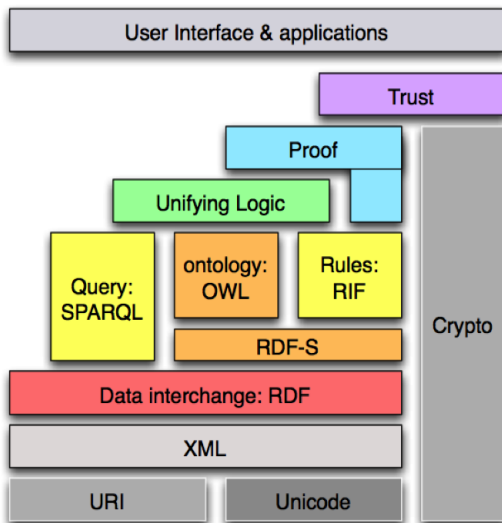




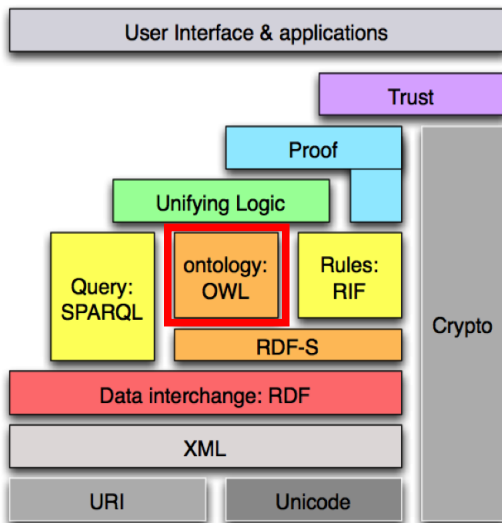
Organisatorisches: Inhalt

Einleitung und XML	17. Okt	SPARQL Syntax & Intuition	12. Dez
Einführung in RDF	20. Okt	Übung 4	15. Dez
RDF Schema	24. Okt	SPARQL Semantik	19. Dez
fällt aus	27. Okt	SPARQL 1.1	22. Dez
Logik – Grundlagen	31. Okt	Übung 5	9. Jan
Übung 1	3. Nov	SPARQL Entailment	12. Jan
Semantik von RDF(S)	7. Nov	SPARQL Implementierung	16. Jan
RDF(S) & Datalog Regeln	10. Nov	Abfragen & RIF	19. Jan
OWL Syntax & Intuition	14. Nov	Übung 6	23. Jan
Übung 2	17. Nov	Ontology Editing	26. Jan
OWL & BLs	21. Nov	Ontology Engineering	30. Jan
OWL 2	24. Nov	Linked Data	2. Feb
Tableau	28. Nov	Übung 7	6. Feb
Übung 3	1. Dez	SemWeb Anwendungen	9. Feb
Blocking & Unravelling	5. Dez	Wiederholung	13. Feb
Hypertableau	8. Dez	Übung 8	16. Feb

OWL 2



OWL 2



Agenda

- ▶ Generelle Idee des Tableaukalküls
- ▶ Propositionales Beispiel
- ▶ Transformation in Negationsnormalform
- ▶ Erfüllbarkeit von \mathcal{ALC} Konzepten
- ▶ Korrektheit und Terminierung
- ▶ Erfüllbarkeit von \mathcal{ALC} Wissensbasen
- ▶ Zusammenfassung

Agenda

- ▶ Generelle Idee des Tableauealküls
- ▶ Propositionales Beispiel
- ▶ Transformation in Negationsnormalform
- ▶ Erfüllbarkeit von \mathcal{ALC} Konzepten
- ▶ Korrektheit und Terminierung
- ▶ Erfüllbarkeit von \mathcal{ALC} Wissensbasen
- ▶ Zusammenfassung

Automatisierbarkeit

- ▶ Bisher Erfüllbarkeit von OWL Axiomen ad-hoc betrachtet
- ▶ Ein Konzept ist erfüllbar, wenn es ein Modell hat
 \rightsquigarrow Idee: Konstruktives Entscheidungsverfahren um Modelle zu bauen
- ▶ Analog zu Wahrheitstabellen in der Aussagenlogik

Automatisierbarkeit

- ▶ Bisher Erfüllbarkeit von OWL Axiomen ad-hoc betrachtet
- ▶ Ein Konzept ist erfüllbar, wenn es ein Modell hat
 \rightsquigarrow Idee: Konstruktives Entscheidungsverfahren um Modelle zu bauen
- ▶ Analog zu Wahrheitstabellen in der Aussagenlogik

$$(p \vee q) \rightarrow (\neg p \vee \neg q)$$

Automatisierbarkeit

- ▶ Bisher Erfüllbarkeit von OWL Axiomen ad-hoc betrachtet
- ▶ Ein Konzept ist erfüllbar, wenn es ein Modell hat
 \rightsquigarrow Idee: Konstruktives Entscheidungsverfahren um Modelle zu bauen
- ▶ Analog zu Wahrheitstabellen in der Aussagenlogik

$$(p \vee q) \rightarrow (\neg p \vee \neg q)$$

Negation vor komplexen Ausdrücken und nicht-atomare Operatoren schwierig, daher umformulieren:

Automatisierbarkeit

- ▶ Bisher Erfüllbarkeit von OWL Axiomen ad-hoc betrachtet
- ▶ Ein Konzept ist erfüllbar, wenn es ein Modell hat
 \rightsquigarrow Idee: Konstruktives Entscheidungsverfahren um Modelle zu bauen
- ▶ Analog zu Wahrheitstabellen in der Aussagenlogik

$$(p \vee q) \rightarrow (\neg p \vee \neg q)$$

Negation vor komplexen Ausdrücken und nicht-atomare Operatoren schwierig, daher umformulieren:

$$\neg(p \vee q) \vee (\neg p \vee \neg q)$$

Automatisierbarkeit

- ▶ Bisher Erfüllbarkeit von OWL Axiomen ad-hoc betrachtet
- ▶ Ein Konzept ist erfüllbar, wenn es ein Modell hat
 \rightsquigarrow Idee: Konstruktives Entscheidungsverfahren um Modelle zu bauen
- ▶ Analog zu Wahrheitstabellen in der Aussagenlogik

$$(p \vee q) \rightarrow (\neg p \vee \neg q)$$

Negation vor komplexen Ausdrücken und nicht-atomare Operatoren schwierig, daher umformulieren:

$$\begin{aligned} & \neg(p \vee q) \vee (\neg p \vee \neg q) \\ & (\neg p \wedge \neg q) \vee (\neg p \vee \neg q) \end{aligned}$$

