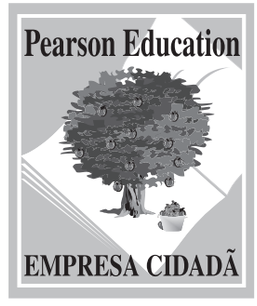
Compiladores

Princípios, técnicas e ferramentas

2ª Edição



Alfred V. Aho Monica S. Lam Ravi Sethi Jeffrey D. Ullman

Compiladores

Princípios, técnicas e ferramentas

2ª Edição

Tradução:
Daniel Vieira

Revisão Técnica:
Mariza Andrade da Silva Bigonha
Professora associado do Departamento de Ciência da Computação
Universidade Federal de Minas Gerais

PEARSON Addison Wesley

São Paulo

Brasil Argentina Colômbia Costa Rica Chile Espanha Guatemala México Peru Porto Rico Venezuela

Gerente editorial: Roger Trimer
Editora sênior: Sabrina Cairo
Editor de desenvolvimento: Marco Pace
Editora de texto: Eugênia Pessotti
Preparação: Andrea Filatro
Revisão: Lucrécia Freitas e Sandra Scapin
Capa: Celso Blanes (sobre projeto original de S. D. Ullman – Strange Tonic Productions)
Composição Editorial: ERJ Composição Editorial e Artes Gráficas Ltda.

Dados Internacionais de Catalogação na Publicação (CIP)
(Câmara Brasileira do Livro, SP, Brasil)

Compiladores : princípios, técnicas e ferramentas / Alfred V. Aho...[et al.]; tradução Daniel Vieira; revisão técnica Mariza Bigonha. – 2. ed. – São Paulo: Pearson Addison-Wesley, 2008.

Outros autores: Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman
Título original: Compilers: principles, techniques, and tools

Bibliografia
ISBN 978-85-88639-24-9

1. Compiladores (Programas de computador)
I. Aho, Alfred V.. II. Lam, Monica S.. III. Sethi, Ravi. IV. Ullman, Jeffrey D..

07-7541 CDD-005.453

Índices para catálogo sistemático:
1. Compiladores : Linguagens de programação : Ciências da computação 005.453

2007

Av. Ermano Marchetti, 1435
CEP: 05038-001 – São Paulo – SP
Fone: (11) 2178-8686 – Fax: (11) 2178-8688
e-mail: vendas@pearsoned.com

SUMÁRIO

PREFÁCIO

Desde a publicação da primeira edição deste livro, em 1986, o mundo do projeto de compiladores mudou significativamente. As linguagens de programação evoluíram a ponto de apresentarem novos problemas de compilação e as arquiteturas de computador passaram a oferecer diversos recursos dos quais o projetista de compilador simplesmente não pode abrir mão. Mais interessante, talvez, seja que a venerável tecnologia de otimização de código encontrou uso fora dos compiladores. Agora, ela é usada em ferramentas que encontram erros no *software* e, mais importante, encontram brechas de segurança no código existente. E grande parte da tecnologia de *front-end* — gramáticas, expressões regulares, analisadores sintáticos e tradutores dirigidos por sintaxe — ainda é muito usada.

Portanto, nossa filosofia, apresentada na edição anterior deste livro, não mudou. Reconhecemos que poucos leitores construirão, ou mesmo realização a manutenção de um compilador para qualquer uma das principais linguagens de programação. Ainda assim, os modelos, a teoria e os algoritmos associados a um compilador podem ser aplicados a uma grande gama de problemas no projeto e desenvolvimento de software. Portanto, enfatizamos problemas que são encontrados mais comumente no projeto de um processador de linguagem, independentemente da linguagem fonte ou da máquina alvo.

Uso do livro

São necessários pelo menos dois trimestres, ou até mesmo dois semestres, para estudar todo ou quase todo o conteúdo deste livro. É comum o uso da primeira metade em um curso de graduação e a segunda metade — enfatizando a otimização do código — em um segundo curso, no nível de pós-graduação. Veja a seguir um resumo de cada um dos capítulos.

Capítulo 1: contém material introdutório e apresenta uma visão geral sobre a arquitetura de computador e os princípios de linguagem de programação.

Capítulo 2: desenvolve um compilador em miniatura e introduz muito dos conceitos importantes, que posteriormente são desenvolvidos em outros capítulos. O compilador em si aparece no Apêndice.

Capítulo 3: abrange a análise léxica, expressões regulares, máquinas de estado finito e ferramentas geradoras de analisadores léxicos. Esse material é fundamental para o processamento de textos de todas as espécies.

Capítulo 4: cobre os principais métodos de análise sintática, descendente (descida recursiva, LL) e ascendente (LR e suas variantes).

Capítulo 5: introduz as principais idéias sobre definições dirigidas por sintaxe e traduções dirigidas por sintaxe.

Capítulo 6: baseia-se na teoria do Capítulo 5 e mostra como usá-la para gerar código intermediário para uma linguagem de programação típica.

Capítulo 7: cobre os ambientes em tempo de execução, especialmente o gerenciamento da pilha de execução e a coleta de lixo.

Capítulo 8: trata da geração de código objeto. Ele abrange a construção de blocos básicos, geração de código a partir de expressões e blocos básicos e técnicas de alocação de registradores.

