Aspectos preliminares

Linguagens de Programação

Marco A L Barbosa

http://github.com/malbarbo/na-lp-copl

Conteúdo

Razões para estudar conceitos de linguagens de programação

Razões para estudar conceitos de linguagens de programação

- Aumentar a capacidade de expressar ideias

Razões para estudar conceitos de linguagens de programação

- Aumentar a capacidade de expressar ideias
- Melhorar as condições de escolha da linguagem apropriada para cada problema

Razões para estudar conceitos de linguagens de programação

- Aumentar a capacidade de expressar ideias
- Melhorar as condições de escolha da linguagem apropriada para cada problema
- Aumentar a capacidade de aprender novas linguagens

Razões para estudar conceitos de linguagens de programação

- Aumentar a capacidade de expressar ideias
- Melhorar as condições de escolha da linguagem apropriada para cada problema
- Aumentar a capacidade de aprender novas linguagens
- Melhorar o uso das linguagens já conhecidas

Razões para estudar conceitos de linguagens de programação

- Aumentar a capacidade de expressar ideias
- Melhorar as condições de escolha da linguagem apropriada para cada problema
- Aumentar a capacidade de aprender novas linguagens
- Melhorar o uso das linguagens já conhecidas
- Entender a importância da implementação

Razões para estudar conceitos de linguagens de programação

- Aumentar a capacidade de expressar ideias
- Melhorar as condições de escolha da linguagem apropriada para cada problema
- Aumentar a capacidade de aprender novas linguagens
- Melhorar o uso das linguagens já conhecidas
- Entender a importância da implementação
- Avanço da área de computação

Domínios de programação

Domínios de programação

- Aplicações científicas

Domínios de programação

- Aplicações científicas
  - Estruturas simples (arranjos e matrizes)
  - Muitas operações com pontos flutuantes
  - Fortran, Algol, C/C++
  - Fortress, Mathlab (Octave), numpy (Python)

Domínios de programação

- Aplicações científicas
  - Estruturas simples (arranjos e matrizes)
  - Muitas operações com pontos flutuantes
  - Fortran, Algol, C/C++
  - Fortress, Mathlab (Octave), numpy (Python)
- Aplicações comerciais

Domínios de programação

- Aplicações científicas
  - Estruturas simples (arranjos e matrizes)
  - Muitas operações com pontos flutuantes
  - Fortran, Algol, C/C++
  - Fortress, Mathlab (Octave), numpy (Python)
- Aplicações comerciais
  - Produção de relatórios
  - Formatação de números decimais e caracteres
  - Cobol

Domínios de programação

- Aplicações científicas
  - Estruturas simples (arranjos e matrizes)
  - Muitas operações com pontos flutuantes
  - Fortran, Algol, C/C++
  - Fortress, Mathlab (Octave), numpy (Python)
- Aplicações comerciais
  - Produção de relatórios
  - Formatação de números decimais e caracteres
  - Cobol
- Inteligência artificial

Domínios de programação

- Aplicações científicas
  - Estruturas simples (arranjos e matrizes)
  - Muitas operações com pontos flutuantes
  - Fortran, Algol, C/C++
  - Fortress, Mathlab (Octave), numpy (Python)
- Aplicações comerciais
  - Produção de relatórios
  - Formatação de números decimais e caracteres
  - Cobol
- Inteligência artificial
  - Manipulação de símbolos (lista ligada)
  - Criação e execução de código
  - Lisp, Prolog
  - C/C++

Domínios de programação

- Software de sistema

Domínios de programação

- Software de sistema
  - Eficiência devido ao uso contínuo
  - C/C++
  - D, Go, Rust

Domínios de programação

- Software de sistema
  - Eficiência devido ao uso contínuo
  - C/C++
  - D, Go, Rust
- Web

Domínios de programação

- Software de sistema
  - Eficiência devido ao uso contínuo
  - C/C++
  - D, Go, Rust
- Web
  - Código dentro do documento
  - Javascript, PHP, Java, Ruby, Python

Critérios para avaliação de linguagens

Critérios para avaliação de linguagens

- Facilidade de leitura (legibilidade)
- Facilidade de escrita
- Confiabilidade
- Custo

Critérios para avaliação de linguagens

- Facilidade de leitura
  - Simplicidade
    - Um conjunto bom de características e construções
    - Poucas formas de expressar cada operação
    - Mínima sobrecarga de operador
    - Muito simples não é bom (assembly)