Automatisierbarkeit

- ▶ Bisher Erfüllbarkeit von OWL Axiomen ad-hoc betrachtet
- ▶ Ein Konzept ist erfüllbar, wenn es ein Modell hat
 \rightsquigarrow Idee: Konstruktives Entscheidungsverfahren um Modelle zu bauen
- ▶ Analog zu Wahrheitstabellen in der Aussagenlogik

$$(p \vee q) \rightarrow (\neg p \vee \neg q)$$

Negation vor komplexen Ausdrücken und nicht-atomare Operatoren schwierig, daher umformulieren:

$$\begin{aligned} & \neg(p \vee q) \vee (\neg p \vee \neg q) \\ & (\neg p \wedge \neg q) \vee (\neg p \vee \neg q) \\ & (\neg p \wedge \neg q) \vee \neg p \vee \neg q \end{aligned}$$

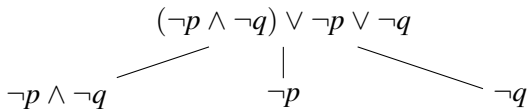
Agenda

- ▶ Generelle Idee des Tableaukalküls
- ▶ Propositionales Beispiel
- ▶ Transformation in Negationsnormalform
- ▶ Erfüllbarkeit von \mathcal{ALC} Konzepten
- ▶ Korrektheit und Terminierung
- ▶ Erfüllbarkeit von \mathcal{ALC} Wissensbasen
- ▶ Zusammenfassung

Einfaches Tableau

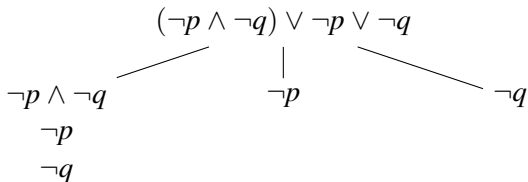
$$(\neg p \wedge \neg q) \vee \neg p \vee \neg q$$

Einfaches Tableau



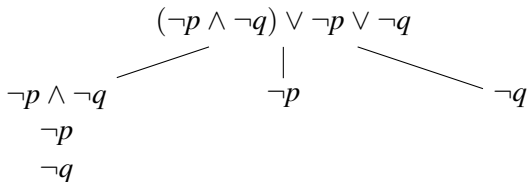
- ▶ Disjunktionen führen zu *Zweigen* (branch) im Tableau
- ▶ Tableau: Endliche Menge von Tableauxzweigen

Einfaches Tableau



- ▶ Disjunktionen führen zu *Zweigen* (branch) im Tableau
- ▶ Tableau: Endliche Menge von Tableauxzweigen

Einfaches Tableau



- ▶ Disjunktionen führen zu *Zweigen* (branch) im Tableau
- ▶ Tableau: Endliche Menge von Tableauxzweigen
- ▶ Vergleich Wahrheitstabelle

$I(p)$	$I(q)$	$I(\neg p)$	$I(\neg q)$	$I(p \vee q)$	$I(\neg p \vee \neg q)$	$I((p \vee q) \rightarrow (\neg p \vee \neg q))$
t	t	f	f	t	f	f
t	f	f	t	t	t	t
f	t	t	f	t	t	t
f	f	t	t	f	t	t

Einfaches Tableau mit Widerspruch

$$(\neg p \vee q) \wedge p \wedge \neg q$$

Einfaches Tableau mit Widerspruch

$$(\neg p \vee q) \wedge p \wedge \neg q$$

$$\neg p \vee q$$

$$p$$

$$\neg q$$

Einfaches Tableau mit Widerspruch

$$(\neg p \vee q) \wedge p \wedge \neg q$$

$$\neg p \vee q$$

$$p$$

$$\neg q$$

$$\neg p$$

$$q$$

Einfaches Tableau mit Widerspruch

$$(\neg p \vee q) \wedge p \wedge \neg q$$

$$\neg p \vee q$$

$$p$$

$$\neg q$$

$$\neg p$$

$$q$$

Einfaches Tableau mit Widerspruch

$$(\neg p \vee q) \wedge p \wedge \neg q$$

$$\neg p \vee q$$

$$p$$

$$\neg q$$

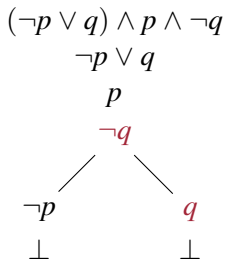
$$\neg p$$

$$q$$

$$\perp$$

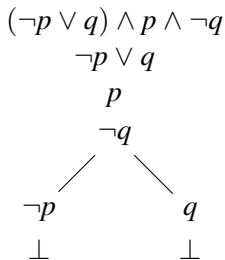
- Befindet sich auf einem Zweig ein atomarer Widerspruch (clash), ist dieser *abgeschlossen*

Einfaches Tableau mit Widerspruch



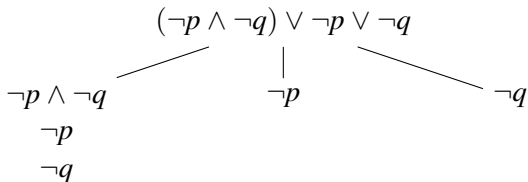
- ▶ Befindet sich auf einem Zweig ein atomarer Widerspruch (clash), ist dieser *abgeschlossen*
- ▶ Ein Tableau ist *abgeschlossen*, wenn alle Zweige abgeschlossen sind

Einfaches Tableau mit Widerspruch



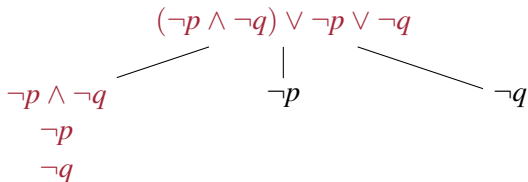
- ▶ Befindet sich auf einem Zweig ein atomarer Widerspruch (clash), ist dieser *abgeschlossen*
- ▶ Ein Tableau ist *abgeschlossen*, wenn alle Zweige abgeschlossen sind
- ▶ Ein komplettes Tableau ohne offenen Zweig zeigt die Unerfüllbarkeit der Formel

