FOL-Decider
Grundlagen und MFOL

VL: Finitismus

PD Dr. Timm Lampert

Humboldt Universität Berlin

FOL-Decider

Ziel: Entscheidung der Widersprüchlichkeit von FOL-Formeln anhand syntaktischer Eigenschaften von FOLDNF.

Methode: Sat-Äquivalenzumformungen

Schritt 1: $\phi \Rightarrow \exists$M-optimierte-FOLDNF.

Schritt 2: endlicher Beweisbaum für Disjunkte D: $\exists$M-$\wedge$I-optimierte D'.

Unifikation

- Unter „Unifikation“ sei ganz allgemein das „gleich Machen“ von Variablen verstanden (keine Skolemisierung vorausgesetzt).
- $\sum 1$-Formeln sind PNF, denen nur Existenzquantoren voranstehen.
- $\sum 1$-Widersprüche sind $\sum 1$-Formeln mit einer DNF-Matrix, in der jedes Disjunkt ein Literal der Form A und ein Literal der Form $\neg A$ enthält.
- Um $\sum 1$-Widersprüche abzuleiten, sind Variablen zu unifizieren.

Maxindizierung

- Wir verwenden für allquantifizierte Variable x-Variablen und für existenzquantifizierte y-Variablen.
- Wir verwenden maxindizierte FOLDNF:

n Quantoren / Disjunkt $\Rightarrow$ n Variablen / Disjunkt

z.B.:

$$\forall y_1(Fy_1 \vee \forall z Gzy_1) \wedge \exists z(\neg Fz \wedge \exists y_1 \neg Gy_1 z \wedge Hz)$$

$$\Rightarrow$$

$$\forall x_1(Fx_1 \vee \forall x_2 Gx_2 x_1) \wedge \exists y_1(\neg Fy_1 \wedge \exists y_2 \neg Gy_2 y_1 \wedge Hy_1)$$

Beweis

- FOLDNF ist widersprüchlich gdw. jedes Disjunkt widersprüchlich ist.
- Ein Disjunkt ist widersprüchlich gdw. aus ihm ein $\sum 1$-Widerspruch abgeleitet werden kann.
- Um $\sum 1$-Widerspruch abzuleiten, sind Variable zu unifizieren: x/y.

$$\forall x_1(Fx_1 \vee \forall x_2 Gx_2x_1) \wedge \exists y_1(\neg Fy_1 \wedge \exists y_2 \neg Gy_2y_1 \wedge Hy_1)$$

$$\Rightarrow$$

$$\exists y_1 \exists y_2 \forall x_1(Fx_1 \wedge \neg Fy_1 \wedge \neg Gx_2x_1 \wedge Hy_1 \vee Gx_2x_1 \wedge \neg Fy_1 \wedge \neg Gy_2y_1 \wedge Hy_1)$$

$$\Rightarrow$$

$$\exists y_1 \exists y_2(\mathbf{Fy_1} \wedge \mathbf{\neg Fy_1} \wedge \neg Gy_2y_1 \wedge Hy_1 \vee \mathbf{Gy_2y_1} \wedge \neg Fy_1 \wedge \mathbf{\neg Gy_2y_1} \wedge Hy_1)$$

$\Rightarrow$ Definiere einen Algorithmus, um aus maxindizierten Disjunkten $\mathcal{D}$ von FOLDNF $\sum 1$-Widersprüche abzuleiten gdw. $\mathcal{D}$ widersprüchlich ist.

Methode

1. Berechne erst die *Möglichkeit* der Unifikation mittels sat-äquivalenter Umformung.
2. Verwende hierfür syntaktische Kriterien maxindizierter optimierter FOLDNF.
3. Gehe von „unten“ nach „oben“ vor:
    1. Notwendige Bedingung nicht erfüllt $\Rightarrow$ sat und ENDE
    2. Hinreichende Bedingung erfüllt $\Rightarrow$ False und ENDE.

