Formale Techniken der Software-Entwicklung

Matthias Hölzl, Christian Kroiß

15. April 2014

Sequenzenkalkül

- Kann man mit den Schlussregeln falsche Aussagen ableiten?
- Gibt es wahre Aussagen, die man nicht ableiten kann?
- Warum braucht man überhaupt Schlussregeln?

Sequenzenkalkül

- Korrektheit
- Vollständigkeit
- (Teil)-Automatisches Schließen, PVS

Korrektheit

- Zu zeigen:
  - Wenn alle Voraussetzungen einer Schlussregel wahr sind
  - Dann ist auch die Konklusion wahr
- Induktion über die Struktur der Herleitung

Vollständigkeit

- Zu zeigen:
  - Wenn eine Aussage eine Tautologie ist
  - Dann kann man das auch mit dem Sequenzenkalkül beweisen
- Beweis: Finden eines Ableitungsbaumes für jede wahre Formel

De Morgan'sche Regeln

$$\neg(\phi \wedge \psi) \iff \neg\phi \vee \neg\psi$$
$$\neg(\phi \vee \psi) \iff \neg\phi \wedge \neg\psi$$

Beweis durch Wahrheitstabelle

Sei $\chi = \neg(\phi \wedge \psi) \iff \neg\phi \vee \neg\psi$

ϕ	ψ	$\neg\phi$	$\neg\psi$	$\phi \wedge \psi$	$\neg(\phi \wedge \psi)$	$\neg\phi \vee \neg\psi$	χ
false	false	true	true	false	true	true	true
false	true	true	false	false	true	true	true
true	false	false	true	false	true	true	true
true	true	false	false	true	false	false	true

Entsprechend zeigt man die Allgemeingültigkeit von
$\neg(\phi \vee \psi) \iff \neg\phi \wedge \neg\psi$.

Beweis im Sequenzenkalkül

Später

Konjunktion/Disjunktion von Formelmengen

Wir definieren

$$\bigwedge\{\phi_1, \ldots, \phi_n\} = \bigwedge_{i=1}^{n} \phi_i = \phi_1 \wedge \cdots \wedge \phi_n$$

und

$$\bigvee\{\phi_1, \ldots, \phi_n\} = \bigvee_{i=1}^{n} \phi_i = \phi_1 \vee \cdots \vee \phi_n$$

Konjunktion/Disjunktion von Formelmengen

Oft lassen wir dabei die Mengenklammern weg und schreiben

$$\bigwedge \phi_1, \ldots, \phi_n$$

oder

$$\bigwedge \Gamma$$

Wir definieren die Spezialfälle

$$\bigwedge \emptyset = \mathsf{true} \qquad \bigvee \emptyset = \mathsf{false}$$

Notation

Formel ϕ ist wahr unter Belegung η

$$\models_\eta \phi \quad \text{oder} \quad \eta \models \phi$$

Formel ϕ ist eine Tautologie (d.h. wahr unter jeder Belegung)

$$\models \phi$$

Semantischer Folgerungsbegriff

ψ folgt semantisch aus ϕ

$$\phi \models \psi$$

genau dann, wenn jede Belegung, die ϕ erfüllt auch ψ erfüllt:

für alle η gilt $\models_\eta \phi$ impliziert $\models_\eta \psi$

Semantischer Folgerungsbegriff

$\psi_1, \ldots, \psi_n$ folgt semantisch aus $\phi_1, \ldots \phi_m$

$$\phi_1, \ldots, \phi_m \models \psi_1, \ldots, \psi_n$$

genau dann, wenn jede Belegung, die alle ϕ_i erfüllt auch (mindestens) ein ψ_j erfüllt, wenn also gilt

$$\bigwedge \phi_1, \ldots, \phi_m \models \bigvee \psi_1, \ldots, \psi_n$$

Deduktionstheorem

Seien Γ eine endliche Menge von Formeln, ϕ und ψ Formeln. Dann gilt

$$\Gamma, \phi \models \psi \tag{1}$$

genau dann, wenn

$$\Gamma \models \phi \implies \psi \tag{2}$$

gilt.

Beweis

(1) $\Rightarrow$ (2): Es gilt $\Gamma, \phi \models \psi$. Sei η eine Belegung, die Γ erfüllt. Wenn η ϕ nicht erfüllt, so gilt $\phi \Rightarrow \psi$ und somit (2). Erfülle also η ϕ. Damit erfüllt η auch Γ, ϕ, und da $\Gamma, \phi \models \psi$ gilt, erfüllt η dann auch ψ. Somit erfüllt η auch $\phi \implies \psi$.

