IMPLEMENTAÇÃO DE UM COMPILADOR QUE GERA UMA REPR. GRÁFICA DE PSEUDOCÓDIGO

Bruno de Brito Coimbra[1], Silvio do Lago Pereira[2]
[1, 2] Departamento de Tecnologia da Informação – FATEC-SP
bbcoimbra@gmail.com, slago@pq.cnpq.br

1. Introdução

O uso de pseudocódigo (e.g., *portugol*) em cursos introdutórios de lógica de programação é uma prática bastante comum atualmente. A despeito dos benefícios que esta prática oferece, por se tratar de uma descrição essencialmente *textual*, o pseudocódigo não é capaz de evidenciar a natureza dinâmica dos algoritmos. De fato, o uso de representação gráfica (e.g., *fluxograma*) tem a vantagem de representar explicitamente o fluxo de execução dos passos do algoritmo, o que facilita sua compreensão para programadores iniciantes. Por exemplo, é muito comum que iniciantes usem "se", em vez de "enquanto", para representar uma repetição (já que a representação textual do algoritmo não é capaz de diferenciar estas duas estruturas de controle); por outro lado, este erro raramente ocorre numa representação gráfica da mesma lógica. Assim, acredita-se que a disponibilidade de uma ferramenta capaz de gerar a representação gráfica da lógica expressa por um pseudocódigo pode facilitar a sua compreensão e acelerar o processo de aprendizagem.

Neste trabalho, o objetivo é descrever a implementação de um compilador que gera duas representações, semanticamente equivalentes, de um pseudocódigo: uma em C [1], que pode ser compilada por um compilador C padrão, e outra em DOT [2] que, ao ser compilada, gera uma representação gráfica da lógica do pseudocódigo.

2. A Linguagem do Pseudocódigo

Neste trabalho, foi considerada uma linguagem para especificação de pseudocódigo bastante simples: todas as constantes são inteiras, todas as variáveis são globais e não é possível definir funções. Um pseudocódigo para exibição do fatorial de um número `n`, é apresentado na Tabela I.

3. Implementação do Compilador

Segundo *Aho et al.* [3], um *compilador* é um programa que transforma um código-fonte num código-objeto. Em geral, os códigos fonte e objeto representam programas equivalentes, porém em linguagens distintas.

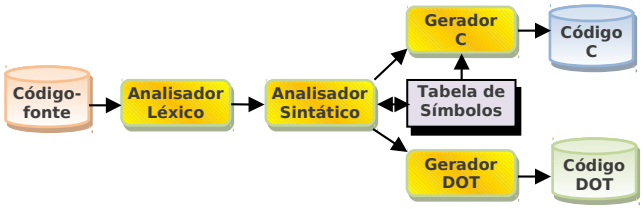

Figura 1 – Módulos do compilador implementado em C.

A Figura 1 exibe o esquema do compilador implementado. O Analisador Léxico foi criado com a ferramenta Flex [4]; o Analisador Sintático foi criado com Bison [5], um gerador de analisadores sintáticos ascendentes que recebe a especificação da gramática livre de contexto a ser reconhecida e produz código C para cada uma de suas regras; a Tabela de Símbolos foi representada por uma tabela de *hash* que utiliza uma lista encadeada para resolver colisões; e os módulos Gerador C e Gerador DOT foram implementados por ações semânticas anotadas nos nós da AST gerada pelo Analisador Sintático. A implementação completa está disponível em http://github.com/bbcoimbra/compiler.

4. Resultados Preliminares Alcançados

O compilador desenvolvido foi testado com vários pseudocódigos. Para todos eles, foram obtidos programas C e representações gráficas semanticamente equivalentes. A Tabela I mostra um exemplo de pseudocódigo e o programa C correspondente gerado, a representação gráfica produzida é apresentada na Figura 2.

Tabela I - Exemplo de pseudocódigo e equivalente em C.

Pseudocódigo	Código gerado em C

leia n;
produto = 1;
enquanto(n>1)
  produto = produto * n;
  n = n -1;
fim;
escreva produto;
	#include <unistd.h>
	#include <stdlib.h>
	#include <stdio.h>
	int main(int argc, char **argv){
	  int produto, n;
	  scanf("%d", &n);
	  produto = 1;
	  while (n > 1) {
	  produto = produto * n;
	  n = n - 1;
	  }
	  printf("%d", produto);
	  exit(EXIT_SUCCESS)
	}

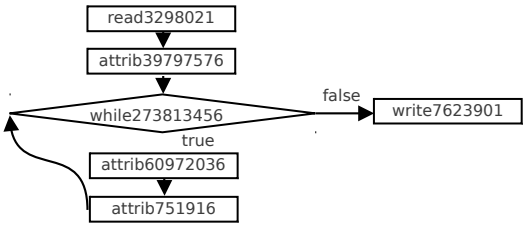

Figura 2 – Representação do pseudocódigo da Tabela I.

5. Conclusões

O compilador criado é capaz de traduzir um pseudocódigo para C e DOT. A representação gráfica, porém, ainda precisa ser melhorada para ficar mais parecida com um fluxograma convencional. Isto será feito futuramente.

6. Referências

[1] B. W. Kernighan & D. M. Ritchie. *C - A Linguagem de Programação*, 2ª ed. RJ: Elsevier, 1999.

[2] E. R. Gansner et al. *Drawing Graphs with DOT*. http://graphviz.org/pdf/dotguide.pdf. Acesso 6/11.

[3] A. V. Aho et al. *Compiladores: Princípios, Técnicas e Ferramentas*. 2ª ed. SP: Pearson, 2008.

[4] Flex Project. http://flex.sourceforge.net. Acesso 3/11.

[5] R. Corbett et al. *Bison Manual*, FSF, 2008.