Kalkülregeln

1. Logische Äquivalenzumformungen: DN, PE, DMG, KOM, ASS, DIS1, DIS2, IP1-4, $\wedge$**I**, PN1-10, SUB, $\neg\forall$Def., $\neg\exists$Def., $\forall$V, $\exists$V
2. sat.-Regel: A $\wedge$ *sat* $\text{-}|_s|\text{-}$ A; A $\vee$ *sat* $\text{-}|_s|\text{-}$ *sat* + Anwendung von sat-Kriterien.
3. $\forall$E: $\exists\mu\ldots\forall\nu\phi(\mu,\nu)$ $|\text{-}$ $\exists\mu\ldots\phi(\mu,\nu/\mu)$
   Diese Regel ist nur zwecks Unifikation im Falle eines Widerspruchsbeweises am Ende des Algorithmus auf optimale PNF anzuwenden.
4. $\exists$M: $\exists\mu(\phi(\mu) \wedge \psi(\mu))$ $|\text{-}$ $\exists\mu_1\exists\mu_2(\phi(\mu_1) \wedge \psi(\mu_2))$
   Diese Regel ist nur auf NNF anzuwenden und wird nur dann angewendet, wenn die Umformung sat-äquivalent ist.

Sat-Äquivalenz

Es werden nur sat-Äquivalenzumformungen zugelassen.

Es steht in Frage, ob ein Algorithmus definiert werden kann, der korrekte sat-Kriterien verwendet, die stark genug sind, damit der Algorithmus auch im sat-Falle terminiert.

Methode „von unten“: Definiere mittels syntaktischer Umformungen möglichst starke sat-Kriterien.

Zeige, dass die Umformungen sat-äquivalent und die sat-Kriterien, die Beweispfade abbrechen, korrekt sind.

Können auf diese Weise im sat-Fall alle Beweispfade abgebrochen werden?

Das logische Problem der Entscheidbarkeit

Brünnler, Probst, Studer, *On Contraction and the Modal Fragment*, S. 1:

From a proof-theoretic perspective, one could say that the rule of contraction is the reason for the undecidability of first-order logic. Contraction basically states that whenever we have derived a premise $A \vee A$, then we are allowed to infer A. Such a structural rule leads to undecidability since in a proof search procedure the following may happen: Assume we have to show that a formula A holds. This formula may have been derived by the rule of contraction. Thus we have to show that $A \vee A$ holds. Again this formula may have been derived by contraction. Now we have to show that $(A \vee A) \vee (A \vee A)$ holds and so on ad infinitum.

∧I-Erweiterung

- Im Baumkalkül stellt sich das Problem in Form der Notwendigkeit, neue Variable einzuführen, so dass Ausdrücke ggf. erst nach mehrmaliger Dekomposition als widersprüchlich erwiesen werden können.
- In dem hier verwendeten Kalkül stellt sich das Problem in Form der Notwendigkeit, mittels ∧I allquantifizierte Ausdrücke zu multiplizieren, um alle notwendigen Unifikationen mittels eindeutiger Anwendung von ∀E vornehmen zu können.

Baumkalkül

- NNF vorausgesetzt

Regel	$\wedge$-R.	$\vee$-R.	$\exists$-R.	$\forall$-R.
Prämisse	$A \wedge B$ $\surd$	$A \vee B$ $\surd$	$\exists v\varphi(v)$ $\surd$	$\forall v\varphi(v)$
Konklusion	A B	$/ \quad \backslash$ $A \quad B$	$\varphi(t)$	$\varphi(t)$

t muss in $\exists$-R. eine neue freie Variable sein.

Mehrmalige Dekomposition

1	$\forall x \exists y (Fx \wedge \neg Fy)$	
2	$\exists y (Fa \wedge \neg Fy)$	1∀-R.
3	$Fa \wedge \neg Fb$	2∃-R.
4	Fa	3∧-R.
5	$\underline{\neg Fb}$	3∧-R.
6	$\exists y (Fb \wedge \neg Fy)$	1∀-R.
7	$Fb \wedge \neg Fc$	6∃-R.
8	$\underline{Fb}$	7∧-R.
	x	

Dekompositionsreduktion

$\forall x \exists y(Fx \wedge \neg Fy)$ -||- $\forall x Fx \wedge \exists y \neg Fy$

1	$\forall x Fx \wedge \exists y \neg Fy$	
2	$\forall x Fx$	$1\wedge$-R.
3	$\exists y \neg Fy$	$1\wedge$-R.
4	$\underline{\neg Fa}$	$3\exists$-R.
5	$\underline{Fa}$	$2\forall$-R.
	x	