(2) $\Rightarrow$ (1): Es gilt $\Gamma \models \phi \implies \psi$. Sei η eine Belegung, die Γ, ϕ (und damit auch Γ und ϕ einzeln) erfüllt. Da $\eta \models \Gamma$ gilt, erfüllt η auch $\phi \Rightarrow \psi$, somit erfüllt η auch ψ, und damit gilt $\Gamma, \phi \models \psi$.

Verallgemeinerte De Morgan'sche Regeln

Man kann die De Morgan'schen Regeln leicht auf Formelmengen verallgemeinern:

$$\neg\Big(\bigwedge_{i=1}^{n} \phi_i\Big) \iff \bigvee_{i=1}^{n} \neg\phi_i$$

$$\neg\Big(\bigvee_{i=1}^{n} \phi_i\Big) \iff \bigwedge_{i=1}^{n} \neg\phi_i$$

(Beweis durch Induktion über n.)

Sequenzen

Eine Sequenz $\Gamma \vdash \Delta$ entspricht einer Implikation: $\bigwedge \Gamma \Rightarrow \bigvee \Delta$

Wir können mit der Definition von $\Rightarrow$ und den De Morgan'schen Regeln folgendermaßen umformen

$$\begin{aligned} \bigwedge \phi_i &\Longrightarrow \bigvee \psi_j \\ \neg(\bigwedge \phi_i) &\vee \bigvee \psi_j \\ \bigvee \neg\phi_i &\vee \bigvee \psi_j \end{aligned}$$

Es ist also $\phi_1, \ldots, \phi_m \vdash \psi_1, \ldots \psi_n$ äquivalent zu

$$\neg\phi_1 \vee \cdots \vee \neg\phi_m \vee \psi_1 \vee \cdots \vee \psi_n$$

Beispiel

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\phi, \psi, \chi \vdash \chi}}{\psi, \chi \vdash \phi \Rightarrow \chi} \quad \dfrac{}{\psi \vdash \psi, \phi \Rightarrow \chi}}{\psi, \psi \Rightarrow \chi \vdash \phi \Rightarrow \chi} \quad \dfrac{\dfrac{}{\phi, \psi \Rightarrow \chi \vdash \phi, \chi}}{\psi \Rightarrow \chi \vdash \phi, \phi \Rightarrow \chi}}{\phi \Rightarrow \psi, \psi \Rightarrow \chi \vdash \phi \Rightarrow \chi}}{(\phi \Rightarrow \psi) \wedge (\psi \Rightarrow \chi) \vdash \phi \Rightarrow \chi}}{\vdash (\phi \Rightarrow \psi) \wedge (\psi \Rightarrow \chi) \implies (\phi \Rightarrow \chi)}$$

Ableitung im Sequenzenkalkül

Eine Formel ψ ist im Sequenzenkalkül ableitbar, wenn es eine Herleitung von ψ mit leerem Antezedens gibt:

$$\begin{aligned} \phi_1, \ldots, \phi_m \vdash \psi_1, \ldots, \psi_n &\equiv \neg\phi_1 \vee \cdots \vee \neg\phi_m \vee \psi_1 \vee \ldots, \vee\psi_n \\ \vdash \psi &\equiv \psi \end{aligned}$$

Eine Implikation $\phi \implies \psi$ ist im Sequenzenkalkül genau dann ableitbar, wenn die Sequenz $\phi \vdash \psi$ ableitbar ist.

$$\begin{aligned} \vdash \left(\phi_1 \wedge \cdots \wedge \phi_m \implies \psi_1 \vee \ldots, \vee\psi_n\right) \\ &\equiv \phi_1 \wedge \cdots \wedge \phi_m \implies \psi_1 \vee \ldots, \vee\psi_n \\ &\equiv \neg\phi_1 \vee \cdots \vee \neg\phi_m \vee \psi_1 \vee \ldots, \vee\psi_n \\ &\equiv \phi_1, \ldots, \phi_m \vdash \psi_1, \ldots, \psi_n \end{aligned}$$

Notation

Eine Sequenz $\Gamma \vdash \Delta$ ist im Sequenzenkalkül ableitbar"

$$\vdash_{\text{seq}} \Gamma \vdash \Delta \quad \text{oder} \quad \Gamma \vdash_{\text{seq}} \Delta$$

Eine Formel ϕ ist im Sequenzenkalkül ableitbar:

$$\vdash_{\text{seq}} \phi$$

Korrektheit und Vollständigkeit

- Korrektheit

$$\Gamma \left|\!\!\frac{}{\text{seq}}\right. \Delta \quad \text{impliziert} \quad \Gamma \models \Delta$$

