Compiladores

Introdução

Bruno Lopes

Instituto de Computação
uff

Apresentação

- Em que período estão?
- O quanto sabem de programação?
- Quais linguagens?
- O quanto sabem de unix?
- O quanto sabem de Linguagens Formais?
- O quanto sabem de Compiladores?

Instituto de Computação

O que se espera do curso?

Instituto de Computação

Objetivo

Conhecer, compreender e implementar todas as fases do processo de compilação.

Instituto de Computação

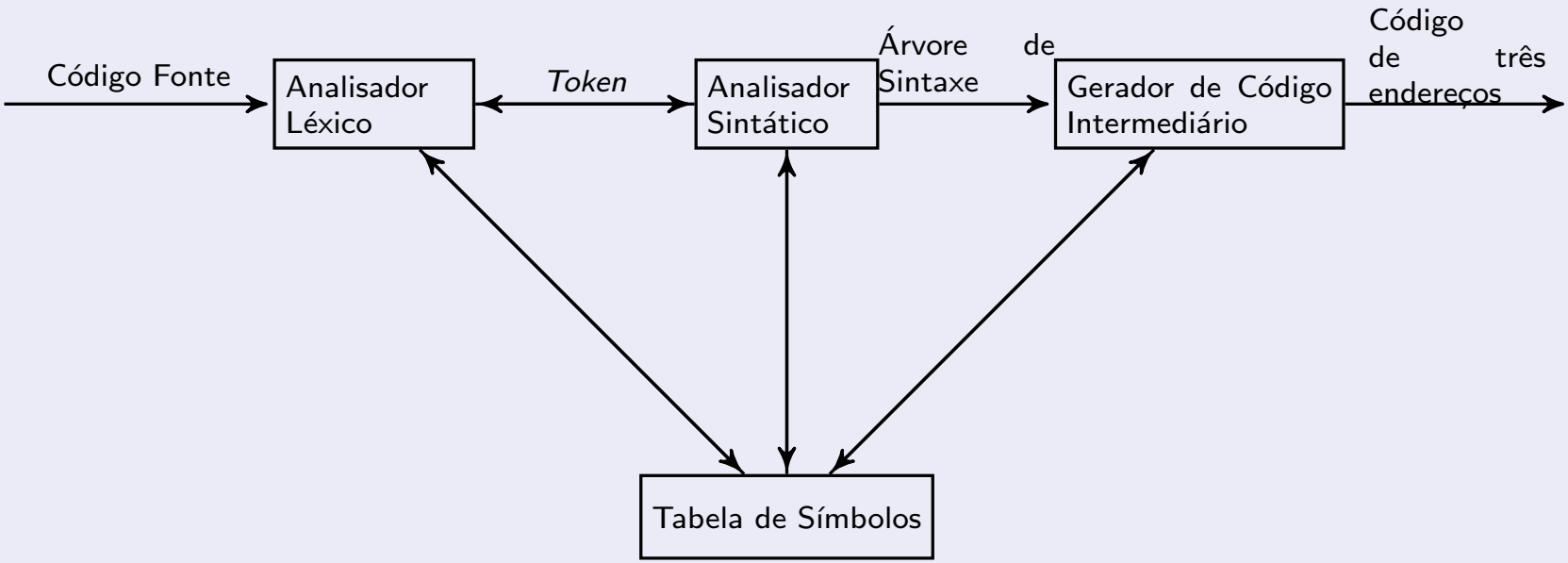
Etapas

- Análise Léxica
- Análise Sintática
- Análise Semântica
- Geração de código
- Otimização de código

Instituto de Computação

Estrutura de um compilador

Front-end



Instituto de Computação

Sobre o estudo de compiladores...

- Interface entre aplicações e arquitetura
- Técnicas de Linguagens Formais
- Técnicas de Programação
- Entender o comportamento de programas

Instituto de Computação

Sobre o estudo de compiladores...

- Interface entre aplicações e arquitetura
- Técnicas de Linguagens Formais
- Técnicas de Programação
- Entender o comportamento de programas

Instituto de Computação

Sobre o estudo de compiladores...

- Interface entre aplicações e arquitetura
- Técnicas de Linguagens Formais
- Técnicas de Programação
- Entender o comportamento de programas

Instituto de Computação

Sobre o estudo de compiladores...

- Interface entre aplicações e arquitetura
- Técnicas de Linguagens Formais
- Técnicas de Programação
- Entender o comportamento de programas

Instituto de Computação

Compilador

Programa que traduz um programa de uma linguagem para outra.

Interpretador

Programa que lê um outro programa e produz o resultado de sua execução.

Instituto de Computação

Compilador

Programa que traduz um programa de uma linguagem para outra.

Interpretador

Programa que lê um outro programa e produz o resultado de sua execução.

Instituto de Computação

Compilador

Programa que traduz um programa de uma linguagem para outra.

Interpretador

Programa que lê um outro programa e produz o resultado de sua execução.