Modellkonstruktion vom Tableau



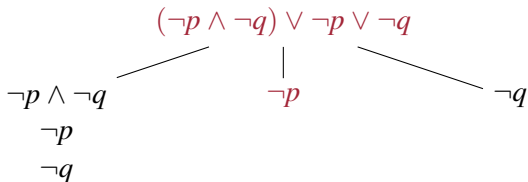
- ▶ Ausgehend von einem offenen Zweig können wir ein Modell bauen

Modellkonstruktion vom Tableau



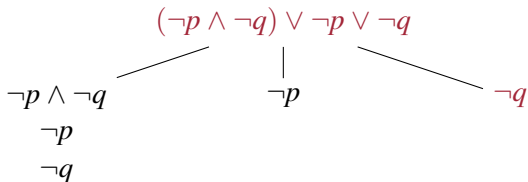
- ▶ Ausgehend von einem offenen Zweig können wir ein Modell bauen
- ▶ Setze $I(p)=\text{false}$ und $I(q)=\text{false}$

Modellkonstruktion vom Tableau



- ▶ Ausgehend von einem offenen Zweig können wir ein Modell bauen
- ▶ Setze $I(p)=\text{false}$ und $I(q)=\text{false}$
- ▶ Setze $I(p)=\text{false}$ ($I(q)$ is irrelevant, da nicht im Tableau, default Belegung false)

Modellkonstruktion vom Tableau



- ▶ Ausgehend von einem offenen Zweig können wir ein Modell bauen
- ▶ Setze $I(p)=\text{false}$ und $I(q)=\text{false}$
- ▶ Setze $I(p)=\text{false}$ ($I(q)$ is irrelevant, da nicht im Tableau, default Belegung false)
- ▶ Setze $I(q)=\text{false}$ ($I(p)$ is irrelevant, da nicht im Tableau, default Belegung false)

Propositionales Tableau

- ▶ Es werden nicht immer exponentiel viele Kombinationen gebildet, wie in der Wahrheitstabelle
- ▶ Zweige können nacheinander gebaut werden \rightsquigarrow nur polynomieller Platz gebraucht
- ▶ Für das Erfüllbarkeitstesten können wir aufhören, sobald ein kompletter offener Zweig gefunden wurde

Konstruktion von je einem Zweig

$$(\neg p \vee q) \wedge p \wedge q$$

Konstruktion von je einem Zweig

$$\begin{array}{c} (\neg p \vee q) \wedge p \wedge q \\ \neg p^{1a} \vee q^{1b} \\ p \\ q \end{array}$$

- ▶ Bei einer Disjunktion setzen wir Wählpunkte (choice points)
- ▶ Alle Tableauerweiterung die von der Wahl abhängen werden ebenfalls markiert

Konstruktion von je einem Zweig

$$\begin{array}{c} (\neg p \vee q) \wedge p \wedge q \\ \neg p^{1a} \vee q^{1b} \\ p \\ q \\ \neg p^{1a} \end{array}$$

- ▶ Bei einer Disjunktion setzen wir Wählpunkte (choice points)
- ▶ Alle Tableauerweiterung die von der Wahl abhängen werden ebenfalls markiert

Konstruktion von je einem Zweig

$$\begin{array}{c}
 (\neg p \vee q) \wedge p \wedge q \\
 \neg p^{1a} \vee q^{1b} \\
 p \\
 q \\
 \neg p^{1a} \\
 \perp^{1a}
 \end{array}$$

- ▶ Bei einer Disjunktion setzen wir Wählpunkte (choice points)
- ▶ Alle Tableauerweiterung die von der Wahl abhängen werden ebenfalls markiert
- ▶ Bei Widerspruch auf Grund einer Wahl: revidiere markierte Formeln und versuche nächste Wahl

Konstruktion von je einem Zweig

$$\begin{array}{c}
 (\neg p \vee q) \wedge p \wedge q \\
 \neg p^{1a} \vee q^{1b} \\
 p \\
 q \\
 \cancel{\neg p^{1a}} \\
 \cancel{\perp^{1a}} \\
 q^{1b}
 \end{array}$$

- ▶ Bei einer Disjunktion setzen wir Wählpunkte (choice points)
- ▶ Alle Tableauerweiterung die von der Wahl abhängen werden ebenfalls markiert
- ▶ Bei Widerspruch auf Grund einer Wahl: revidiere markierte Formeln und versuche nächste Wahl

Vom Propositionalen Tableau zu Tableaux für BLs

Wie kann das Tableau zum Testen der Erfüllbarkeit von \mathcal{ALC} Konzepten erweitert werden?

- ▶ Tableau repräsentiert ein Element in der Domäne

Vom Propositionalen Tableau zu Tableaux für BLs

Wie kann das Tableau zum Testen der Erfüllbarkeit von \mathcal{ALC} Konzepten erweitert werden?

- ▶ Tableau repräsentiert ein Element in der Domäne
- ▶ Tableauezweig: Endliche Menge von Aussagen der Form $C(a)$, $\neg C(a)$, $r(a, b)$
- ▶ Für Existenzquantoren werden neue Elemente eingeführt
- ▶ Allquantoren propagieren Formeln zu benachbarten Elementen

Agenda

- ▶ Generelle Idee des Tableauealküls
- ▶ Propositionales Beispiel
- ▶ Transformation in Negationsnormalform
- ▶ Erfüllbarkeit von \mathcal{ALC} Konzepten
- ▶ Korrektheit und Terminierung
- ▶ Erfüllbarkeit von \mathcal{ALC} Wissensbasen
- ▶ Zusammenfassung

Aussagenlogik – Einige logische Äquivalenzen

- Unser Ziel ist es, Negation nur vor atomaren Konzepten zu haben

↪ siehe Vorlesung 5: Logik – Grundlagen

$$\varphi \wedge \psi \equiv \psi \wedge \varphi$$

$$\varphi \rightarrow \psi \equiv \neg\varphi \vee \psi$$

$$\varphi \vee \psi \equiv \psi \vee \varphi$$

$$\varphi \leftrightarrow \psi \equiv (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$$

$$\varphi \wedge (\psi \wedge \omega) \equiv (\varphi \wedge \psi) \wedge \omega$$

$$\neg(\varphi \wedge \psi) \equiv \neg\varphi \vee \neg\psi$$

$$\varphi \vee (\psi \vee \omega) \equiv (\varphi \vee \psi) \vee \omega$$

$$\neg(\varphi \vee \psi) \equiv \neg\varphi \wedge \neg\psi$$

$$\varphi \wedge \varphi \equiv \varphi$$

$$\neg\neg\varphi \equiv \varphi$$

$$\varphi \vee \varphi \equiv \varphi$$

$$\varphi \wedge (\psi \vee \varphi) \equiv \varphi$$

$$\varphi \vee (\psi \wedge \omega) \equiv (\varphi \vee \psi) \wedge (\varphi \vee \omega)$$

$$\varphi \vee (\psi \wedge \varphi) \equiv \varphi$$

$$\varphi \wedge (\psi \vee \omega) \equiv (\varphi \wedge \psi) \vee (\varphi \wedge \omega)$$

