Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Luka Prijatelj

Porazdeljeno upodabljanje v spletu

diplomsko delo

univerzitetni študijski program
prve stopnje
računalništvo in informatika

Mentor: doc. dr. Matija Marolt
Somentor: izr. prof. dr. Ciril Bohak

Ljubljana, 2017

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Besedilo teme diplomskega dela študent prepiše iz študijskega informacijskega sistema, kamor ga je vnesel mentor. V nekaj stavkih bo opisal, kaj pričakuje od kandidatovega diplomskega dela. Kaj so cilji, kakšne metode uporabiti, morda bo zapisal tudi ključno literaturo.

Družini in prijateljem.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
GUI	Graphical User Interface	Grafični uporabniški vmesnik
ALL	Adaptive LL	Prilagodljiv LL
PEG	Parsing Expression Grammar	Razčlenjevalno izrazna gramatika
GCC	GNU Compiler Collection	GNU zbirka prevajalnikov
CFG	Context-Free Grammar	Kontekstno neodvisna gramatika
DCFG	Deterministic Context-Free Grammar	Deterministično kontekstno neodvisna gramatika
YACC	Yet Another Compiler-Compiler	Še en izdelovalnik prevajalnika
CFL	Context-Free Language	Kontekstno neodvisni jezik
XML	eXtensible Markup Language	Razširljiv označevalni jezik
DTD	Document Type Definition	Definicija tipa dokumenta
DCFL	Deterministic Context-Free Language	Deterministično kontekstno neodvisni jezik
ANTLR	ANother Tool for Language Recognition	Še eno orodje za razpoznavo jezika
BNF	Backus-Naur Form	Backus-Naur oblika
EBNF	Extended Backus-Naur Form	Razširjena Backus-Naur oblika
ABNF	Augmented Backus-Naur Form	Povečana Backus-Naur oblika
WSN	Wirth Syntax Notation	Wirth sintaksna notacija
DCF	Definite Clause Grammar	Razločna stavčna gramatika
LALR	LookAhead LR	Vnaprejšnji LR
SLR	Simple LR	Preprosti LR
CYK	Cocke, Younger, Kasami	Cocke, Younger, Kasami
LL	Left to right, performing Left-most derivation	Iz leve v desno izvršujoč najbolj levo izpeljavo

Povzetek

Naslov: Porazdeljeno upodabljanje v spletu

Avtor: Luka Prijatelj

V diplomski nalogi bom predstavil delovanje sintaksnega analizatorja v odprtokodnih prevajalnikih. Prav tako bom opisal in razložil gramatike, ki jih prevajalniki uporabljajo za različne programske jezike. Za opis ter testiranje sintaksnih analizatorjev sem se odločil vzeti zelo popularen prevajalnik imenovan GCC. Čeprav je GCC skupek prevajalnikov za prevajanje različnih programskih jezikov, pa ga bom v diplomski nalogi imenoval kar GCC prevajalnik. GCC prevajalnik bom tudi primerjal in sem zato kot drugi prevajalnik vzel Antlr orodje. Antlr orodje ni celoten prevajalnik tako kot GCC, vendar je le orodje za generiranje parserjev iz gramatik. Ker pa je pri obeh prisoten sintaksni analizator, to ne predstavlja problem. Iz obeh prevajalnikov bom namreč predstavil le sintaksni analizator in gramatiko.

Ključne besede: Gramatike, Sintakse, Analizatorji, Odprtokodnost.

Abstract

Title: Web-based distributed rendering

Author: Luka Prijatelj

In my diploma assignment I will present functions of the syntax analyser in the open-source compilers. I will describe context-free grammars that compilers use for compiling different programming languages. As a subject of study I decided to use GCC compiler. GCC is really a collection of compilers that can compile different programming languages but I will call it simply GCC compiler. Reason for this is simplicity. Because I will examine and compare GCC compiler, I will need second compiler for comparison. I decided to use Antlr tool. Antlr tool is not a typical compiler. It is only a tool for generating parsers from context-free grammars. But because both compilers have syntax analysers this will not be a problem.

Keywords: Grammars, Syntax, Analyzer, Open-Source.