Instituto de Computação

Compiladores: princípios

- O significado do programa deve ser preservado.
- O código gerado deve ser melhor que o de entrada. (Como?)

Instituto de Computação

Histórico

- Linguagem de máquina
- Linguagem de montagem
- Linguagem de programação

Instituto de Computação

Histórico

- Speedcoding
- Fortran
- Hierarquia de Chomsky
- Algoritmos para reconhecer Linguagens Livres de Contexto
- Técnicas de otimização de código

Instituto de Computação

Funcionalidades

- Reconhecer código
- Gerar código correto
- Gerenciar armazenamento de variáveis e código

Instituto de Computação

Fases

- Análise Léxica
- Análise Sintática
- Análise Semântica
- Otimização
- Geração de código

Front-end, Middle-end, back-end

Instituto de Computação

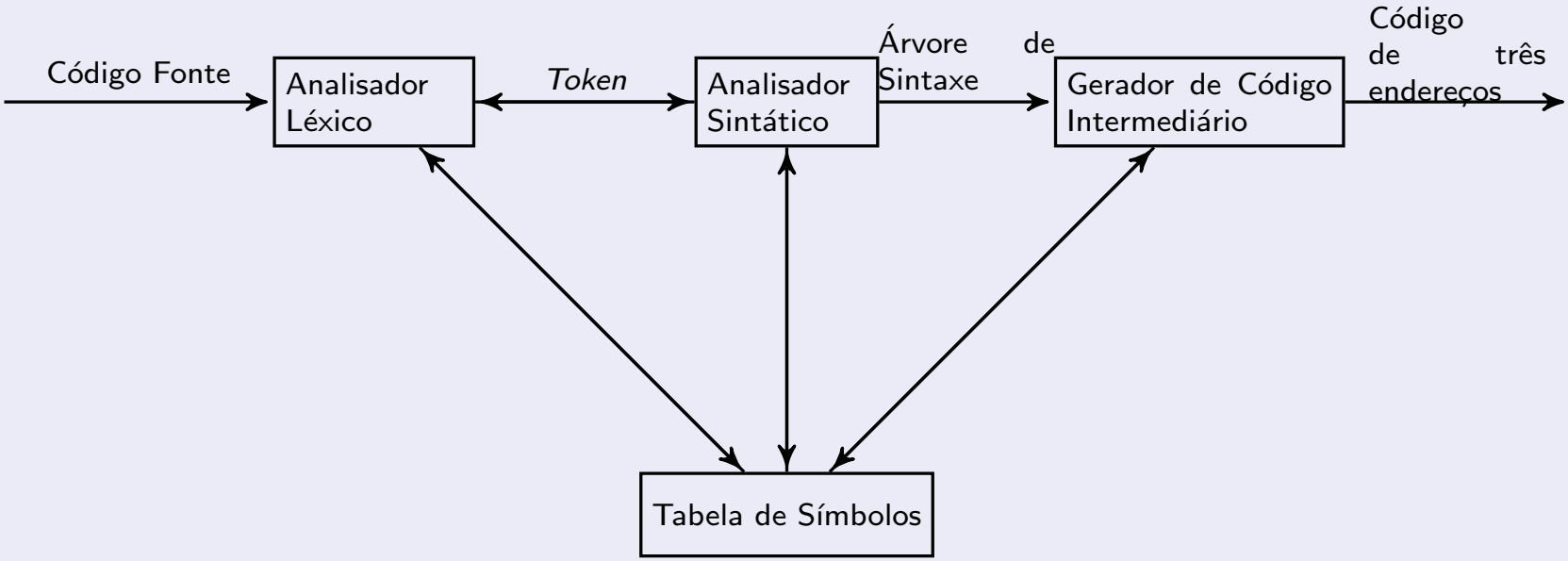
Fases

- Análise Léxica
- Análise Sintática
- Análise Semântica
- Otimização
- Geração de código

Front-end, Middle-end, back-end

Instituto de Computação

Front-end



Instituto de Computação

Front-end

- Reconhece programas legais
- Reporta erros
- Produz código intermediário
- Produz de gerenciamento de memória e código

Instituto de Computação

Scanner

Análise Léxica

Organização em *tokens*.

Este é um programa

```
int i = 10;
```

Instituto de Computação

Scanner

Análise Léxica

Organização em *tokens*.

Este é um programa

```
int i = 10;
```

Instituto de Computação

Scanner

Análise Léxica

Organização em *tokens*.

Este é um programa

```
int i = 10;
```

Instituto de Computação

Quantos *tokens?*

```
#include<stdio.h>

int main (int argc, char *argv[]) {
  printf("Hello world!");
  return 0;
}
```

Instituto de Computação

Análise Léxica

- Recohecimento de *tokens*
- Gerenciamento da tabela de símbolos (que operações? quando efetuá-las?)
- Uso de expressões regulares
- Ferramentas: Lex, JFlex etc.