Weitere logische Äquivalenzen

$$\neg(C \sqcap D) \rightsquigarrow \neg C \sqcup \neg D$$

$$\neg(D \sqcup D) \rightsquigarrow \neg C \sqcap \neg D$$

$$\neg\neg C \rightsquigarrow C$$

$$\neg(\forall r.C) \rightsquigarrow \exists r.(\neg C)$$

$$\neg(\exists r.C) \rightsquigarrow \forall r.(\neg C)$$

$$\neg(\leq n s.C) \rightsquigarrow \geq n + 1 s.C$$

$$\neg(\geq n s.C) \rightsquigarrow \leq n - 1 s.C, \quad n \geq 1$$

$$\neg(\geq 0 s.C) \rightsquigarrow \perp$$

- ▶ Wende die folgenden Regeln so lange an, bis es nicht mehr geht

\rightsquigarrow Äquivalentes Konzept in Negationsnormalform

NNF Transformation

Induktive Definition einer NNF Transformation:

Wenn C atomar:

$$NNF(C) := C$$

$$NNF(\neg C) := \neg C$$

Sonst:

$$NNF(\neg\neg C) := NNF(C)$$

$$NNF(C \sqcap D) := NNF(C) \sqcap NNF(D)$$

$$NNF(\neg(C \sqcap D)) := NNF(\neg C) \sqcup NNF(\neg D)$$

$$NNF((C) \sqcup D) := NNF(C) \sqcup NNF(D)$$

$$NNF(\neg(C \sqcup D)) := NNF(\neg C) \sqcap NNF(\neg D)$$

$$NNF(\forall r.C) := \forall r.(NNF(C))$$

$$NNF(\neg(\forall r.C)) := \exists r.(NNF(\neg C))$$

$$NNF(\exists r.C) := \exists r.(NNF(C))$$

$$NNF(\neg(\exists r.C)) := \forall r.(NNF(\neg C))$$

$$NNF(\leq n s.C) := \leq n s.(NNF(C))$$

$$NNF(\neg(\leq n s.C)) := \geq n + 1 s.(NNF(C))$$

$$NNF(\geq n s.C) := \geq n s.(NNF(C))$$

$$NNF(\neg(\geq n s.C)) := \leq n - 1 s.(NNF(C))$$

wenn $n \geq 1$

$$NNF(\geq 0 s.C) := \top$$

$$NNF(\neg(\geq 0 s.C)) := \perp \quad \text{sonst}$$

Hervorgehoben sind Fehler in der ersten Skriptversion

NNF Transformation – Beispiel

$$\begin{aligned} & NNF(\neg(\neg C \sqcap (\neg D \sqcup E))) \\ = & NNF(\neg\neg C) \sqcup NNF(\neg(\neg D \sqcup E)) \\ = & NNF(C) \sqcup NNF(\neg(\neg D \sqcup E)) \\ = & C \sqcup NNF(\neg(\neg D \sqcup E)) \\ = & C \sqcup (NNF(\neg\neg D) \sqcap NNF(\neg E)) \\ = & C \sqcup (NNF(D) \sqcap NNF(\neg E)) \\ = & C \sqcup (D \sqcap NNF(\neg E)) \\ = & C \sqcup (D \sqcap \neg E) \end{aligned}$$

Agenda

- ▶ Generelle Idee des Tableaukalküls
- ▶ Propositionales Beispiel
- ▶ Transformation in Negationsnormalform
- ▶ **Erfüllbarkeit von \mathcal{ALC} Konzepten**
- ▶ Korrektheit und Terminierung
- ▶ Erfüllbarkeit von \mathcal{ALC} Wissensbasen
- ▶ Zusammenfassung

Tableau für \mathcal{ALC} Konzepte

- ▶ Tableau für eine propositionelle Formel α : ein Element, dass mit Teilformeln von α markiert ist
- ▶ Tableau für ein \mathcal{ALC} Konzept C : Graph (genauer: Baum) in dem Knoten mit Teilformeln von C markiert sind
- ▶ Wurzel markiert mit C
- ▶ Repräsentiert Modell für C
- ▶ Neue Knoten durch existentielle Quantoren

Definition

Sei C ein \mathcal{ALC} Konzept, $\text{TF}(C)$ die Menge aller Teilformeln von C und $\text{Rol}(C)$ die Menge der Rollen in C . Ein *Tableau für C* ist ein Baum $G = \langle V, E, L \rangle$, mit V einer Menge von Knoten, $E \subseteq V \times V$ einer Menge von Kanten und $L: V \rightarrow 2^{\text{TF}(C)}$ und $L: V \times V \rightarrow 2^{\text{Rol}(C)}$.

Eigenschaften des \mathcal{ALC} Tableau Algorithmus

- ▶ Der Algorithmus ist spezifiziert als eine Menge von Regeln
- ▶ Jede Regel bricht ein Konzept herunter in Teilkonzepte
- ▶ Anwendung von Regeln in beliebiger Ordnung
- ▶ Der Algorithmus ist nichtdeterministisch (durch Disjunktionen, bildet jeweils einen Tableau Zweig zur Zeit).
- ▶ Prüfen von atomaren Widersprüchen

Tableau Algorithmus zum Testen der Erfüllbarkeit von \mathcal{ALC} Konzepten

Eingabe: Ein \mathcal{ALC} Konzept in NNF

Ausgabe: wahr wenn ein widerspruchsfreies Tableau existiert
 in dem keine Regeln anwendbar sind
falsch andernfalls (Tableau geschlossen)

