UNISUL
UNIVERSIDADE DO SUL DE SANTA CATARINA
Aqui o seu futuro acontece

Linguagens Formais e Programação

Aula 2 – Introdução ao Compilador

Prof. Flávio Ceci, Dr.

flavio.ceci@unisul.br

1

Definições

- Tradutor:
  - É um programa que traduz um programa fonte escrito em uma linguagem qualquer (denominada linguagem fonte) para um programa objeto equivalente escrito em outra linguagem (denominada linguagem objeto)

Fonte: Furtado (2014)

2

Definições

- Compilador:
  - É um Tradutor em que a linguagem fonte é uma linguagem de alto nível e a linguagem objeto é uma linguagem de baixo nível (assembly ou máquina)

Fonte: Furtado (2014)

3

Definições

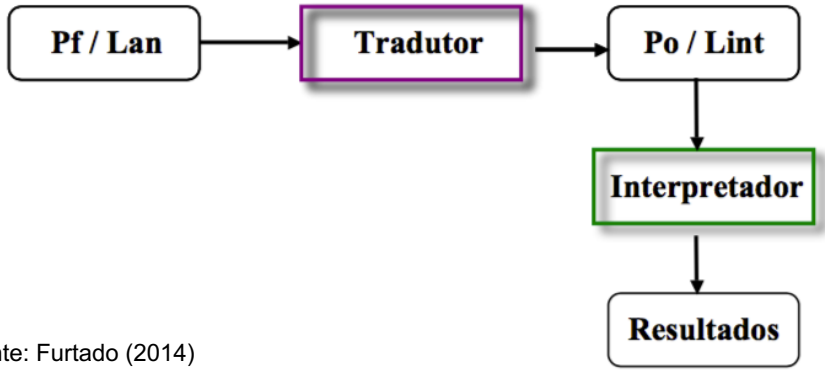
- Interpretador:
  - É um programa que interpreta diretamente as instruções do programa fonte, gerando o resultado.

Fonte: Furtado (2014)

4

Definições

- Tradutor / Interpretador:
  - Esquema hibrido para implementação de linguagens de programação.



Fonte: Furtado (2014)

5

Definições

- Montador:
  - É um Tradutor em que o programa fonte está escrito em linguagem assembly e o programa objeto resultante está em linguagem de máquina

Fonte: Furtado (2014)

6

Definições

- Pré-processador:
  - É um Tradutor em que tanto o programa fonte quanto o programa objeto estão escritos em linguagens de alto nível.
- *Cross – Compiler*
  - Compilador que gera código para uma máquina diferente da utilizada na compilação.

Fonte: Furtado (2014)

7

Estrutura de um Compilador Moderno

8

Estrutura de um Compilador Moderno

- Front-end:
  - **Análise léxica**: identifica símbolos que compõem o programa.
  - **Análise sintática**: identifica como estes símbolos se relacionam entre si.
  - **Análise semântica**: identifica o significado destas relações.
  - **Geração de código intermediário**: produz a estrutura na representação intermediária.

Fonte: Malaquias (2012)

9

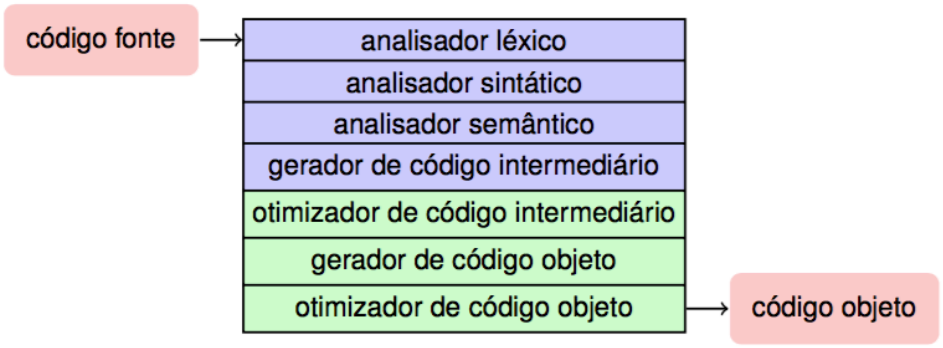
Estrutura de um Compilador Moderno

- Back-end:
  - **Otimização do código intermediário**: simplifica as estruturas intermediárias geradas.
  - **Geração de código objeto**: produz a estrutura na linguagem objeto.
  - **Otimização de código objeto**: melhora a estrutura produzida.