Poglavje 1

Uvod

V diplomski nalogi bom predstavil delovanje GCC prevajalnika in delovanje orodja Antlr. Začel bom z kratkim opisom prevajalnikov in njihovo zgodovino. Na kratko bom predstavil tudi razvoj prevajalnikov. Naslednje poglavje bo govorilo o podrobni analizi GCC prevajalnika. Vključevalo bo opis gramatike in njeno strukturo, opis sintaksne analize, prednosti in slabosti sintaksne analize, testiranje prevajalnika ter časovno zahtevnost. Sledilo bo poglavje o analizi Antlr orodja. Tudi to poglavje bo vključevalo enaka podpoglavja kot pri poglavju Analiza GCC prevajalnika [**??**].
Po podrobnejši analizi bo sledila primerjava gramatik in sintaksnih analiz med GCC prevajalnikom in Antlr orodjem. Naštete in opisane bodo tudi prednosti in slabosti GCC prevajalnika ter Antlr orodja. Sledila bo primerjava časovne zahtevnosti med GCC in Antlr.
V poglavju Rezultati [**??**] bodo opisani rezultati analize in nato bo sledilo še zadnje poglavje Zaključek [**??**].

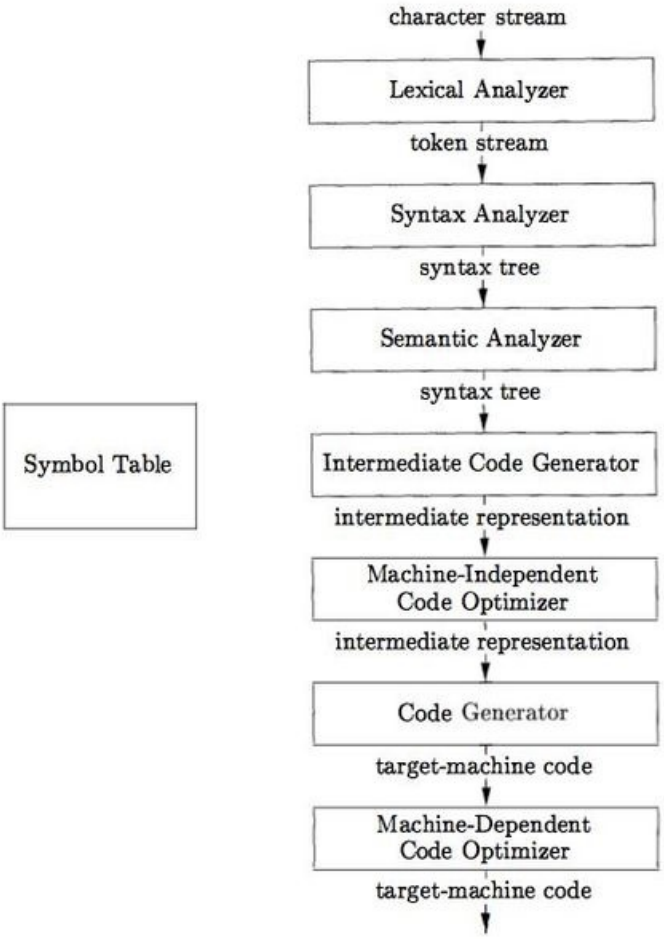
Poglavje 2

Leksikalna analiza

Sintaksni analizator (angl. Syntax Analyzer) je sestavljen iz leksikalne analize in sintaksne analize. To sta prvi dve stopnji prevajalnikovega cevovoda.

Glavne stopnje prevajalnikovega cevovoda so prikazane na sliki [2.1]:

Leksikalna analiza (angl. Lexical Analysis) je proces pretvarjanja niza znakov v niz žetonov. Žeton je niz znakov, ki mu je določen pomen. Program, ki izvaja lexikalno analizo se imenuje lekser (angl. Lexer). Lexer je običajno združen s parserjem, s katerim skupaj analizirata sintakso programskega jezika.



Slika 2.1: Glavne stopnje cevovoda prevajalnika [**?**]

Poglavje 3

Gramatika

Programski jezik je opisan z kombinacijo semantike in gramatike. Semantika določa pomen vsakega konstrukta, ki je možen v programskem jeziku, gramatika pa določa strukturo konstrukta.[**?**] Gramatika - lahko tudi formalna gramatika - je predstavljena kot skupek produkcijskih pravil. Pravila opisujejo, na kakšen način lahko tvorimo nize, da ustrezajo veljavni sintaksi jezika. Gramatika sama po sebi ne opisuje pomena nizov, opisuje le njihovo obliko. S pomočjo gramatike lahko predstavimo določen program v obliki sintaksnega drevesa.

