Logik-Programmierung und Negation

Sebastian Marius Kirsch
skirsch@moebius.inka.de

Überblick

- Motivation und Vorbetrachtungen
- Rückblick: Herbrand-Theorie, Vervollständigung, SLDNF-Resolution
- SLD-CNF-Resolution
- Dreiwertige Logik, dreiwertige Herbrandmodelle
- Wohlfundierte Semantik
- SLS-Resolution

Überblick

- Motivation und Vorbetrachtungen
- Rückblick: Herbrand-Theorie, Vervollständigung, SLDNF-Resolution
- SLD-CNF-Resolution
- Dreiwertige Logik, dreiwertige Herbrandmodelle
- Wohlfundierte Semantik
- SLS-Resolution

Überblick

- Motivation und Vorbetrachtungen
- Rückblick: Herbrand-Theorie, Vervollständigung, SLDNF-Resolution
- SLD-CNF-Resolution
- Dreiwertige Logik, dreiwertige Herbrandmodelle
- Wohlfundierte Semantik
- SLS-Resolution

Überblick

- Motivation und Vorbetrachtungen
- Rückblick: Herbrand-Theorie, Vervollständigung, SLDNF-Resolution
- SLD-CNF-Resolution
- Dreiwertige Logik, dreiwertige Herbrandmodelle
- Wohlfundierte Semantik
- SLS-Resolution

Back
Close

Überblick

- Motivation und Vorbetrachtungen
- Rückblick: Herbrand-Theorie, Vervollständigung, SLDNF-Resolution
- SLD-CNF-Resolution
- Dreiwertige Logik, dreiwertige Herbrandmodelle
- Wohlfundierte Semantik
- SLS-Resolution

Überblick

- Motivation und Vorbetrachtungen
- Rückblick: Herbrand-Theorie, Vervollständigung, SLDNF-Resolution
- SLD-CNF-Resolution
- Dreiwertige Logik, dreiwertige Herbrandmodelle
- Wohlfundierte Semantik
- SLS-Resolution

Überblick

- Motivation und Vorbetrachtungen
- Rückblick: Herbrand-Theorie, Vervollständigung, SLDNF-Resolution
- SLD-CNF-Resolution
- Dreiwertige Logik, dreiwertige Herbrandmodelle
- Wohlfundierte Semantik
- SLS-Resolution

Motivation

- Monotonie: $\Gamma \vdash \mathrm{L} \Rightarrow \Gamma \cup \Gamma' \vdash \mathrm{L}$
- klassische Logik ist monoton
- Negation führt zu Nichtmonotonie.
- Semantik mit Negation teilw. berechenbar
- Rationalität: $\Gamma \not\vdash \neg A$ und $\Gamma \vdash \mathrm{L} \Rightarrow \Gamma \cup \{A\} \vdash \mathrm{L}$
- Datenbanken (negative Information implizit)
- „gesunder Menschenverstand“ ist nichtmonoton (vergl. Default-Logiken)

Motivation

- Monotonie: $\Gamma \vdash \mathrm{L} \Rightarrow \Gamma \cup \Gamma' \vdash \mathrm{L}$
- klassische Logik ist monoton
- Negation führt zu Nichtmonotonie.
- Semantik mit Negation teilw. berechenbar
- Rationalität: $\Gamma \not\vdash \neg A$ und $\Gamma \vdash \mathrm{L} \Rightarrow \Gamma \cup \{A\} \vdash \mathrm{L}$
- Datenbanken (negative Information implizit)
- „gesunder Menschenverstand“ ist nichtmonoton (vergl. Default-Logiken)

Motivation

- Monotonie: $\Gamma \vdash \mathrm{L} \Rightarrow \Gamma \cup \Gamma' \vdash \mathrm{L}$
- klassische Logik ist monoton
- Negation führt zu Nichtmonotonie.
- Semantik mit Negation teilw. berechenbar
- Rationalität: $\Gamma \not\vdash \neg A$ und $\Gamma \vdash \mathrm{L} \Rightarrow \Gamma \cup \{A\} \vdash \mathrm{L}$
- Datenbanken (negative Information implizit)
- „gesunder Menschenverstand“ ist nichtmonoton (vergl. Default-Logiken)

Motivation

- Monotonie: $\Gamma \vdash \mathrm{L} \Rightarrow \Gamma \cup \Gamma' \vdash \mathrm{L}$
- klassische Logik ist monoton
- Negation führt zu Nichtmonotonie.
- Semantik mit Negation teilw. berechenbar
- Rationalität: $\Gamma \nvdash \neg A$ und $\Gamma \vdash \mathrm{L} \Rightarrow \Gamma \cup \{A\} \vdash \mathrm{L}$
- Datenbanken (negative Information implizit)
- „gesunder Menschenverstand“ ist nichtmonoton (vergl. Default-Logiken)

Motivation

- Monotonie: $\Gamma \vdash \mathrm{L} \Rightarrow \Gamma \cup \Gamma' \vdash \mathrm{L}$
- klassische Logik ist monoton
- Negation führt zu Nichtmonotonie.
- Semantik mit Negation teilw. berechenbar
- Rationalität: $\Gamma \nvdash \neg A$ und $\Gamma \vdash \mathrm{L} \Rightarrow \Gamma \cup \{A\} \vdash \mathrm{L}$
- Datenbanken (negative Information implizit)
- „gesunder Menschenverstand“ ist nichtmonoton (vergl. Default-Logiken)

Motivation

- Monotonie: $\Gamma \vdash \mathrm{L} \Rightarrow \Gamma \cup \Gamma' \vdash \mathrm{L}$
- klassische Logik ist monoton
- Negation führt zu Nichtmonotonie.
- Semantik mit Negation teilw. berechenbar
- Rationalität: $\Gamma \not\vdash \neg A$ und $\Gamma \vdash \mathrm{L} \Rightarrow \Gamma \cup \{A\} \vdash \mathrm{L}$
- Datenbanken (negative Information implizit)
- „gesunder Menschenverstand“ ist nichtmonoton (vergl. Default-Logiken)

Motivation

- Monotonie: $\Gamma \vdash \mathrm{L} \Rightarrow \Gamma \cup \Gamma' \vdash \mathrm{L}$
- klassische Logik ist monoton
- Negation führt zu Nichtmonotonie.
- Semantik mit Negation teilw. berechenbar
- Rationalität: $\Gamma \not\vdash \neg A$ und $\Gamma \vdash \mathrm{L} \Rightarrow \Gamma \cup \{A\} \vdash \mathrm{L}$
- Datenbanken (negative Information implizit)
- „gesunder Menschenverstand“ ist nichtmonoton (vergl. Default-Logiken)

Back
Close

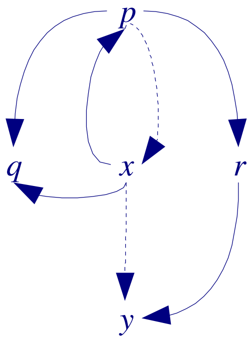
Motivation

- Monotonie: $\Gamma \vdash \mathrm{L} \Rightarrow \Gamma \cup \Gamma' \vdash \mathrm{L}$
- klassische Logik ist monoton
- Negation führt zu Nichtmonotonie.
- Semantik mit Negation teilw. berechenbar
- Rationalität: $\Gamma \not\vdash \neg A$ und $\Gamma \vdash \mathrm{L} \Rightarrow \Gamma \cup \{A\} \vdash \mathrm{L}$
- Datenbanken (negative Information implizit)
- „gesunder Menschenverstand“ ist nichtmonoton (vergl. Default-Logiken)

Abhängigkeitsgraphen

- gerichteter Graph mit positiven und negativen Kanten
- Kanten von der Relation im Kopf einer Klausel zu jeder Relation im Rumpf
- p hängt gerade (ungerade) von q ab, wenn es einen Pfad mit einer geraden (ungeraden) Anzahl negativer Kanten gibt.
- *call consistency*: Keine Relation hängt ungerade von sich selbst ab.

