Übungsaufgaben automatisch korrigieren mit autotool

Johannes Waldmann
Institut für Informatik
Universität Leipzig

Aufgabe (WS 00/01)

2.) Geben Sie kontextsensitive Grammatiken für die folgenden Sprachen an:

b) $\{a^{2^n} | n \in N\}$ c) $\{a^{n^2} | n \in N\}$

Lösung

```
student = Grammatik
   { terminale = "a"
   , nichtterminale = "SHKFLRAB"
   , startsymbol = 'S'
   , regeln = [ ("S", "a"), ("S", "aa") -- kleine Wörter von Hand
              , ("S", "LHAR") -- hier gehts los
              , ("HA", "BBH") -- nach rechts fahren
              , ("HR", "KR")  -- umschalten
              , ("BK", "KA") -- nach links fahren
              , ("LK", "LH") -- zurückschalten oder
              , ("LHA", "aaF") -- finalisieren
              , ("FA", "aaaaF") -- nach rechts fahren
              , ("FR", "aa")  -- abschalten
              ]
   }
```


Antwort

```
Das ist eine Typ-0-Grammatik
Das ist eine monotone Grammatik.
Das ist eine Typ-1-Grammatik

Ihre Grammatik G ist ...

Ich zeige Ihnen jetzt einige Ableitungen in G

in 0 Schritten ableitbar: ["S"]
in 1 Schritten ableitbar: ["LHAR","a","aa"]
in 2 Schritten ableitbar: ["LBBHR","aaFR"]
...
Die damit erzeugten Wörter aus Terminalzeichen sind
in 0 Schritten ableitbar: []
in 1 Schritten ableitbar: ["a","aa"]
...
Ich vergleiche jetzt Ihre Sprache L(G) mit der Zielsprache
und berücksichtige nur Wörter bis zur Länge 16,
die durch Ableitungen enstehen, bei denen alle Wortformen
höchstens die Länge 16 haben.

Ich prüfe, ob alle (kleinen) Wörter in L(G) zur Zielsprache gehören.
Das ist der Fall.

Ich prüfe, ob alle (kleinen) Wörter der Zielsprache zu L(G) gehören.
Das ist der Fall.

Glückwunsch. Soweit ich sehen kann,
haben Sie die Aufgabe gelöst.
```


Fehlermeldung

```
...
Ich prüfe, ob alle (kleinen) Wörter in L(G)
zur Zielsprache gehören.
Wenigstens diese w in L(G) sind falsch ["aaaaaaaaaaaa"]

Ich denke, Sie haben damit die Aufgabe nicht gelöst.
Denken Sie drüber nach,
und probieren Sie es einfach nochmal.

Wenn Sie Hilfe benötigen, oder einen Fehler
im Korrekturprogramm vermuten,
dann schreiben Sie mir bitte eine Mail.
```

autotool–Themen

Automaten und Sprachen:

- Grammatiken (alle Typen)
- Kellerautomaten
- Turingmaschinen

Berechenbarkeit:

- primitiv und partiell rekursive Funktionen

autotool–Dokumentation

http://www.informatik.uni-leipzig.de/~autotool/doc/fun.html

```
plus = Ind 2 [ Proj 1 1 , Sub 3 [ Succ 1, Proj 3 3 ] ]
mal  = Ind 2 [ Zero 1   , Sub 3 [ plus, Proj 3 1, Proj 3 3 ] ]
hoch = Ind 2 [ Sub 1 [ Succ 1, Zero 1 ]
             , Sub 3 [ mal, Proj 3 1, Proj 3 3 ] ]
```

Zero k	die k-stellige Null-Funktion
Succ 1	die einstellige Nachfolger-Funktion
Proj i j	die Projektion (eines i-Tupels auf die j-te Komponente)
Builtin i “name”	eine i-stellige eingebaute Funktion
Sub i [h, g1, ..]	In h werden die Funktionen g1, .. eingesetzt (es entsteht eine i-stellige Funktion)
Ind i [g, h]	die i-stellige Funktion, die durch das Induktionsschema ...
Min i [g]	die i-stellige Funktion, die durch die Minimum-Bildung ... entsteht

