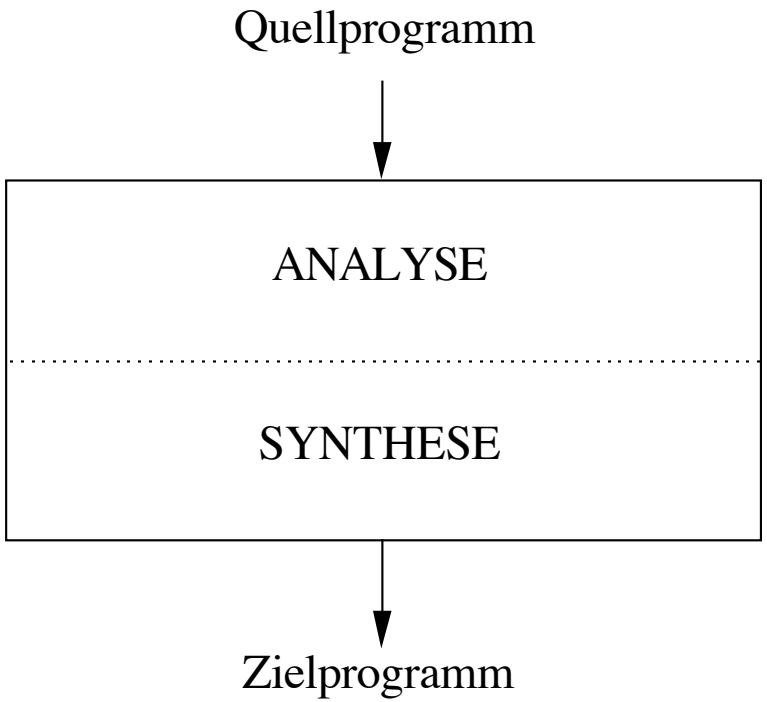
Kapitel 1: Struktur von Compilern

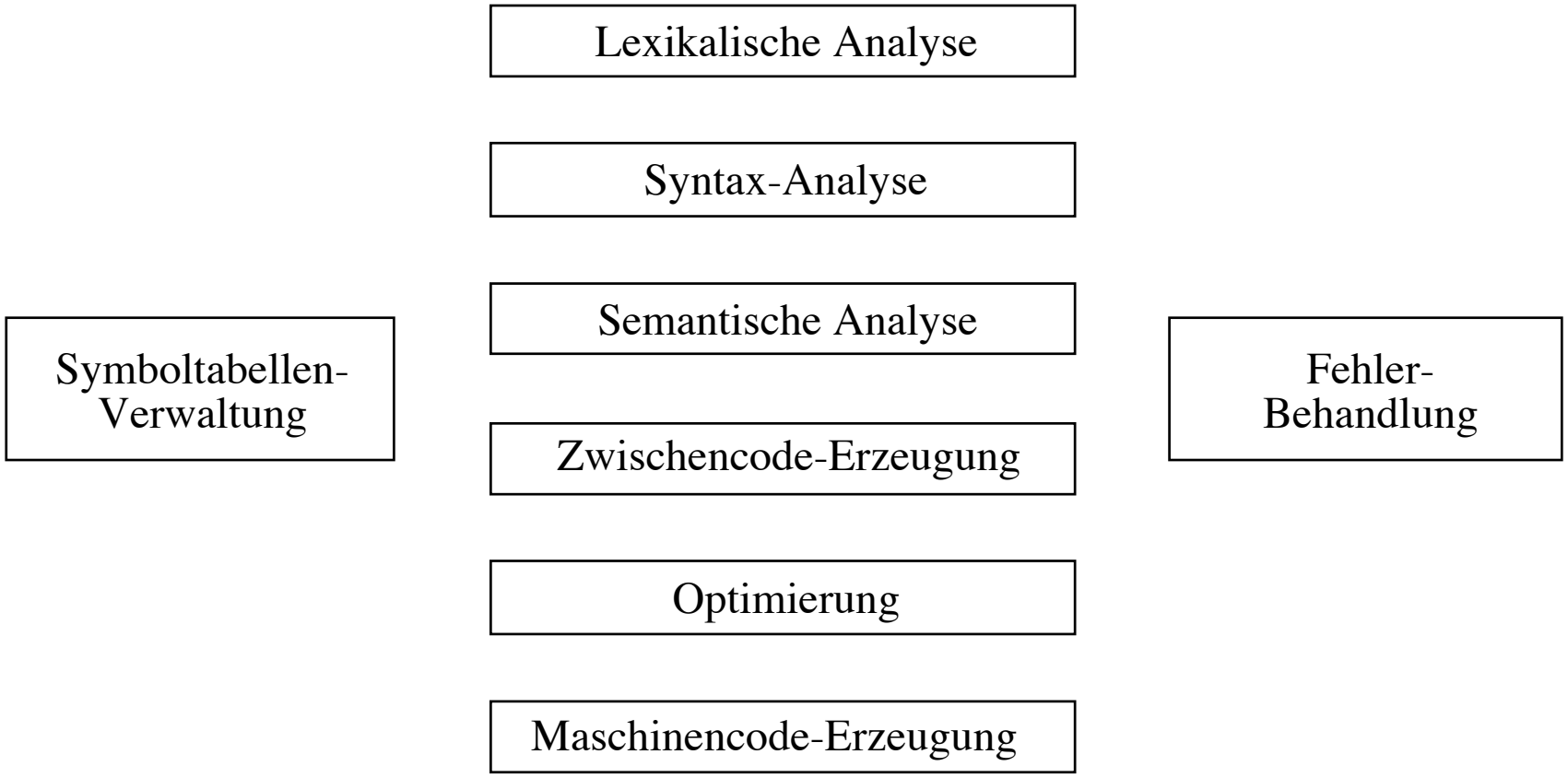
Themen

- Statische Struktur (Teilaufgaben)
- Dynamische Struktur (Einpass/Mehrpass)

Allgemeine Grundstruktur