- Vollständigkeit

$$\Gamma \models \Delta \quad \text{impliziert} \quad \Gamma \left|\!\!\frac{}{\text{seq}}\right. \Delta$$

Sequenzenkalkül (1)

$$\frac{\Gamma_1 \vdash \Delta_1}{\Gamma_2 \vdash \Delta_2}\mathbf{W} \qquad \text{if } \Gamma_1 \subseteq \Gamma_2 \wedge \Delta_1 \subseteq \Delta_2$$

$$\frac{}{\Gamma, \phi \vdash \phi, \Delta}\mathbf{Ax} \qquad \frac{}{\Gamma, \bot \vdash \Delta}\bot \qquad \frac{}{\Gamma \vdash \top, \Delta}\top$$

$$\frac{\Gamma \vdash \phi, \Delta}{\Gamma, \neg\phi \vdash \Delta}\neg\vdash \qquad \frac{\Gamma, \phi \vdash \Delta}{\Gamma \vdash \neg\phi, \Delta}\vdash\neg$$

Sequenzenkalkül (2)

$$\frac{\phi, \psi, \Gamma \vdash \Delta}{\phi \wedge \psi, \Gamma \vdash \Delta} \wedge\vdash \qquad \frac{\Gamma \vdash \phi, \Delta \quad \Gamma \vdash \psi, \Delta}{\Gamma \vdash \phi \wedge \psi, \Delta} \vdash\wedge$$

$$\frac{\phi, \Gamma \vdash \Delta \quad \psi, \Gamma \vdash \Delta}{\phi \vee \psi, \Gamma \vdash \Delta} \vee\vdash \qquad \frac{\Gamma \vdash \phi, \psi, \Delta}{\Gamma \vdash \phi \vee \psi, \Delta} \vdash\vee$$

$$\frac{\psi, \Gamma \vdash \Delta \quad \Gamma \vdash \phi, \Delta}{\phi \Rightarrow \psi, \Gamma \vdash \Delta} \Rightarrow\vdash \qquad \frac{\Gamma, \phi \vdash \psi, \Delta}{\Gamma \vdash \phi \Rightarrow \psi, \Delta} \vdash\Rightarrow$$

Abschwächung (Weakening)

$$\frac{\Gamma_1 \vdash \Delta_1}{\Gamma_2 \vdash \Delta_2}\mathbf{W} \qquad \text{if } \Gamma_1 \subseteq \Gamma_2 \wedge \Delta_1 \subseteq \Delta_2$$

$$\frac{\phi_1, \ldots, \phi_m \vdash \psi_1, \ldots \psi_n}{\phi_1, \ldots, \phi_m, \phi_{m+1}, \ldots, \phi_p \vdash \psi_1, \ldots \psi_n, \psi_{n+1}, \ldots, \psi_q}$$

Abschwächung (Weakening)

$$\frac{\phi_1, \ldots, \phi_m \vdash \psi_1, \ldots \psi_n}{\phi_1, \ldots, \phi_m, \phi_{m+1}, \ldots, \phi_p \vdash \psi_1, \ldots \psi_n, \psi_{n+1}, \ldots, \psi_q}$$

$$\neg\phi_1 \vee \cdots \vee \neg\phi_m \vee \neg\phi_{m+1} \vee \cdots \vee \neg\phi_p \vee \psi_1 \vee \cdots \vee \psi_n \vee \psi_{n+1} \vee \cdots \vee \psi_q$$

Wahrheitstabelle

ϕ_1	...	ϕ_{m+1}	...	ψ_1	...	ψ_{n+1}	...	ϕ	ψ
?	...	false	...	?	...	false	...	?	true
?	...	false	...	?	...	true	...	?	true
?	...	true	...	?	...	false	...	?	?
?	...	true	...	?	...	true	...	?	true

Axiom (propositional)

$$\frac{}{\Gamma, \phi \vdash \phi, \Delta}\,\mathbf{Ax}$$

$$\neg\Gamma \vee \neg\phi \vee \phi \vee \Delta$$

Negation

$$\frac{\Gamma \vdash \phi, \Delta}{\Gamma, \neg\phi \vdash \Delta}\neg\vdash$$

$$\neg\Gamma \vee \phi \vee \Delta$$

$$\neg\Gamma \vee \neg\neg\phi \vee \Delta$$

$$\frac{\Gamma, \phi \vdash \Delta}{\Gamma \vdash \neg\phi, \Delta}\vdash\neg$$

$$\neg\Gamma \vee \neg\phi \vee \Delta$$

$$\neg\Gamma \vee \neg\phi \vee \Delta$$

Konjunktion

$$\frac{\phi, \psi, \Gamma \vdash \Delta}{\phi \wedge \psi, \Gamma \vdash \Delta} \wedge\vdash$$

