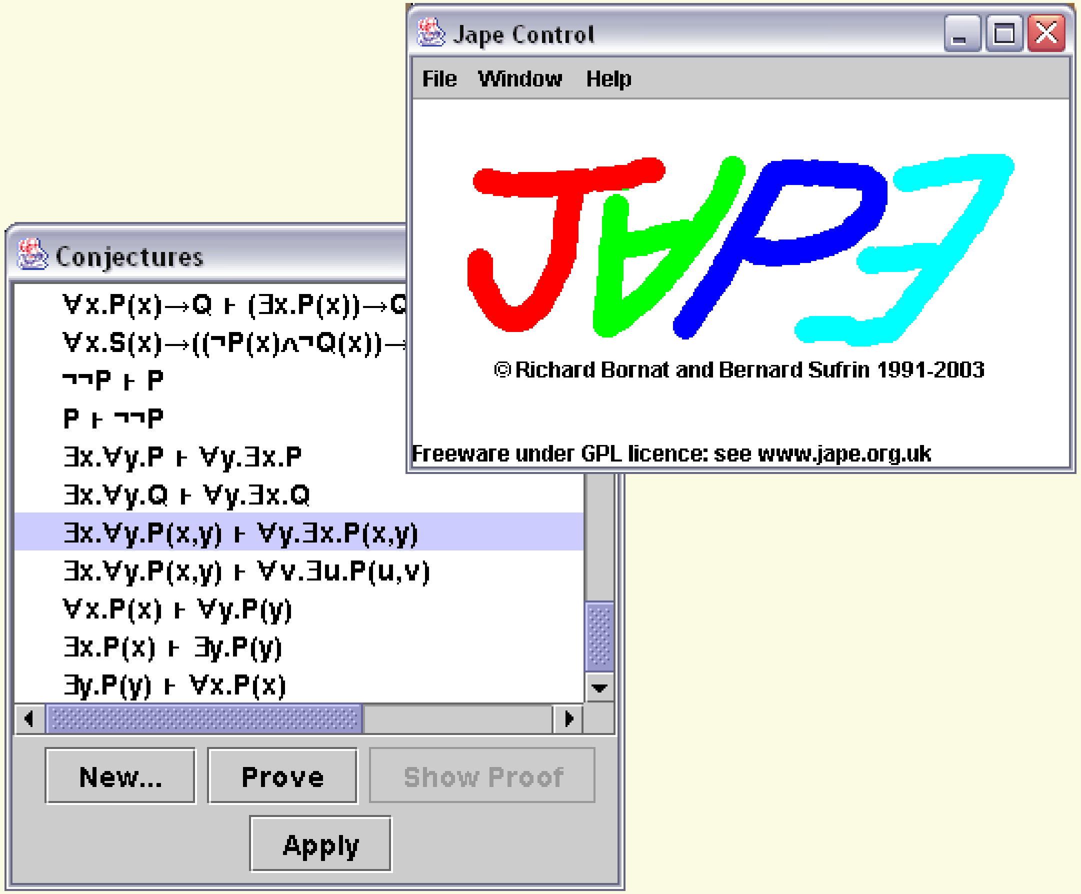
Jape

Just another proof editor

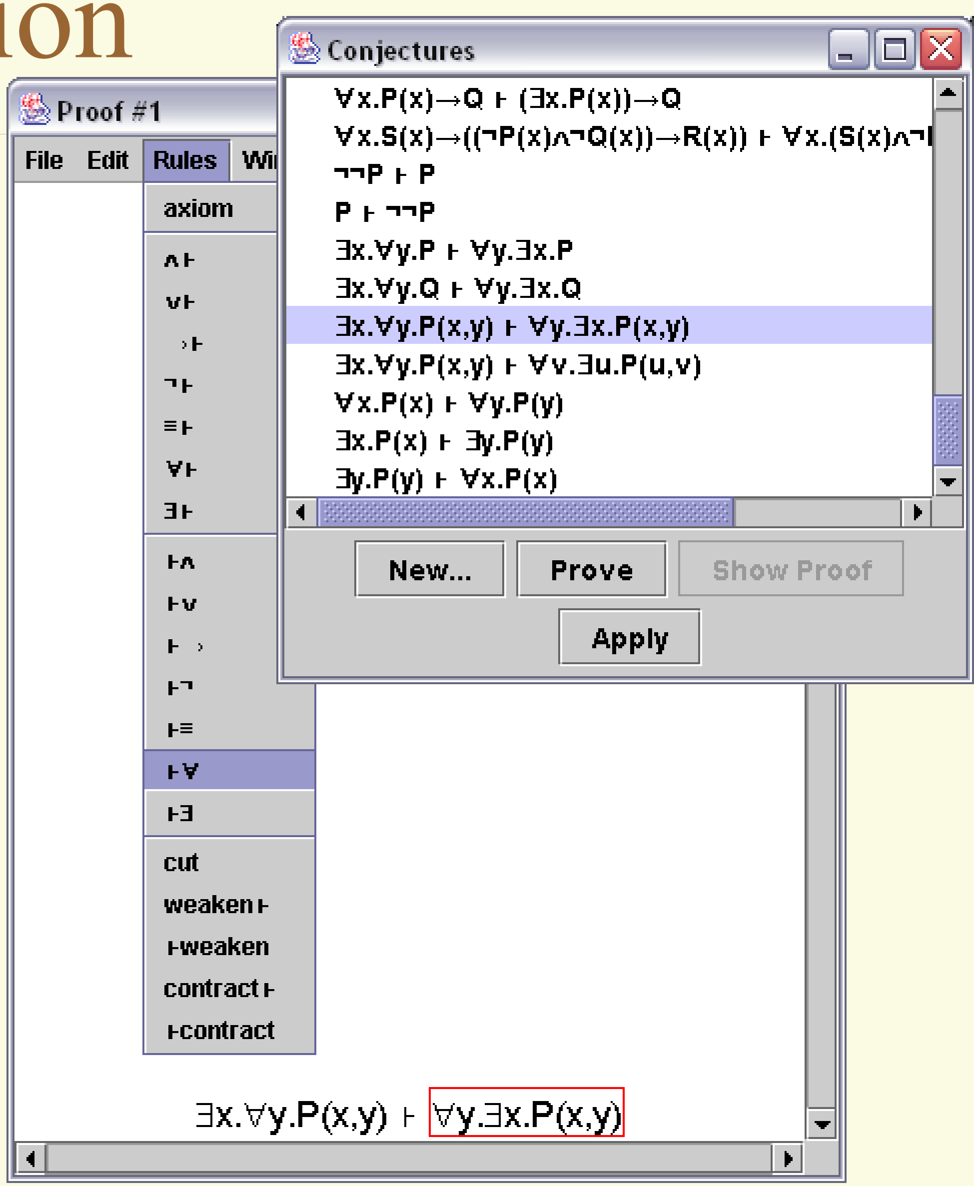
Jape – Just another proof editor

- Jape ist ein System zur interaktiven Erstellung von Beweisen
- Jape erlaubt
  - selber Logiken zu erfinden
  - Beweisregeln einzuführen
  - interaktiv Beweise zu führen
- Beweisregeln können durch
  - Mouseclicks
  - Gestures (nur unter Apple)
  - ausgeführt werden
- Beweise können durch Unifikation erleichtert werden

