Vejledning

D602F13

24. juni 2013

1 Forudsætninger

Programmet er skrevet i programmeringssproget Haskell. For at afvikle programmet kræves en interaktiv Haskell-fortolker á la GHCi. Vi har udviklet og testet programmet i GHCi på en Linux platform.

GHC er en forkortelse for "Glasgow Haskell Compiler", mens i'et i GHCi stor for "interactive". GHC samt GHCI kan downloades via pakkemanageren i stort set alle Linux distributioner.

Programmet indeholder en lille parser, som gør det nemt at indtaste og evaluerer udsagn direkte i terminalen. Parseren er bygget vha. det generiske Parsec-bibliotek i Haskell. Når GHC samt GHCi er installeret kan Parsec-biblioteket nemt installeres via kommandolinjen på følgende måde:

```
cabal install parsec
```

2 Indhold i arkiv-fil

Kildekoden kan downloades via `http://dhil.net/public/edu/aau/d602f13/TheoremProver.tar.gz` som et tar.gz-arkiv. Arkivet indeholder følgende

```
TheoremProver.tar.gz
 \-TheoremProver
  |-Parser.hs    [Blackbox]
  |-Lexer.hs     [Blackbox]
  |-Evaluator.hs [Blackbox]
  |-Language.hs
  |-Tablaeu.hs
```

Det er kun de to filer "Language.hs" og "Tableau.hs" der er relevante og refereres til i rapporten. De resterende filer udgør blot den omtalte parser, og kan betragtes som en "gimmick".

3 Afvikling af programmet

Først download og udpak arkiv-filen, dernæst start en terminal op og skift arbejdssti til den netop udpakkede folder. Herefter start GHCi ved at indtaste kommandoen "ghci" i terminalen og endelig benyt GHCi-kommandoen ":l" til at indlæse filen "Evaluator.hs". Følgende er et eksempel på opstart af GHCi, samt indlæsning af Evaluator.hs.

```
[user@host TheoremProver]$ ghci
GHCi, version 7.6.3: http://www.haskell.org/ghc/  :? for help
Loading package ghc-prim ... linking ... done.
Loading package integer-gmp ... linking ... done.
Loading package base ... linking ... done.
ghci>:l Evaluator.hs
[1 of 5] Compiling Language         ( Language.hs, interpreted )
[2 of 5] Compiling Tableau          ( Tableau.hs, interpreted )
[3 of 5] Compiling Lexer            ( Lexer.hs, interpreted )
[4 of 5] Compiling Parser           ( Parser.hs, interpreted )
[5 of 5] Compiling Evaluator        ( Evaluator.hs, interpreted )
Ok, modules loaded: Lexer, Parser, Evaluator, Language, Tableau.
ghci>
```

Nu kan funktionen "eval" benyttes til at konstruere et tableau fra et udsagn. Eksempelvis

```
ghci>eval "p -> ((p -> bot) -> bot)"
```

hvilket udskriver tableauet i en "fladstruktur". Ved at kombinere "eval" med funktionen "isSuccessful", kan vi få at vide om tableauet er vellykket eller ej.

```
ghci>isSuccessful $ eval "p -> ((p -> bot) -> bot)"
True
ghci>
```

4 Syntaks

Det sprog vi har implementeret har en simpel og enkel syntaks.

- Alle udsagn skal have et lowercase begyndelsesbogstav.
- bot er nøgleordet for $\bot$.
- -> er den syntaktiske konstruktion for $\rightarrow$.
- and er nøgleordet for $\wedge$.
- or er nøgleordet for $\vee$.