
INF05516 - Semântica formal N
Ciência da Computação - UFRGS
2006-2

Marcus Ritt
mrpritt@inf.ufrgs.br

06/08/2006

Administrativa	2
Bem vindo!	3
Informações gerais	4
Elementos e avaliação	5
Elementos de avaliação...	6
Livros básicos	7
Livros adicionais	8
Introdução	9
Objetivos da disciplina	10
O panorama	11
Projeto de linguagens na prática	12
Paradigmas de programação	13
Paradigmas de programação (2)	14
Distribuição das linguagens.	15
Por que tantas linguagens?	16
Linguagens	17
O que caracteriza uma boa linguagem?	18
Critérios desejáveis	19
Critérios desejáveis	20
Como projetar linguagens?	21
Como cozinhar uma linguagem? Caraterísticas	22
Lembrança: Sintaxe e semântica	23
Estudando caraterísticas	24
Semântica - por quê?	25
Exemplo 1 – Pascal	26
Exemplo 2 – C	27
Exemplo 3 – C	28
Semântica - quem precisa?	29
Semântica - quem quer?	30
Visto geral	31
Semânticas.	32

Semânticas.	33
Exemplo: Semântica translacional	34
Bibliografia.	35

Bem vindo!

- Meu nome: Marcus Ritt
<http://www.inf.ufrgs.br/mrpritt>
- Página da disciplina
<http://www.inf.ufrgs.br/mrpritt/doku.php?id=inf05516>

v1771

Semântica formal N, aula 1 – 3 / 35

Informações gerais**Pre-requisitos**

Teoria da computação ← Matemática discreta
← Lógica para computação

Súmula Introdução ao cálculo lambda. Domínios de Scott; pontos fixos de funções contínuas. Semântica denotacional. Semântica algébrica. Semântica de ações. Semântica axiomática. Semântica operacional.

Horários de consulta Quarta 13–15

v1771

Semântica formal N, aula 1 – 4 / 35

Elementos e avaliação

A disciplina tem os seguintes elementos

- Aulas: Frequência obrigatória de 75%
- Exercícios opcionais

Sucesso sem exercícios é pouco provável:
Ler sem caneta é sonhar!

- Duas provas obrigatórias com notas $p_1, p_2 \in [0, 10]$
 - ◆ A provas são individuais
- Um trabalho obrigatório com nota $t \in [0, 10]$
 - ◆ O trabalho consiste em: (a) um produto (b) uma apresentação
 - ◆ O trabalho é permitido em grupos até três

v1771

Semântica formal N, aula 1 – 5 / 35

Observação

Produtos possíveis: Um texto (sumário, crítico, etc.), uma implementação com documentação

v1771

Semântica formal N, aula 1 – note 1 of slide 5

Elementos de avaliação...

■ Uma nota final $n = (4p_1 + 4p_2 + 2t)/10$

■ Um conceito final de

$$\begin{cases} A & 9 \leq m \leq 10 \wedge f \geq 75\% \\ B & 7.5 \leq m < 9 \wedge f \geq 75\% \\ C & 6 \leq m < 7.5 \wedge f \geq 75\% \\ D & m < 6 \wedge f \geq 75\% \\ FF & f < 75\% \end{cases}$$

■ Conceito D? Chance de uma prova de recuperação que substitui a menor nota das provas (p_1 ou p_2), se tu entregou os exercícios até o dia da realização das provas.

v1771

Semântica formal N, aula 1 – 6 / 35

Livros básicos

[1] Hanne Riis Nielson and Flemming Nielson. *Semantics with applications – A formal introduction*. July 1999. INF: 681.3.01 N669s.

[2] Glynn Winskel. *The formal semantics of programming languages*. MIT Press, February 1993. Biblioteca: 681.32.06 W777f.

v1771

Semântica formal N, aula 1 – 7 / 35

Livros adicionais

- [1] Kim B. Bruce. *Foundations of Object-Oriented Programming Languages: Types and Semantics*. MIT Press, 2002. INF: 681.32.06 B887f.
- [2] Daniel P. Friedman, Mitchell Wand, and Christopher T. Haynes. *Essentials of Programming Languages*. MIT Press, second edition, 2001. INF: 681.32.06 F911e (1º edição).
- [3] Hanne Riis Nielson and Flemming Nielson. *Semantics with applications: an appetizer*. Wiley, 2006.
- [4] Benjamin C. Pierce. *Types and programming languages*. MIT Press, 2002. INF: 681.32.06 P615t.
- [5] Kenneth Slonneger and Barry Kurtz. *Formal syntax and semantics of programming languages: a laboratory based approach*. Addison-Wesley, 1995. INF: 681.32.06 S634f.

v1771

Semântica formal N, aula 1 – 8 / 35

Introdução

9 / 35

Objetivos da disciplina

- Aprender sobre linguagens e análise de programas
 - ◆ Varias semânticas: operacional, axiomática, denotacional,...
 - ◆ Sistemas de tipos
- Aprender sobre algumas linguagens e características de linguagens
- Aprender um pouco sobre tópicos de pesquisa

v1771

Semântica formal N, aula 1 – 10 / 35

O panorama

- Linguagens de programação é uma das áreas mais antigas de Ciência de Computação
- Permanece uma área ativa e vibrante
- Algumas tendências
 - ◆ A web renova interesse em projeto de linguagens: veja Java, por exemplo
 - ◆ Tipos são reconhecidas como tendo um papel importante em segurança
 - ◆ Análise de programas se torna um componente significativo em Engenharia de Software

v1771

Semântica formal N, aula 1 – 11 / 35

Projeto de linguagens na prática

- Linguagens são adotadas para preencher uma necessidade: abstração
 - ◆ Permitir uma aplicação antes difícil ou impossível
 - ◆ Executar em uma nova plataforma
- Treinamento de programador tem custo muito alto
 - ◆ Linguagens com muitos usuários raramente são substituídas
 - ◆ Linguagens populares se tornam estagnadas
 - ◆ Novas aplicações e novas plataformas apresentam oportunidades para inovação

v1771

Semântica formal N, aula 1 – 12 / 35

Plano

Quais linguagens os estudantes conhecem? Escreve na tábua.

v1771

Semântica formal N, aula 1 – note 1 of slide 12

Paradigmas de programação

Imperativa Estado e transformação dele.