$$\neg\phi \vee \neg\psi \vee \Gamma \vee \Delta$$

$$\neg(\phi \wedge \psi) \vee \Gamma \vee \Delta$$

$$\frac{\Gamma \vdash \phi, \Delta \quad \Gamma \vdash \psi, \Delta}{\Gamma \vdash \phi \wedge \psi, \Delta} \vdash\wedge$$

$$(\neg\Gamma \vee \phi \vee \Delta) \wedge (\neg\Gamma \vee \psi \vee \Delta)$$

$$\neg\Gamma \vee (\phi \wedge \psi) \vee \Delta$$

Konjunktion

$$\frac{\phi, \psi, \Gamma \vdash \Delta}{\phi \wedge \psi, \Gamma \vdash \Delta} \wedge\vdash$$

$$\neg\phi \vee \neg\psi \vee \Gamma \vee \Delta$$

$$\neg(\phi \wedge \psi) \vee \Gamma \vee \Delta$$

$$\frac{\Gamma \vdash \phi, \Delta \quad \Gamma \vdash \psi, \Delta}{\Gamma \vdash \phi \wedge \psi, \Delta} \vdash\wedge$$

$$(\phi \vee (\neg\Gamma \vee \Delta)) \wedge (\psi \vee (\neg\Gamma \vee \Delta))$$

$$(\phi \wedge \psi) \vee (\neg\Gamma \vee \Delta)$$

Implikation

$$\frac{\psi, \Gamma \vdash \Delta \quad \Gamma \vdash \phi, \Delta}{\phi \Rightarrow \psi, \Gamma \vdash \Delta} \Rightarrow\vdash$$

$$(\neg\psi \vee \neg\Gamma \vee \Delta) \wedge (\neg\Gamma \vee \phi \vee \Delta)$$

$$\neg(\neg\phi \vee \psi) \vee \neg\Gamma \vee \Delta$$

$$(\phi \wedge \neg\psi) \vee \neg\Gamma \vee \Delta$$

$$\frac{\Gamma, \phi \vdash \psi, \Delta}{\Gamma \vdash \phi \Rightarrow \psi, \Delta} \vdash\Rightarrow$$

$$\neg\Gamma \vee \neg\phi \vee \psi \vee \Delta$$

$$\neg\Gamma \vee (\neg\phi \vee \psi) \vee \Delta$$

Sequenzenkalkül

$$\frac{\phi, \psi, \Gamma \vdash \Delta}{\phi \wedge \psi, \Gamma \vdash \Delta} \wedge\vdash \qquad \frac{\Gamma \vdash \phi, \Delta \quad \Gamma \vdash \psi, \Delta}{\Gamma \vdash \phi \wedge \psi, \Delta} \vdash\wedge$$

$$\frac{\phi, \Gamma \vdash \Delta \quad \psi, \Gamma \vdash \Delta}{\phi \vee \psi, \Gamma \vdash \Delta} \vee\vdash \qquad \frac{\Gamma \vdash \phi, \psi, \Delta}{\Gamma \vdash \phi \vee \psi, \Delta} \vdash\vee$$

$$\frac{\psi, \Gamma \vdash \Delta \quad \Gamma \vdash \phi, \Delta}{\phi \Rightarrow \psi, \Gamma \vdash \Delta} \Rightarrow\vdash \qquad \frac{\Gamma, \phi \vdash \psi, \Delta}{\Gamma \vdash \phi \Rightarrow \psi, \Delta} \vdash\Rightarrow$$

PVS

- Theorien
- Syntax
- Umgang mit dem Theorembeweiser

PVS Theorien

- (Momentan) Sammlung von aussagenlogischen Formeln
- Axiome $A_1, \ldots, A_n$
- Propositionen $P_1, \ldots, P_n$ (Lemmata, Theoreme, ...)
- $A_1, \ldots, A_m \models P_1 \wedge \cdots \wedge P_n$
- Oft keine Axiome

Syntax

```
name : THEORY
  BEGIN

  V1, V2: bool

  P1: PROPOSITION
      V1 OR NOT(V1)
  P2: THEOREM
      (V1 => V2) <=> (NOT(V2) => NOT(V1))

  END name
```