Critérios para avaliação de linguagens

- Facilidade de leitura
  - Simplicidade
    - Um conjunto bom de características e construções
    - Poucas formas de expressar cada operação
    - Mínima sobrecarga de operador
    - Muito simples não é bom (assembly)
  - Ortogonalidade
    - Poucas características podem ser combinadas de várias maneiras
    - Uma característica deve ser independente do contexto que é usada (exceções a regra são ruins)
    - Muito ortogonalidade não é bom (Algol68)
    - Linguagens funcionais oferecem uma boa combinação de simplicidade e ortogonalidade

Critérios para avaliação de linguagens

- Facilidade de leitura
  - Tipos de dados
    - Tipos pré-definidos adequados

Critérios para avaliação de linguagens

- Facilidade de leitura
  - Tipos de dados
    - Tipos pré-definidos adequados
  - Sintaxe
    - Flexibilidade para nomear identificadores
    - Forma de criar instruções compostas
    - A forma deve ter relação com o significado

Critérios para avaliação de linguagens

- Facilidade de escrita
  - Simplicidade e ortogonalidade
    - Poucas construções e um conjunto consistente de formas de combinação

Critérios para avaliação de linguagens

- Facilidade de escrita
  - Simplicidade e ortogonalidade
    - Poucas construções e um conjunto consistente de formas de combinação
  - Suporte para abstração
    - Definir e usar estruturas e operações de maneira que os detalhes possam ser ignorados
    - Suporte a subprogramas
    - Suporte a tipos abstratos de dados

Critérios para avaliação de linguagens

- Facilidade de escrita
  - Simplicidade e ortogonalidade
    - Poucas construções e um conjunto consistente de formas de combinação
  - Suporte para abstração
    - Definir e usar estruturas e operações de maneira que os detalhes possam ser ignorados
    - Suporte a subprogramas
    - Suporte a tipos abstratos de dados
  - Expressividade
    - Maneira conveniente de expressar a computação

Critérios para avaliação de linguagens

- Confiabilidade
  - Verificação de tipos

Critérios para avaliação de linguagens

- Confiabilidade
  - Verificação de tipos
  - Manipulação de exceções

Critérios para avaliação de linguagens

- Confiabilidade
  - Verificação de tipos
  - Manipulação de exceções
  - Apelidos

Critérios para avaliação de linguagens

- Confiabilidade
  - Verificação de tipos
  - Manipulação de exceções
  - Apelidos
  - Facilidade de leitura e escrita

Critérios para avaliação de linguagens

- Custo
  - Treinar programadores

Critérios para avaliação de linguagens

- Custo
  - Treinar programadores
  - Escrever programas

Critérios para avaliação de linguagens

- Custo
  - Treinar programadores
  - Escrever programas
  - Compilar programas

Critérios para avaliação de linguagens

- Custo
  - Treinar programadores
  - Escrever programas
  - Compilar programas
  - Executar programas

Critérios para avaliação de linguagens

- Custo
  - Treinar programadores
  - Escrever programas
  - Compilar programas
  - Executar programas
  - Confiabilidade

Critérios para avaliação de linguagens

- Custo
  - Treinar programadores
  - Escrever programas
  - Compilar programas
  - Executar programas
  - Confiabilidade
  - Manutenção

Critérios para avaliação de linguagens

- Custo
  - Treinar programadores
  - Escrever programas
  - Compilar programas
  - Executar programas
  - Confiabilidade
  - Manutenção
  - Maior peso no custo: escrita, manutenção e confiabilidade

Critérios para avaliação de linguagens

- Outros critérios
  - Portabilidade
  - Padronização

Critérios para avaliação de linguagens

- Outros critérios
  - Portabilidade
  - Padronização
- Diferentes visões
  - Programador
  - Projetista da linguagem
  - Implementador da linguagem

Influências no projeto de linguagens

Influências no projeto de linguagens

- Arquitetura do Computador
  - Arquitetura de von Neumann
  - Arquiteturas multicore
  - Outras?

Influências no projeto de linguagens

- Arquitetura do Computador
  - Arquitetura de von Neumann
  - Arquiteturas multicore
  - Outras?
- Metodologias de Programação
  - Orientada a processos
  - Orientada a dados
  - Orientação a objetos

Categorias de linguagens

Categorias de linguagens

- Imperativas:

Categorias de linguagens

- Imperativas: Algol68, Fortran, Cobol, Ruby, Python, Go, Java, Pascal, C/C++, ...