Beweis im FOL-Decider

```
Timing[decide[A[x, E[y, F[x] && ! F[y]]]]]

------------------------------

new expression: ∀x ∃y (F(x) ∧ ¬ F(y))

em-optimized sat-equivalent FOLDNF:

∃y(1) ¬ F(y(1)) ∧ ∀x(1) F(x(1))

proof found: False

final sat-equivalent expression:

∃y(1) ¬ F(y(1)) ∧ ∀x(1) F(x(1))

final kpairs:

( F(x(1)) ¬ F(y(1)) )

final sublist:

{{x[1], y[1]}}

final prefix:

{y[1], x[1]}

total number of iteration steps: 0

{0.156, False}
```

Optimierte PNF

- Optimierte PNF werden aus antipränexen Disjunkten $\mathcal{D}$ gebildet, indem Existenzquantoren möglichst links von Allquantoren stehen. Hierdurch wird die Anwendbarkeit von $\forall$E maximiert.
- Optimale PNF besitzen Pränexe derart, dass alle $\forall\nu$ rechts von $\exists\mu$, wenn ν/μ zwecks Unifikation zu ersetzen ist, um ein $\sum 1$-Widerspruch abzuleiten.
- Wir geben optimale Pränexe in Form von xy-Listen aus.

Konnektierte Literale

- Zwei Literale von $\mathcal{D}$ sind konnektiert, wenn
  1. das eine negiert ist, das andere nicht negiert ist,
  2. beide denselben Prädikatbuchstaben mit derselben Anzahl Argumentstellen enthalten,
  3. Nicht an derselben Position unterschiedliche *y*-Variable vorkommen.

Zwei konnektierte Literale bilden ein k-Paar. k-Paare sind potentielle Widerspruchspaare.

Minimale k-Paarmengen

- Eine minimale k-Paarmenge ist eine Menge $\mathcal{M}$ an k-Paaren einer DNF-Matrix einer optimierten PNF, derart dass
  1. Jedes Disjunkt der DNF-Matrix mindestens ein k-Paar enthält, und
  2. Kein k-Paar aus $\mathcal{M}$ gelöscht werden kann, ohne dass Bedingung 1 nicht mehr erfüllt ist.

Minimale k-Paarmenge sind minimale Mengen, deren Unifikation für eine Widerspruchsbeweis hinreichen.

Beweis im FOL-Decider

1. $\phi \Rightarrow \exists$M-optimierte, sat-äquivalente FOLDNF
2. Für jedes Disjunkt $\mathcal{D}$:
   1. Minimale, maxindizierte, sat-äquivalente, anti-pränexe $\mathcal{D}'$,
   2. Minimale k-Paarmenge,
   3. Eindeutige Substitutionsliste („Unifikator“),
   4. Optimales Pränex.

∧I-Notwendigkeit

```
Timing[decide[didiformel]]

------------------------------
```

new expression:

$$\exists_{y(1)} \left(\exists_{y(2)} \left(\exists_{y(3)} \left(g(y(1), y(3)) \wedge g(y(3), y(2))\right) \wedge \neg f(y(2))\right) \bigwedge f(y(1))\right) \bigwedge$$
$$\forall_{x(1)} \left(\forall_{x(2)} \left(f(x(2)) \vee \neg g(x(1), x(2))\right) \vee \neg f(x(1))\right)$$

em-optimized sat-equivalent FOLDNF:

$$\exists_{y(1)} \left(\exists_{y(2)} \left(\exists_{y(3)} \left(g(y(1), y(3)) \wedge g(y(3), y(2))\right) \wedge \neg f(y(2))\right) \bigwedge f(y(1))\right) \bigwedge$$
$$\forall_{x(1)} \left(\forall_{x(2)} \left(f(x(2)) \vee \neg g(x(1), x(2))\right) \vee \neg f(x(1))\right)$$

proof found: False

final sat-equivalent expression:

$$\exists_{y(1)} \left(\exists_{y(2)} \left(\exists_{y(3)} \left(g(y(1), y(3)) \wedge g(y(3), y(2))\right) \wedge \neg f(y(2))\right) \bigwedge f(y(1))\right) \bigwedge$$
$$\forall_{x(1,1)} \left(\forall_{x(2,1)} \left(f(x(2, 1)) \vee \neg g(x(1, 1), x(2, 1))\right) \vee \neg f(x(1, 1))\right) \bigwedge$$
$$\forall_{x(1,2)} \left(\forall_{x(2,2)} \left(f(x(2, 2)) \vee \neg g(x(1, 2), x(2, 2))\right) \vee \neg f(x(1, 2))\right)$$

final kpairs:

$$\begin{pmatrix} f(x(2, 1)) & \neg f(x(1, 2)) \\ f(y(1)) & \neg f(x(1, 1)) \\ f(x(2, 2)) & \neg f(y(2)) \\ g(y(1), y(3)) & \neg g(x(1, 1), x(2, 1)) \\ g(y(3), y(2)) & \neg g(x(1, 2), x(2, 2)) \end{pmatrix}$$

final sublist:

```
{{x[1, 1], y[1]}, {x[1, 2], y[3]}, {x[2, 1], y[3]}, {x[2, 2], y[2]}}
```

final prefix:

```
{y[1], y[2], y[3], x[1, 1], x[1, 2], x[2, 1], x[2, 2]}
```

total number of iteration steps: 1

```
{0.437, False}
```

MFOLDNF: $\wedge$I-freies Fragment

- Sei ϕ eine Formel der monadischen FOL (MFOL), dann ist jedes Disjunkt der FOLDNF von ϕ eine Konjunktion aus Ausdrücken der Form $\forall\nu A(\nu)$ ($A(\nu)$ = Disjunktion von Literalen, die nur ν als Variable enthalten) sowie der Form $\exists\mu A(\mu)$ ($A(\mu)$ = Konjunktion von Literalen, die nur μ als Variable enthalten).
- Derartige Ausdrücke sind ein Fragment des sogenannten „modal fragment“ von FOL, für das gilt: Wenn eine Formel beweisbar ist, dann ohne Kontraktion bzw. ohne blocking oder $\wedge$I-Anwendungen (vgl. Brünnler u.a.).

Algorithmus MFOL

1. ϕ von MFOL $\Rightarrow$ FOLDNF.
2. Gehe alle $\mathcal{D}$ durch:
   1. Bilde eine optimierte PNF, in der alle $\exists$ links von $\forall$ stehen.
   2. Bilde alle minimale k-Paarmengen $\mathcal{M}$ und gehe diese durch:
      1. Ist $\mathcal{M}$ leer, dann ist $\mathcal{D}$ und damit FOLDNF und damit ϕ sat.
      2. Können alle k-Paare aus *einer* $\mathcal{M}$ mittels $\forall$E unifiziert werden, dann ist $\mathcal{D}$ False und es kann zum nächsten $\mathcal{D}$ übergegangen werden.
      3. Können in *allen* k-Paarmengen $\mathcal{M}$ nicht alle k-Paare mittels $\forall$E unifiziert werden, dann ist $\mathcal{D}$ und damit FOLDNF und damit ϕ sat.
   3. Sind alle $\mathcal{D}$ False, dann ist FOLDNF und damit ϕ False.

Beispiel 1: 1. Schritt

```
Timing[decide[bruennler]]

------------------------------
```

new expression:

$$\exists_y P(y) \wedge \forall_x ((\exists_y Q(y) \wedge \neg R(x)) \vee \neg P(x)) \wedge \forall_x (\exists_y (P(y) \wedge R(y)) \vee \neg Q(x))$$

em-optimized sat-equivalent FOLDNF:

$$(\exists_{y(1)} (P(y(1)) \wedge R(y(1))) \wedge \forall_{x(1)} (\neg P(x(1)) \vee \neg R(x(1)))) \vee$$
$$(\exists_{y(1)} P(y(1)) \wedge \forall_{x(1)} \neg P(x(1))) \vee$$
$$(\exists_{y(1)} Q(y(1)) \wedge \forall_{x(1)} \neg Q(x(1)))$$

