Um compilador simples de uma passagem

Um tradutor para expressões simples

Prof. Edson Alves

Faculdade UnB Gama

Tradutor dirigido pela sintaxe

- Quando não há associação de atributos aos não-terminais, um tradutor dirigido pela sintaxe pode ser construído a partir da extensão de um analisador gramatical preditivo
- Para isso, inicialmente construa o analisador gramatical preditivo
- Em seguida, copie as ações sintáticas do tradutor nas posições adequadas no analisador gramatical preditivo
- Se a gramática tiver uma ou mais produções recursivas à esquerda, é preciso modificar a gramatica para eliminar esta recursão antes de proceder com a construção do analisador gramatical preditivo

Transformação de produções recursivas à esquerda

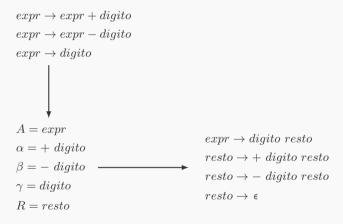
Transformação de produção recursiva à esquerda

Seja $A \to A\alpha \mid A\beta \mid \gamma$ uma produção recursiva à esquerda. Esta produção equivale às produções recursivas à direta

$$\begin{array}{l} A \rightarrow \gamma R \\ R \rightarrow \alpha R \mid \beta R \mid \epsilon \end{array}$$

onde α e β é uma cadeia de terminais e não-terminais que não começam com A e nem terminam com R.

Exemplo de transformação de produção recursiva à esquerda



Esquema de tradução da gramática para expressões para a notação posfixa

```
expr \rightarrow digito \ resto
 resto \rightarrow + digito \{imprimir('+')\}\ resto
 resto \rightarrow -digito \{imprimir('-')\}\ resto
 resto \rightarrow \epsilon
digito \rightarrow 0 \{imprimir('o')\}
digito \rightarrow 1 \{imprimir('1')\}
diaito \rightarrow 9 \quad \{imprimir('9')\}
```

Rotina de extração do próximo token em C++

```
1#include <iostream>
2
3using token = char;
4
5token proximo_token()
6{
7   auto t = std::cin.get();
8
9   return (token) t;
10}
```

Rotina de tratamento de erro e declaração de lookahead em C++

```
12token lookahead;
13
14void erro()
15{
16    std::cerr << "\nErro de sintaxe! lookahead = " << lookahead << '\n';
17    exit(-1);
18}</pre>
```

Rotina de reconhecimento de tokens em C++

```
20 void reconhecer(token t)
21 {
22    if (lookahead == t)
23         lookahead = proximo_token();
24    else
25         erro();
26}
```

Rotina associada ao não-terminal expr em C++

```
52 void expr()
53 {
54     digito();
55     resto();
56}
```

Rotina associada ao não-terminal digito em C++

```
28 void digito()
29 {
      if (isdigit(lookahead))
30
3.1
           std::cout.put(lookahead);
32
           reconhecer(lookahead);
33
      } else
34
35
           erro();
36
37
38 }
```

Rotina associada ao não-terminal resto em C++

Rotina principal do tradutor em C++

```
58int main()
59{
60     lookahead = proximo_token();
61
62     expr();
63
64     std::cout.put('\n');
65
66     return 0;
67}
```

Referências

- 1. AHO, Alfred V, SETHI, Ravi, ULLMAN, Jeffrey D. Compiladores: Princípios, Técnicas e Ferramentas, LTC Editora, 1995.
- 2. GNU.org. GNU Bison, acesso em 23/05/2022.
- 3. Wikipédia. Flex (lexical analyser generator), acesso em 23/05/2022.