Categorias de linguagens

- Imperativas: Algol68, Fortran, Cobol, Ruby, Python, Go, Java, Pascal, C/C++, ...
- Funcionais ou Aplicativas: Lisp, Haskell, ML, Scheme, Erlang, Ocaml, F#, Miranda, ...

Categorias de linguagens

- Imperativas: Algol68, Fortran, Cobol, Ruby, Python, Go, Java, Pascal, C/C++, ...
- Funcionais ou Aplicativas: Lisp, Haskell, ML, Scheme, Erlang, Ocaml, F#, Miranda, ...
- Lógicas ou Declarativas: Prolog, Planner, QA-4, Popler, Conniver, QLISP, Mercury, Oz, Frill, ...

Métodos de implementação

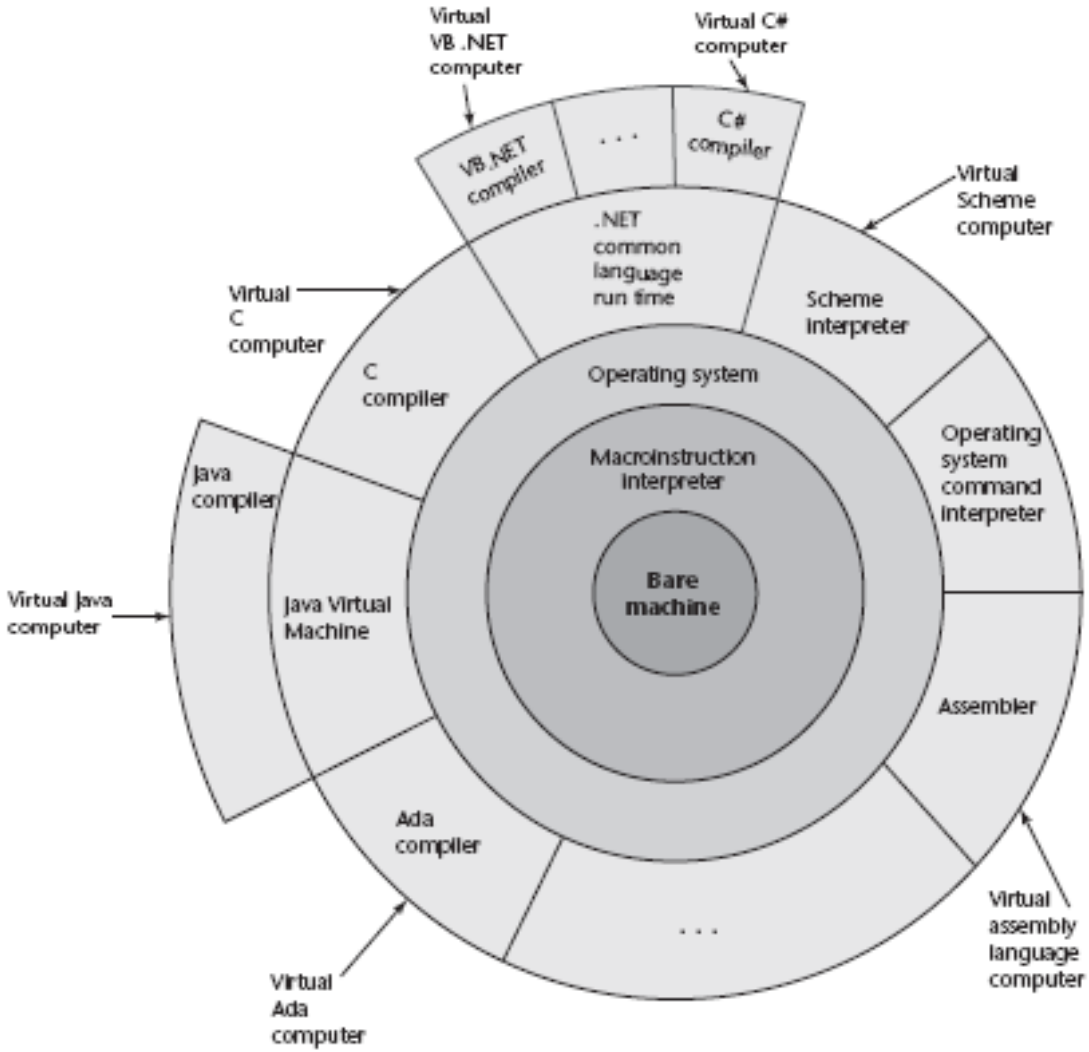
Métodos de implementação

- Compilação
- Interpretação
- Híbrido

Métodos de implementação

Figure 1.2

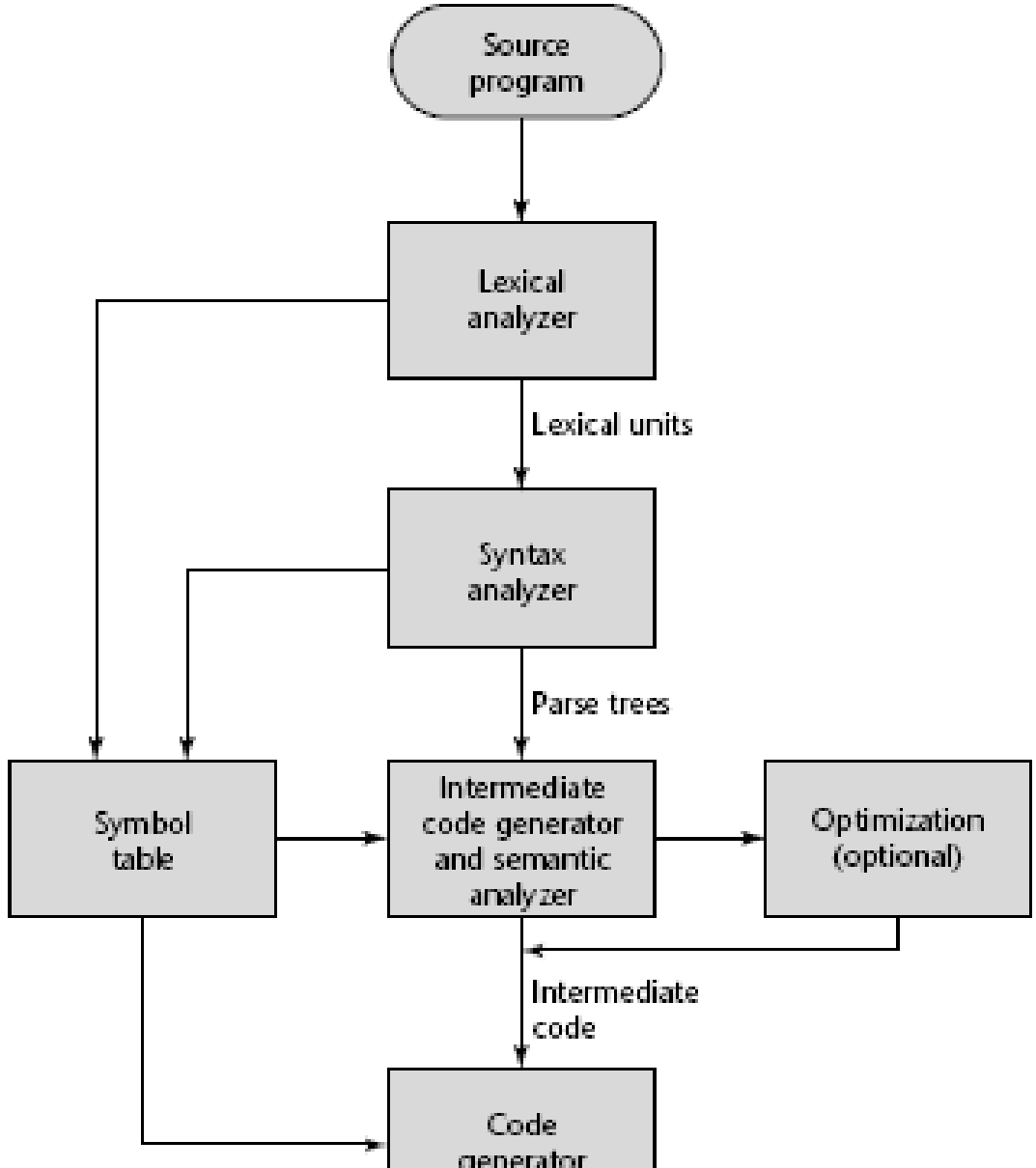
Layered interface of virtual computers, provided by a typical computer system