Teilaufgaben (Phasen)



Lexikalische Analyse

Ausschnitt eines Modula-Quellprogramms

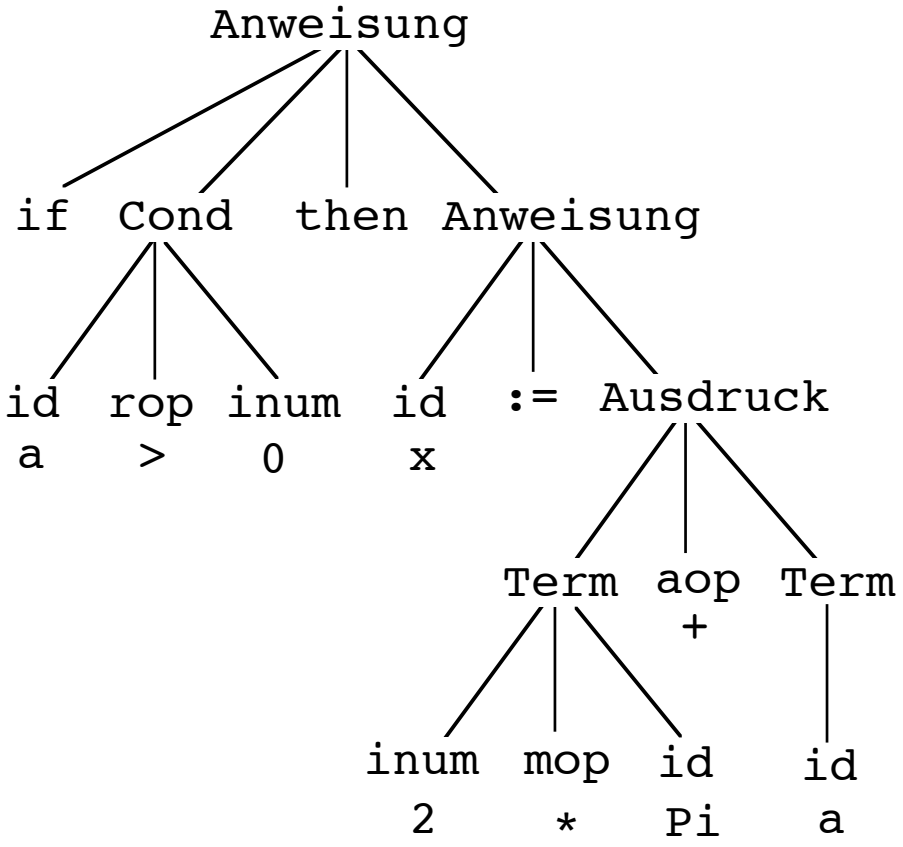
```
... CONST Pi=3.14;
    VAR a:INTEGER; x:REAL;
... IF a > 0 THEN
      x:=2*Pi+a ...
```