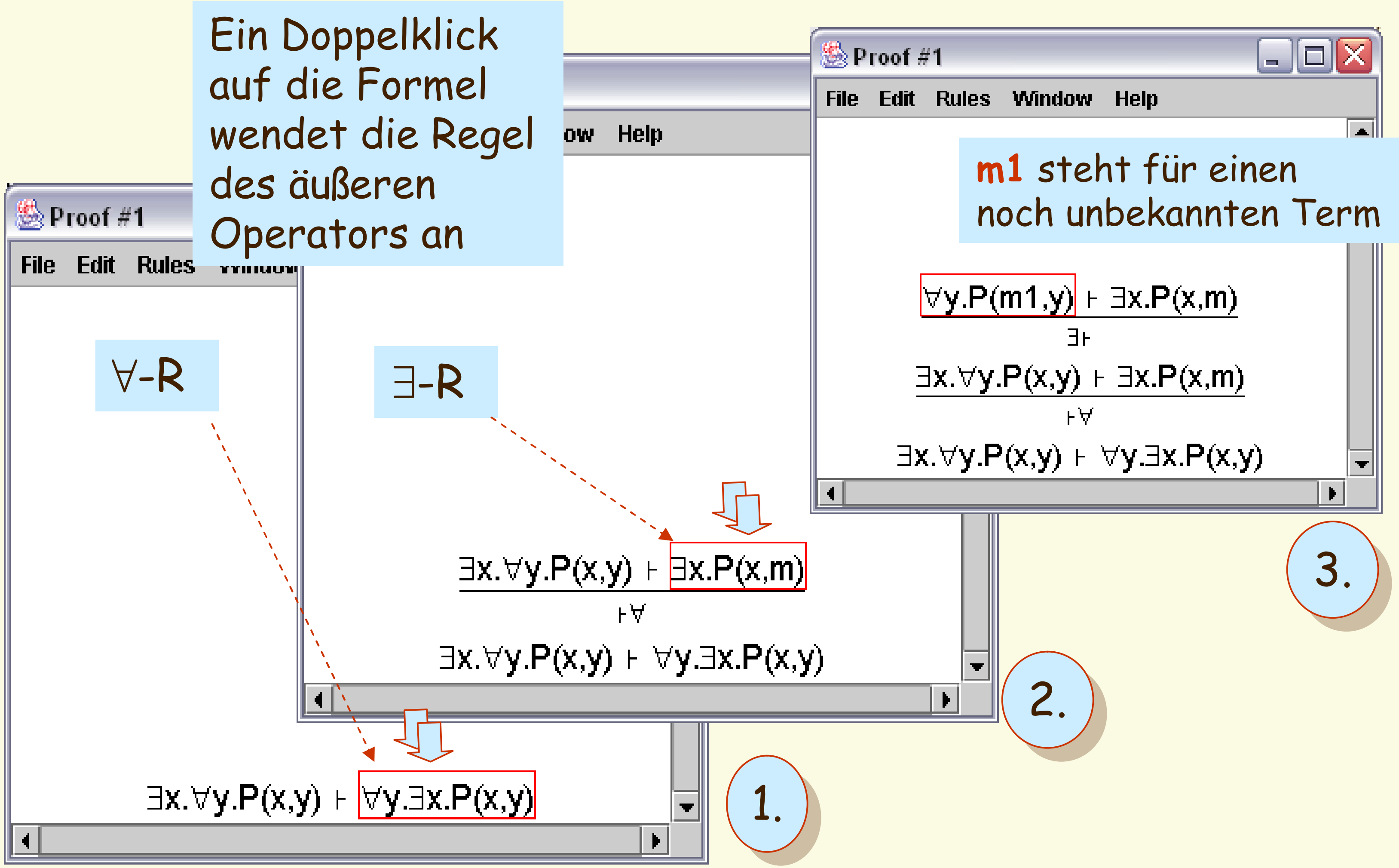


Jape Interaktion

- In einer „Conjecture-Datei“ werden zu beweisende Sätze aufgeschrieben
- In einem „Conjecture-Panel“ wählt man die zu beweisende Formel
- Die Anwendung der Beweisregeln wählt man
  - aus einem Menü, oder
  - durch Anklicken des zu bearbeitenden logischen Operators

