

Apresentação da Disciplina Fundamentos Teóricos da Computação

Professor

Mark Alan Junho Song – song@pucminas.br

Sumário

1 Apresentação da Disciplina

- Introdução
- Conteúdo Programático
- Bibliografia
- Avaliações

2 Computabilidade e Complexidade

- Introdução
- Histórico

Ementa

Fundamentos Teóricos da Computação

- Linguagens.
- Expressões regulares.
- Gramáticas.
- Autômatos.
- Aplicação de autômatos nas análises léxica e sintática (descendente e ascendente).
- Máquinas de Turing.
- Hierarquia de Chomsky.
- Decidibilidade.

Conteúdo Programático

- 1 Introdução / Conceitos matemáticos preliminares
- 2 Autômatos finitos / Linguagens regulares
 - $AFD \times AFN \times AFN-\lambda$
 - Gramática linear e regular
 - Expressão regular
 - Propriedades de fechamento / Lema do bombeamento
- 3 Autômatos de pilha / LLC
 - $APD \times APN$
 - GLC / Derivação
 - Ambiguidade
 - Propriedades de fechamento / Lema do bombeamento
 - Parsing descendente e ascendente

Conteúdo Programático

- 1 Introdução / Conceitos matemáticos preliminares
- 2 Autômatos finitos / Linguagens regulares
 - $AFD \times AFN \times AFN-\lambda$
 - Gramática linear e regular
 - Expressão regular
 - Propriedades de fechamento / Lema do bombeamento
- 3 Autômatos de pilha / LLC
 - $APD \times APN$
 - GLC / Derivação
 - Ambiguidade
 - Propriedades de fechamento / Lema do bombeamento
 - Parsing descendente e ascendente

Conteúdo Programático

- 1 Introdução / Conceitos matemáticos preliminares
- 2 Autômatos finitos / Linguagens regulares
 - $AFD \times AFN \times AFN-\lambda$
 - Gramática linear e regular
 - Expressão regular
 - Propriedades de fechamento / Lema do bombeamento
- 3 Autômatos de pilha / LLC
 - $APD \times APN$
 - GLC / Derivação
 - Ambiguidade
 - Propriedades de fechamento / Lema do bombeamento
 - Parsing descendente e ascendente

Conteúdo Programático

- 4 Máquinas de Turing / Ling. Recursivas e LRE
 - MTD \times MTN (e outras variações de MT)
 - Gramática irrestrita
 - Autômatos linearmente limitados
 - LSC / GSC
 - Hierarquia de Chomsky
- 5 Decidibilidade / Tratabilidade
 - Problemas de decisão / MT para PD
 - Tese de Church-Turing
 - MT Universal
 - Problemas Decidíveis \times Problemas Indecidíveis
 - Redução de um problema a outro

Conteúdo Programático

- ④ Máquinas de Turing / Ling. Recursivas e LRE
 - MTD \times MTN (e outras variações de MT)
 - Gramática irrestrita
 - Autômatos linearmente limitados
 - LSC / GSC
 - Hierarquia de Chomsky
- ⑤ Decidibilidade / Tratabilidade
 - Problemas de decisão / MT para PD
 - Tese de Church-Turing
 - MT Universal
 - Problemas Decidíveis \times Problemas Indecidíveis
 - Redução de um problema a outro

Bibliografia

- VIEIRA, Newton José. Introdução aos fundamentos da computação: linguagens e máquinas. São Paulo: Pioneira Thomson Learning, 2006.
- SIPSER, Michael. Introdução à teoria da computação. São Paulo: Thomson Learning, 2007.
- HOPCROFT, John E.; ULLMAN, Jeffrey D.; MOTWANI, Rajeev. Introdução à teoria de autômatos, linguagens e computação. Rio de Janeiro: Campus, 2003.
- SUDKAMP, Thomas A. Languages and machines: an introduction to the theory of computer science. 3rd ed. Boston: Pearson, Addison Wesley, 2006.
- GAREY, Michael R.; JOHNSON, David S. Computers and intractability: a guide to the theory of NP-Completeness. New York: W. H. Freeman, 1979.
- ARORA, Sanjeev; BARAK, Boaz. Computational complexity: a modern approach. Cambridge, UK: Cambridge University Press, 2009.
- GOLDBREICH, Oded. Computational complexity: a conceptual perspective. Cambridge, UK: Cambridge University Press, 2008.
- PAPADIMITRIOU, Christos H.; STEIGLITZ, Kenneth. Combinatorial optimization: algorithms and complexity. Mineola: Dover, 1998.
- MENEZES, Paulo Fernando Blauth. Linguagens formais e autômatos. 5. ed. Porto Alegre: Sagra-Luzzatto, 2005.
- DIVERIO, Tiarajú Asmuz. Teoria da computação: máquinas universais e computabilidade. 2. ed. Porto Alegre: Sagra Luzzatto, 2000.

Introdução

Computabilidade e Complexidade

- O que é um computador?
- Quais são suas capacidades e limitações?
- Que classe de problemas são computáveis?
- Como determinar a complexidade da solução?

Como obter um modelo geral para o conceito de computabilidade?

Introdução

“Algoritmo”

Um processo finito de ações executar uma determinada tarefa

“Algoritmos” também possuem um papel importante na matemática

Exemplo: algoritmo de Euclides para MDC

Histórico

A noção de algoritmo só foi definida, precisamente, no século XX

Antes disso, os matemáticos baseavam-se na noção intuitiva de “algoritmo”

Qual o problema de se usar uma noção intuitiva?

Histórico

Uma noção intuitiva não é suficiente para se obter provas sobre propriedades de algoritmos!

Exemplo

Em 1900, David Hilbert propôs uma série de problemas matemáticos:

- 10º Problema de Hilbert era o de encontrar um “algoritmo” para determinar, dado um polinômio com coeficientes inteiros, se o mesmo possuía raízes inteiras

Histórico

Aparentemente, Hilbert supunha que tal “algoritmo” existia

Esse “algoritmo” não existe - esse problema é indecidível

Provar que um algoritmo não existe requer uma definição formal
do conceito de algoritmo!

Histórico

A definição formal do conceito de algoritmo veio em 1936, a partir dos trabalhos de Church, Turing e Gödel/Kleene:

- Church utilizou uma notação denominada λ -calculus para definir o conceito de algoritmo;
- Turing definiu o conceito através de um modelo de máquina - máquina de Turing;
- Gödel/Kleene definiram a classe das funções computáveis como sendo a classe das funções recursivas parciais.

Histórico

As 3 definições são equivalentes:

- A conexão, entre a noção informal de algoritmo e essas definições formais, é conhecida como Tese de Church-Turing.

Modelos possibilitam demonstrar a existência de problemas que não podem ser resolvidos utilizando computadores:

- Determinar se uma asserção matemática é verdadeira;
- Determinar a parada de um programa;
- Testar a existência de solução inteira para polinômios;
- ...

Histórico

Aplicação de Modelos

- Modelos matemáticos para descrever um computador genérico
- Caracterizar classes de problemas computáveis
- Analisar complexidade de algoritmos

Modelos matemáticos para classes de computadores

- Autômatos de Estados Finitos - processamento de texto, reconhecimento de voz e de padrões, projeto de hardware
- Gramáticas Livres de Contexto - construção de compiladores, projeto de linguagens de programação