Gramatika je sestavljena iz niza simbolov. Vsak simbol je lahko končen ali nekončen. Končen pomeni, da je žeton abecede nizov programskega jezika. Lahko pa je nekončen kar pomeni, da se pojavi na levi strani določenega pravila. Noben žeton pa se ne more pojaviti na levi strani produkcijskega pravila. [**?**]

- **Backus-Naur oblika (BNF)**
  Backus-Naur oblika ali tudi Backusova normalna oblika je notacijska tehnika za kontekstno neodvisne gramatike. Pogosto se uporablja za opisovanje sintakse jezika v računalništvu. Uporablja se pri: programskih jezikih, formatih dokumentov, naborih ukazov in komunikacijskih protokolih. [**?**]

```
<exp> ::= <exp> "+" <exp>
<exp> ::= <exp> "*" <exp>
<exp> ::= "(" <exp> ")"
<exp> ::= "a"
<exp> ::= "b"
<exp> ::= "c"
```

Izvorna koda 3.1: Primer preproste gramatike v BNF obliki

- **Razširjena Backus-Naur oblika (EBNF)**
  Vsaka gramatika definirana v EBNF obliki je lahko predstavljena v BNF obliki, vendar so običajno gramatike nato daljše. [**?**] Razširjena Backus-Naur oblika gramatike je družina metasintaksnih notacij, ki se uporabljajo za izražanje kontekstno neodvisne slovnice (gramatike).

```
Exp ::= Exp "+" Exp
      | Exp "*" Exp
      | "(" Exp ")"
      | "a"
      | "b"
      | "c"
```

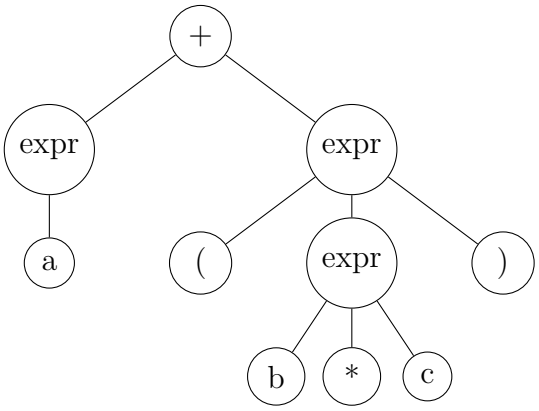
Izvorna koda 3.2: Primer preproste gramatike v EBNF obliki

- **Povečana Backus-Naur oblika (ABNF)**

Izvorni kodi [3.1] in [3.2] vsebujeta preprost primer gramatike artimetičnega izraza, ki je sestavljen iz operatorjev množenja in seštevanja, zapisanega v različnih oblikah gramatike.

3.1 Kontekstno neodvisna gramatika

Kontekstno neodvisna gramatika je določen tip formalne gramatike. Vsebuje produkcijska pravila, ki opisujejo vse možne nize za podan formalni jezik. Produkcijska pravila so preproste zamenjave. V CFG-ju je lahko vsako produkcijsko pravilo lahko le tipa:



Slika 3.1: Primer zgornje gramatike v obliki sintaksnega drevesa

- Ena proti ena
- Ena proti mnogo
- Ena proti nič

Jeziki, ki so generirani s pomočjo kontekstno neodvisne gramatike, se imenujejo kontekstno neodvisni jeziki (Context-Free Languages ali CFL). Različno zapisani CFG-ji lahko generirajo enak kontekstno neodvisni jezik. Zelo dober primer uporabe CFG-ja je pri XML (eXtensible Markup Language), ki uporablja definicijo tipa dokumenta (Document Type Definition ali DTD). V računalniški znanosti je popularen zapis CFG-ja kot BNF.

3.2 Deterministična kontekstno neodvisna gramatika

Je podmnožica kontekstno neodvisne gramatike, ki generira deterministično neodvisen programski jezik (angl. *Deterministic Context-Free Language ali DCFL*). DCFG-ji so vedno nedvoumni in so pomemben podrazred nedvoumnih CFG-jev. So tudi zelo praktični, saj so lahko parsani v linearnem času. Parser je lahko celo avtomatsko generiran iz gramatike z uporabo generatorja parserjev (Yacc, Antlr, Bison, JavaCC).

3.3 Dvoumna gramatika

Dvoumna gramatika je kontekstno neodvisna gramatika za katero lahko obstaja niz, ki ima več kot eno najbolj levo izpeljavo ali parsno drevo. Pri nedvoumni gramatiki, pa ima lahko vsak niz le unikatno najbolj levo izpeljavo ali parsno drevo. Mnogo programskih jezikov dovoli tako dvoumne kot nedvoumne gramatike vendar nekateri programski jeziki pa le nedvoumne.