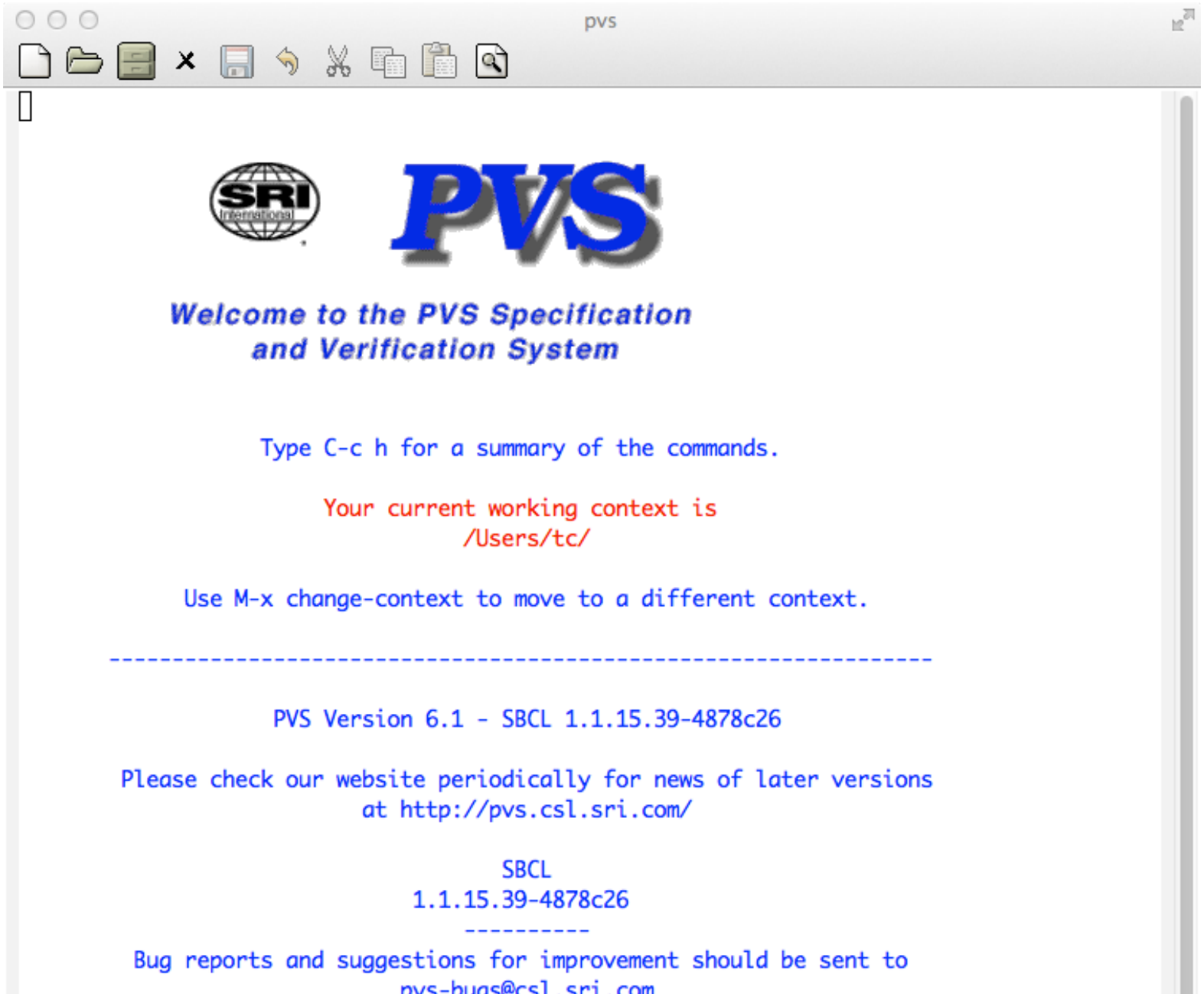
Beispiel

```
lecture_1 : THEORY
  BEGIN

  A, B, C: bool

  tertium_non_datur: PROPOSITION A OR NOT (A)
  double_negation_1: PROPOSITION A => NOT(NOT(A))
  double_negation_2: PROPOSITION A <=> NOT(NOT(A))
  counterpositive:   PROPOSITION
                       (A => B) <=> (NOT(B) => NOT(A))
  transitivity:      PROPOSITION
                       (A => B) AND (B => C) => (A => C)
  END lecture_1
```

PVS Emacs


pvs
Welcome to the PVS Specification
and Verification System
Type C-c h for a summary of the commands.
Your current working context is
/Users/tc/
Use M-x change-context to move to a different context.
PVS Version 6.1 - SBCL 1.1.15.39-4878c26
Please check our website periodically for news of later versions
at http://pvs.csl.sri.com/
SBCL
1.1.15.39-4878c26
Bug reports and suggestions for improvement should be sent to