Tableau Regeln für \mathcal{ALC} Konzepte

- \sqcap -Regel: Für ein beliebiges $v \in V$ mit $C \sqcap D \in L(v)$ und $\{C, D\} \not\subseteq L(v)$, setze $L(v) := L(v) \cup \{C, D\}$.
- \sqcup -Regel: Für ein beliebiges $v \in V$ mit $C \sqcup D \in L(v)$ und $\{C, D\} \cap L(v) = \emptyset$, wähle $X \in \{C, D\}$ und setze $L(v) := L(v) \cup \{X\}$.
- \exists -Regel: Für ein beliebiges $v \in V$ mit $\exists r.C \in L(v)$ so dass es keinen r -Nachfolger v' für v mit $C \in L(v')$ gibt, setze $V = V \cup \{v'\}$, $E = E \cup \{\langle v, v' \rangle\}$, $L(v') := \{C\}$ und $L(v, v') := \{r\}$ für v' ein neuer Knoten.
- \forall -Regel: Für beliebige $v, v' \in V$, v' r -Nachbar von v , $\forall r.C \in L(v)$ und $C \notin L(v')$, setze $L(v') := L(v') \cup \{C\}$.
- ▶ Ein Knoten v' ist ein r -Nachbar eines Knotens v wenn $\langle v, v' \rangle \in E$ und $r \in L(v, v')$

Tableau Regeln für \mathcal{ALC} Konzepte

- \sqcap -Regel: Für ein beliebiges $v \in V$ mit $C \sqcap D \in L(v)$ und $\{C, D\} \not\subseteq L(v)$, setze $L(v) := L(v) \cup \{C, D\}$.
- \sqcup -Regel: Für ein beliebiges $v \in V$ mit $C \sqcup D \in L(v)$ und $\{C, D\} \cap L(v) = \emptyset$, wähle $X \in \{C, D\}$ und setze $L(v) := L(v) \cup \{X\}$.
- \exists -Regel: Für ein beliebiges $v \in V$ mit $\exists r.C \in L(v)$ so dass es keinen r -Nachfolger v' für v mit $C \in L(v')$ gibt, setze $V = V \cup \{v'\}$, $E = E \cup \{\langle v, v' \rangle\}$, $L(v') := \{C\}$ und $L(v, v') := \{r\}$ für v' ein neuer Knoten.
- \forall -Regel: Für beliebige $v, v' \in V$, v' r -Nachbar von v , $\forall r.C \in L(v)$ und $C \notin L(v')$, setze $L(v') := L(v') \cup \{C\}$.
- ▶ Ein Knoten v' ist ein r -Nachbar eines Knotens v wenn $\langle v, v' \rangle \in E$ und $r \in L(v, v')$
 - ▶ Regel Reihenfolge: “don’t care” Nichtdeterminismus

Tableau Regeln für \mathcal{ALC} Konzepte

- \sqcap -Regel: Für ein beliebiges $v \in V$ mit $C \sqcap D \in L(v)$ und $\{C, D\} \not\subseteq L(v)$, setze $L(v) := L(v) \cup \{C, D\}$.
- \sqcup -Regel: Für ein beliebiges $v \in V$ mit $C \sqcup D \in L(v)$ und $\{C, D\} \cap L(v) = \emptyset$, wähle $X \in \{C, D\}$ und setze $L(v) := L(v) \cup \{X\}$.
- \exists -Regel: Für ein beliebiges $v \in V$ mit $\exists r.C \in L(v)$ so dass es keinen r -Nachfolger v' für v mit $C \in L(v')$ gibt, setze $V = V \cup \{v'\}$, $E = E \cup \{\langle v, v' \rangle\}$, $L(v') := \{C\}$ und $L(v, v') := \{r\}$ für v' ein neuer Knoten.
- \forall -Regel: Für beliebige $v, v' \in V$, v' r -Nachbar von v , $\forall r.C \in L(v)$ und $C \notin L(v')$, setze $L(v') := L(v') \cup \{C\}$.
- ▶ Ein Knoten v' ist ein r -Nachbar eines Knotens v wenn $\langle v, v' \rangle \in E$ und $r \in L(v, v')$
 - ▶ Regel Reihenfolge: “don’t care” Nichtdeterminismus
 - ▶ Wahl der Disjunktion: “don’t know” Nichtdeterminismus

Tableau Algorithmus Beispiel

$$C = \exists r.(A \sqcup \exists r.B) \sqcap \exists r.\neg A \sqcap \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))$$

u

$$L(u) = \{C\}$$

Tableau Algorithmus Beispiel

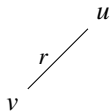
$$C = \exists r.(A \sqcup \exists r.B) \sqcap \exists r.\neg A \sqcap \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))$$

u

$$L(u) = \{C, \exists r.(A \sqcup \exists r.B), \\ \exists r.\neg A, \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))\}$$

Tableau Algorithmus Beispiel

$$C = \exists r.(A \sqcup \exists r.B) \sqcap \exists r.\neg A \sqcap \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))$$

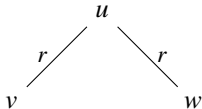


$$L(u) = \{C, \exists r.(A \sqcup \exists r.B), \\ \exists r.\neg A, \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))\}$$

$$L(v) = \{A \sqcup \exists r.B\}$$

Tableau Algorithmus Beispiel

$$C = \exists r.(A \sqcup \exists r.B) \sqcap \exists r.\neg A \sqcap \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))$$



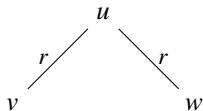
$$L(u) = \{C, \exists r.(A \sqcup \exists r.B), \\ \exists r.\neg A, \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))\}$$

$$L(v) = \{A \sqcup \exists r.B\}$$

$$L(w) = \{\neg A\}$$

Tableau Algorithmus Beispiel

$$C = \exists r.(A \sqcup \exists r.B) \sqcap \exists r.\neg A \sqcap \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))$$



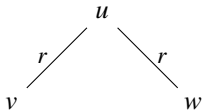
$$L(u) = \{C, \exists r.(A \sqcup \exists r.B), \\ \exists r.\neg A, \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))\}$$

$$L(v) = \{A \sqcup \exists r.B, \neg A, \forall r.(\neg B \sqcup A)\}$$

$$L(w) = \{\neg A\}$$

Tableau Algorithmus Beispiel

$$C = \exists r.(A \sqcup \exists r.B) \sqcap \exists r.\neg A \sqcap \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))$$



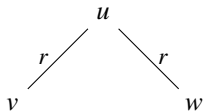
$$L(u) = \{C, \exists r.(A \sqcup \exists r.B), \\ \exists r.\neg A, \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))\}$$

$$L(v) = \{A \sqcup \exists r.B, \neg A, \forall r.(\neg B \sqcup A)\}$$

$$L(w) = \{\neg A, \forall r.(\neg B \sqcup A)\}$$

Tableau Algorithmus Beispiel

$$C = \exists r.(A \sqcup \exists r.B) \sqcap \exists r.\neg A \sqcap \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))$$



