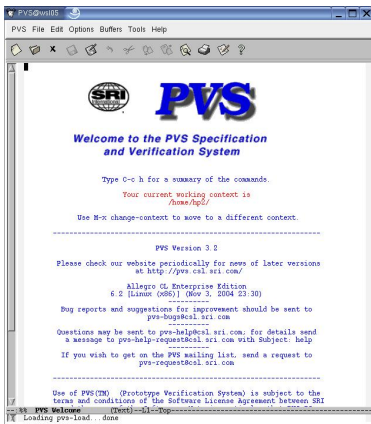


Umgang mit PVS

Maschinelles Beweisen mit PVS – WS 05/06

27. Oktober 2005

- Im Linux-Pool:
 - `use pvs`
 - Aufruf mit `pvs`
- Auf eigenem Rechner:
 - `http://pvs.csl.sri.com` oder `http://www.informatik.uni-ulm.de/ki/PVS/mirror.html`
 - Dateien in Installationsverzeichnis entpacken, `dort` das Skript `./bin/relocate` ausführen
 - evtl. Installationsverzeichnis in `PATH`-Variable aufnehmen, oder symbolischer Link in `/usr/local/bin/` auf `pvs`
 - Aufruf mit `pvs`

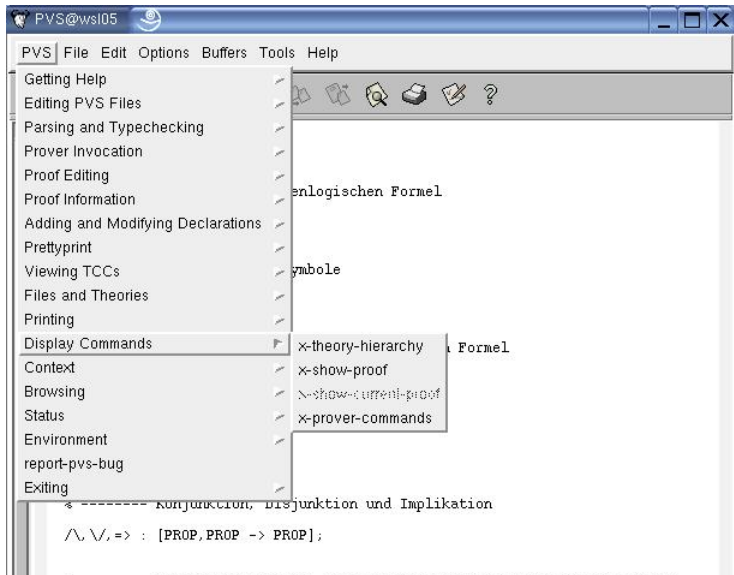


- in ein Arbeitsverzeichnis (PVS: **context**) wechseln:
 - **M-x cc**, Verzeichnisnamen eingeben
- neue PVS-Theorie erstellen:
 - **M-x nf**, Theorienamen eingeben
- bestehende Theorie laden:
 - **C-c C-f**, Theorienamen eingeben
- Spezifikation, Typprüfung, Beweis
 - im folgenden beschrieben ...
- PVS beenden:
 - **C-x C-c**, zur Bestätigung **y** eingeben

Die wichtigsten Emacs-Kommandos für PVS

- PVS Hilfe: `C-c h`
 - ... zur Spezifikationsprache: `C-c C-h l`
 - ... zum Beweiser: `C-c C-h p`
 - ... zu Beweiserkommandos `C-c C-h c`
 - ... zu Emacs-Kommandos im Beweiser `C-c C-h e`
- Theorie laden: `C-c C-f` und speichern: `C-x C-s`
- Typprüfung: `M-x tc` oder `C-c t`
 - ... mit Beweis der TCCs: `M-x tcp`
- Formel beweisen: `M-x pr` oder `C-c p`
 - ... mit Beweisbaum: `M-x x-prove`, `M-x xpr`, oder `C-c C-p x`
 - ... schrittweise: `M-x step-prove`, `M-x prs` oder `C-c C-p s`
 - ... schrittweise mit Beweisbaum `M-x x-step-prove`, `M-x xsp` oder `C-c C-p X`
- Typkorrektheitsbedingungen
 - ... anzeigen: `M-x show-tccs` oder `C-c C-q s`
 - ... beweisen: `M-x prove-tccs-theory` oder `M-x prtt`
- Beweise bearbeiten: `M-x edit-proof`
 - ... installieren (speichern): `M-x install-proof` oder `C-c C-i`

Viele der PVS-Kommandos sind auch über Menüs erreichbar



- Cursor auf zu beweisende Formel, Kommando `M-x pr` oder `C-c p`
- weiterer Emacs-Buffer wird geöffnet:

```
--:-- aussagenlogik.pvs (PVS :ready)--L52--Bot-----
Starting pvs-allegro6.2 -qq ...
Allegro CL Enterprise Edition
6.2 [Linux (x86)] (Nov 3, 2004 23:30)
Copyright (C) 1985-2002, Franz Inc., Berkeley, CA, USA. All Rights Reserved.

This dynamic runtime copy of Allegro CL was built by:
  [TC8101] SRI International

;; Optimization settings: safety 1, space 1, speed 3, debug 1.
;; For a complete description of all compiler switches given the
;; current optimization settings evaluate (explain-compiler-settings).
;;---
;; Current reader case mode: :case-sensitive-lower
pvs(1):
pvs(2):

thml :

  |-----
  {1}  valid(A /\ B => A)

Rule? █
--:** *pvs* (ILISP :ready)--L22--All-----
X aussagenlogik typechecked in 0.04s: No TCCs generated; 2 msgs
```

- `Rule?`-Prompt erwartet Beweiserkommandos

- PVS homepage: <http://pvs.csl.sri.com>
- Dokumentation:
<http://pvs.csl.sri.com/documentation.shtml>
 - Systembeschreibung
 - Language Reference
 - Prover Guide
 - Tutorials, viele Papers, etc.
- Hilfesystem von PVS: [C-c h](#)
- unsere Kurzanleitung(en)
- Kommilitonen ...