Häufig gefragte Fragen (Beispiel)

http://www.informatik.uni-leipzig.de/~autotool/doc/faq.html

```
Unresolved top-level overloading

Fehler:

Type checking
ERROR "Student****.hs" (line **):
    Unresolved top-level overloading
*** Binding             : student
*** Outstanding context : (Num Fun, Enum Fun)

Ursache:

student =
    Ind 1 [ Sub 0 [ Builtin 2 "paar"
                  , Sub 0 [ succ 1, Zero 0 ] , ...

Der Bezeicher "succ" bezeichnet gar nicht das,
was Sie meinen.

Abhilfe: Es muß Succ (mit großem Anfangsbuchstaben)
         heißen.
```


autotool–Architektur

Ablauf:

- Mail–Interface
- Verwalter
- Korrektor (je Aufgabe)
- Simulator (je Modell)

Quelltext: autotool/

- bin/ , config/, doc/
- fun/ , pda/, grammatik/, turing/, util/
- ss01/aufgaben/, ss01/loesungen/
- work/, log/

autotool – Eingebettete Sprachen

eingesandte Lösungen sind
Haskell–Ausdrücke:

Vorteile:

- benutzen Syntax (Parser) und Typsystem von Haskell
- nicht ad hoc, sondern einheitlich

Nachteile:

- Studenten müssen etwas Haskell–Syntax kennen
- teilweise unspezifische Fehlermeldungen

Synergien

autotool–Module als Praktikum fktl. Prog.

Simulator für (primitiv) rekursive Funktionen = Interpreter für fktl. Sprache → Beispiel für Semantik–Vorlesung

```
really_eval0 :: Builtins -> Exp -> Trace Cache Integer
really_eval0 b e = case e of
        ...

        App (Succ 1) [ arg ] -> do
            x <- eval b arg
            return $ x + 1

        App (Proj oben unten) args -> do
            x <- eval b $ args !! (unten - 1)
            return x


        App f @ ( Ind a [ g, h ] ) args -> do
            let (xs, [y]) = splitAt (a - 1) args
            yy <- eval b y
            if 0 == yy
                then eval b $ App g xs
                else do let prev = xs ++ [ Zahl $ yy - 1 ]
                        eval b $ App h $ prev ++ [ App f prev ]
```


Typ–Checker für rekursive Funktionen

```
typeof f @ (Ind a fs) = do

    when ( a < 0 ) $ fail
        $ "negativ-stellige Funktionen gibt es nicht: "
        ++ show f

    when ( length fs /= 2 ) $  fail
        $ "Ind benötigt genau zwei Argumente: "
        ++ show f

    [ t1, t2 ] <- sequence $ map typeof fs

    when ( t1 /= a - 1 ) $ fail
        $ "Durch Ind soll eine " ++ show a
        ++ "-stellige Funktion entstehen,\n"
        ++ " aber das erste Argument ist "
        ++ show t1 ++ "-stellig: " ++ show f

    when ( t2 /= a + 1 ) $ fail
        $ "Durch Ind soll eine " ++ show a
        ++ "-stellige Funktion entstehen,\n"
        ++ " aber das zweite Argument ist "
        ++ show t2 ++ "-stellig: " ++ show f

    return a
```

Mail/Web–Server–Programmierung

Sicherheit

Angriffspunkte

- IO–Aktionen
- Haskell: unsafePerformIO
- procmail/shell
- Maschine überlasten
- Mails an Unbeteiligte

Vorsichtsmaßnahmen

- strenges Typsystem
- Verbot von “import IOExts”
- separater Account (/Rechner)
- ulimits
- Benutzer–Liste

Authentifizierung

Ist der Absender wirklich er selbst?

Hat er die Arbeit selbst angefertigt?

Darf autotool (Teile von) Prüfungen korrigieren?

Wie weiter?

neue Aufgaben

- Komplexität
- endliche Automaten, reguläre Ausdrücke
- Mengenlehre? Logik?

Test–Modus

Dokumentation (auch für private Installation)

Web–Interface