Dokumentation Ableitungsprogramm in C++

von: Marco Möller
Matr. Nr: 23220320
Stand: 29.04.04 - 17:16:22
für: Übungsaufgabe I (Theoretische Informatik 2)

Einleitung

Das folgende Programm kann über den Kommandozeilenaufruf

```
g++ -o gramma main.cpp
```
compiliert, und über

```
./gramma
```
gestartet werden. In der Datei ./grammatik.conf werden alle nötigen Informationen für die Grammatik gespeichert. Diese Datei wird beim Start auf syntaktische Korrektheit geprüft. Nun wird die Grammatik angezeigt und nach den Bedingung, die eine Grammatik erfüllen muss, untersuch. In beiden Schritten wird bei Misserfolg eine entsprechende Meldung ausgegeben, und das Programm abgebrochen. Danach wird der Benutzer aufgefordert ein Wort einzugeben. Wenn dieses ausschließlich aus Variablen und Terminalen besteht, werden von diesem nun alle möglichen 1-Schritt Ableitungen erzeugt und ausgegeben. Als nächstes wird der Benutzer nach einem zweiten Wort gefragt. Für diese wird nun in der Liste der Ableitungen der 1. Eingabe n nach einem Treffer gesucht. Wenn vorhanden wird die Produktion zurückgegeben, mit der diese Ableitungen zustande gekommen ist, wenn nicht eine entspr. Meldung.

Dateiformat

Die Datei ./grammatik.conf hat enthält die zum Ableiten erforderliche Grammatik, und hat folgendes Format.
Hier einmal ein Muster:

```
/ Das hier ist ein nettes kommentar
[V]
A;
S;

[T]
a;
b;
 ;/ Leerzeichen als Symbol
;/Leeres Symbol Epsilon

[P]
S>A|S;/Alernative Ableitungen
S>A,b;
A>b,b,c;
A,b>b,a;

[S]
S;
```

Leerzeilen (OHNE Spaces!!!) werden in der Datei Ignoriert. "/" Kennzeichnet Kommentare bis

Zeilenende ("\n"), in anderen Situationen werden Zeilenenden ignoriert. [*] leitet einen Abschnitt ein. "V" steht für Variablen, "T" für Terminale, "S" für die Startvariable und "P" für die Produktionen. Die einzelnen Elemente dieser Mengen sind mit ";" separiert und können jeweils mehrere druckbare Zeichen aus dem Bereich enthalten (außer ,>|/[];). Bei den Produktionen sind die Einzelnen Zeichen durch "," getrennt, und durch ">" die Linke und Rechte Seite der Produktionen. Durch | werden alternative Rechte Seiten einer Produktion getrennt. "" ist für Epsilon reservier, und kann nur als Terminal aufgenommen werden.

Durch diese Methode ist es möglich sogar Typ-0 Grammatiken zu implementieren, deren Buchstaben aus mehr als einem ASCII-Zeichen bestehen können.

Bedienung

Nach dem Programmstart (./gramma) wird man aufgefordert ein erstes Word einzugeben. Bei diesem müssen die einzelnen Variablen und Terminale mit "," voneinander getrennt sein. Aus diesem werden dann alle Ableitungen gebildet und Angezeigt. Bei der zweiten Abfrage kann nun wieder ein Wort (im selben Format) eingegeben werden. Diese wird nun in den Ableitungen des 1. Wortes gesucht. Bei einem Treffer wird die Produktion ausgegeben, mit der man aus dem 1. Wort das 2.te erhält. Falls keine solche Produktion existiert wird dies ebenfalls mitgeteilt.

Beispiel Programmlauf

```
caller@linux:~/Documents/theoinf/aufgabe2> ./gramma

Grammatik (vom Typ 2 - nach der Chomsky-Hierarchie):

Variablen:      "A,B,C,S"
Startvariable:  "S"

Terminale:      ", ,a,a b,b,c,d"

Produktionen:
"S"     =>      "a"
"S"     =>      "c"
"S"     =>      "a b,c,, "
"S"     =>      "a,A"
"A"     =>      "a"
"A"     =>      ""
"A"     =>      "b"
"A"     =>      "b,C"
"C"     =>      "c,C"


Welches Wort möchtest du abgeleitet haben: S,A,S
Alle Ableitungen von "S,A,S" bzgl. Grammatik

1)      "a,A,S"  ("S"   =>      "a")
2)      "S,A,a"  ("S"   =>      "a")
3)      "c,A,S"  ("S"   =>      "c")
4)      "S,A,c"  ("S"   =>      "c")
5)      "a b,c, ,A,S"    ("S"   =>      "a b,c,, ")
6)      "S,A,a b,c, "    ("S"   =>      "a b,c,, ")
7)      "a,A,A,S"        ("S"   =>      "a,A")
```

```
8)      "S,A,a,A"        ("S"   =>      "a,A")
9)      "S,a,S"  ("A"   =>      "a")
10)     "S,S"    ("A"   =>      "")
11)     "S,b,S"  ("A"   =>      "b")
12)     "S,b,C,S"        ("A"   =>      "b,C")


Welches Wort möchtest du gerne aus deiner Eingabe erzeugen: S,S

Passende Produktion gefunden:
"A"     =>      ""

caller@linux:~/Documents/theoinf/aufgabe2>
```