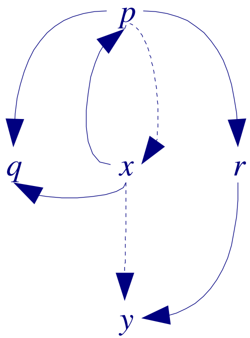
$$
\begin{array}{c}
p \leftarrow q \\
p \leftarrow r, \neg x \\
q \\
r \leftarrow y \\
x \leftarrow q \\
x \leftarrow p \\
x \leftarrow \neg y, r
\end{array}
$$



Abhängigkeitsgraphen

- gerichteter Graph mit positiven und negativen Kanten
- Kanten von der Relation im Kopf einer Klausel zu jeder Relation im Rumpf
- p hängt gerade (ungerade) von q ab, wenn es einen Pfad mit einer geraden (ungeraden) Anzahl negativer Kanten gibt.
- *call consistency*: Keine Relation hängt ungerade von sich selbst ab.

$$
\begin{array}{c}
p \leftarrow q \\
p \leftarrow r, \neg x \\
q \\
r \leftarrow y \\
x \leftarrow q \\
x \leftarrow p \\
x \leftarrow \neg y, r
\end{array}
$$

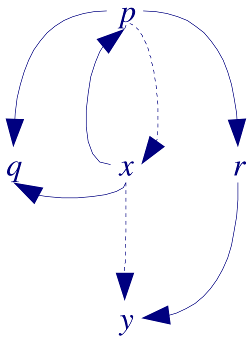


Back
Close

Abhängigkeitsgraphen

- gerichteter Graph mit positiven und negativen Kanten
- Kanten von der Relation im Kopf einer Klausel zu jeder Relation im Rumpf
- p hängt gerade (ungerade) von q ab, wenn es einen Pfad mit einer geraden (ungeraden) Anzahl negativer Kanten gibt.
- *call consistency*: Keine Relation hängt ungerade von sich selbst ab.

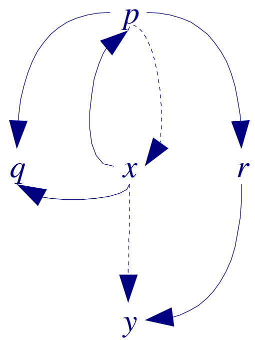
$$
\begin{gathered}
p \leftarrow q \\
p \leftarrow r, \neg x \\
q \\
r \leftarrow y \\
x \leftarrow q \\
x \leftarrow p \\
x \leftarrow \neg y, r
\end{gathered}
$$



Abhängigkeitsgraphen

- gerichteter Graph mit positiven und negativen Kanten
- Kanten von der Relation im Kopf einer Klausel zu jeder Relation im Rumpf
- p hängt gerade (ungerade) von q ab, wenn es einen Pfad mit einer geraden (ungeraden) Anzahl negativer Kanten gibt.
- *call consistency*: Keine Relation hängt ungerade von sich selbst ab.

$$
\begin{gathered}
p \leftarrow q \\
p \leftarrow r, \neg x \\
q \\
r \leftarrow y \\
x \leftarrow q \\
x \leftarrow p \\
x \leftarrow \neg y, r
\end{gathered}
$$



Rückblick: Herbrand-Theorie

- Idee: keine beliebigen Interpretationen, stattdessen werden Konstanten und Funktionsanwendungen durch sich selbst interpretiert.
- Herbrand-Universum $\mathcal{U}_{\mathrm{P}}$: Menge aller variablenfreien Terme, die aus Funktions- und Konstantensymbolen in P gebildet werden können
- Alternative zum Herbrand-Universum: Postuliere eine universelle Sprache, in der alle Programme und Fragen ausgedrückt werden.
- Vorteil: Löst das Problem von Sprachelementen, die in der Frage, aber nicht im Programm vorkommen.
- Herbrand-Basis $\mathcal{B}_{\mathrm{P}}$: Menge aller Grundatome
- Repräsentiere Herbrand-Interpretation I als Menge aller Grundatome, die unter I wahr sind.

Rückblick: Herbrand-Theorie

- Idee: keine beliebigen Interpretationen, stattdessen werden Konstanten und Funktionsanwendungen durch sich selbst interpretiert.
- Herbrand-Universum $\mathcal{U}_{\mathrm{P}}$: Menge aller variablenfreien Terme, die aus Funktions- und Konstantensymbolen in P gebildet werden können
- Alternative zum Herbrand-Universum: Postuliere eine universelle Sprache, in der alle Programme und Fragen ausgedrückt werden.
- Vorteil: Löst das Problem von Sprachelementen, die in der Frage, aber nicht im Programm vorkommen.
- Herbrand-Basis $\mathcal{B}_{\mathrm{P}}$: Menge aller Grundatome
- Repräsentiere Herbrand-Interpretation I als Menge aller Grundatome, die unter I wahr sind.

Rückblick: Herbrand-Theorie

- Idee: keine beliebigen Interpretationen, stattdessen werden Konstanten und Funktionsanwendungen durch sich selbst interpretiert.
- Herbrand-Universum $\mathcal{U}_{\mathrm{P}}$: Menge aller variablenfreien Terme, die aus Funktions- und Konstantensymbolen in P gebildet werden können
- Alternative zum Herbrand-Universum: Postuliere eine universelle Sprache, in der alle Programme und Fragen ausgedrückt werden.
- Vorteil: Löst das Problem von Sprachelementen, die in der Frage, aber nicht im Programm vorkommen.
- Herbrand-Basis $\mathcal{B}_{\mathrm{P}}$: Menge aller Grundatome
- Repräsentiere Herbrand-Interpretation I als Menge aller Grundatome, die unter I wahr sind.

Rückblick: Herbrand-Theorie

- Idee: keine beliebigen Interpretationen, stattdessen werden Konstanten und Funktionsanwendungen durch sich selbst interpretiert.
- Herbrand-Universum $\mathcal{U}_{\mathrm{P}}$: Menge aller variablenfreien Terme, die aus Funktions- und Konstantensymbolen in P gebildet werden können
- Alternative zum Herbrand-Universum: Postuliere eine universelle Sprache, in der alle Programme und Fragen ausgedrückt werden.
- Vorteil: Löst das Problem von Sprachelementen, die in der Frage, aber nicht im Programm vorkommen.
- Herbrand-Basis $\mathcal{B}_{\mathrm{P}}$: Menge aller Grundatome
- Repräsentiere Herbrand-Interpretation I als Menge aller Grundatome, die unter I wahr sind.