Capítulo 9: introduz a tecnologia de otimização de código, incluindo grafo de fluxo, estruturas de fluxo de dados e algoritmos iterativos para solucionar essas estruturas.

Capítulo 10: abrange a otimização em nível de instrução. A ênfase é dada à extração do paralelismo a partir de pequenas seqüências de instruções e seu escalonamento em processadores individuais que podem efetuar mais de uma tarefa de uma só vez.

Capítulo 11: trata de detecção e exploração de paralelismo em escala maior. Neste capítulo, a ênfase é sobre códigos de aplicação numéricas que possuem muitos laços estreitos que iteram sobre arranjos multidimensionais.

Capítulo 12: apresenta a análise entre procedimentos. Ele aborda a análise de apontador, sinônimos e a análise de fluxo de dados que leva em consideração a seqüência de chamadas de procedimento que alcançam um determinado ponto no código.

Os cursos de compiladores das universidades de Columbia, Harvard e Stanford são baseados neste livro. Em Columbia, uma disciplina de fim de curso de graduação ou de primeiro ano de pós-graduação sobre linguagens de programação e tradutores tem sido oferecido regularmente, usando o material dos oito primeiros capítulos. Um destaque desse curso é um trabalho de

um semestre no qual os alunos se reúnem em pequenas equipes para criar e implementar uma pequena linguagem com seu próprio projeto. As linguagens criadas pelos alunos têm incluído diversos domínios de aplicação, incluindo cálculo quântico, síntese musical, computação gráfica, jogos, operações com matriz e muitas outras áreas. Os alunos utilizam geradores de componente de compilador, como ANTLR, Lex e Yacc, e as técnicas de tradução dirigidas por sintaxe, discutidas nos capítulos 2 e 5, para criar seus compiladores. Um outro curso de graduação utilizou o material dos capítulos de 9 a 12, enfatizando a geração e otimização de código para máquinas contemporâneas, incluindo processadores em rede e arquiteturas de multiprocessadores.

Em Stanford, um curso introdutório de um trimestre abrange parte do material dos Capítulos de 1 a 8, embora exista uma introdução à otimização de código global do Capítulo 9. O segundo curso de compiladores abrange os capítulos de 9 a 12 e o material mais avançado sobre coleta de lixo, do Capítulo 7. Os alunos usam um sistema desenvolvido localmente, baseado em Java, chamado Joeq, para implementar os algoritmos de análise de fluxo de dados.

Pré-requisitos

O leitor deverá possuir alguma 'sofisticação em ciência da computação', incluindo pelo menos um segundo curso sobre programação, e cursos sobre estruturas de dados e matemática discreta. O conhecimento de diversas linguagens de programação diferentes é útil.

Exercícios

O livro contém muitos exercícios, em quase todas as seções. Indicamos os exercícios mais difíceis, ou as partes mais complexas de alguns, com um ponto de exclamação. Os exercícios ainda mais difíceis possuem um ponto de exclamação duplo.

Companion Website

No site de apoio do livro, que pode ser acessado no endereço www.aw.com/aho_br, estão disponíveis o fonte do Apêndice A e uma biblioteca de imagens.

Agradecimentos

A arte da capa é de S. D. Ullman, da Strange Tonic Productions.

John Bentley nos forneceu muitos comentários sobre vários capítulos de um rascunho preliminar deste livro. Recebemos sugestões úteis e erratas da edição anterior das seguintes pessoas: Domenico Bianculli, Peter Bosch, Marcio Buss, Marc Eaddy, Stephen Edwards, Vibhav Garg, Kim Hazelwood, Gaurav Kc, Wei Li, Mike Smith, Art Stamness, Krysta Svore, Olivier Tardieu, e Jia Zeng. Agradecemos a ajuda de todas elas. Os erros que restaram são nossos, naturalmente.

Além disso, Monica gostaria de agradecer aos seus colegas da equipe de compiladores do SUIF por uma lição de 18 anos sobre compilação: Gerald Aigner, Dzintars Avots, Saman Amarasinghe, Jennifer Anderson, Michael Carbin, Gerald Cheong, Amer Diwan, Robert French, Anwar Ghuloum, Mary Hall, John Hennessy, David Heinem, Shih-Wei Liao, Amy Lim, Benjamin Livshits, Michael Martin, Dror Maydan, Todd Mowry, Brian Murphy, Jeffrey Oplinger, Karen Pieper, Martin Rinard, Olatunji Ruwase, Constantine Sapuntzakis, Patrick Sathyanathan, Michael Smith, Steven Tjiang, Chau-Wen Tseng, Christopher Unkel, John Whaley, Robert Wilson, Christopher Wilson e Michael Wolf.

A. V. A., Chatham NJ
M. S. L., Menlo Park CA
R. S., Far Hills NJ
J. D. U., Stanford CA
Junho de 2006.

Agradecimento dos editores brasileiros

Agradecemos aos professores Sandra Maria Aluíso, da Universidade de São Paulo, campus São Carlos, Hélio Aparecido Navarro, da Universidade Estadual de São Paulo e Pedro S. Nicolletti, da Universidade Federal de Campina Grande, por sua colaboração no processo de avaliação desta obra.