisse (Vorführung oder Programm-Demonstration) an den Projektterminen.

Einen Projektschein

- erhält, wer darüber hinaus einen Projektbericht erstellt, der die Planung des Vorgehens, dessen Verlauf und die Durchführung sowie die erreichten Ergebnisse zusammenfaßt.

Soll der Projektbericht als **Baccalaureatsarbeit**

- **anerkannt werden**, so muss dies vor Beginn des Projektes beim Prüfungsausschuss beantragt werden.

Termine und TeilnehmerInnen

**Terminprobleme?
Lösungen?**

Grober Ablauf

25.10.–15.11.	Einarbeitung: Grundlagen, Arbeitspaketdefinition
17.11.–6.12.	Arbeit in Kleingruppen
8.12.–13.12.	Zusammenführung der Resultate
15.12.–22.12.	Optimierungen: Kennenlernen, Arbeitspaketdefinition
10.1.–26.1.	Arbeit in Kleingruppen
31.1.–2.2.	Zusammenführung der Resultate
7.2.–9.2.	Ergebnispräsentation und Abschluss

Einarbeitung: Grundlagen, Arbeitspaketdefinition

- diese Woche: Freitag: Erstellung einer Ontologie**
- 2. Woche: Mittwoch: Tableau-Verfahren (Grundlagen)**
- 2. Woche: Freitag: Kennenlernen des AL-Beweisers**
- 3. Woche: Mittwoch: Beschreibungslogiken (Grundlagen)**
- 3. Woche: Freitag: Beschreibungslogische Tableau-Beweiser**
- 4. Woche: Mittwoch: Klärung der Arbeitspakete 1. Phase**
- 4. Woche: Freitag: Beginn der Arbeit in Kleingruppen**