Folge von Symbolen (Token)

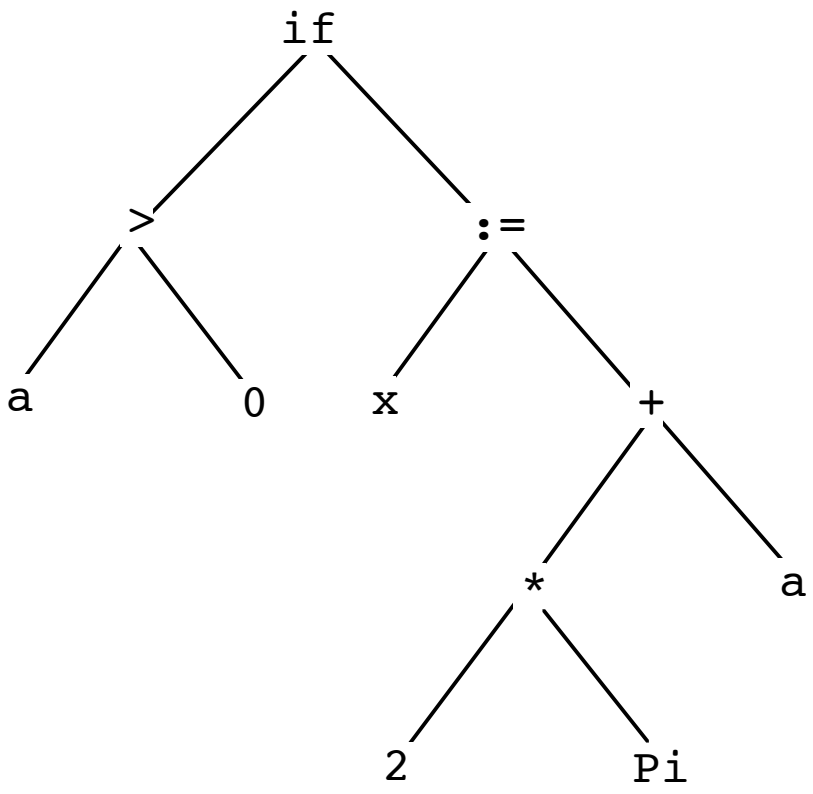
```
... (const)(id,Pi)(=)(rnum,3.14)(;)
    (var)(id,a)(:)(int)(;)(id,x)(:)(real)(;)
... (if)(id,a)(rop,>)(inum,0)(then)
    (id,x)(:=)(inum,2)(mop,*)(id,Pi)(aop,+)(id,a) ...
```