Fonte: Malaquias (2012)

10

Estrutura de um Compilador Moderno



Fonte: Malaquias (2012)

11

Analisador Léxico

- Interface entre o programa fonte e o compilador;
- Funções básicas:
  - Ler o programa fonte
  - Agrupar caracteres em itens léxicos (tokens)
    - Identificadores
    - Palavras Reservadas
    - Constantes (numéricas e literais)
    - Símbolos especiais (simples, duplos, ...)
  - Continua...

Fonte: Furtado (2014)

12

Analisador Léxico

- Interface entre o programa fonte e o compilador;
- Funções básicas:
  - Ignorar elementos sem valor sintático
    - Espaços em brancos, comentários e caracteres de controle
  - Detectar e diagnosticar erros léxicos
    - Símbolos inválidos, elementos mal formados

Fonte: Furtado (2014)

13

Analisador Léxico

- Exemplo:

Programa Fonte	Tokens Reconhecidos	
program exemplo; var A, B : integer; begin (* Inicio do programa *) read (A); B := A + 2.5; . . . end.	program exemplo ; var A , ... end .	1 - PR 2 - ID 3 - SE 4 - PR 2 - ID 5 - SE ... 37 - PR 38 - SE

Fonte: Furtado (2014)

14

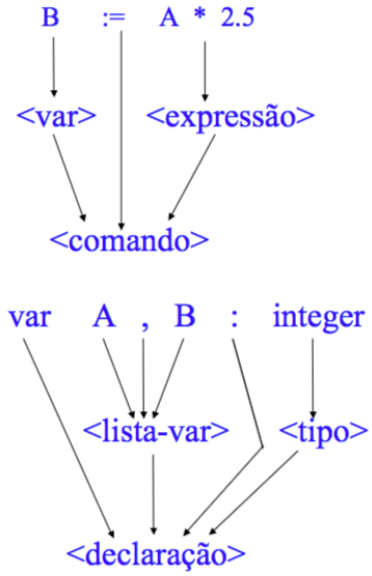
Analisador Sintático

- Funções básicas
  - Agrupar TOKENS em estruturas sintáticas (expressões, comandos, declarações, etc. ...)
  - Verificar se a sintaxe da linguagem na qual o programa foi escrito está sendo respeitada
  - Detectar/Diagnosticar erros sintáticos

Fonte: Furtado (2014)

15

Analisador Sintático



Fonte: Furtado (2014)

16

Analisador Semântico

- Funções básicas:
  - Verificar se as construções utilizadas no P.F. estão semanticamente corretas
  - Detectar e diagnosticar erros semânticos
  - Extrair informações do programa fonte que permitam a geração de código

Fonte: Furtado (2014)

17

Analisador Semântico

- Verificações Semânticas Usuais
  - Análise de escopo
    - Declaração de variáveis
  - Compatibilidade de tipos
  - Coerência entre declaração e uso de identificadores
  - Correlação entre parâmetros formais e atuais
  - Referências não resolvidas.

Fonte: Furtado (2014)

18

Gerador de Código Intermediário

- Consiste na geração de um conjunto de instruções (equivalentes ao programa fonte de entrada) para uma máquina hipotética (virtual):
- Exemplo: **E := (A + B) * (C + D)**

Quadrupla	Máquina de acumulador
(+ , A , B , T1) (+ , C , D , T2) (+ , T1 , T2 , E)	carregue A some B armazene T1 carregue C some D armazene T2 carregue T1 multiplique T2 armazene E

Fonte: Furtado (2014)

19

Otimizador de Código

- Melhorar o código, de forma que a execução seja mais eficiente quanto ao tempo e/ou espaço ocupado.
- Otimizações mais comuns
  - Agrupamento de sub-expressões comuns;
  - Eliminação de desvios para a próxima instrução;
  - Retirada de comandos invariantes ao LOOP;
  - Eliminação de código inalcançável;

Fonte: Furtado (2014)

20

Otimizador de Código

- Otimizações mais comuns
  - Redução em força;
  - Transformação/avaliação parcial; e
  - Alocação ótima de registradores.