Beispiel 1: 1. Disjunkt

disjunct no. 1 to evaluate:

$$\exists_{y(1)} \, (P(y(1)) \wedge R(y(1))) \wedge \forall_{x(1)} \, (\neg P(x(1)) \vee \neg R(x(1)))$$

proof found: False

final sat-equivalent expression:

$$\exists_{y(1)} \, (P(y(1)) \wedge R(y(1))) \wedge \forall_{x(1)} \, (\neg P(x(1)) \vee \neg R(x(1)))$$

final kpairs:

$$\begin{pmatrix} P(y(1)) & \neg P(x(1)) \\ R(y(1)) & \neg R(x(1)) \end{pmatrix}$$

final sublist:

{{x[1], y[1]}}

final prefix:

{y[1], x[1]}

Beispiel 1: 2. Disjunkt

disjunct no. 2 to evaluate:

$\exists_{y(1)}\ P(y(1)) \wedge \forall_{x(1)} \neg P(x(1))$

proof found: False

final sat-equivalent expression:

$\exists_{y(1)}\ P(y(1)) \wedge \forall_{x(1)} \neg P(x(1))$

final kpairs:

$(\ P(y(1))\ \neg P(x(1))\ )$

final sublist:

{{x[1], y[1]}}

final prefix:

{y[1], x[1]}

Beispiel 1: 3. Disjunkt

disjunct no. 3 to evaluate:

$\exists_{y(1)}\, Q(y(1)) \wedge \forall_{x(1)} \neg Q(x(1))$

proof found: False

final sat-equivalent expression:

$\exists_{y(1)}\, Q(y(1)) \wedge \forall_{x(1)} \neg Q(x(1))$

final kpairs:

$(\, Q(y(1)) \;\; \neg Q(x(1)) \,)$

final sublist:

{{x[1], y[1]}}

final prefix:

{y[1], x[1]}

Beispiel 2: 1. Schritt

new expression: $\forall_{x(1)} \exists_{y(1)} \exists_{y(2)} (F(y(1)) \wedge G(y(2)) \wedge (\neg F(x(1)) \vee \neg G(x(1))))$

em-optimized sat-equivalent FOLDNF:

$\exists_{y(1)} F(y(1)) \wedge \exists_{y(2)} G(y(2)) \wedge \forall_{x(1)} (\neg F(x(1)) \vee \neg G(x(1)))$

Beispiel 2: 2. Schritt

$$\mathcal{M}: \begin{pmatrix} F(y(1)) & \neg F(x(1)) \\ G(y(2)) & \neg G(x(1)) \end{pmatrix}$$

$\wedge$I:

$$\exists_{y(1)}\ F(y(1)) \bigwedge \exists_{y(2)}\ G(y(2)) \bigwedge$$
$$\forall_{x(1,1)}\ (\neg F(x(1,1)) \vee \neg G(x(1,1))) \bigwedge$$
$$\forall_{x(1,2)}\ (\neg F(x(1,2)) \vee \neg G(x(1,2)))$$

$\mathcal{M}'$:

$$\begin{pmatrix} F(y(1)) & \neg F(x(1,1)) \\ G(y(2)) & \neg G(x(1,2)) \end{pmatrix}$$

Erweiterung notwendig:

$$\begin{pmatrix} F(y(1)) & \neg F(x(1,1)) \\ G(y(2)) & \neg G(x(1,2)) \end{pmatrix} \quad \text{o.} \quad \begin{pmatrix} F(y(1)) & \neg F(x(1,1)) \\ G(y(2)) & \neg G(x(1,2)) \end{pmatrix}$$
$$G(y(2)) \quad \neg G(x(1,1)) \qquad\qquad F(y(1)) \quad \neg F(x(1,2))$$

Keine minimale Erweiterung möglich!

Kernfrage

- Kann die Anwendung von $\wedge$I *in jedem Fall* beschränkt werden, um zu entscheiden, ob aus einem Disjunkt $\mathcal{D}$ einer FOLDNF ein $\sum$1-Widerspruch abgeleitet werden kann?

Vorgehen: Berechnung der *minimal notwendigen* $\wedge$I-Anwendungen vor Unifikation mittels $\forall$E.

Dies setzt den syntaktischen Rahmen von FOLDNF voraus.