Back
Close

Rückblick: Herbrand-Theorie

- Idee: keine beliebigen Interpretationen, stattdessen werden Konstanten und Funktionsanwendungen durch sich selbst interpretiert.
- Herbrand-Universum $\mathcal{U}_{\mathrm{P}}$: Menge aller variablenfreien Terme, die aus Funktions- und Konstantensymbolen in P gebildet werden können
- Alternative zum Herbrand-Universum: Postuliere eine universelle Sprache, in der alle Programme und Fragen ausgedrückt werden.
- Vorteil: Löst das Problem von Sprachelementen, die in der Frage, aber nicht im Programm vorkommen.
- Herbrand-Basis $\mathcal{B}_{\mathrm{P}}$: Menge aller Grundatome
- Repräsentiere Herbrand-Interpretation I als Menge aller Grundatome, die unter I wahr sind.

Rückblick: Herbrand-Theorie

- Idee: keine beliebigen Interpretationen, stattdessen werden Konstanten und Funktionsanwendungen durch sich selbst interpretiert.
- Herbrand-Universum $\mathcal{U}_{\mathrm{P}}$: Menge aller variablenfreien Terme, die aus Funktions- und Konstantensymbolen in P gebildet werden können
- Alternative zum Herbrand-Universum: Postuliere eine universelle Sprache, in der alle Programme und Fragen ausgedrückt werden.
- Vorteil: Löst das Problem von Sprachelementen, die in der Frage, aber nicht im Programm vorkommen.
- Herbrand-Basis $\mathcal{B}_{\mathrm{P}}$: Menge aller Grundatome
- Repräsentiere Herbrand-Interpretation I als Menge aller Grundatome, die unter I wahr sind.

Rückblick: Herbrand-Theorie

- Idee: keine beliebigen Interpretationen, stattdessen werden Konstanten und Funktionsanwendungen durch sich selbst interpretiert.
- Herbrand-Universum $\mathcal{U}_{\mathrm{P}}$: Menge aller variablenfreien Terme, die aus Funktions- und Konstantensymbolen in P gebildet werden können
- Alternative zum Herbrand-Universum: Postuliere eine universelle Sprache, in der alle Programme und Fragen ausgedrückt werden.
- Vorteil: Löst das Problem von Sprachelementen, die in der Frage, aber nicht im Programm vorkommen.
- Herbrand-Basis $\mathcal{B}_{\mathrm{P}}$: Menge aller Grundatome
- Repräsentiere Herbrand-Interpretation I als Menge aller Grundatome, die unter I wahr sind.

Rückblick: Programmvervollständigung

- *closed world assumption* (CWA): $\neg A$ wird angenommen, wenn A nicht bewiesen werden kann.
- i. A. unentscheidbar
- Vervollständigung ist Formalisierung der *closed world assumption*
- Verstärkung der Semantik von Logikprogrammen
- Implikation wird ersetzt durch Äquivalenz
- Beispiel: $\mathrm{P} = \{\mathrm{p} \leftarrow \mathrm{q}; \mathrm{p} \leftarrow \mathrm{r}\}, \mathrm{Comp}(\mathrm{P}) = \{\mathrm{p} \leftrightarrow \mathrm{q} \vee \mathrm{r}\}$
- $\mathrm{Comp}(\mathrm{P})$ kann inkonsistent sein: $\mathrm{P} = \{\mathrm{p} \leftarrow \neg\mathrm{p}\}, \mathrm{Comp}(\mathrm{P}) = \{\mathrm{p} \leftrightarrow \neg\mathrm{p}\}$
- wenn P *call-consistent* ist, hat $\mathrm{Comp}(\mathrm{P})$ ein Herbrand-Modell

Rückblick: Programmvervollständigung

- *closed world assumption* (CWA): $\neg A$ wird angenommen, wenn A nicht bewiesen werden kann.
- i. A. unentscheidbar
- Vervollständigung ist Formalisierung der *closed world assumption*
- Verstärkung der Semantik von Logikprogrammen
- Implikation wird ersetzt durch Äquivalenz
- Beispiel: $\mathrm{P} = \{\mathrm{p} \leftarrow \mathrm{q}; \mathrm{p} \leftarrow \mathrm{r}\}, \mathrm{Comp}(\mathrm{P}) = \{\mathrm{p} \leftrightarrow \mathrm{q} \vee \mathrm{r}\}$
- $\mathrm{Comp}(\mathrm{P})$ kann inkonsistent sein: $\mathrm{P} = \{\mathrm{p} \leftarrow \neg\mathrm{p}\}, \mathrm{Comp}(\mathrm{P}) = \{\mathrm{p} \leftrightarrow \neg\mathrm{p}\}$
- wenn P *call-consistent* ist, hat $\mathrm{Comp}(\mathrm{P})$ ein Herbrand-Modell

Rückblick: Programmvervollständigung

- *closed world assumption* (CWA): $\neg A$ wird angenommen, wenn A nicht bewiesen werden kann.
- i. A. unentscheidbar
- Vervollständigung ist Formalisierung der *closed world assumption*
- Verstärkung der Semantik von Logikprogrammen
- Implikation wird ersetzt durch Äquivalenz
- Beispiel: $\mathrm{P} = \{\mathrm{p} \leftarrow \mathrm{q}; \mathrm{p} \leftarrow \mathrm{r}\}, \mathrm{Comp}(\mathrm{P}) = \{\mathrm{p} \leftrightarrow \mathrm{q} \vee \mathrm{r}\}$
- $\mathrm{Comp}(\mathrm{P})$ kann inkonsistent sein: $\mathrm{P} = \{\mathrm{p} \leftarrow \neg\mathrm{p}\}, \mathrm{Comp}(\mathrm{P}) = \{\mathrm{p} \leftrightarrow \neg\mathrm{p}\}$
- wenn P *call-consistent* ist, hat $\mathrm{Comp}(\mathrm{P})$ ein Herbrand-Modell

Rückblick: Programmvervollständigung

- *closed world assumption* (CWA): $\neg A$ wird angenommen, wenn A nicht bewiesen werden kann.
- i. A. unentscheidbar
- Vervollständigung ist Formalisierung der *closed world assumption*
- Verstärkung der Semantik von Logikprogrammen
- Implikation wird ersetzt durch Äquivalenz
- Beispiel: $\mathrm{P} = \{\mathrm{p} \leftarrow \mathrm{q}; \mathrm{p} \leftarrow \mathrm{r}\}, \mathrm{Comp}(\mathrm{P}) = \{\mathrm{p} \leftrightarrow \mathrm{q} \vee \mathrm{r}\}$
- $\mathrm{Comp}(\mathrm{P})$ kann inkonsistent sein: $\mathrm{P} = \{\mathrm{p} \leftarrow \neg\mathrm{p}\}, \mathrm{Comp}(\mathrm{P}) = \{\mathrm{p} \leftrightarrow \neg\mathrm{p}\}$
- wenn P *call-consistent* ist, hat $\mathrm{Comp}(\mathrm{P})$ ein Herbrand-Modell

Rückblick: Programmvervollständigung

- *closed world assumption* (CWA): $\neg A$ wird angenommen, wenn A nicht bewiesen werden kann.
- i. A. unentscheidbar
- Vervollständigung ist Formalisierung der *closed world assumption*
- Verstärkung der Semantik von Logikprogrammen
- Implikation wird ersetzt durch Äquivalenz
- Beispiel: $\mathrm{P} = \{\mathrm{p} \leftarrow \mathrm{q}; \mathrm{p} \leftarrow \mathrm{r}\}, \mathrm{Comp}(\mathrm{P}) = \{\mathrm{p} \leftrightarrow \mathrm{q} \vee \mathrm{r}\}$
- $\mathrm{Comp}(\mathrm{P})$ kann inkonsistent sein: $\mathrm{P} = \{\mathrm{p} \leftarrow \neg\mathrm{p}\}, \mathrm{Comp}(\mathrm{P}) = \{\mathrm{p} \leftrightarrow \neg\mathrm{p}\}$
- wenn P *call-consistent* ist, hat $\mathrm{Comp}(\mathrm{P})$ ein Herbrand-Modell