Instituto de Computação

Parsing

- Como as palavras estão estruturadas?
- Estão bem estruturadas?
- Retorno: estrutura diagramizada

```
Este é um programa
```

```
int i = 10;
```

Instituto de Computação

Parsing

- Como as palavras estão estruturadas?
- Estão bem estruturadas?
- Retorno: estrutura diagramizada

Este é um programa

```
int i = 10;
```

Instituto de Computação

Parsing

- Como as palavras estão estruturadas?
- Estão bem estruturadas?
- Retorno: estrutura diagramizada

Este é um programa

```
int i = 10;
```

Instituto de Computação

Parser

- Reconhece a sintaxe
- Reporta erros
- Código intermediário

Instituto de Computação

Parser

- Determina elementos estruturais do código e seus relacionamentos
- Como especificar a sintaxe?
- Produz uma árvore sintática

Instituto de Computação

Resultado do *parsing?*

```
#include<stdio.h>

int main (int argc, char *argv[]) {
  printf("Hello world!");
  return 0;
}
```

Instituto de Computação

Parser

- Estrutura definida por regras recursivas (como?)
- Regras recursivas definidas por uma gramática
- Geradores: YACC, Bison, SableCC, JavaCC etc.

Instituto de Computação

Análise Semântica

Busca entender o significado.

- Atributos semânticos podem ser analisados antes da execução
- Verificação de tipos
- Eliminação de ambiguidades
- Construções Sensíveis ao Contexto

Instituto de Computação

Análise Semântica

Busca entender o significado.

- Atributos semânticos podem ser analisados antes da execução
- Verificação de tipos
- Eliminação de ambiguidades
- Construções Sensíveis ao Contexto

Instituto de Computação

Otimização de código

- Como utilizar menos recursos?
- Como melhorar o desempenho?
- Heurísticas

Instituto de Computação

```
for(i=0; i<n; i++)
  for(j=0; j<n; j++)
    A[i][j] = 0;

for(i=0; i<n; i++)
  for(j=0; j<n; j++)
    A[j][i] = 0;

p = &A[0][0];
t = n * n;
for(i=0; i<t; i++)
  *p++ = 0;
```

Instituto de Computação

Back-end

- Traduz código intermediário em código de máquina
- Escolhe instruções para implementar cada operação
- Gerencia registradores

Instituto de Computação

Registradores

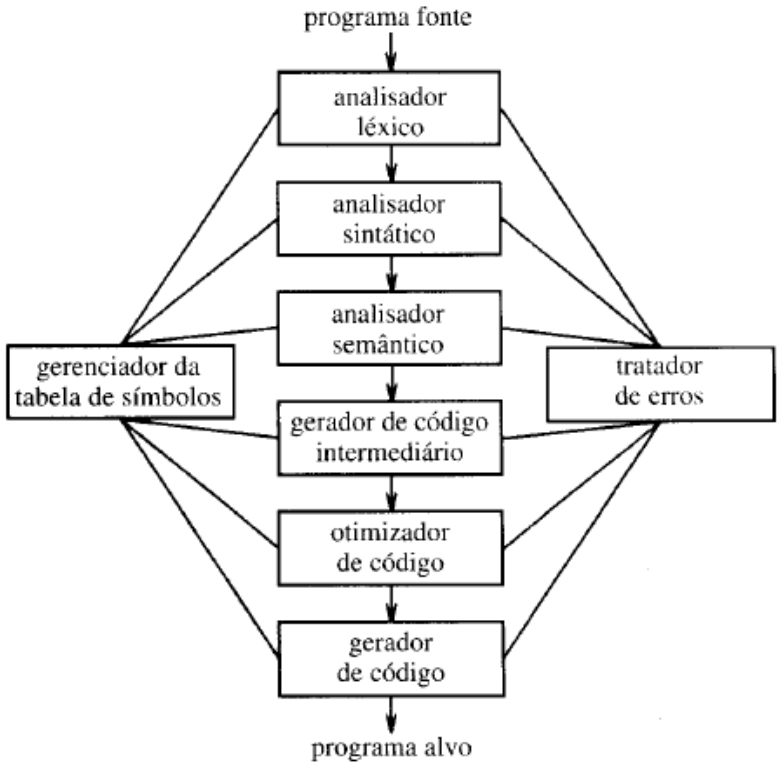
- Inserir no Load
- Inserir no Store
- Define conjunto de registradores que serão utilizados

Instituto de Computação

Escalonamento de instruções

- Usar recursos de forma produtiva
- Alocação ótima é NP-Completo
- Heurísticas

Instituto de Computação



Instituto de Computação

Linguagem Tiny

- Variáveis inteiras
- Declaração a partir da atribuição de valores
- Controle com if e repeat
- if termina com end e tem um else opcional
- Comentários entre chaves
- Expressões aritméticas e booleanas

Instituto de Computação

```
{Exemplo de programa em Tiny - Fatorial}
read x; {inteiro de entrada}
if x > 0 then {expressão booleana}
fact := 1;
repeat
fact := fact * x;
x := x + 1;
until x = 0;
write fact;
end
```

Instituto de Computação