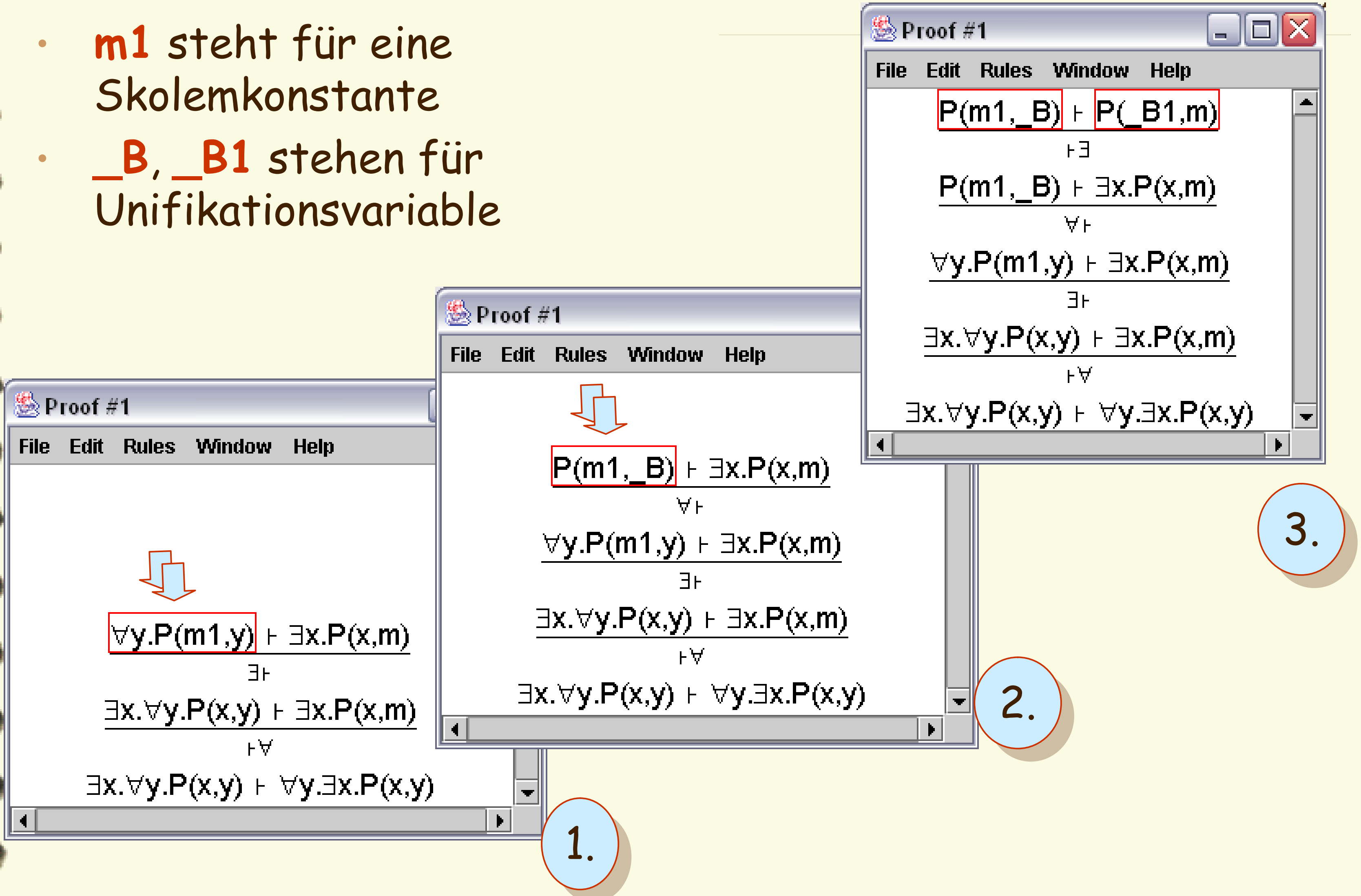


Beweis durch point and click



Beweis

- **m1** steht für eine Skolemkonstante
- **_B, _B1** stehen für Unifikationsvariable



