

# Attributierte Grammatiken

O.Univ.-Prof. Dipl.-Ing. Dr. Wolfgang Pree  
Universität Salzburg  
[www.SoftwareResearch.net](http://www.SoftwareResearch.net)

© Copyright Software Research Lab, All Rights Reserved



## Inhalt

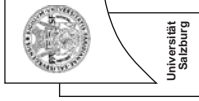
- Merkmale attributierter Grammatiken
- Beispiel



# Merkmale attributierter Grammatiken (ATG)

3

© 2003, W. Pree

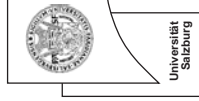


## Allgemeine Merkmale von ATG

- Datenorientierte Entwurfsmethode
- Grundidee: Eingangsinformation steuert Verarbeitungsprozess maßgeblich
- Beschreibung der Daten durch eine Grammatik
- Erweitern der Grammatik durch "Übersetzungsaktionen"
- Verwendung bewährter Techniken des Übersetzerbaus (lexikalische Analyse, Syntaxanalyse)

4

© 2003, W. Pree



## Wann sind ATG anwendbar?

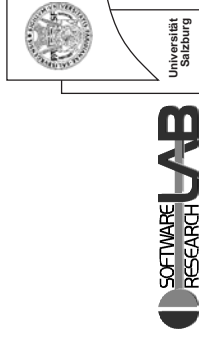
- wenn im wesentlichen *ein* Eingabestrom vorliegt
- wenn Eingabeinformation genügend strukturiert ist

ATG sind geeignet für

- Entwurf von Programmen
- Spezifikation von Programmen
- Dokumentation von Programmen

5

© 2003, W. Pree



## Algorithmische Interpretation von ATG (I)

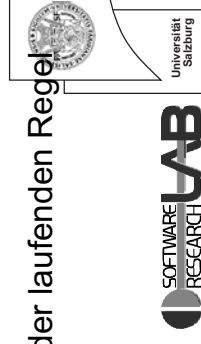
- Grammatikregeln werden als Erkennungsprozesse aufgefaßt.
- Das aus der Grammatik abgeleitete Programm heißt *Syntaxanalyser* (*Parser*)
- Es setzt einen *lexikalischen Analyser* voraus, der Terminalsymbole erkennt und liefert.

Arbeitsweise des Syntaxanalyzers

- Jede Ableitungsregel wird von links nach rechts durchlaufen
- Zur Alternativenauswahl wird das nächste Eingabesymbol verwendet
- Bei jedem Terminalsymbol fordert der Syntaxanalyser vom lexikalischen Analyser das nächste Eingabesymbol an
- Bei jedem Nonterminalsymbol
  - unterbricht der Syntaxanalyser die Abarbeitung der laufenden Regel
  - arbeitet das angetroffene Nonterminalsymbol ab
  - setzt dann an der Abbruchstelle fort.

6

© 2003, W. Pree



## Semantische Aktionen in Ableitungsregeln

Schreibweise: Pseudocode zwischen "sem" und "endsem"

Beispiel 1: Zählung der Buchstaben eines Wortes

```
Wort = Buchstabe    sem b:=1 endsem
{ Buchstabe    sem b:=b+1 endsem
}    sem Write(b," Buchstaben" ) endsem.
```

Beispiel 2: Binärzahl in umgekehrte Reihenfolge transformieren

```
BinZahl = "0" [BinZahl]    sem Write("0" ) endsem
| "1" [BinZahl]    sem Write("1" ) endsem.
```

7

© 2003, W. Pree

## Semantische Werte der Grammatiksymbole

- **Ausgangsattribute**
  - ! entstehen bei der Erkennung eines Symbols
  - ! entsprechen Ausgangsparametern
- **Eingangsattribute**
  - ! werden einem Symbol zu seiner Erkennung mitgegeben
  - ! entsprechen Eingangsparametern