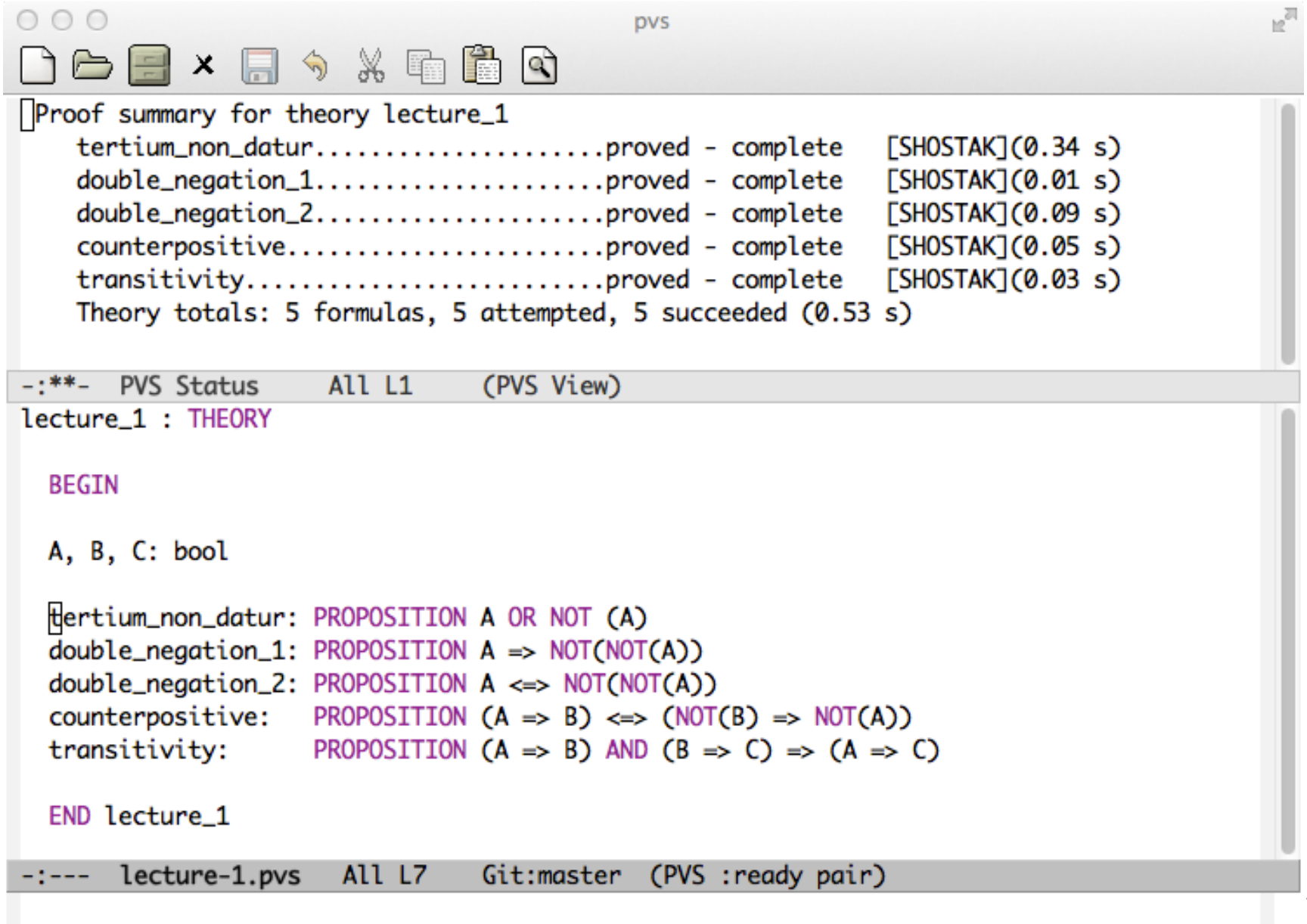
PVS Context

- Alle Interaktionen mit PVS erfolgen innerhalb eines Kontexts
- Der Kontext enthält (Hilfs-)Theorien, angefangene Beweise, etc.
- Am Anfang jeder PVS-Session muss mit `M-x change-context` in den richtigen Kontext gewechselt werden

Interaktives Beweisen

- Nach dem Laden einer PVS-Theorie kann man sie auf syntaktische und Typfehler überprüfen und die darin enthaltenen Aussagen interaktiv beweisen
- Der Theorembeweiser wird mit dem Kommando `M-x prove-theory` aufgerufen
- Falls im Kontext schon vorhergehende Beweise oder Beweisversuche existieren können diese nochmals ausgeführt werden