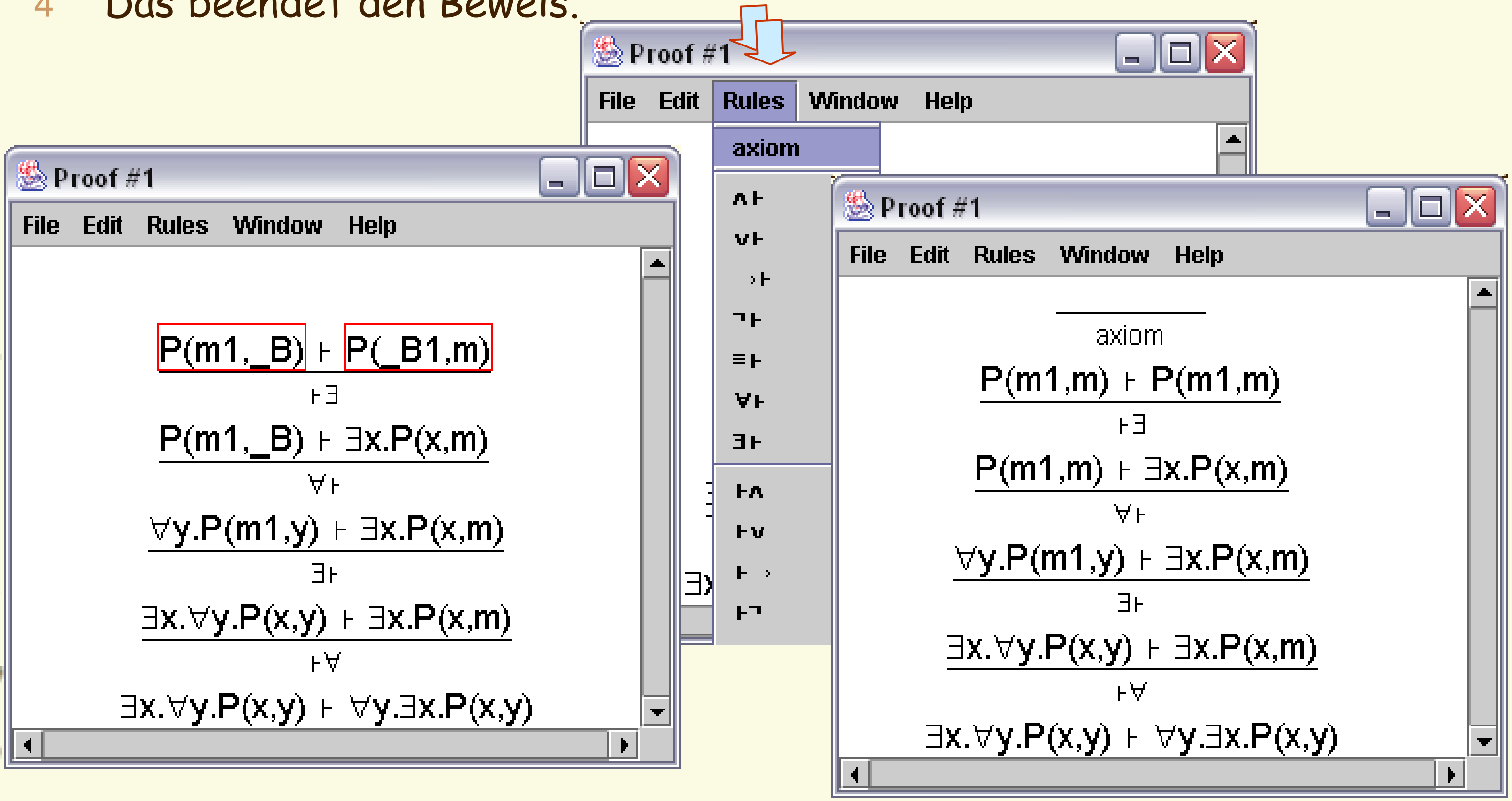
Unifikation

4 Eine Anwendung des Axioms

$$\Gamma,\ p \vdash \Delta\ ,\ p$$

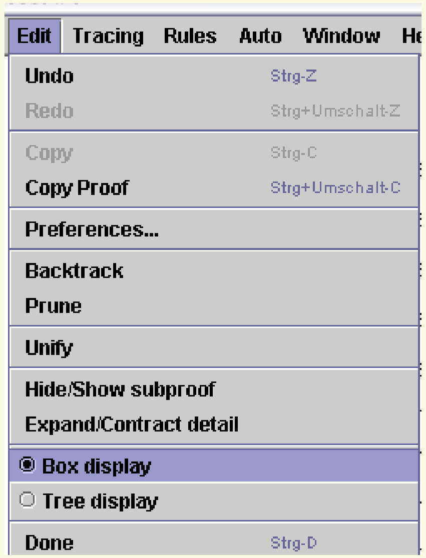
besteht in der Unifikation von **P(m1,_B)** mit **P(_B1,m)**. Das ist möglich mit σ = { B = m , _B1=m1 }