Rückblick: Programmvervollständigung

- *closed world assumption* (CWA): $\neg A$ wird angenommen, wenn A nicht bewiesen werden kann.
- i. A. unentscheidbar
- Vervollständigung ist Formalisierung der *closed world assumption*
- Verstärkung der Semantik von Logikprogrammen
- Implikation wird ersetzt durch Äquivalenz
- Beispiel: $\mathrm{P} = \{\mathrm{p} \leftarrow \mathrm{q}; \mathrm{p} \leftarrow \mathrm{r}\}, \mathrm{Comp}(\mathrm{P}) = \{\mathrm{p} \leftrightarrow \mathrm{q} \vee \mathrm{r}\}$
- $\mathrm{Comp}(\mathrm{P})$ kann inkonsistent sein: $\mathrm{P} = \{\mathrm{p} \leftarrow \neg\mathrm{p}\}, \mathrm{Comp}(\mathrm{P}) = \{\mathrm{p} \leftrightarrow \neg\mathrm{p}\}$
- wenn P *call-consistent* ist, hat $\mathrm{Comp}(\mathrm{P})$ ein Herbrand-Modell

Rückblick: Programmvervollständigung

- *closed world assumption* (CWA): $\neg A$ wird angenommen, wenn A nicht bewiesen werden kann.
- i. A. unentscheidbar
- Vervollständigung ist Formalisierung der *closed world assumption*
- Verstärkung der Semantik von Logikprogrammen
- Implikation wird ersetzt durch Äquivalenz
- Beispiel: $\mathrm{P} = \{\mathrm{p} \leftarrow \mathrm{q}; \mathrm{p} \leftarrow \mathrm{r}\}, \mathrm{Comp}(\mathrm{P}) = \{\mathrm{p} \leftrightarrow \mathrm{q} \vee \mathrm{r}\}$
- $\mathrm{Comp}(\mathrm{P})$ kann inkonsistent sein: $\mathrm{P} = \{\mathrm{p} \leftarrow \neg \mathrm{p}\}, \mathrm{Comp}(\mathrm{P}) = \{\mathrm{p} \leftrightarrow \neg \mathrm{p}\}$
- wenn P *call-consistent* ist, hat $\mathrm{Comp}(\mathrm{P})$ ein Herbrand-Modell

Rückblick: SLDNF

- endliche, berechenbare Alternative zur CWA: *negation by finite failure*
- nur variablenfreie negative Literale können ausgewählt werden
- Knoten im SLDNF-Baum, die keine positives oder variablenfreies negatives Literal enthalten, sind blockiert
- i. A. unentscheidbar, ob der SLDNF-Baum für ein Programm P und eine Frage F blockierte Knoten enthalten wird
- syntaktische Kriterien dafuer, dass keine blockierten Knoten vorkommen
- Beispiel: Jede Variable der Frage und einer Klausel kommt in einem positiven Literal vor

Rückblick: SLDNF

- endliche, berechenbare Alternative zur CWA: *negation by finite failure*
- nur variablenfreie negative Literale können ausgewählt werden
- Knoten im SLDNF-Baum, die keine positives oder variablenfreies negatives Literal enthalten, sind blockiert
- i. A. unentscheidbar, ob der SLDNF-Baum für ein Programm P und eine Frage F blockierte Knoten enthalten wird
- syntaktische Kriterien dafuer, dass keine blockierten Knoten vorkommen
- Beispiel: Jede Variable der Frage und einer Klausel kommt in einem positiven Literal vor

Rückblick: SLDNF

- endliche, berechenbare Alternative zur CWA: *negation by finite failure*
- nur variablenfreie negative Literale können ausgewählt werden
- Knoten im SLDNF-Baum, die keine positives oder variablenfreies negatives Literal enthalten, sind blockiert
- i. A. unentscheidbar, ob der SLDNF-Baum für ein Programm P und eine Frage F blockierte Knoten enthalten wird
- syntaktische Kriterien dafuer, dass keine blockierten Knoten vorkommen
- Beispiel: Jede Variable der Frage und einer Klausel kommt in einem positiven Literal vor

Rückblick: SLDNF

- endliche, berechenbare Alternative zur CWA: *negation by finite failure*
- nur variablenfreie negative Literale können ausgewählt werden
- Knoten im SLDNF-Baum, die keine positives oder variablenfreies negatives Literal enthalten, sind blockiert
- i. A. unentscheidbar, ob der SLDNF-Baum für ein Programm P und eine Frage F blockierte Knoten enthalten wird
- syntaktische Kriterien dafuer, dass keine blockierten Knoten vorkommen
- Beispiel: Jede Variable der Frage und einer Klausel kommt in einem positiven Literal vor

Rückblick: SLDNF

- endliche, berechenbare Alternative zur CWA: *negation by finite failure*
- nur variablenfreie negative Literale können ausgewählt werden
- Knoten im SLDNF-Baum, die keine positives oder variablenfreies negatives Literal enthalten, sind blockiert
- i. A. unentscheidbar, ob der SLDNF-Baum für ein Programm P und eine Frage F blockierte Knoten enthalten wird
- syntaktische Kriterien dafuer, dass keine blockierten Knoten vorkommen
- Beispiel: Jede Variable der Frage und einer Klausel kommt in einem positiven Literal vor

Rückblick: SLDNF

- endliche, berechenbare Alternative zur CWA: *negation by finite failure*
- nur variablenfreie negative Literale können ausgewählt werden
- Knoten im SLDNF-Baum, die keine positives oder variablenfreies negatives Literal enthalten, sind blockiert
- i. A. unentscheidbar, ob der SLDNF-Baum für ein Programm P und eine Frage F blockierte Knoten enthalten wird
- syntaktische Kriterien dafuer, dass keine blockierten Knoten vorkommen
- Beispiel: Jede Variable der Frage und einer Klausel kommt in einem positiven Literal vor

Rückblick: SLDNF

- endliche, berechenbare Alternative zur CWA: *negation by finite failure*
- nur variablenfreie negative Literale können ausgewählt werden
- Knoten im SLDNF-Baum, die keine positives oder variablenfreies negatives Literal enthalten, sind blockiert
- i. A. unentscheidbar, ob der SLDNF-Baum für ein Programm P und eine Frage F blockierte Knoten enthalten wird
- syntaktische Kriterien dafuer, dass keine blockierten Knoten vorkommen
- Beispiel: Jede Variable der Frage und einer Klausel kommt in einem positiven Literal vor

SLD-CNF: Konstruktive Negation

- SLDNF: Nur positive Literale können berechnete Antworten generieren
- Idee: Ersetze Substitution $\theta = \{x_1/t_1, \ldots, x_n/t_n\}$ durch Formel $\hat{\theta} = \exists \mathbf{y}[x_1 = t_1 \wedge \cdots \wedge x_n = t_n]$
- für berechnete Antworten $\theta_1, \ldots, \theta_l$ ist $F_F = \hat{\theta}_1 \vee \cdots \vee \hat{\theta}_l$
- $Comp(P) \models \bar{\forall}(F \leftrightarrow F_F)$ bzw. $Comp(P) \models \bar{\forall}(\neg F \leftrightarrow \neg F_F)$
- $\neg F_F$ entspricht berechneter Antwort
- $\neg F_F$ ist keine Menge von Substitutionen, kann also nicht einfach auf F angewandt werden
- erweitere Sprache um $=$ und $\neq$
- Algorithmus zur Normalisierung von $\neg F_F$