Syntax–Analyse

Ableitungsbaum

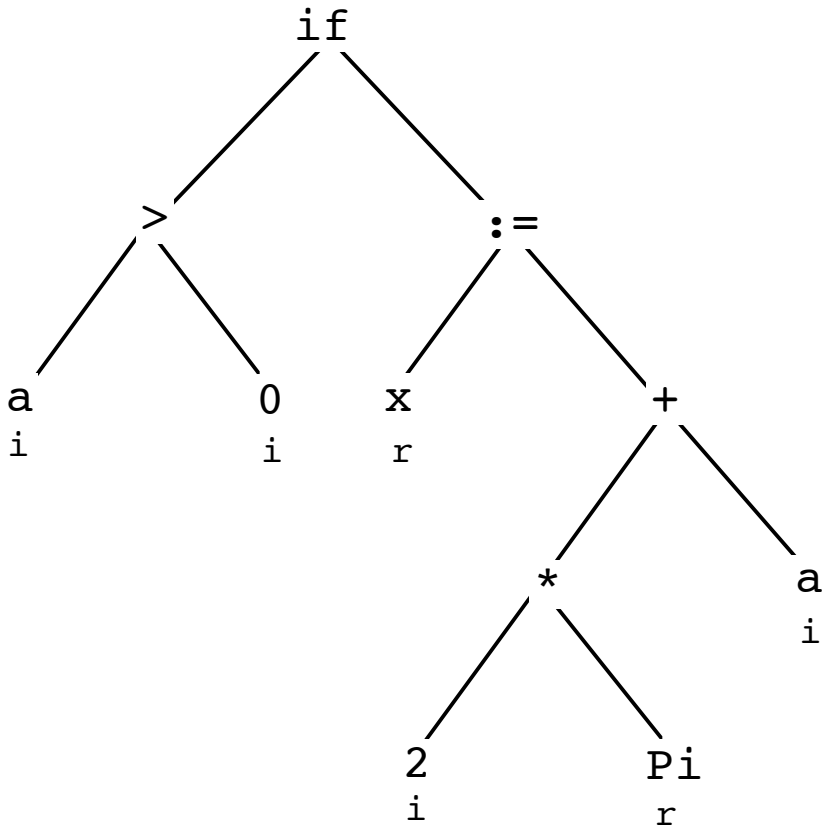


Abstrakter Syntaxbaum



Semantische Analyse

Attributierter Syntaxbaum

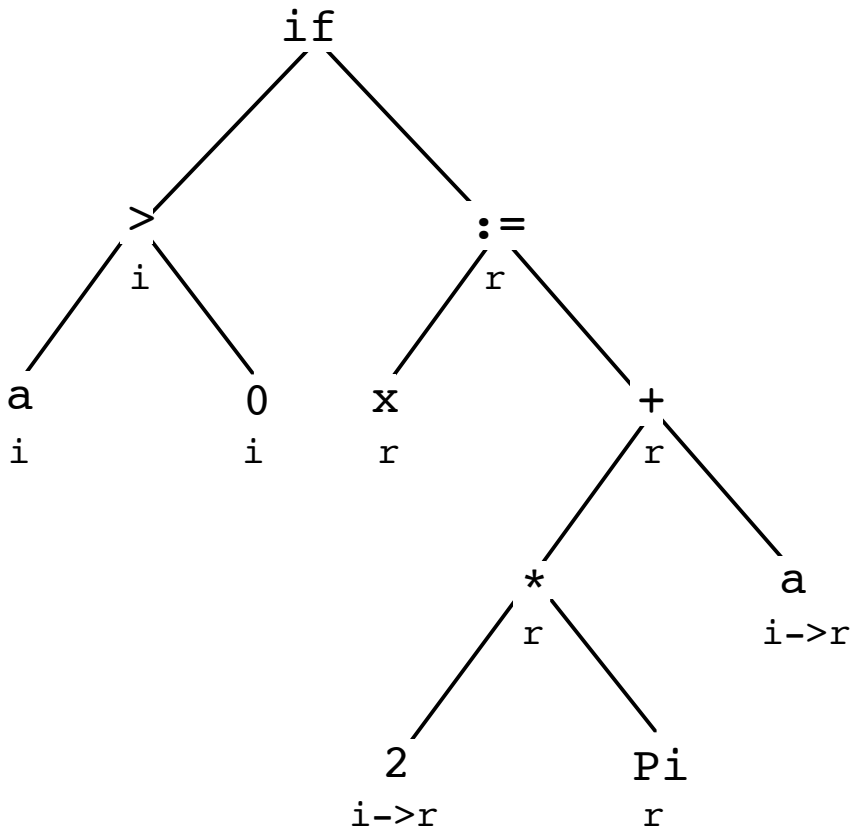


Symboltabelle

Name	Art	Typ	Wert/Adr
Pi	c	r	3.14
a	v	i	0
x	v	r	4

Zwischencode-Erzeugung

Attributierter Syntaxbaum



Linearer Zwischencode

```
   if a >i 0 goto L1
   goto L2
L1:t1 =r intreal(2)
   t2 =r t1 *r 3.14
   t3 =r intreal(a)
   t4 =r t2 +r t3
    x =r t4
L2:
```

Optimierung

Zwischencode

```
    if a >i 0 goto L1
    goto L2
L1:t1 =r intreal(2)
    t2 =r t1 *r 3.14