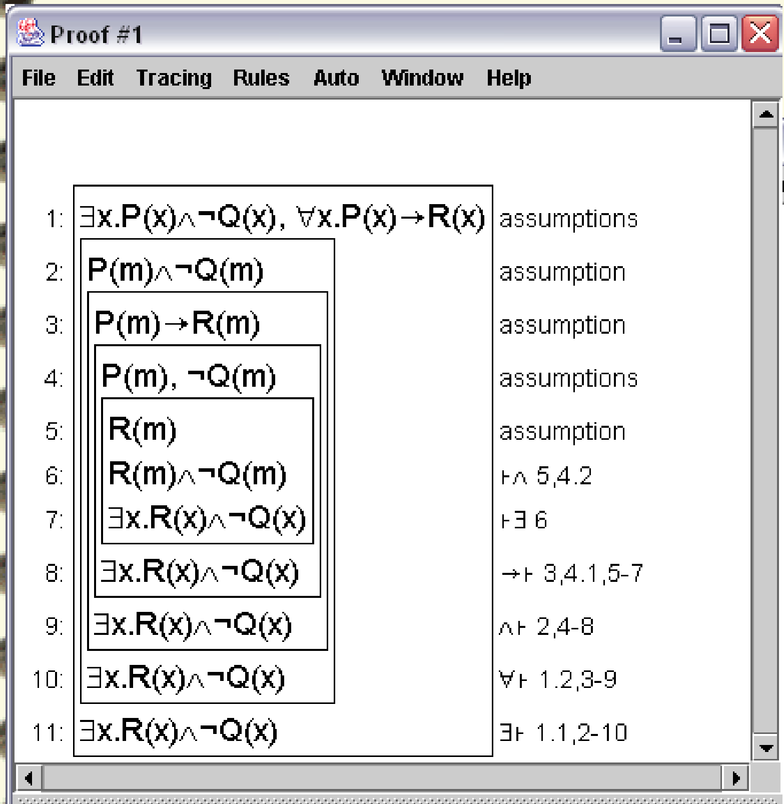
4 Das beendet den Beweis.