SLD-CNF: Konstruktive Negation

- SLDNF: Nur positive Literale können berechnete Antworten generieren
- Idee: Ersetze Substitution $\theta = \{x_1/t_1, \ldots, x_n/t_n\}$ durch Formel $\hat{\theta} = \exists \mathbf{y}[x_1 = t_1 \wedge \cdots \wedge x_n = t_n]$
- für berechnete Antworten $\theta_1, \ldots, \theta_l$ ist $F_F = \hat{\theta}_1 \vee \cdots \vee \hat{\theta}_l$
- $Comp(P) \models \bar{\forall}(F \leftrightarrow F_F)$ bzw. $Comp(P) \models \bar{\forall}(\neg F \leftrightarrow \neg F_F)$
- $\neg F_F$ entspricht berechneter Antwort
- $\neg F_F$ ist keine Menge von Substitutionen, kann also nicht einfach auf F angewandt werden
- erweitere Sprache um $=$ und $\neq$
- Algorithmus zur Normalisierung von $\neg F_F$

SLD-CNF: Konstruktive Negation

- SLDNF: Nur positive Literale können berechnete Antworten generieren
- Idee: Ersetze Substitution $\theta = \{x_1/t_1, \ldots, x_n/t_n\}$ durch Formel $\hat{\theta} = \exists \mathbf{y}[x_1 = t_1 \wedge \cdots \wedge x_n = t_n]$
- für berechnete Antworten $\theta_1, \ldots, \theta_l$ ist $F_F = \hat{\theta}_1 \vee \cdots \vee \hat{\theta}_l$
- $\mathrm{Comp}(P) \models \bar{\forall}(F \leftrightarrow F_F)$ bzw. $\mathrm{Comp}(P) \models \bar{\forall}(\neg F \leftrightarrow \neg F_F)$
- $\neg F_F$ entspricht berechneter Antwort
- $\neg F_F$ ist keine Menge von Substitutionen, kann also nicht einfach auf F angewandt werden
- erweitere Sprache um $=$ und $\neq$
- Algorithmus zur Normalisierung von $\neg F_F$

Back
Close

Dreiwertige Logik

- Werte 0 – falsch, $\frac{1}{2}$ – unbestimmt, 1 – wahr

$$
\begin{aligned}
\mu(\neg A) &= 1 - \mu(A) \\
\mu(A \wedge B) &= \min(\mu(A), \mu(B)) \\
\mu(A \vee B) &= \max(\mu(A), \mu(B)) \\
\mu(A \leftarrow B) &= \begin{cases} 1 & \text{wenn } \mu(A) \geq \mu(B) \\ 0 & \text{sonst} \end{cases} \\
\mu(A \leftrightarrow B) &= \begin{cases} 1 & \text{wenn } \mu(A) = \mu(B) \\ 0 & \text{sonst} \end{cases}
\end{aligned}
$$

Dreiwertige Herbrand-Interpretationen

- Dreiwertige Herbrand-Interpretation $\mathrm{I} = (\mathrm{I}^+, \mathrm{I}^-), \mathrm{I}^+, \mathrm{I}^- \subseteq \mathcal{B}_\mathrm{P}$
- I ist total, wenn $\mathrm{I}^+ \cup \mathrm{I}^- = \mathcal{B}_\mathrm{P}$
- I ist konsistent, wenn $\mathrm{I}^+ \cap \mathrm{I}^- = \emptyset$
- zweiwertige Interpretation I entspricht dreiwertiger $(\mathrm{I}, \mathcal{B}_\mathrm{P} \backslash \mathrm{I})$
- Informationsordnung: $\mathrm{I} \subseteq \mathrm{J}$ gdw. $\mathrm{I}^+ \subseteq \mathrm{J}^+$ und $\mathrm{I}^- \subseteq \mathrm{J}^-$
- $\subseteq$ ist eine induktive Ordnung auf der Menge der konsistenten dreiwertigen Herbrand-Interpretationen
- konsistente dreiwertige Herbrand-Interpretationen bilden damit einen Verband
- Wahrheit und Falschheit von Grundatomen verhalten sich monoton bzgl. $\subseteq$

Back
Close

Dreiwertige Herbrand-Interpretationen

- Dreiwertige Herbrand-Interpretation $\mathrm{I} = (\mathrm{I}^+, \mathrm{I}^-), \mathrm{I}^+, \mathrm{I}^- \subseteq \mathcal{B}_\mathrm{P}$
- I ist total, wenn $\mathrm{I}^+ \cup \mathrm{I}^- = \mathcal{B}_\mathrm{P}$
- I ist konsistent, wenn $\mathrm{I}^+ \cap \mathrm{I}^- = \emptyset$
- zweiwertige Interpretation I entspricht dreiwertiger $(\mathrm{I}, \mathcal{B}_\mathrm{P} \backslash \mathrm{I})$
- Informationsordnung: $\mathrm{I} \subseteq \mathrm{J}$ gdw. $\mathrm{I}^+ \subseteq \mathrm{J}^+$ und $\mathrm{I}^- \subseteq \mathrm{J}^-$
- $\subseteq$ ist eine induktive Ordnung auf der Menge der konsistenten dreiwertigen Herbrand-Interpretationen
- konsistente dreiwertige Herbrand-Interpretationen bilden damit einen Verband
- Wahrheit und Falschheit von Grundatomen verhalten sich monoton bzgl. $\subseteq$

Dreiwertige Herbrand-Interpretationen

- Dreiwertige Herbrand-Interpretation $\mathrm{I} = (\mathrm{I}^+, \mathrm{I}^-), \mathrm{I}^+, \mathrm{I}^- \subseteq \mathcal{B}_\mathrm{P}$
- I ist total, wenn $\mathrm{I}^+ \cup \mathrm{I}^- = \mathcal{B}_\mathrm{P}$
- I ist konsistent, wenn $\mathrm{I}^+ \cap \mathrm{I}^- = \emptyset$
- zweiwertige Interpretation I entspricht dreiwertiger $(\mathrm{I}, \mathcal{B}_\mathrm{P} \backslash \mathrm{I})$
- Informationsordnung: $\mathrm{I} \subseteq \mathrm{J}$ gdw. $\mathrm{I}^+ \subseteq \mathrm{J}^+$ und $\mathrm{I}^- \subseteq \mathrm{J}^-$
- $\subseteq$ ist eine induktive Ordnung auf der Menge der konsistenten dreiwertigen Herbrand-Interpretationen
- konsistente dreiwertige Herbrand-Interpretationen bilden damit einen Verband
- Wahrheit und Falschheit von Grundatomen verhalten sich monoton bzgl. $\subseteq$