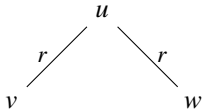
$$L(u) = \{C, \exists r.(A \sqcup \exists r.B), \\ \exists r.\neg A, \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))\}$$

$$L(v) = \{A \sqcup \exists r.B, \neg A, \forall r.(\neg B \sqcup A), A\}$$

$$L(w) = \{\neg A, \forall r.(\neg B \sqcup A)\}$$

Tableau Algorithmus Beispiel

$$C = \exists r.(A \sqcup \exists r.B) \sqcap \exists r.\neg A \sqcap \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))$$



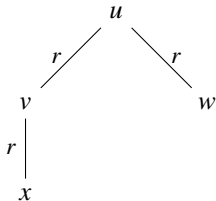
$$L(u) = \{C, \exists r.(A \sqcup \exists r.B), \\ \exists r.\neg A, \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))\}$$

$$L(v) = \{A \sqcup \exists r.B, \neg A, \forall r.(\neg B \sqcup A), \text{X}\}$$

$$L(w) = \{\neg A, \forall r.(\neg B \sqcup A)\}$$

Tableau Algorithmus Beispiel

$$C = \exists r.(A \sqcup \exists r.B) \sqcap \exists r.\neg A \sqcap \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))$$



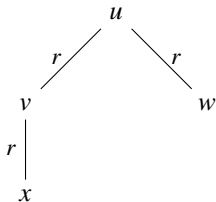
$$L(u) = \{C, \exists r.(A \sqcup \exists r.B), \\ \exists r.\neg A, \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))\}$$

$$L(v) = \{A \sqcup \exists r.B, \neg A, \forall r.(\neg B \sqcup A), \cancel{A}, \cancel{\exists r.B}\}$$

$$L(w) = \{\neg A, \forall r.(\neg B \sqcup A)\}$$

Tableau Algorithmus Beispiel

$$C = \exists r.(A \sqcup \exists r.B) \sqcap \exists r.\neg A \sqcap \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))$$



$$L(u) = \{C, \exists r.(A \sqcup \exists r.B), \\ \exists r.\neg A, \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))\}$$

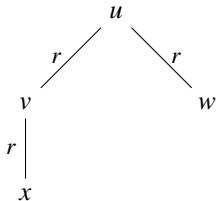
$$L(v) = \{A \sqcup \exists r.B, \neg A, \forall r.(\neg B \sqcup A), \exists r.B\}$$

$$L(w) = \{\neg A, \forall r.(\neg B \sqcup A)\}$$

$$L(x) = \{B\}$$

Tableau Algorithmus Beispiel

$$C = \exists r.(A \sqcup \exists r.B) \sqcap \exists r.\neg A \sqcap \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))$$



$$L(u) = \{C, \exists r.(A \sqcup \exists r.B), \\ \exists r.\neg A, \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))\}$$

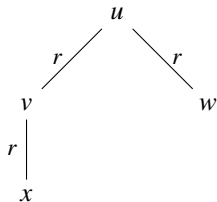
$$L(v) = \{A \sqcup \exists r.B, \neg A, \forall r.(\neg B \sqcup A), \exists r.B\}$$

$$L(w) = \{\neg A, \forall r.(\neg B \sqcup A)\}$$

$$L(x) = \{B, \neg B \sqcup A\}$$

Tableau Algorithmus Beispiel

$$C = \exists r.(A \sqcup \exists r.B) \sqcap \exists r.\neg A \sqcap \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))$$



$$L(u) = \{C, \exists r.(A \sqcup \exists r.B), \\ \exists r.\neg A, \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))\}$$

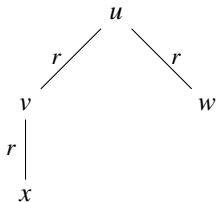
$$L(v) = \{A \sqcup \exists r.B, \neg A, \forall r.(\neg B \sqcup A), \exists r.B\}$$

$$L(w) = \{\neg A, \forall r.(\neg B \sqcup A)\}$$

$$L(x) = \{B, \neg B \sqcup A, \neg B\}$$

Tableau Algorithmus Beispiel

$$C = \exists r.(A \sqcup \exists r.B) \sqcap \exists r.\neg A \sqcap \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))$$



$$L(u) = \{C, \exists r.(A \sqcup \exists r.B), \\ \exists r.\neg A, \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))\}$$

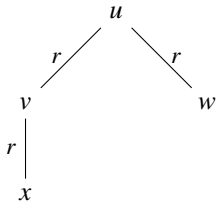
$$L(v) = \{A \sqcup \exists r.B, \neg A, \forall r.(\neg B \sqcup A), \exists r.B\}$$

$$L(w) = \{\neg A, \forall r.(\neg B \sqcup A)\}$$

$$L(x) = \{B, \neg B \sqcup A, B\}$$

Tableau Algorithmus Beispiel

$$C = \exists r.(A \sqcup \exists r.B) \sqcap \exists r.\neg A \sqcap \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))$$



$$L(u) = \{C, \exists r.(A \sqcup \exists r.B), \\ \exists r.\neg A, \forall r.(\neg A \sqcap \forall r.(\neg B \sqcup A))\}$$

$$L(v) = \{A \sqcup \exists r.B, \neg A, \forall r.(\neg B \sqcup A), \exists r.B\}$$

$$L(w) = \{\neg A, \forall r.(\neg B \sqcup A)\}$$

$$L(x) = \{B, \neg B \sqcup A, B, A\}$$

Tableau Algorithmus Beispiel

Das vom Algorithmus gefundene Modell \mathcal{I} ist wie folgt:

$$\Delta^{\mathcal{I}} = \{u, v, w, x\}$$

$$A^{\mathcal{I}} = \{x\}$$

$$B^{\mathcal{I}} = \{x\}$$

$$r^{\mathcal{I}} = \{\langle u, v \rangle, \langle u, w \rangle, \langle v, x \rangle\}$$

Übung: Prüfe, dass $C^{\mathcal{I}} = \{u\}$ unter der definierten Semantik für \mathcal{ALC}

Tableau Algorithmus Eigenschaften

1. Das Modell ist **endlich**: nur endlich viele Elemente in der Domäne
2. Das Modell ist **baumförmig**: das Tableau ist ein markierter Baum

Der Algorithmus wird *immer* endliche Bäume konstruieren

- ▶ Von einem widerspruchsfreien Tableau können wir ein endliches Modell konstruieren
- ▶ Wenn es kein widerspruchsfreies Tableau gibt, dann gibt es kein Modell