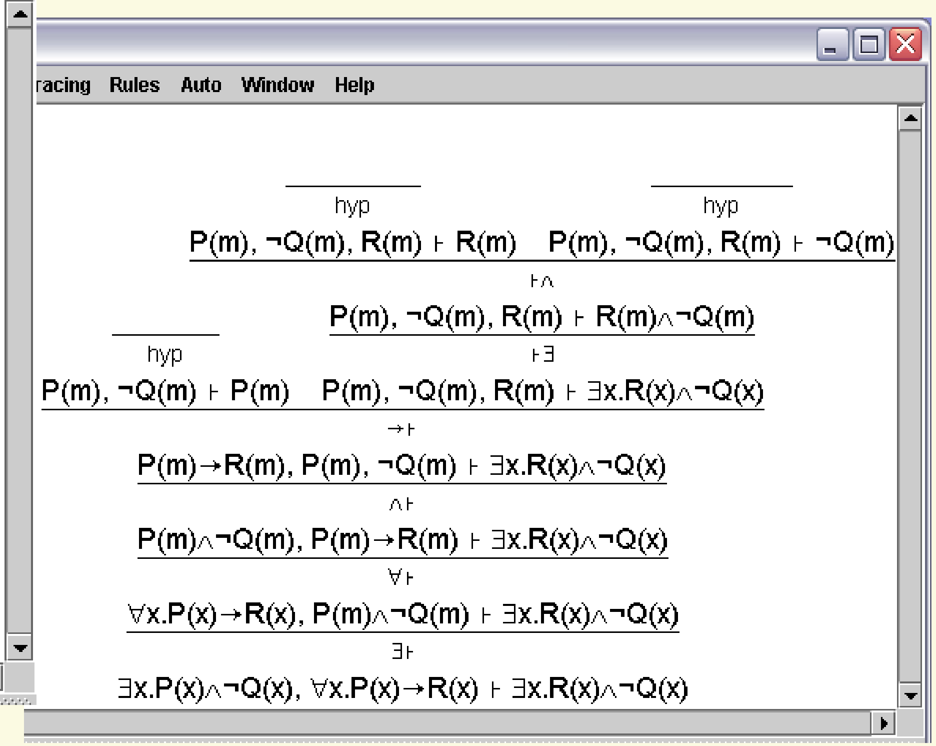


Beweisdarstellung

4 Jape kann Beweise in
  – Box-Darstellung oder
  – Baumdarstellung anzeigen







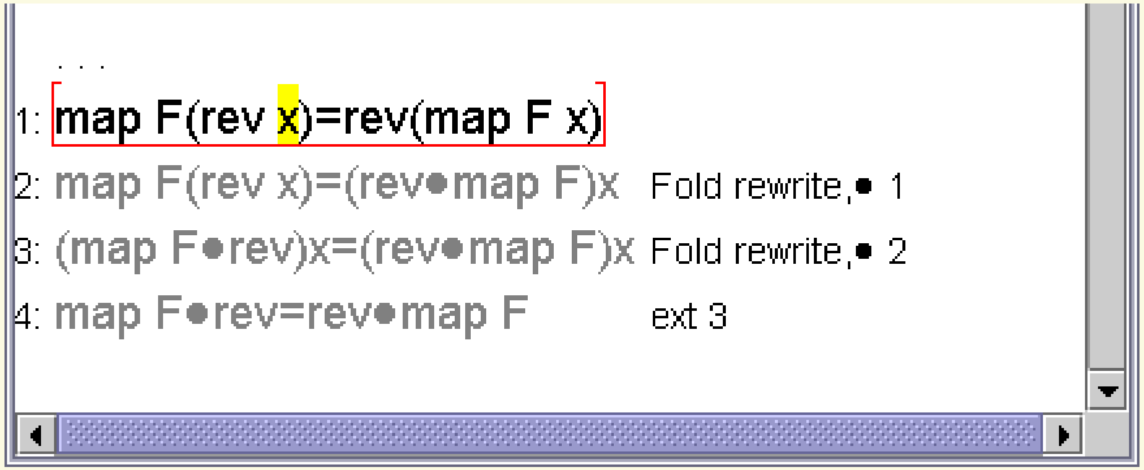
Theorie funktionaler Programme

- Beginn des Beweises von

  map F o rev = rev o map F
- **Teilterme können mit**

  Alt-Linke Maustaste

  markiert werdeen



Eigene Logiken in Jape

- In Jape können beliebige Logiken selber erstellt werden
  - Syntax der Formeln
  - Regeln
  - Menüs für Regelauswahl
  - Mausinteraktionen

- Vorhanden sind u.a.
  - Sequenzenkalkül (MCSC)
  - Single-Conclusion Sequenzenkalkül (SCSC)
  - Natürliches Schließen
  - BAN-Kalkül (für Verifikation von Sicherheitsprotokollen)
  - Funktionales Programmieren
  - etc. ...

Beispiel für eine Jape-Definition:

- Regeln für Natürliche Herleitung

```
RULE "→-E"(A) IS FROM A AND A→B INFER B
RULE "∧-E(L)"(B) IS FROM A ∧ B INFER A
RULE "∧-E(R)"(A) IS FROM A ∧ B INFER B
RULE "∨-E"(A,B) IS FROM A ∨ B AND A ⊢ C AND B ⊢ C INFER C
RULE "¬-E" IS FROM ¬¬A INFER A
RULE "∀-E"(B) IS FROM ∀x. A(x) INFER A(B)
RULE "∃-E"(OBJECT c) WHERE FRESH c AND c NOTIN ∃x.A IS FROM ∃x.A(x) AND A(c) ⊢ C INFER C

RULE "→-I" IS FROM A ⊢ B INFER A→B
RULE "∧-I" IS FROM A AND B INFER A ∧ B
RULE "∨-I(L)"(B) IS FROM A INFER A ∨ B
RULE "∨-I(R)"(A) IS FROM B INFER A ∨ B
RULE "¬-I"(B) IS FROM A ⊢ B ∧ ¬B INFER ¬A
RULE "∀-I"(OBJECT c) WHERE FRESH c IS FROM A(c) INFER ∀x .A(x)
RULE "∃-I"(B) IS FROM A(B) INFER ∃x.A(x)
```

