Rechnerübung zu Theorembeweiser und ihre Anwendungen

Prof. Dr.-Ing. Gregor Snelting
Dipl.-Inf. Univ. Daniel Wasserrab

Lehrstuhl Programmierparadigmen
IPD Snelting
Universität Karlsruhe (TH)

Teil VI

Kombination von Regeln

Variablen in Regeln spezifizieren mittels `of`

Manchmal nötig, dass Variablen vor Regelanwendung festgelegt (z.B. Isabelle kann passende Terme nicht inferieren), dann:

- eckige Klammer hinter Regelnamen
- Schlüsselwort `of`, danach einer oder mehrere Terme
- müssen natürlich zu Typ der Variable passen
- Reihenfolge wie erstes Auftreten in Regel
- _ für Variablen, die man nicht instantiieren möchte

Beispiel:

```
iffE:          ⟦?P = ?Q; ⟦?P ⟶ ?Q; ?Q ⟶ ?P⟧ ⟹ ?R⟧ ⟹ ?R
iffE[of X]:    ⟦X = ?Q; ⟦X ⟶ ?Q; ?Q ⟶ X⟧ ⟹ ?R⟧ ⟹ ?R
iffE[of X Y]:  ⟦X = Y; ⟦X ⟶ Y; Y ⟶ X⟧ ⟹ ?R⟧ ⟹ ?R
iffE[of X Y Z]: ⟦X = Y; ⟦X ⟶ Y; Y ⟶ X⟧ ⟹ Z⟧ ⟹ Z
```

Variablen in Regeln spezifizieren mittels `of`

Manchmal nötig, dass Variablen vor Regelanwendung festgelegt (z.B. Isabelle kann passende Terme nicht inferieren), dann:

- eckige Klammer hinter Regelnamen
- Schlüsselwort `of`, danach einer oder mehrere Terme
- müssen natürlich zu Typ der Variable passen
- Reihenfolge wie erstes Auftreten in Regel
- _ für Variablen, die man nicht instantiieren möchte

Beispiel:

`iffE:`	$[\![?P = ?Q;\ [\![?P \longrightarrow ?Q;\ ?Q \longrightarrow ?P]\!] \Longrightarrow ?R]\!] \Longrightarrow ?R$
`iffE[of X]:`	$[\![X = ?Q;\ [\![X \longrightarrow ?Q;\ ?Q \longrightarrow X]\!] \Longrightarrow ?R]\!] \Longrightarrow ?R$
`iffE[of X Y]:`	$[\![X = Y;\ [\![X \longrightarrow Y;\ Y \longrightarrow X]\!] \Longrightarrow ?R]\!] \Longrightarrow ?R$
`iffE[of X Y Z]:`	$[\![X = Y;\ [\![X \longrightarrow Y;\ Y \longrightarrow X]\!] \Longrightarrow Z]\!] \Longrightarrow Z$

Prämissen in Regeln spezifizieren mittels *OF*

Analog: Ganze Prämissen instantiieren

- ebenso eckige Klammer,
- Schlüsselwort *OF*, danach Regelname
- Konklusion der Regel und entspr. Prämisse müssen unifizieren
- entspr. Prämisse mit Prämissen der eingefügten Regel ersetzt
- bei mehreren *OF*s: erst linkeste Prämisse, dann die rechts davon usw.
- auch hier _ für Überspringen von Prämissen
- vor allem bei Induktionshypothesen einsetzbar

Beispiel:

```
conjI:                 ⟦?P; ?Q⟧ ⟹ ?P ∧ ?Q
ccontr:                (¬ ?P ⟹ False) ⟹ ?P
conjI[OF ccontr]:      ⟦¬ ?P ⟹ False; ?Q⟧ ⟹ ?P ∧ ?Q
conjI[OF ccontr,of X]: ⟦¬ X ⟹ False; ?Q⟧ ⟹ X ∧ ?Q
```

Prämissen in Regeln spezifizieren mittels *OF*

Analog: Ganze Prämissen instantiieren

- ebenso eckige Klammer,
- Schlüsselwort *OF*, danach Regelname
- Konklusion der Regel und entspr. Prämisse müssen unifizieren
- entspr. Prämisse mit Prämissen der eingefügten Regel ersetzt
- bei mehreren *OF*s: erst linkeste Prämisse, dann die rechts davon usw.
- auch hier _ für Überspringen von Prämissen
- vor allem bei Induktionshypothesen einsetzbar

Beispiel:

conjI:	$[\![?P;\ ?Q]\!] \Longrightarrow ?P \wedge ?Q$
ccontr:	$(\neg\ ?P \Longrightarrow False) \Longrightarrow ?P$
conjI[OF ccontr]:	$[\![\neg\ ?P \Longrightarrow False;\ ?Q]\!] \Longrightarrow ?P \wedge ?Q$
conjI[OF ccontr,of X]:	$[\![\neg\ X \Longrightarrow False;\ ?Q]\!] \Longrightarrow X \wedge ?Q$

Regeln kombinieren mittels *THEN*

statt zwei Regel nacheinander auszuführen, Kombination dieser Regeln

- Konklusion der ersten passt auf eine Prämisse der zweiten Regel
- Variablen entsprechend zweiter Regel unifiziert
- erster Regelname gefolgt von eckiger Klammer
- dann Schlüsselwort *THEN* gefolgt von zweitem Regelnamen
- gut einsetzbar, falls Isabelle bei Substitution scheitert

Beispiel:

```
iffI:                    ⟦?P ⟹ ?Q; ?Q ⟹ ?P⟧ ⟹ ?P = ?Q
sym:                     ?s = ?t ⟹ ?t = ?s
mp:                      ⟦?P ⟶ ?Q; ?P⟧ ⟹ ?Q
iffI[THEN sym]:          ⟦?s ⟹ ?t; ?t ⟹ ?s⟧ ⟹ ?t = ?s
iffI[THEN sym,of P Q]:   ⟦P ⟹ Q; Q ⟹ P⟧ ⟹ Q = P
iffI[THEN sym,OF mp,of P R Q]:
                         ⟦P ⟹ R ⟶ Q; P ⟹ R; Q ⟹ P⟧ ⟹ Q = P
```

Regeln kombinieren mittels *THEN*

statt zwei Regel nacheinander auszuführen, Kombination dieser Regeln

- Konklusion der ersten passt auf eine Prämisse der zweiten Regel
- Variablen entsprechend zweiter Regel unifiziert
- erster Regelname gefolgt von eckiger Klammer
- dann Schlüsselwort *THEN* gefolgt von zweitem Regelnamen
- gut einsetzbar, falls Isabelle bei Substitution scheitert

Beispiel:

iffI: $\llbracket ?P \Longrightarrow ?Q;\ ?Q \Longrightarrow ?P \rrbracket \Longrightarrow ?P = ?Q$

sym: $?s = ?t \Longrightarrow ?t = ?s$

mp: $\llbracket ?P \longrightarrow ?Q;\ ?P \rrbracket \Longrightarrow ?Q$

iffI[THEN sym]: $\llbracket ?s \Longrightarrow ?t;\ ?t \Longrightarrow ?s \rrbracket \Longrightarrow ?t = ?s$

iffI[THEN sym,of P Q]: $\llbracket P \Longrightarrow Q;\ Q \Longrightarrow P \rrbracket \Longrightarrow Q = P$

iffI[THEN sym,OF mp,of P R Q]:

$$\llbracket P \Longrightarrow R \longrightarrow Q;\ P \Longrightarrow R;\ Q \Longrightarrow P \rrbracket \Longrightarrow Q = P$$