Métodos de implementação - Compilação

Figure 1.3

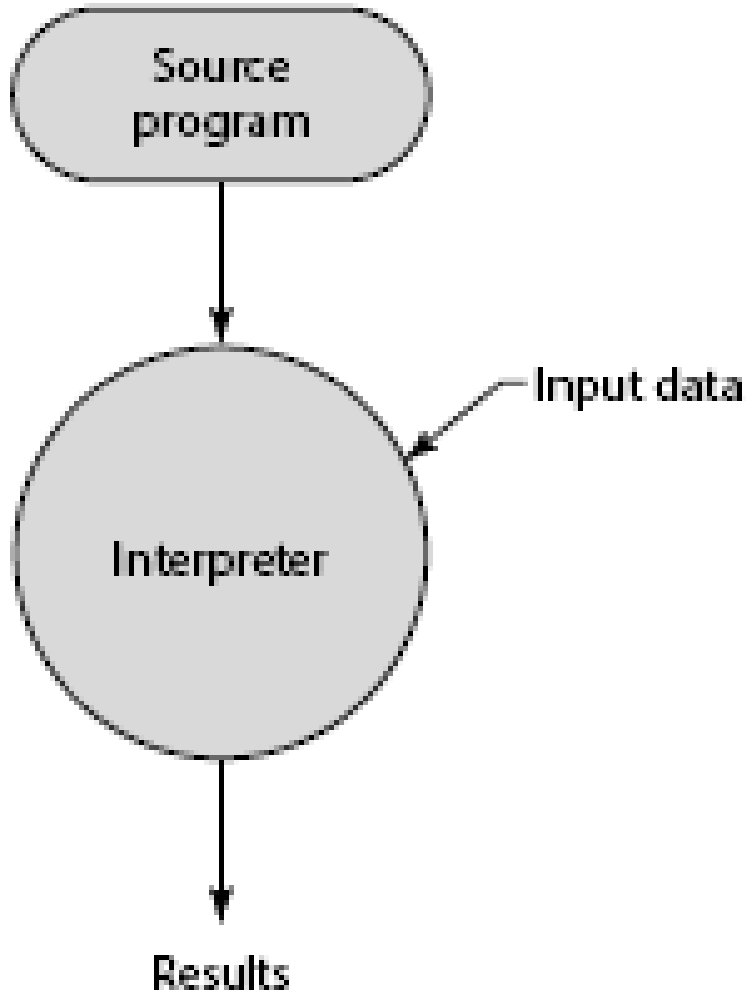
The compilation process



Métodos de implementação - Interpretação

Figure 1.4

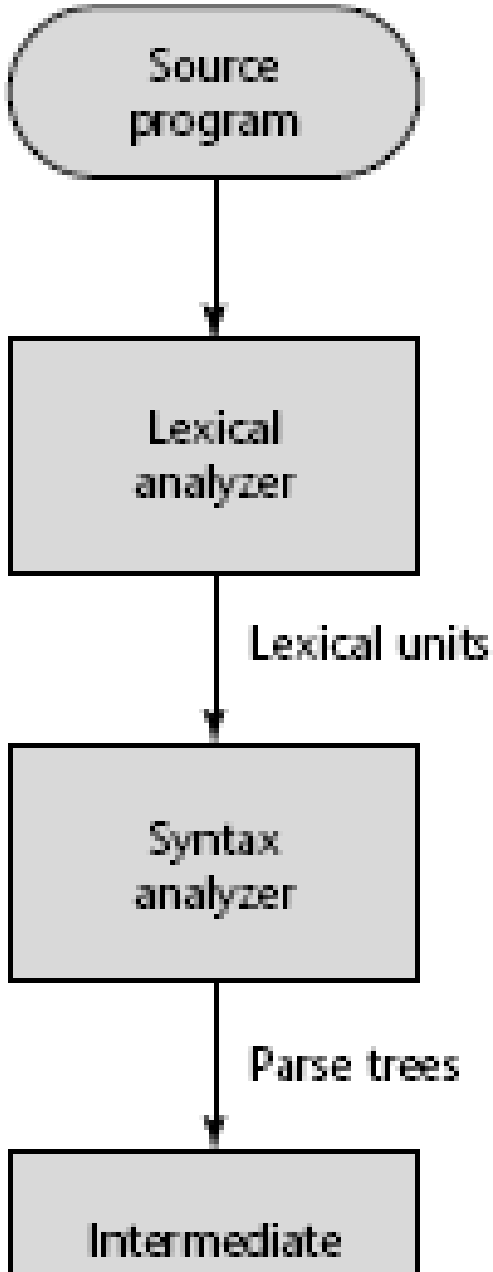
Pure interpretation



Métodos de implementação - Híbrido

Figure 1.5

Hybrid implementation system



Referências

Referências

- Robert Sebesta, Concepts of programming languages, 9ª edição. Capítulo 1.