Anwendung von Regeln durch Mausclick

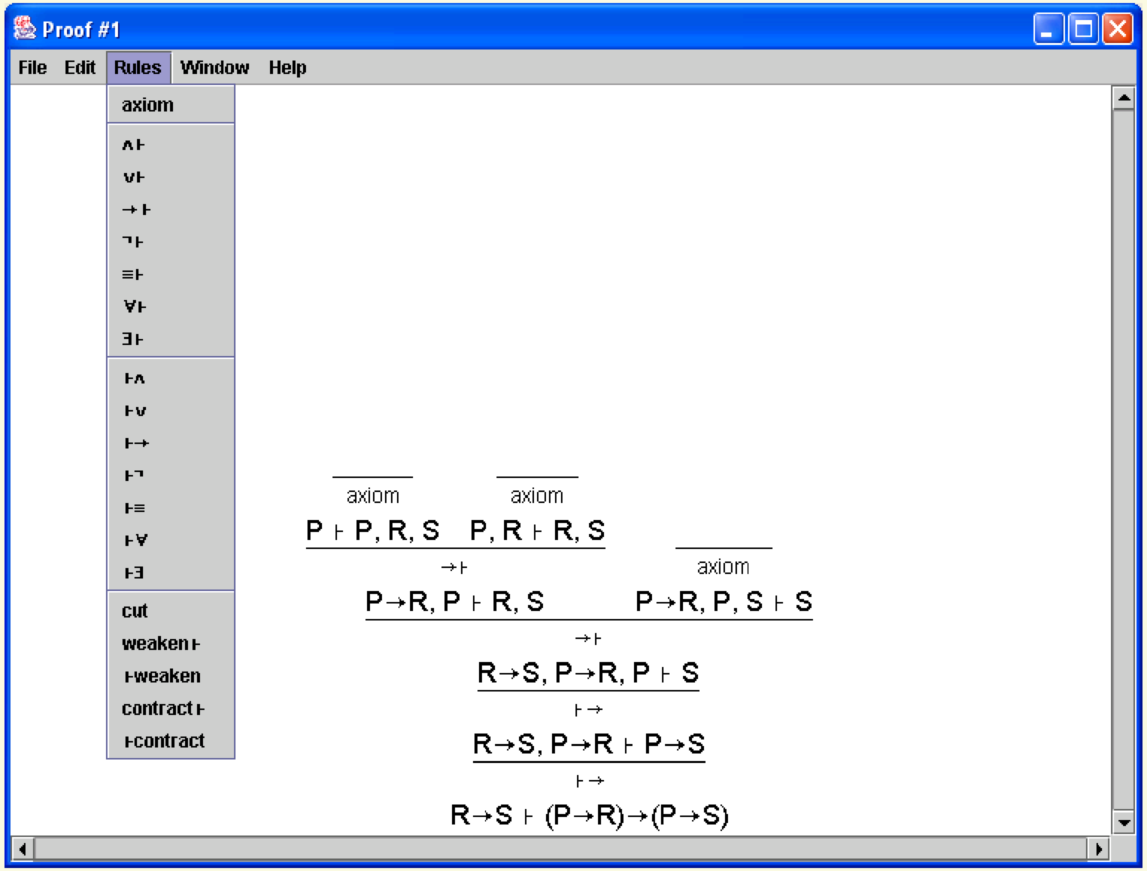
HYPHIT	A→B ⊢	IS "→⊢"
HYPHIT	A∨B ⊢	IS "∨⊢"
HYPHIT	A∧B ⊢	IS "∧⊢"
HYPHIT	¬A ⊢	IS "¬⊢"
HYPHIT	∀x.A ⊢	IS "∀⊢"
HYPHIT	∃x.A ⊢	IS "∃⊢"

... im Antezedenz

CONCHIT	B∧C	IS "⊢∧"
CONCHIT	B∨C	IS "⊢∨"
CONCHIT	B→C	IS "⊢→"
CONCHIT	¬B	IS "⊢¬"
CONCHIT	∀x.B	IS "⊢∀"
CONCHIT	∃x.B	IS "⊢∃"

... im Succedenz

Die definierte Jape-Logik in Aktion



Die Menüdefinition ...

```
MENU Rules IS
    ENTRY axiom
    SEPARATOR
```

...

```
    ENTRY "⊢∀"
    ENTRY "⊢∃"
    SEPARATOR
    ENTRY cut
    ENTRY "weaken⊢"
    ENTRY "⊢weaken"
    ENTRY "contract⊢"
    ENTRY "⊢contract"
END
```