    t3 =r intreal(a)
    t4 =r t2 +r t3
     x =r t4
L2:
```

Optimierter Zwischencode

```
   if a <=i 0 goto L
   t1 =r intreal(a)
    x =r 6.28 +r t1
L:
```

Maschinencode-Erzeugung

Zwischencode

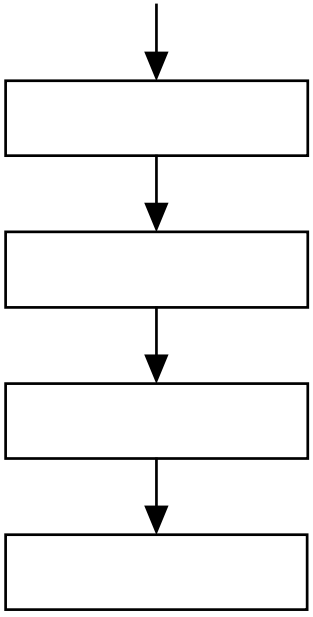
```
   if a <=i 0 goto L
   t1 =r intreal(a)
    x =r 6.28 +r t1
L:
```

Maschinencode (68000)

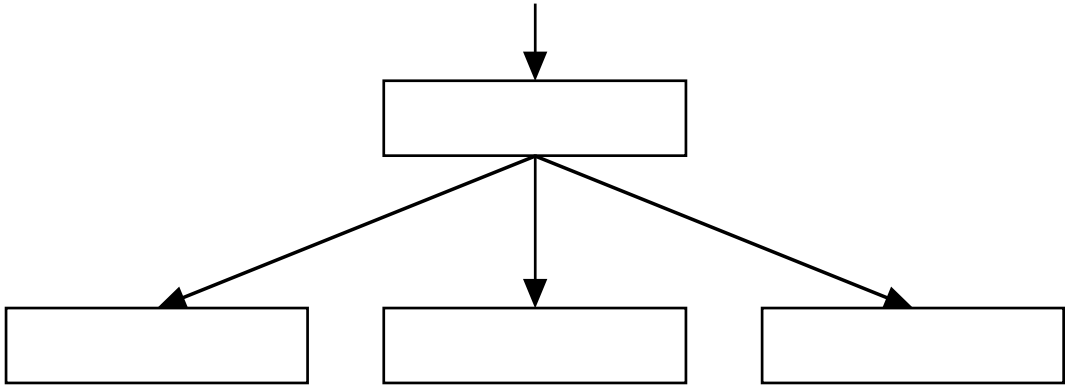
```
   MOVi 0(A),R1
   CMPi #0,R1
   BLE  L
   FLT  R1,F1
   ADDr #6.28,F1
   MOVr F1,4(A)
L:
```

Dynamische Struktur

Mehrpass-Compiler



Einpass-Compiler



Hauptprogramm: Syntax-Analyse