Dreiwertige Herbrand-Interpretationen

- Dreiwertige Herbrand-Interpretation $\mathrm{I} = (\mathrm{I}^+, \mathrm{I}^-), \mathrm{I}^+, \mathrm{I}^- \subseteq \mathcal{B}_\mathrm{P}$
- I ist total, wenn $\mathrm{I}^+ \cup \mathrm{I}^- = \mathcal{B}_\mathrm{P}$
- I ist konsistent, wenn $\mathrm{I}^+ \cap \mathrm{I}^- = \emptyset$
- zweiwertige Interpretation I entspricht dreiwertiger $(\mathrm{I}, \mathcal{B}_\mathrm{P} \setminus \mathrm{I})$
- Informationsordnung: $\mathrm{I} \subseteq \mathrm{J}$ gdw. $\mathrm{I}^+ \subseteq \mathrm{J}^+$ und $\mathrm{I}^- \subseteq \mathrm{J}^-$
- $\subseteq$ ist eine induktive Ordnung auf der Menge der konsistenten dreiwertigen Herbrand-Interpretationen
- konsistente dreiwertige Herbrand-Interpretationen bilden damit einen Verband
- Wahrheit und Falschheit von Grundatomen verhalten sich monoton bzgl. $\subseteq$

Dreiwertige Herbrand-Interpretationen

- Dreiwertige Herbrand-Interpretation $\mathrm{I} = (\mathrm{I}^+, \mathrm{I}^-), \mathrm{I}^+, \mathrm{I}^- \subseteq \mathcal{B}_\mathrm{P}$
- I ist total, wenn $\mathrm{I}^+ \cup \mathrm{I}^- = \mathcal{B}_\mathrm{P}$
- I ist konsistent, wenn $\mathrm{I}^+ \cap \mathrm{I}^- = \emptyset$
- zweiwertige Interpretation I entspricht dreiwertiger $(\mathrm{I}, \mathcal{B}_\mathrm{P} \backslash \mathrm{I})$
- Informationsordnung: $\mathrm{I} \subseteq \mathrm{J}$ gdw. $\mathrm{I}^+ \subseteq \mathrm{J}^+$ und $\mathrm{I}^- \subseteq \mathrm{J}^-$
- $\subseteq$ ist eine induktive Ordnung auf der Menge der konsistenten dreiwertigen Herbrand-Interpretationen
- konsistente dreiwertige Herbrand-Interpretationen bilden damit einen Verband
- Wahrheit und Falschheit von Grundatomen verhalten sich monoton bzgl. $\subseteq$

Bottom-up-Charakterisierung von Herbrand-Modellen

- $H_{i+1}(P) = H_i(P) \cup$ Menge aller Grundatome $A = p(u_1, \ldots, u_n)$, für die es eine Grundinstanz $A \leftarrow A_1, \ldots, A_l$ einer Regel von P gibt, so daß $A_j \in H_i(P), j = 1, \ldots, l$
- Andere Formulierung: Operator $T_p(I) = \{A | A \leftarrow A_1, \ldots, A_l \in \text{ground}(P), I \models A_1, \ldots, A_l\}$
- H_{min} ist der kleinste Fixpunkt von T_P und kann mit endlich vielen Schritten von $T_P(\emptyset)$ erreicht werden.

Back
Close

Bottom-up-Charakterisierung von Herbrand-Modellen

- $H_{i+1}(P) = H_i(P) \cup$ Menge aller Grundatome $A = p(u_1, \dots, u_n)$, für die es eine Grundinstanz $A \leftarrow A_1, \dots, A_l$ einer Regel von P gibt, so daß $A_j \in H_i(P), j = 1, \dots, l$
- Andere Formulierung: Operator $T_p(I) = \{A | A \leftarrow A_1, \dots, A_l \in \text{ground}(P), I \models A_1, \dots, A_l\}$
- H_{min} ist der kleinste Fixpunkt von T_P und kann mit endlich vielen Schritten von $T_P(\emptyset)$ erreicht werden.

Back
Close

Bottom-up-Charakterisierung von Herbrand-Modellen

- $H_{i+1}(P) = H_i(P) \cup$ Menge aller Grundatome $A = p(u_1, \ldots, u_n)$, für die es eine Grundinstanz $A \leftarrow A_1, \ldots, A_l$ einer Regel von P gibt, so daß $A_j \in H_i(P), j = 1, \ldots, l$
- Andere Formulierung: Operator $T_p(I) = \{A | A \leftarrow A_1, \ldots, A_l \in \text{ground}(P), I \models A_1, \ldots, A_l\}$
- H_{min} ist der kleinste Fixpunkt von T_P und kann mit endlich vielen Schritten von $T_P(\emptyset)$ erreicht werden.

Bottom-up-Charakterisierung von Herbrand-Modellen

- $H_{i+1}(P) = H_i(P) \cup$ Menge aller Grundatome $A = p(u_1, \ldots, u_n)$, für die es eine Grundinstanz $A \leftarrow A_1, \ldots, A_l$ einer Regel von P gibt, so daß $A_j \in H_i(P), j = 1, \ldots, l$
- Andere Formulierung: Operator $T_p(I) = \{A | A \leftarrow A_1, \ldots, A_l \in \text{ground}(P), I \models A_1, \ldots, A_l\}$
- H_{min} ist der kleinste Fixpunkt von T_P und kann mit endlich vielen Schritten von $T_P(\emptyset)$ erreicht werden.

Analogon für dreiwertige Logik

$T3_P(I) = (T, F)$

$$T = \{A \mid \exists A_1, \ldots, A_l (A \leftarrow A_1, \ldots, A_l \in \text{ground}(P), \\ A_1, \ldots, A_l \text{ ist wahr in } I)\}$$

$$F = \{A \mid \exists A_1, \ldots, A_l (A \leftarrow A_1, \ldots, A_l \in \text{ground}(P), \\ A_1, \ldots, A_l \text{ impliziert, dass } A \text{ in } I \text{ falsch ist})\}$$

T_P und Vervollständigung

- für Herbrandinterpretationen gilt $I \models \mathrm{Comp}(P)$ gdw. $T_P(I) = I$
- wenn T_P einen Fixpunkt hat, ist dieser ein Modell fuer $\mathrm{Comp}(P)$
- bei dreiwertiger Logik gilt analog $I \models_3 \mathrm{Comp}(P)$ gdw. $T3_P(I) = I$
- $T3_P$ ist im Gegensatz zu T_P monoton
- Wenn I konsistent ist (z. B. $I = (\emptyset, \emptyset)$), so ist $T3_P(I)$ konsistent.
- $T3_P$ hat einen kleinsten Fixpunkt auf dem Verband der konsistenten dreiwertigen Herbrand-Interpretationen (Knaster-Tarski-Theorem)
- Dieser kleinste Fixpunkt von $T3_P$ ist ein konsistentes Modell von $\mathrm{Comp}(P)$
- Das bedeutet: $\mathrm{Comp}(P)$ kann nach zweiwertiger Logik inkonsistent sein, hat aber trotzdem ein konsistentes dreiwertiges Modell.

T_P und Vervollständigung

- für Herbrandinterpretationen gilt $I \models \mathrm{Comp}(P)$ gdw. $T_P(I) = I$
- wenn T_P einen Fixpunkt hat, ist dieser ein Modell fuer $\mathrm{Comp}(P)$
- bei dreiwertiger Logik gilt analog $I \models_3 \mathrm{Comp}(P)$ gdw. $T3_P(I) = I$
- $T3_P$ ist im Gegensatz zu T_P monoton
- Wenn I konsistent ist (z. B. $I = (\emptyset, \emptyset)$), so ist $T3_P(I)$ konsistent.
- $T3_P$ hat einen kleinsten Fixpunkt auf dem Verband der konsistenten dreiwertigen Herbrand-Interpretationen (Knaster-Tarski-Theorem)
- Dieser kleinste Fixpunkt von $T3_P$ ist ein konsistentes Modell von $\mathrm{Comp}(P)$
- Das bedeutet: $\mathrm{Comp}(P)$ kann nach zweiwertiger Logik inkonsistent sein, hat aber trotzdem ein konsistentes dreiwertiges Modell.

