

# SYNTAX UND SEMANTIK DER AUSSAGENLOGIK

## Thesepapier

Nach dem Einbinden der Vorarbeiten von HARRISON vermöge des Befehls:

```
1 # #use "init.ml";;
```

muss noch der Parser auf aussagenlogische Formel umgestellt werden:

```
1 # let default_parser = parse_prop_formula;;
```

### Zeichentabelle

| Deutsch                  | Symbolisch            | ASCII                | OCaml    |
|--------------------------|-----------------------|----------------------|----------|
| falsch                   | $\perp$               | false                | False    |
| wahr                     | $\top$                | true                 | True     |
| nicht $p$                | $\neg p$              | $\sim p$             | Not p    |
| $p$ und $q$              | $p \wedge q$          | $p /\ \ q$           | And(p,q) |
| $p$ oder $q$             | $p \vee q$            | $p \ \backslash \ q$ | Or(p,q)  |
| $p$ impliziert $q$       | $p \Rightarrow q$     | $p ==> q$            | Imp(p,q) |
| $p$ genau dann, wenn $q$ | $p \Leftrightarrow q$ | $p <=> q$            | Iff(p,q) |

Formeln werden in ASCII-Notation in frz. Anführungszeichen eingegeben: <<...>>.

### Syntax

```
1 type ('a)formula = False
2 | True
3 | Atom of 'a
4 | Not of ('a)formula
5 | And of ('a)formula * ('a)formula
6 | Or of ('a)formula * ('a)formula
7 | Imp of ('a)formula * ('a)formula
8 | Iff of ('a)formula * ('a)formula
9 | Forall of string * ('a)formula
10 | Exists of string * ('a)formula;;
```

## Semantik

```
1 let rec eval fm v =  
2   match fm with  
3     False -> false  
4   | True -> true  
5   | Atom(x) -> v(x)  
6   | Not(p) -> not(eval p v)  
7   | And(p,q) -> (eval p v) & (eval q v)  
8   | Or(p,q) -> (eval p v) or (eval q v)  
9   | Imp(p,q) -> not(eval p v) or (eval q v)  
10  | Iff(p,q) -> (eval p v) = (eval q v);;
```

## Nützliche Funktionen

| Befehl              | Funktion                                                                                                                                                  |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| atoms fm            | Gibt für eine Formel fm die Menge ihrer Atome zurück.                                                                                                     |
| print_truthtable fm | Schreibt die Wahrheitstafel zur Formel fm.                                                                                                                |
| tautology fm        | Entscheidet, ob fm eine Tautologie ist.                                                                                                                   |
| psubst subfn        | Ersetzt Atome $x$ durch Formeln $\varphi$ für geg. Formeln fm.<br>subfn hat die Form: $(P"x" \Rightarrow \langle\langle\varphi\rangle\rangle) \text{ fm}$ |

## Zwei Substitutionssätze

**Satz.** Für jede atomare Aussage  $x$  und beliebige Formeln  $p$  und  $q$  sowie eine beliebige Bewertungsfunktion  $v$  gilt:

$$\text{eval}(\text{psubst}(x \Rightarrow q) p) v = \text{eval } p((x \mapsto \text{eval } q v) v).$$

**Satz.** Seien  $p$  und  $q$  Formeln und  $v$  eine Bewertungsfunktion, sodass  $\text{eval } p v = \text{eval } q v$ . Dann gilt für jedes Atom  $x$  und jede Formel  $r$ :

$$\text{eval}(\text{psubst}(x \Rightarrow p) r) v = \text{eval}(\text{psubst}(x \Rightarrow q) r) v.$$