

Übungen zur Logik

SAT-Solver `semprop` und TPTP-Syntax

Login auf der Sun-Halle

Die für diese Vorlesung verwendeten Programme und Werkzeuge laufen zum größten Teil nur unter SUN Solaris (auch die unten beschriebenen). Es ist daher erforderlich, daß alle Teilnehmer der Vorlesung ein Login auf der Sun-Halle haben.

Studierende, die noch keinen solchen Account besitzen, wenden sich bitte an Gerwin Klein (E-Mail: `kleing@in.tum.de`, Telefon: 289-28490, Raum-Nr.: 1528).

Es gibt ein Verzeichnis zur Vorlesung unter `/usr/proj/logik/ss01`. Dort befinden sich die verwendeten Programme (unter `bin`, am besten `/usr/proj/logik/ss01/bin` mit in die Environment-Variablen `$PATH` aufnehmen!).

SAT-Solver `semprop`

`semprop` ist ein SAT-Solver, basierend auf dem Davis-Putnam-Verfahren, und wurde von Reinhold Letz geschrieben. Er akzeptiert als Eingabe eine Formel in konjunktiver Normalform. Die Eingabedatei erhält pro Zeile eine LISP-Liste der Form $(l_1 l_2 \dots l_n)$, l_i ist entweder eine positive oder negative ganze Zahl und repräsentiert entsprechend ein positives oder negatives Literal.

Eine Liste repräsentiert die Disjunktion ihrer Literale, die gesamte Datei die Konjunktion der Listen.

Beispiel: die Formel $(x_2 \vee x_5 \vee x_7) \wedge (\neg x_2 \vee x_7)$ kann folgendermaßen dargestellt werden:

```
(2 5 7)
(-2 7)
```

Bei großen Ausdrücken empfiehlt es sich ein Programm (in einer beliebigen Programmiersprache) zu schreiben, das den Ausdruck erzeugt.

Ohne Optionen überprüft `semprop`, ob die Formel unerfüllbar (unsatisfiable) oder erfüllbar (satisfiable) ist, und gibt im zweiten Fall ein mögliches Modell an. Mit der Option `-allmodels` kann man erreichen, daß *alle* Modelle ausgegeben werden.

`semprop` wird mit dem Befehl `semprop <file>` bzw. `semprop -allmodels <file>` aufgerufen.

TPTP-Syntax

Die Syntax von `semprop` ist für den Benutzer schwer lesbar und es ist auch nicht möglich, damit Formeln zu beschreiben, die sich nicht in konjunktiver Normalform befinden. Deshalb benutzen wir im folgenden auch noch eine andere Syntax—die sogenannte TPTP-Syntax—zur Beschreibung aussagenlogischer Formeln, die später auch für die Prädikatenlogik erweitert werden wird.

In TPTP-Syntax gibt es prinzipiell zwei Möglichkeiten, eine Formel zu beschreiben: in konjunktiver Normalform oder in allgemeiner Form. Wir werden im folgenden die Syntax für die allgemeine Form beschreiben.

Eine TPTP-Datei `<file>.p` besteht aus einer Folge von Ausdrücken der form

```
input_formula(<name>,hypothesis,<formula>).
```

Dabei ist `<name>` eine beliebige Folge von Zeichen. `<formula>` ist ein aussagenlogischer Ausdruck, der aus (kleingeschriebenen!) Atomen, Klammern, dem Präfix-Operator `~` (not, \neg) und den Infix-Operatoren `|` (or, \vee), `&` (and, \wedge), `<=>` (Biimplikation, \leftrightarrow), `=>` (Implikation, \rightarrow), `<~>` (xor), `~|` (nor), `~&` (nand) besteht.

Der gesamte von `<file>.p` beschriebene Ausdruck ist dann die Konjunktion der einzelnen Formeln.

Beispiel: Der Ausdruck $\neg(A1 \leftrightarrow (B \rightarrow A2)) \wedge (B \vee (C \wedge A1))$ kann folgendermaßen dargestellt werden:

```
input_formula(formel1,hypothesis,
              ~(a1 <=> (b => a2))).
```

```
input_formula(formel2,hypothesis,
              b | (c & a1)).
```

Umwandeln von TPTP-Syntax nach semprop

Ein Ausdruck in der oben beschriebenen TPTP-Syntax kann mit Hilfe von `tptp2sem_cls` in die Syntax von `semprop` umgewandelt werden. Enthalten in dieser Umwandlung ist die Konvertierung des Ausdrucks in konjunktive Normalform, da `semprop` nur diese Form akzeptiert.

Der Konverter wird folgendermaßen aufgerufen:

```
tptp2sem_cls <infile>.p > <outfile>
```

`tptp2sem_cls` weist den vorkommenden Atom entsprechend ihrer alphabetischen Reihenfolge natürliche Zahlen (beginnend mit 1) zu.

Vorsicht: die Umwandlung kann sehr aufwendig sein und die Größe der Formel exponentiell aufblähen.