PVS Bewiesene Theorie



Interaktives Beweisen

- Mit `M-x prove` oder `C-c p` kann der interaktive Theorembeweiser aufgerufen werden.
- Die zu beweisende Sequenz wird in einem Fenster angezeigt, Beweiskommandos werden im gleichen Fenster eingegeben:

```
Installing rewrite rule sets.singleton_rew
(all instances)
tertium_non_datur :

  |-------
{1}   A OR NOT (A)

Rule? (flatten 1)
Applying disjunctive simplification to flatten
sequent,
Q.E.D.
```

Sequenzenkalkül in PVS

$$\frac{\phi, \psi, \Gamma \vdash \Delta}{\phi \wedge \psi, \Gamma \vdash \Delta} \wedge\vdash \qquad \frac{\Gamma \vdash \phi, \Delta \quad \Gamma \vdash \psi, \Delta}{\Gamma \vdash \phi \wedge \psi, \Delta} \vdash\wedge$$

$$\frac{\Gamma \vdash \phi, \psi, \Delta}{\Gamma \vdash \phi \vee \psi, \Delta} \vdash\vee \qquad \frac{\phi, \Gamma \vdash \Delta \quad \psi, \Gamma \vdash \Delta}{\phi \vee \psi, \Gamma \vdash \Delta} \vee\vdash$$

$$\frac{\Gamma, \phi \vdash \psi, \Delta}{\Gamma \vdash \phi \Rightarrow \psi, \Delta} \vdash\Rightarrow \qquad \frac{\psi, \Gamma \vdash \Delta \quad \Gamma \vdash \phi, \Delta}{\phi \Rightarrow \psi, \Gamma \vdash \Delta} \Rightarrow\vdash$$

`(flatten)` `(split)`

Sequenzenkalkül in PVS

- Der Beweisbaum wird *von unten nach oben* abgearbeitet
- Man arbeitet immer an einem Pfad durch den Beweis
- Durch `(postpone)` kann man vorübergehend in einen anderen Pfad wechseln
- Manche Vereinfachungen werden immer durchgeführt
- Mit `(undo)` kann man falsche Beweisschritte rückgängig machen
- Neben den einfachen Regeln stehen auch komplexe Strategien zur Verfügung
- Beweiskommandos können auf einzelne Terme (durch Angabe der Nummer), den Antezedens/Sukzedens (durch −, +) oder alle Terme angewendet werden

Wichtige Beweiskommandos

- `(flatten)`: Führt Regeln aus, die den Beweisbaum nicht weiter aufspalten
- `(flatten-disjunct)`: Weniger leistungsfähige Version von `(flatten)`, gut zum Verstehen der Arbeitsweise des Theorembeweisers
- `(split)`: Führt Regeln aus, die den Beweis in mehrere Äste aufspalten
- `(prop)`: Führt propositionale Vereinfachung durch
- `(grind)`: Leistungsfähigste Strategie, kann auch viele nicht-propositionale Theoreme beweisen

PVS

- Veranschaulichen der Korrektheit und Vollständigkeit des von PVS-verwendeten Sequenzenkalküls mit PVS
- Das ist natürlich kein gültiger Beweis, aber für den Umgang mit PVS instruktiv

Erfüllbarkeit (SAT)

- Beschreibung des SAT Problems
- NP-Vollständigkeit
- Normalformen
- Praktische Lösbarkeit
- Ansätze zur Lösung

SAT

Gegeben sei eine aussagenlogische Formel ϕ. Das *Erfüllbarkeitsproblem (SAT)* ist die Frage nach einer erfüllenden Belegung: Gibt es ein η mit

$$\models_\eta \phi?$$

SAT war das erste Problem, das als *NP-vollständig* bewiesen wurde.

NP

Ein *Prüfprogramm (Verifier)* für eine Sprache $\mathcal{A}$ ist ein Algorithmus V, für den gilt

$$\mathcal{A} = \{w \mid V \text{ akzeptiert } \langle w, c \rangle \text{ für eine Zeichenkette } c\}$$

c heißt *Zertifikat* oder *Beweis*.

NP ist die Klasse der Sprachen, die in polynomialer Zeit verifiziert werden können. (Das heißt, dass Zertifikate maximal polynomiale Länge haben dürfen.)

NP-Vollständigkeit

Eine Sprache B ist *NP-vollständig*, wenn

- sie in NP enthalten ist und
- jedes A in NP in polynomialer Zeit auf B reduziert werden kann.

Cook-Levin Theorem

SAT ist *NP*-vollständig

- Beweisskizze: Tableau der zertifikatsprüfenden Turing-Maschine lässt sich in Aussagenlogik codieren
- Konsequenzen für Algorithmen:
  - Wahrscheinlich gibt es keinen Algorithmus, der im schlechtesten Fall besser ist als Wahrheitstabellen
  - In der Praxis lassen sich fast alle *SAT*-Probleme gut lösen

Normalformen

- Warum Normalformen?
- Konjunktive Normalform (CNF)
- Disjunktive Normalform (DNF)
- ...
- Algorithmus zur Umwandlung in CNF

Warum Normalformen?