Fonte: Furtado (2014)

21

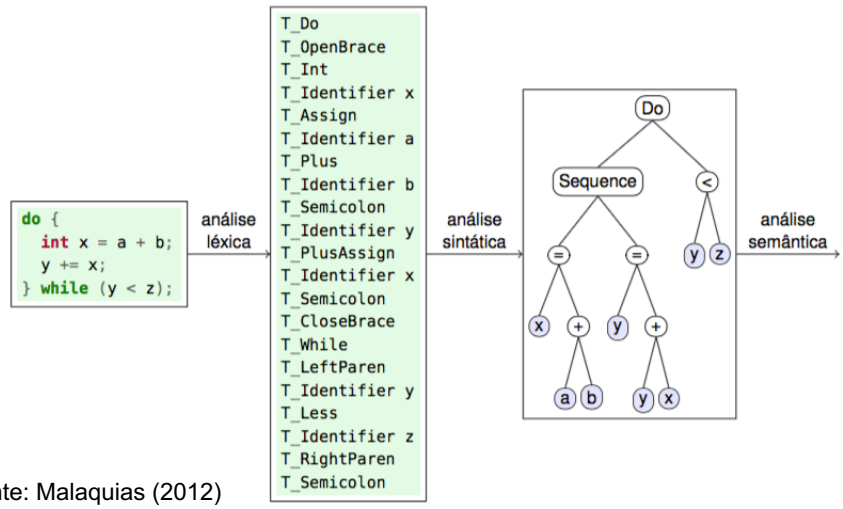
Gerador de Código

- Converter o programa fonte (diretamente ou a partir de sua representação na forma de código intermediário) para uma sequência de instruções (*assembler* ou máquina) de uma máquina real.

Fonte: Furtado (2014)

22

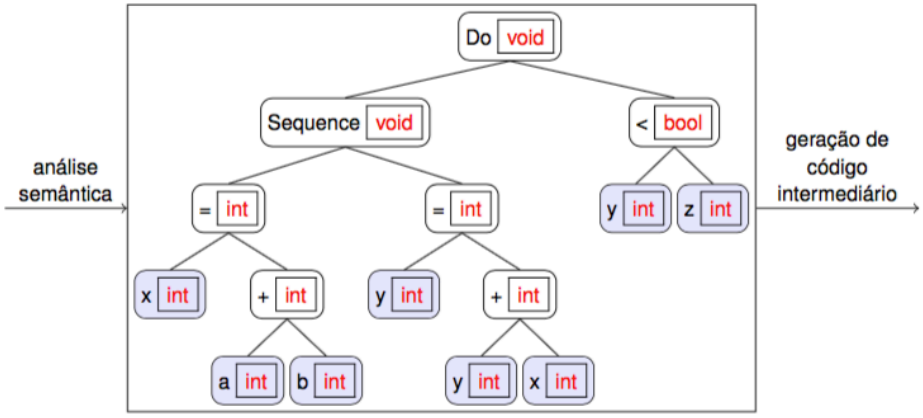
Exemplo de um Compilador Moderno



Fonte: Malaquias (2012)

23

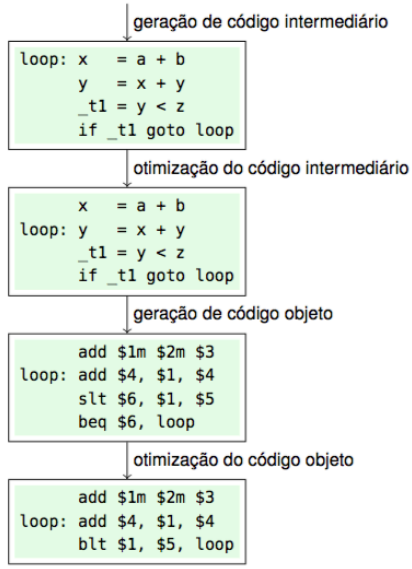
Exemplo de um Compilador Moderno



Fonte: Malaquias (2012)

24

Exemplo de um Compilador Moderno



Fonte: Malaquias (2012)

25

Referencial Teórico

FURTADO, Olinto José Varela. INE5622 – **INTRODUÇÃO A COMPILADORES**. INE-UFSC, 2014.

MALAQUIAS, José Romildo. **Construção de Compiladores**. Capitulo 1, UFOP, 2012.

26