${\bf Programmier praktikum}$

Einfacher? Crosscompiler von Python nach C++

Felix Helsch Julian Buchhorn

29. Dezember 2014

Betreuer Dr. Holger Arndt

Inhaltsverzeichnis

1	Einleitung	3
2	Verwendung	5
3	Programme	5
4	Features	5
5	Codestruktur	5
Αŀ	Abbildungsverzeichnis	
Ta	Tabellenverzeichnis	
Lis	Listings	

1 Einleitung 3

1 Einleitung

Im Rahmen eines Programmierpraktikums an der bergischen Universität Wuppertal haben wir einen Compiler von Python zu C++ geschrieben. Dabei ging es in erster Linie darum, zu verstehen, wie ein Compiler funktioniert und nicht einen vollständigen Compiler zu schreiben.

Als vorläufiges Ziel haben wir un gesetzt ein einfaches Testprogramm mit grundlegenden Syntaxelementen zu übersetzen (s. Lst. 1).

Listing 1: Python Testprogramm für den Compiler

```
from __future__ import division
2
3
   Test program for codegeneration
4
5
6
   def f(x):
7
        """calculate"""
8
        return x**2 + x*4 - (x-3) / x -1.3e2+2
9
                +5 #comment
10
11
12
13
   if __name__ == '__main__ ':
14
15
        L = [0] * 10
16
17
        for i in range(len(L)):
18
19
            if i>3:
20
                 L[i] = f(i+1)
21
             else:
22
                 L[i] += 1
23
        for x in L:
24
25
             print x
26
        print 'program', 'end'
27
```

Das Ziel haben wir erreicht, unser Compiler kann in seinem momentanen Zustand das Python Programm übersetzen und liefert das C++ Programm in Listing 2. Dabei mussten wie erwartet einige Annahmen über das Programm gemacht werden um die Übersetzung zu vereinfachen.

Listing 2: Vom Compiler erzeugtes C++ Programm

```
#include <array>
#include <cmath>
#include <iostream>

using namespace std;
```

1 Einleitung 4

```
7
8
    Test program for codegeneration
9
10
    double f(double x) {
11
12
        /*calculate*/
        return pow(x, 2) + x*4 - (x-3) / (double) x-1.3e2+2
13
14
                 +5; //comment
15
16
17
18
19
    int main() {
20
21
        array < double , 10 > L={};
22
23
         for (int i= 0; i<L.size(); i++) {</pre>
24
             if (i>3) {
25
                  L[i] = f(i+1);
26
             else {
27
                  L[i]++; // com
28
             }
29
        }
30
31
32
        for (auto x : L) {
33
             cout << x << endl;</pre>
34
35
        cout << "program" <<' ' '<< "end" << endl;</pre>
36
37
        return 0;
38
39
    }
40
```

In den folgenden Kapiteln werden wir noch erklären wie man den Compiler verwenden kann, welche Programme man dafür benötigt und welche Features der Compiler im Moment unterstützt. Zum Schluss werden wir noch kurz auf die Struktur des Codes eingehen, den wir geschrieben haben.

Für eine detailliertere Erklärung wird auf die Bachelorarbeit von Felix Helsch verwiesen..., in welcher die Erstellung des Compilers fortgeführt wird.

5 Codestruktur 5

2 Verwendung

3 Programme

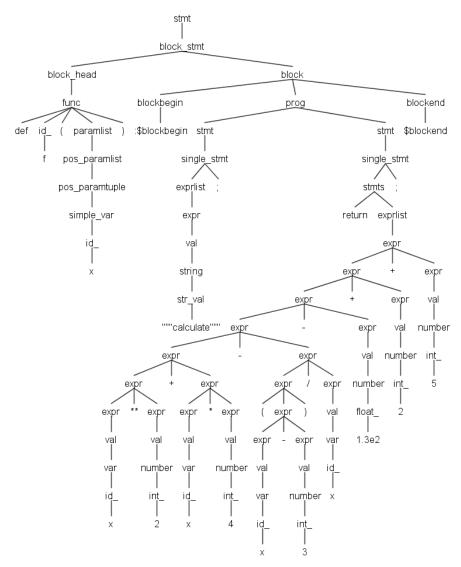


Abbildung 1: Beispiel Parsetree

4 Features

5 Codestruktur

Abbildungsverzeichnis					
Abbi	ldungsverzeichnis				
1	Beispiel Parsetree	5			

Tabellenverzeichnis 7

Tabellenverzeichnis

Listings _____8

		•	
LI	sti	ing	S

1	Python Testprogramm für den Compiler	3
2	Vom Compiler erzeugtes C++ Programm	3