- Einfacher für Computer
- Theoretische Erkenntnisse

Konjunktive Normalform

- Ein *Literal* L ist eine Variable A oder eine negierte Variable $\neg A$

$$L = A \mid \neg A$$

- Eine Klausel K ist eine Disjunktion von Literalen:

$$K = L_1 \vee L_2 \vee \cdots \vee L_m$$

- Eine Formel C ist in konjunktiver Normalform (KNF, CNF), wenn sie eine Konjunktion von Klauseln ist:

$$\begin{aligned} C &= K_1 \wedge K_2 \wedge \cdots \wedge K_n \\ &= (L_{11} \vee \cdots \vee L_{1p}) \wedge \cdots \wedge (L_{k1} \vee \cdots \vee L_{kp}) \end{aligned}$$

Anwendung von CNF

- Überprüfen von Allgemeingültigkeit
- Eine Klausel ist allgemeingültig genau dann, wenn sie Literale L_i, L_j enthält, so dass $L_i = \neg L_j$
- Eine Formel in CNF ist allgemeingültig, wenn jede ihrer Klauseln allgemeingültig ist
- Beispiel:

$$(A \vee B \vee \neg C \vee \neg B) \wedge (B \vee \neg C \vee E \vee F)$$

ist nicht allgemeingültig, da die zweite Klausel nicht allgemeingültig ist

Konvertierung in CNF

- Gibt es zu jeder Formel ϕ eine äquivalente Formel ψ in CNF?
- Ja, aber ψ ist nicht eindeutig:
- $\psi_1 = \phi$ und $\psi_2 = \phi \wedge (\phi \vee \chi)$ sind beide CNF für ϕ
- Die Entwicklung von Algorithmen, die ein ψ mit möglichst geringen "Kosten" berechnen ist ein wichtiges Problem für formale Methoden

Algorithmus zur Konvertierung in CNF

- Rekursion über die Struktur der Formel

$$\text{Wenn} \quad \phi = \phi_1 \wedge \phi_2 \quad \text{dann} \quad \psi = \psi_1 \wedge \psi_2$$

- Beweis der Korrektheit durch strukturelle Induktion

Algorithmus zur Konvertierung in CNF

- Ersetze $\phi \iff \psi$ durch $\phi \implies \psi \wedge \psi \implies \phi$
- Ersetze $\phi \implies \psi$ durch $\neg\phi \vee \psi$
- Schiebe Negationen nach innen; streiche doppelte Negation (Negationsnormalform)
- Dann: Ist ϕ ein Literal so ist ψ gleich ϕ
- Hat ϕ die Form $\phi_1 \wedge \phi_2$, so hat ψ die Form $\psi_1 \wedge \psi_2$ wobei ψ_1 und ψ_2 rekursiv berechnet werden
- Hat ϕ die Form $\phi_1 \vee \phi_2$, so berechne ψ rekursiv als $\psi_1 \vee \psi_2$ und wende das Distributivgesetz an

Beispiel

$$\phi \Leftrightarrow (\psi \vee \chi)$$

$$\phi \Rightarrow (\psi \vee \chi) \wedge (\psi \vee \chi) \Rightarrow \phi$$

$$(\neg\phi \vee \psi \vee \chi) \wedge (\neg(\psi \vee \chi) \vee \phi)$$

$$(\neg\phi \vee \psi \vee \chi) \wedge ((\neg\psi \wedge \neg\chi) \vee \phi)$$

$$(\neg\phi \vee \psi \vee \chi) \wedge (\neg\psi \vee \phi) \wedge (\neg\chi \vee \phi)$$

Praktische Lösungsverfahren

- Analyse von Problemen: nur wenige sind schwer
- DPLL
- WalkSat

Implementierung eines DPLL-Solvers

- Rekursiver Algorithmus
- Effizientere Implementierung
  - Iterativer Algorithmus
  - Heuristik zur Variablenauswahl
  - Lernen von Klauseln
  - Conflict-directed Backjumping
  - Beobachtete Literale

Rekursives DPLL

- Grundidee: Rekursive Berechnung der Wahrheitstabelle
- Eingabe in CNF
- Frühzeitiger Abbruch
- “Pure Symbol” Heuristik
- “Unit Clause” Heuristik

Implementierung eines WalkSat-Solvers

- Unvollständige Methode
- Random restarts