T_P und Vervollständigung

- für Herbrandinterpretationen gilt $I \models Comp(P)$ gdw. $T_P(I) = I$
- wenn T_P einen Fixpunkt hat, ist dieser ein Modell fuer $Comp(P)$
- bei dreiwertiger Logik gilt analog $I \models_3 Comp(P)$ gdw. $T3_P(I) = I$
- $T3_P$ ist im Gegensatz zu T_P monoton
- Wenn I konsistent ist (z. B. $I = (\emptyset, \emptyset)$), so ist $T3_P(I)$ konsistent.
- $T3_P$ hat einen kleinsten Fixpunkt auf dem Verband der konsistenten dreiwertigen Herbrand-Interpretationen (Knaster-Tarski-Theorem)
- Dieser kleinste Fixpunkt von $T3_P$ ist ein konsistentes Modell von $Comp(P)$
- Das bedeutet: $Comp(P)$ kann nach zweiwertiger Logik inkonsistent sein, hat aber trotzdem ein konsistentes dreiwertiges Modell.

T_P und Vervollständigung

- für Herbrandinterpretationen gilt $I \models \mathrm{Comp}(P)$ gdw. $T_P(I) = I$
- wenn T_P einen Fixpunkt hat, ist dieser ein Modell fuer $\mathrm{Comp}(P)$
- bei dreiwertiger Logik gilt analog $I \models_3 \mathrm{Comp}(P)$ gdw. $T3_P(I) = I$
- $T3_P$ ist im Gegensatz zu T_P monoton
- Wenn I konsistent ist (z. B. $I = (\emptyset, \emptyset)$), so ist $T3_P(I)$ konsistent.
- $T3_P$ hat einen kleinsten Fixpunkt auf dem Verband der konsistenten dreiwertigen Herbrand-Interpretationen (Knaster-Tarski-Theorem)
- Dieser kleinste Fixpunkt von $T3_P$ ist ein konsistentes Modell von $\mathrm{Comp}(P)$
- Das bedeutet: $\mathrm{Comp}(P)$ kann nach zweiwertiger Logik inkonsistent sein, hat aber trotzdem ein konsistentes dreiwertiges Modell.

Wohlfundierte Semantik

- *ein* dreiwertiges Modell $\mathrm{WFM}(\mathrm{P})$ statt mehreren zweiwertigen
- dreiwertige Logik zur Erzeugung nicht nötig.
- Idee: Manche Atome *müssen* wahr sein, unabhängig von der Semantik negativer Literale (Fakten), und manche Atome *müssen* falsch sein (weil sie nicht mit dem Kopf einer Klausel unfizieren.)
- benutze diese Information, um das Programm zu vereinfachen
- wenn der Wahrheitswert von allen Atomen so zu entscheiden ist, ist das Programm *effektiv stratifizierbar*
- wenn nicht: versuche nicht, den Wahrheitswert von Atomen zu raten, sondern gebe einfach ein dreiwertiges Modell zurück
- Nicht alle erwarteten Atome werden inferiert

Wohlfundierte Semantik

- *ein* dreiwertiges Modell WFM(P) statt mehreren zweiwertigen
- dreiwertige Logik zur Erzeugung nicht nötig.
- Idee: Manche Atome *müssen* wahr sein, unabhängig von der Semantik negativer Literale (Fakten), und manche Atome *müssen* falsch sein (weil sie nicht mit dem Kopf einer Klausel unfizieren.)
- benutze diese Information, um das Programm zu vereinfachen
- wenn der Wahrheitswert von allen Atomen so zu entscheiden ist, ist das Programm *effektiv stratifizierbar*
- wenn nicht: versuche nicht, den Wahrheitswert von Atomen zu raten, sondern gebe einfach ein dreiwertiges Modell zurück
- Nicht alle erwarteten Atome werden inferiert

Wohlfundierte Semantik

- *ein* dreiwertiges Modell WFM(P) statt mehreren zweiwertigen
- dreiwertige Logik zur Erzeugung nicht nötig.
- Idee: Manche Atome *müssen* wahr sein, unabhängig von der Semantik negativer Literale (Fakten), und manche Atome *müssen* falsch sein (weil sie nicht mit dem Kopf einer Klausel unfizieren.)
- benutze diese Information, um das Programm zu vereinfachen
- wenn der Wahrheitswert von allen Atomen so zu entscheiden ist, ist das Programm *effektiv stratifizierbar*
- wenn nicht: versuche nicht, den Wahrheitswert von Atomen zu raten, sondern gebe einfach ein dreiwertiges Modell zurück
- Nicht alle erwarteten Atome werden inferiert

Wohlfundierte Semantik

- *ein* dreiwertiges Modell WFM(P) statt mehreren zweiwertigen
- dreiwertige Logik zur Erzeugung nicht nötig.
- Idee: Manche Atome *müssen* wahr sein, unabhängig von der Semantik negativer Literale (Fakten), und manche Atome *müssen* falsch sein (weil sie nicht mit dem Kopf einer Klausel unfizieren.)
- benutze diese Information, um das Programm zu vereinfachen
- wenn der Wahrheitswert von allen Atomen so zu entscheiden ist, ist das Programm *effektiv stratifizierbar*
- wenn nicht: versuche nicht, den Wahrheitswert von Atomen zu raten, sondern gebe einfach ein dreiwertiges Modell zurück
- Nicht alle erwarteten Atome werden inferiert

Fixpunkt-Charakterisierung von WFM

- $I_3(P) = (H_{min}(P^+), \overline{H_{min}(P^-)})$
  P^+: P ohne Klauseln mit negativen Literalen
  P^-: P ohne negative Literale
- $\Phi_P(I) = I_3(P \backslash I)$
  $P \backslash I$ aus ground(P), indem man alle Klauseln mit Literalen löscht, die falsch sind in I, sowie alle Literale, die wahr sind in I
- $\Phi_P(I)$ ist monoton bzgl. der Informationsordnung $\subseteq$
- also existiert ein kleinster Fixpunkt von Φ_P
- kann von $(\emptyset, \emptyset)$ erreicht werden
- dieser kleinste Fixpunkt ist das *wohlfundierte Modell* WFM(P)

Fixpunkt-Charakterisierung von WFM

- $I_3(P) = (H_{min}(P^+), \overline{H_{min}(P^-)})$
  P^+: P ohne Klauseln mit negativen Literalen
  P^-: P ohne negative Literale
- $\Phi_P(I) = I_3(P \backslash I)$
  $P \backslash I$ aus ground(P), indem man alle Klauseln mit Literalen löscht, die falsch sind in I, sowie alle Literale, die wahr sind in I
- $\Phi_P(I)$ ist monoton bzgl. der Informationsordnung $\subseteq$
- also existiert ein kleinster Fixpunkt von Φ_P
- kann von $(\emptyset, \emptyset)$ erreicht werden
- dieser kleinste Fixpunkt ist das *wohlfundierte Modell* WFM(P)