- **Beispiele**

Ziffer ↑ wert

Telegramm ↑ words ↑ longWords

BewegungsSatz ↓ stammNr

8

© 2003, W. Pree

## Entwurfsvorgang mit ATG

- Syntaktische Struktur der Eingabe als kontextfreie Grammatik darstellen
- Attribute für den Verarbeitungsprozeß finden
  - Semantische Werte von Symbolen überlegen
  - Angabe mit Name, Typ und Bedeutung
- Attributierte Grammatiksymbole spezifizieren
- Semantische Prozeduren spezifizieren
  - ausserhalb der Grammatik definierte Prozeduren
  - Attributierte Grammatik aufstellen

## ATG-Beispiel: Bewegung eines Roboterarms

## ATG-Beispiel

### Verbale Formulierung

- Die Eingabe beginnt mit den Koordinaten des Startpunkts, auf die beliebig viele lineare und kreisförmige Teilbewegungen folgen können. Die Eingabe wird mit dem Code "E" abgeschlossen.
- Die Beschreibung einer linearen Teilbewegung beginnt mit der Anzahl der Einzelschritte, die benutzt werden sollen, um den Zielpunkt anzusteuern. Darauf folgen der Code "L" und die Koordinaten des Zielpunkts.
- Die Beschreibung einer Kreisbewegung beginnt ebenfalls mit der Anzahl der (linearen) Einzelschritte, auf die der Code "C" und die Koordinaten eines Übergangspunktes und des Zielpunkts folgen. Der Zielpunkt soll in einer Kreisbahn angesteuert werden, auf der der Übergangspunkt liegt.

11

© 2003, W. Pree

## ATG-Beispiel

- Punkt-Koordinaten werden durch drei ganze Zahlen zwischen 0 und 2000 angegeben, die in runde Klammern eingeschlossen und durch Kommas voneinander getrennt werden. Jede Zahl bedeutet eine absolute Millimeterangabe. Wenn ihr ein "+" oder "-" vorangestellt ist, bedeutet die Zahl eine (zum jeweiligen Ausgangspunkt der Teilbewegung) relative Distanz.
- Beispiel:  
(200,100,100)  
5 L (-100,+200,0)  
20 C (+700,+700,+1200) (+0,+1400,+0)  
E

12

© 2003, W. Pree

## ATG-Beispiel

- Bedeutung:  
Start beim Punkt  
     $x=200, y=100, z=100$ .  
Lineare Bewegung in 5 Schritten zum Punkt  
     $x=100, y=300, z=0$ .  
Kreisbewegung in 20 Schritten zum Punkt  
     $x=100, y=1700, z=0$ ,  
wobei der Punkt  
     $x=800, y=1000, z=1200$   
auf der Kreisbahn liegen soll.
- Nachteile von verbalen Beschreibungen:
  - oft unklar; nur durch Beispiele verständlich
  - ungenau (wo sind Leerzeichen vorgeschrieben, bzw. wo dürfen welche stehen?)
  - Formulierungenregeln mit Nebenbedingungen und Bedeutungsangaben vermischt

13

© 2003, W. Pree

## ATG-Beispiel

Robo = Position { Move } "E" .

Position = "(" Coordinate " , " Coordinate " , " Coordinate ")" .

Move = Number ( "L" Position | "C" Position Position ) .

Coordinate = [ "+" | "-" ] Number .

Number = digit { digit } .

Terminalsymbole: "L", "C", "E", "+", "-", "(", ")", ",", " , ", " , " , " , "

Terminalklasse: digit

Nonterminalsymbole: Robo, Position, Move, Coordinate, Number, Space

14

© 2003, W. Pree

## ATG-Beispiel

Robo erkennt eine vollständige Bewegungsbeschreibung

Position  $\langle x0, y0, z0, x, y, z \rangle$   
erkennt eine Position mit den Koordinaten  $x, y$  und  $z$ .  
 $x0, y0$  und  $z0$  bedeuten die Ursprungsposition, auf die sich relative Koordinatenangaben beziehen.