Agenda

- ▶ Generelle Idee des Tableaukalküls
- ▶ Propositionales Beispiel
- ▶ Transformation in Negationsnormalform
- ▶ Erfüllbarkeit von \mathcal{ALC} Konzepten
- ▶ **Korrektheit und Terminierung**
- ▶ Erfüllbarkeit von \mathcal{ALC} Wissensbasen
- ▶ Zusammenfassung

Tableau Eigenschaften

- ▶ Die Tiefe (Anzahl der Verschachtelten Quantoren) nimmt in jedem Knoten ab
- ▶ Jeder Knoten ist nur mit Teilformeln von C markiert
- ▶ C hat nur polynomiell viele Teilformeln
- ▶ Bei Ausgabe `wahr` können wir an Hand des konstruierten Tableaus ein Modell konstruieren
- ▶ Wir können ein Modell für ein erfüllbares Konzept nutzen, um ein widerspruchsfreies Tableau für das Konzept zu konstruieren

Tableau Algorithmus für \mathcal{ALC} -Konzepte

Theorem

1. *Der Algorithmus terminiert für jede Eingabe.*
2. *Wenn die Ausgabe `wahr` lautet, dann ist das Eingabekonzept erfüllbar.*
3. *Wenn das Eingabekonzept erfüllbar ist, lautet die Ausgabe `wahr`.*

Tableau Algorithmus für \mathcal{ALC} -Konzepte

Theorem

1. *Der Algorithmus terminiert für jede Eingabe.*
2. *Wenn die Ausgabe `wahr` lautet, dann ist das Eingabekonzept erfüllbar.*
3. *Wenn das Eingabekonzept erfüllbar ist, lautet die Ausgabe `wahr`.*

Corollary

Ein \mathcal{ALC} Konzept C hat die folgenden Eigenschaften:

1. ***Endliche Modell Eigenschaft** (Finite Model Property):
Wenn C ein Modell hat, dann hat C ein endliches Modell*
2. ***Baum-Modell Eigenschaft** (Tree Model Property): Wenn C ein Modell hat, dann hat C ein baumförmiges Modell*

Agenda

- ▶ Generelle Idee des Tableaukalküls
- ▶ Propositionales Beispiel
- ▶ Transformation in Negationsnormalform
- ▶ Erfüllbarkeit von \mathcal{ALC} Konzepten
- ▶ Korrektheit und Terminierung
- ▶ Erfüllbarkeit von \mathcal{ALC} Wissensbasen
- ▶ Zusammenfassung

Tableau Algorithmus für TBoxen

Wir erweitern nun den Tableau Algorithmus für den Umgang mit \mathcal{ALC} TBoxen

- ▶ Eine TBox enthält Axiome (GCIs) der Form $C \sqsubseteq D$
- ▶ Annahme: $C \equiv D$ ist ersetzt durch $C \sqsubseteq D$ und $D \sqsubseteq C$
- ▶ Jedes GCI ist äquivalent zu $\top \sqsubseteq \neg C \sqcup D$

Wir können die gesamte TBox in einem Axiom zusammenfassen (internalisieren, engl. internalize):

$$\mathcal{T} = \{C_i \sqsubseteq D_i \mid 1 \leq i \leq n\}$$

ist äquivalentes zu:

$$\mathcal{T}' = \{\top \sqsubseteq \prod_{1 \leq i \leq n} \neg C_i \sqcup D_i\}$$

Sei $C_{\mathcal{T}}$ das Konzept auf der rechten Seite des GCI in NNF.

Tableau Algorithmus for TBoxes

Wir erweitern die Regeln des \mathcal{ALC} Tableau Algorithmus mit der Regel:

\mathcal{T} -Regel: Für ein beliebiges $v \in V$ mit $C_{\mathcal{T}} \notin L(v)$,
setze $L(v) := L(v) \cup \{C_{\mathcal{T}}\}$.

Beispiel: Sei $\mathcal{T} = A \sqsubseteq \exists r.A$. Ist A erfüllbar bzgl. \mathcal{T} ?

Tableau Algorithmus for TBoxes

Wir erweitern die Regeln des \mathcal{ALC} Tableau Algorithmus mit der Regel:

\mathcal{T} -Regel: Für ein beliebiges $v \in V$ mit $C_{\mathcal{T}} \notin L(v)$,
setze $L(v) := L(v) \cup \{C_{\mathcal{T}}\}$.

Beispiel: Sei $\mathcal{T} = A \sqsubseteq \exists r.A$. Ist A erfüllbar bzgl. \mathcal{T} ?

Der Tableau Algorithmus terminiert nicht mehr!

Tableau Algorithmus for TBoxes

Wir erweitern die Regeln des \mathcal{ALC} Tableau Algorithmus mit der Regel:

\mathcal{T} -Regel: Für ein beliebiges $v \in V$ mit $C_{\mathcal{T}} \notin L(v)$,
setze $L(v) := L(v) \cup \{C_{\mathcal{T}}\}$.

Beispiel: Sei $\mathcal{T} = A \sqsubseteq \exists r.A$. Ist A erfüllbar bzgl. \mathcal{T} ?

Der Tableau Algorithmus terminiert nicht mehr!

Die Tiefe der Quantoren nimmt nicht mehr ab auf einem Pfad im Tableau

Tableau Algorithmus for TBoxes

Wir erweitern die Regeln des \mathcal{ALC} Tableau Algorithmus mit der Regel:

\mathcal{T} -Regel: Für ein beliebiges $v \in V$ mit $C_{\mathcal{T}} \notin L(v)$,
setze $L(v) := L(v) \cup \{C_{\mathcal{T}}\}$.

Beispiel: Sei $\mathcal{T} = A \sqsubseteq \exists r.A$. Ist A erfüllbar bzgl. \mathcal{T} ?

Der Tableau Algorithmus terminiert nicht mehr!

Die Tiefe der Quantoren nimmt nicht mehr ab auf einem Pfad im Tableau

Lösung: Wir werden **Zyklen erkennen** (sich wiederholende Knotenmarkierungen)

Tableau Algorithmus für TBoxen

Definition (Blocking)

Ein Knoten $v \in V$ **blockiert** einen Knoten $v' \in V$ **direkt**, wenn:

1. v' von v erreichbar ist,
2. $L(v') \subseteq L(v)$; und
3. es keinen direkt blockierten Knoten v'' gibt so dass v' von v'' erreichbar ist.