Fixpunkt-Charakterisierung von WFM

- $I_3(P) = (H_{min}(P^+), \overline{H_{min}(P^-)})$
  P^+: P ohne Klauseln mit negativen Literalen
  P^-: P ohne negative Literale
- $\Phi_P(I) = I_3(P \backslash I)$
  $P \backslash I$ aus ground(P), indem man alle Klauseln mit Literalen löscht, die falsch sind in I, sowie alle Literale, die wahr sind in I
- $\Phi_P(I)$ ist monoton bzgl. der Informationsordnung $\subseteq$
- also existiert ein kleinster Fixpunkt von Φ_P
- kann von $(\emptyset, \emptyset)$ erreicht werden
- dieser kleinste Fixpunkt ist das *wohlfundierte Modell* WFM(P)

SLS-Resolution

- Resolution für stratifizierte Logikprogramme (*linear resolution for stratified clauses*)
- auch für normale Logikprogramme anwendbar
- Idee: ersetze Negation per *finite failure* durch Negation per *failure*
- damit nicht mehr berechenbar, Implementationen können nur approximieren
- Beispiel: Implementiere Zyklentest für unendliche missglückte Äste
- korrekt bzgl. *well-founded semantics*

SLS-Resolution

- Resolution für stratifizierte Logikprogramme (*linear resolution for stratified clauses*)
- auch für normale Logikprogramme anwendbar
- Idee: ersetze Negation per *finite failure* durch Negation per *failure*
- damit nicht mehr berechenbar, Implementationen können nur approximieren
- Beispiel: Implementiere Zyklentest für unendliche missglückte Äste
- korrekt bzgl. *well-founded semantics*

SLS-Resolution

- Resolution für stratifizierte Logikprogramme (*linear resolution for stratified clauses*)
- auch für normale Logikprogramme anwendbar
- Idee: ersetze Negation per *finite failure* durch Negation per *failure*
- damit nicht mehr berechenbar, Implementationen können nur approximieren
- Beispiel: Implementiere Zyklentest für unendliche missglückte Äste
- korrekt bzgl. *well-founded semantics*

SLS-Resolution

- Resolution für stratifizierte Logikprogramme (*linear resolution for stratified clauses*)
- auch für normale Logikprogramme anwendbar
- Idee: ersetze Negation per *finite failure* durch Negation per *failure*
- damit nicht mehr berechenbar, Implementationen können nur approximieren
- Beispiel: Implementiere Zyklentest für unendliche missglückte Äste
- korrekt bzgl. *well-founded semantics*

Back
Close

SLS-Resolution

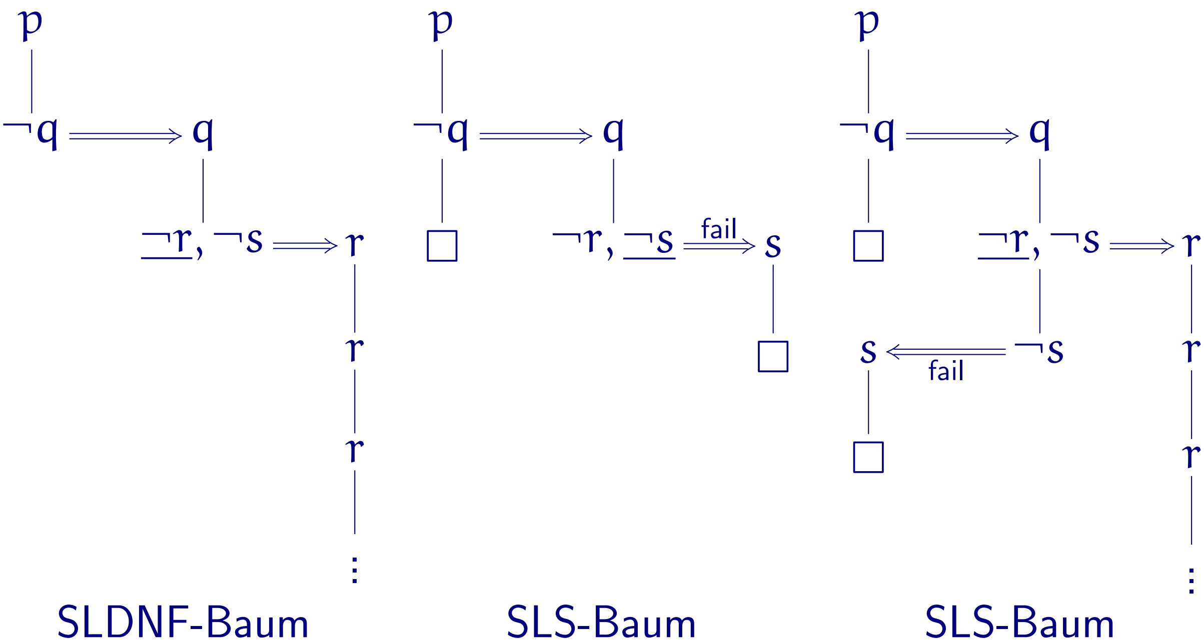
- Resolution für stratifizierte Logikprogramme (*linear resolution for stratified clauses*)
- auch für normale Logikprogramme anwendbar
- Idee: ersetze Negation per *finite failure* durch Negation per *failure*
- damit nicht mehr berechenbar, Implementationen können nur approximieren
- Beispiel: Implementiere Zyklentest für unendliche missglückte Äste
- korrekt bzgl. *well-founded semantics*

SLS-Resolution

- Resolution für stratifizierte Logikprogramme (*linear resolution for stratified clauses*)
- auch für normale Logikprogramme anwendbar
- Idee: ersetze Negation per *finite failure* durch Negation per *failure*
- damit nicht mehr berechenbar, Implementationen können nur approximieren
- Beispiel: Implementiere Zyklentest für unendliche missglückte Äste
- korrekt bzgl. *well-founded semantics*

SLS-Resolution: Beispiel

$P = \{p \leftarrow \neg q;\, q \leftarrow \neg r, \neg s;\, r \leftarrow r;\, s\}$



Schlussstrich

- SLDNF-Resolution ist nicht der Weisheit letzter Schluss
- Dreiwertige Logik führt zu konsequenteren Beschreibungsmöglichkeiten
- Logikprogrammierung mit Negation ist ein weites und sehr aktives Forschungsfeld. . .

Schlussstrich

- SLDNF-Resolution ist nicht der Weisheit letzter Schluss
- Dreiwertige Logik führt zu konsequenteren Beschreibungsmöglichkeiten
- Logikprogrammierung mit Negation ist ein weites und sehr aktives Forschungsfeld. . .

Schlussstrich

- SLDNF-Resolution ist nicht der Weisheit letzter Schluss
- Dreiwertige Logik führt zu konsequenteren Beschreibungsmöglichkeiten
- Logikprogrammierung mit Negation ist ein weites und sehr aktives Forschungsfeld…

Schlussstrich

- SLDNF-Resolution ist nicht der Weisheit letzter Schluss
- Dreiwertige Logik führt zu konsequenteren Beschreibungsmöglichkeiten
- Logikprogrammierung mit Negation ist ein weites und sehr aktives Forschungsfeld. . .

Schlussstrich

- SLDNF-Resolution ist nicht der Weisheit letzter Schluss
- Dreiwertige Logik führt zu konsequenteren Beschreibungsmöglichkeiten
- Logikprogrammierung mit Negation ist ein weites und sehr aktives Forschungsfeld...

Literatur

[1] Krysztof R. Apt, Roland N. Bol: *Logic Programming and Negation: A Survey.* In: Journal of Logic Programming 20, 9-71, 1994.