Move  $\langle x0, y0, z0, x, y, z \rangle$   
erkennt eine Bewegung beginnend bei  $(x0, y0, z0)$  und liefert den Endpunkt  $(x, y, z)$  der Bewegung.

Coordinate  $\langle v0, v \rangle$   
erkennt eine Koordinate mit dem Wert  $v$ .  $v0$  ist ein Ursprungswert, auf den sich relative Angaben beziehen.

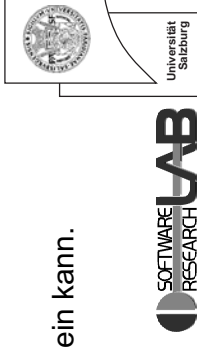
Number  $\langle v \rangle$   
erkennt eine ganze Dezimalzahl mit dem Wert  $v$ .

Space  
erkennt eine beliebig lange Folge von Leerzeichen, die auch leer sein kann.

digit  $\langle d \rangle$  erkennt eine Ziffer mit dem Wert  $d$ .

15

© 2003, W. Pree

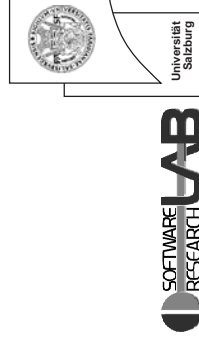


## ATG-Beispiel

```
Robo =  
Space  
Position  $\langle 0, 0, 0, x0, y0, z0 \rangle$   
sem  
    InitMoves( $x0, y0, z0$ )  
endsem  
{ Move  $\langle x0, y0, z0, x, y, z \rangle$   
  sem  
     $x0:=x; y0:=y; z0:=z$   
  endsem  
} "E" Space .
```

16

© 2003, W. Pree





## ATG-Beispiel

```
Position <x0, y0, z0, x, y, z> =  
  "(" Space Coordinate <x0, x>  
  " , "  
  " , "  
  " , "  
  " ) " Space .  
  
Coordinate <v0, v> =  
  sem sign:=' ' endsem  
  [ " + " sem sign:='+' endsem  
  | " - " sem sign:='- ' endsem  
  ]  
  Number <v> sem  
  if sign='+' then v:=v0+v;  
  else if sign='- ' then v:=v0-v  
  endsem  
  .
```

17

© 2003, W. Pree

## ATG-Beispiel

```
Move <x0, y0, z0, x, y, z> =  
  Number <steps>  
  ( " L " Space  
    Position <x0, y0, z0, x, y, z>  
    sem  
    AddLine(x, y, z, steps)  
  )  
  endsem  
  | " C " Space  
    Position <x0, y0, z0, x1, y1, z1>  
    Position <x0, y0, z0, x, y, z>  
    sem  
    AddArc(x, y, z,  
            x1, y1, z1, steps)  
  )  
  endsem  
  .
```

18

© 2003, W. Pree

## ATG-Beispiel

```
Number <v> =  
  digit <d>      sem v:=d endsem  
  {  digit <d>      sem v:=10*v+d endsem  
  }  
  Space .  
  
Space =  
  { " " } .
```

19

© 2003, W. Pree

## ATG-Beispiel

Semantische Routinen:

Datenkapsel Moves zur Verwaltung einer Liste von Teilbewegungen mit folgenden Zugriffsroutinen:

InitMoves(x0, y0, z0)

Initialisiert die Teilbewegungsliste mit dem Startpunkt (x0,y0,z0).

AddLine(x, y, z, steps)

Erweitert die Teilbewegungsliste um eine lineare Bewegung zum Punkt (x,y,z) in steps Schritten.

AddArc(x, y, z, x1, y1, z1, steps)

Erweitert die Teilbewegungsliste um eine kreisförmige Bewegung über den Punkt (x1,y1,z1) zum Punkt (x,y,z) in steps Schritten.

20

© 2003, W. Pree