3.4 Razčlenjevalno izrazna gramatika

Je tip analitično formalne gramatike kar pomeni, da opisuje formalni jezik v odvisnosti od množice pravil za prepoznavanje nizov v jeziku. Razčlenjevalno izrazna gramatika (Parsing Expression Grammar ali PEG) je zelo povezana z družino odzgoraj-navzdol parsnih jezikov. Sintaktično PEG izgleda zelo podobno kot CFG vendar imajo DRUGAČNO interpretacijo. Izbirni operator izbere prvo ujemanje v PEG, medtem ko je izbira dvoumna pri CFG. Ta način je bližje temu, kako so prepoznani nizi v praksi s pomočjo rekurzivno spustnega parserja.

Poglavje 4

Sintaksna analiza

Parsanje oziroma sintaksna analiza je proces analiziranja niza simbolov (žetonov) ali ustrezajo pravilom formalne gramatike. Sintaksna analiza dobi iz leksikalne analize kot vhod niz simbolov iz katerih nato zgradi sintaksno drevo.

<žeton, leksem, lokacija>

Slika 4.1: Prikaz strukture posameznega simbola [**?**]

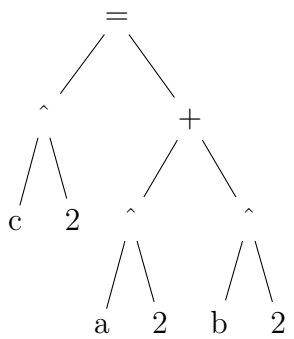
Spodaj je prikazan primer [4.2] preprostega programa za pitagorov izrek in njegova drevesna struktura za sintaktično analizo [4.4].

$$c^2 = a^2 + b^2$$

Slika 4.2: Primer programa za pitgorov izrek

<SPREMENLJIVKA, 'c', [1:1-1:1]>
<POTENCA, '^', [1:2-1:2]>
<KONSTANTA, '2', [1:3-1:3]>
<ENAČAJ, '=', [1:4-1:4]>
<SPREMENLJIVKA, 'a', [1:5-1:5]>
<POTENCA, '^', [1:6-1:6]>
<KONSTANTA, '2', [1:7-1:7]>
<PLUS, '+', [1:8-1:8]>
<SPREMENLJIVKA, 'b', [1:9-1:9]>
<POTENCA, '^', [1:10-1:10]>
<KONSTANTA, '2', [1:11-1:11]>

Slika 4.3: Seznam simbolov za program [4.2]



Slika 4.4: Sintaksno drevo sestavljeno iz simbolov [4.3]

Poglavje 5

Vrste razčlenjevalnih metod

Razčlenjevalniki tipa od zgoraj navzdol razpoznajo oziroma razčlenijo vhod iz oblike programskega jezika v notranjo predstavitev tako, da ujema prihajoče simbole v produkcijska pravila. Produkcijsko pravila so definirana z BNF. Spodaj so našteti in opisani različni parserji, ki delujejo po principu od zgoraj navzdol.

	LL(*)	LL(1)	LL(k)	ALL(*)	Pred-LL(k)	Packart parser (PEG)	Earley parser	RDP- Handwritten
Zbirni jezik				Antlr4	Antlr3			
Action script				Antlr4				
Ada		Coco/R		Antlr4				
Basic				Antlr4				
C		Coco/R		Antlr4	Antlr3			GCC4 CLang APG

Tabela 5.1: Tabela odzgoraj-navzdol razčlenjevalnih metod v povezavi z programskimi jeziki in razčlenjevalniki

Poglavje 6

Razčlenjevalniki

random text

6.1 GCC

random text

6.2 CLion

random text

6.3 Roslyn

random text

6.4 Esprima

random text

6.5 Nashorn

random text

6.6 Chrome V8

random text

6.7 SpiderMonkey

random text

6.8 Rhino

random text

Poglavje 7

Generatorji razčlenjevalnikov

random text

7.1 Antlr

random text

7.1.1 Programski jeziki

random text

7.1.2 Gramatika

random text

7.1.3 ALL(*)

random text

7.1.4 Pred-LL(k)

ranodm text

7.2 Yacc

random text

- Lex
- Flex

7.3 GNU Bison

random text

7.4 JavaCC

random text

Poglavje 8

Sklepne ugotovitve

random text