Ein Knoten $v' \in V$ ist **blockiert** wenn entweder

1. v' direkt blockiert ist oder
2. es einen direkt blockierten Knoten v gibt, so dass v' von v erreichbar ist.

Tableau Algorithmus für TBoxen

Definition (Blocking)

Ein Knoten $v \in V$ **blockiert** einen Knoten $v' \in V$ **direkt**, wenn:

1. v' von v erreichbar ist,
2. $L(v') \subseteq L(v)$; und
3. es keinen direkt blockierten Knoten v'' gibt so dass v' von v'' erreichbar ist.

Ein Knoten $v' \in V$ ist **blockiert** wenn entweder

1. v' direkt blockiert ist oder
2. es einen direkt blockierten Knoten v gibt, so dass v' von v erreichbar ist.

Die Anwendung der \exists -Regel ist beschränkt auf Knoten die **nicht blockiert** sind.

Tableau Algorithmus mit Blocking

Beispiel: Sei $\mathcal{T} = A \sqsubseteq \exists r.A$. Ist A erfüllbar bzgl. \mathcal{T} ?

Wir erhalten das folgende widerspruchsfreie Tableau:

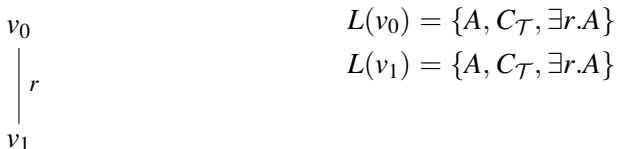
$$\begin{array}{c} v_0 \\ \left| \begin{array}{c} \\ r \\ \end{array} \right. \\ v_1 \end{array} \qquad \begin{array}{l} L(v_0) = \{A, C_{\mathcal{T}}, \exists r.A\} \\ L(v_1) = \{A, C_{\mathcal{T}}, \exists r.A\} \end{array}$$

in dem v_1 **direkt** von v_0 **blockiert** ist.

Tableau Algorithmus mit Blocking

Beispiel: Sei $\mathcal{T} = A \sqsubseteq \exists r.A$. Ist A erfüllbar bzgl. \mathcal{T} ?

Wir erhalten das folgende widerspruchsfreie Tableau:



in dem v_1 **direkt** von v_0 **blockiert** ist.

Der Algorithmus konstruiert wieder endliche Bäume

- ▶ Von einem widerspruchsfreien Tableau können wir ein Modell konstruieren
- ▶ Wenn es kein widerspruchsfreies Tableau gibt, dann gibt es kein Modell

Vom Tableau zum Modell

Wir können wiederum ein endliches Modell von einem widerspruchsfreien Tableau konstruieren:

$$\Delta^{\mathcal{I}} = \{v_0\}$$

$$A^{\mathcal{I}} = \Delta^{\mathcal{I}}$$

$$r^{\mathcal{I}} = \{\langle v_0, v_0 \rangle\}$$

- ▶ Blockierte Knoten repräsentieren kein Element im Modell
- ▶ Eine Kante von einem Knoten v zu einem direkt blockierten Knoten v' wird im Modell zu einer "Kante" von v zu dem Knoten, der v' direkt blockiert

Vom Tableau zum Modell

Wir können wiederum ein endliches Modell von einem widerspruchsfreien Tableau konstruieren:

$$\Delta^{\mathcal{I}} = \{v_0\}$$

$$A^{\mathcal{I}} = \Delta^{\mathcal{I}}$$

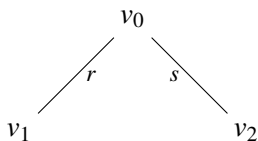
$$r^{\mathcal{I}} = \{\langle v_0, v_0 \rangle\}$$

- ▶ Blockierte Knoten repräsentieren kein Element im Modell
- ▶ Eine Kante von einem Knoten v zu einem direkt blockierten Knoten v' wird im Modell zu einer "Kante" von v zu dem Knoten, der v' direkt blockiert
- ↪ Wir haben die **finite model property**
- ↪ Modell nicht mehr baumförmig

Tableau Algorithmus mit Blocking II

Beispiel: Sei $\mathcal{T} = A \sqsubseteq \exists r.A \sqcap \exists s.B$. Ist A erfüllbar bzgl. \mathcal{T} ?

Wir erhalten das folgende widerspruchsfreie Tableau:



$$L(v_0) = \{A, C_{\mathcal{T}}, \exists r.A \sqcap \exists s.B, \exists r.A, \exists s.B\}$$

$$L(v_1) = \{A, C_{\mathcal{T}}, \exists r.A \sqcap \exists s.B, \exists r.A, \exists s.B\}$$

$$L(v_2) = \{B, C_{\mathcal{T}}, \neg A\}$$

in dem v_1 wieder **direkt** von v_0 **blockiert** ist.

Vom Tableau zum Modell II

Wir können wiederum ein endliches Modell von einem widerspruchsfreien Tableau konstruieren:

$$\Delta^{\mathcal{I}} = \{v_0, v_2\}$$

$$A^{\mathcal{I}} = \Delta^{\mathcal{I}}$$

$$r^{\mathcal{I}} = \{\langle v_0, v_0 \rangle\}$$

$$s^{\mathcal{I}} = \{\langle v_0, v_2 \rangle\}$$

Agenda

- ▶ Generelle Idee des Tableauealküls
- ▶ Propositionales Beispiel
- ▶ Transformation in Negationsnormalform
- ▶ Erfüllbarkeit von \mathcal{ALC} Konzepten
- ▶ Korrektheit und Terminierung
- ▶ Erfüllbarkeit von \mathcal{ALC} Wissensbasen
- ▶ Zusammenfassung

Zusammenfassung

- ▶ Wir haben nun ein konstruktives Verfahren zum Konstruieren von Modellabstraktionen
- ▶ \mathcal{ALC} Konzepte haben immer ein endliches Modell, das wir auch konstruieren können
- ▶ \mathcal{ALC} Konzepte bzgl. Wissenbasen haben ebenfalls endliche Modelle, aber wir können diese nur mit Mechanismen zur Zyklen-Erkennung konstruieren
- ▶ Die dann erstellten Modelle sind nicht mehr baumförmig
- ▶ Nächste Vorlesung: Wir können auch baumförmige (aber dann unendliche) Modelle konstruieren
- ▶ Der Algorithmus ist korrekt, vollständig und terminiert
- ▶ Grundlage für in der Praxis implementierte Algorithmen