Funcional Avaliação sem side-effects, funções são dados.

Lógico Declaração de regras lógicas

Orientada a objetos Encapsulação, heredição, polimorfismo.

Concorrentes ou distribuídas Distribuição, sincronização e comunicação

v1771

Semântica formal N, aula 1 – 13 / 35

Plano

Ordena as linguagens conhecidas em paradigmas. Escreve na tabela.

v1771

Semântica formal N, aula 1 – note 1 of slide 13

Paradigmas de programação (2)

Imperativas Fortran, Algol, Cobol, C, Pascal

Funcionais Lisp, Scheme, ML, Haskell, XSLT

Lógicas Prolog, λ Prolog

Orientadas a objetos Smalltalk, Eiffel, Self, C++, Java

Concorrentes ou distribuídas Fortran90, HPF, CSP

v1771

Semântica formal N, aula 1 – 14 / 35

Plano

Aumenta a tábua.

v1771

Semântica formal N, aula 1 – note 1 of slide 14

Distribuição das linguagens

- Linguagens usadas na prática (imagem)
- Mais que x linguagens estimados
- Por exemplo “99 Bottles of Beer” tem uma programa em 900 linguagens

v1771

Semântica formal N, aula 1 – 15 / 35

Por que tantas linguagens?

- Muitas linguagens foram criadas para aplicações específicas
- Aplicações com necessidades distintas ou conflitantes
 - ◆ IA: computação simbólica (Lisp, Prolog)
 - ◆ Computação científica: alto desempenho (Fortran)
 - ◆ Negócios: geração de relatórios (Cobol)
 - ◆ Programação de sistemas: acesso ao baixo nível (C)
 - ◆ Customização: Scripts (Perl, Javascript)
 - ◆ Sistemas distribuídos: computação móvel (Java, C#)
 - ◆ Outras (LaTeX, awk, ...)

v1771

Semântica formal N, aula 1 – 16 / 35

Linguagens

17 / 35

O que caracteriza uma boa linguagem?

- Não há métricas aceitas universalmente para projeto de linguagens
- Uma linguagem boa é uma linguagem que as pessoas usam?

Seria Cobol a melhor linguagem?

v1771

Semântica formal N, aula 1 – 18 / 35

Critérios desejáveis

Queremos uma linguagem

- compreensível, com sintaxe e semântica simples,
- legível e fácil para escrever,
- poderoso e abstrato, com muitas funções,
- segura e robusta,
- portátil (independente da máquina)
- escalável, que suporte programação de sistemas grandes,
- eficiente na execução e compilação.

v1771

Semântica formal N, aula 1 – 19 / 35

Critérios desejáveis

- Estes objetivos são quase sempre conflitantes
 - ◆ Segurança e independência de máquina em geral custam eficiência
 - ◆ Sistemas de tipos restringem estilo de programação para obter garantias
 - ◆ Muitas funções restringem a compreensão
 - ◆ Portabilidade pode restringir a funcionalidade
- Projeto de boas linguagens é uma tarefa difícil
- Não se chega a uma boa linguagem por acidente

v1771

Semântica formal N, aula 1 – 20 / 35

Como projetar linguagens?

- Seguinda Hoare [2], o projetista deve
 - ◆ ser familiar com várias caraterísticas
 - ◆ ter bom discernimento na escolha e na combinação de caraterísticas
 - ◆ conhecer o domínio de aplicação da nova linguagem
 - ◆ saber o quão complicada a linguagem pode ficar
 - ◆ ter recursos para implementação, documentação e ambiente de desenvolvimento
- Uma coisa que ele não deve fazer é incluir ideias suas não testadas.

“Sua tarefa é consolidação, não inovação”.

v1771

Semântica formal N, aula 1 – 21 / 35

Como cozinhar uma linguagem? Caraterísticas

- Tipos e estruturas de dados
- Operadores, predicados e expressões
- Contrô (goto?)
- Abstração: funções anônimas, tipos de dados abstratos (objetos), modules, polimorfismo e genericidade, reflexão, ...
- Tratamento dos erros
- Concorrência, paralelismo?
- Entrada/saída
- Commentários
- Biblioteca padrão

v1771

Semântica formal N, aula 1 – 22 / 35

Lembrança: Sintaxe e semântica

- A *sintaxe* descreve como construir frases corretas
 - ◆ Sintaxe concreta
 - ◆ Sintaxe abstrata
- A *semântica* descreve o significado destas frases
 - ◆ Semântica estática
 - ◆ Semântica dinâmica

v1771

Semântica formal N, aula 1 – 23 / 35

Exemplo

Exemplo: $(2+3)*5$ é sintaxe concreta: precisamos os parenteses. $\text{vezes}(\text{mais}(2,3),5)$ ou um árvore de parse é sintaxe abstrata: prioridades, associatividade e outras características que são importante para escrever, entender e “parser” a linguagem são fora.

v1771

Semântica formal N, aula 1 – note 1 of slide 23

Estudando características

- Uma maneira de estudar um conceito ou construção é considerá-lo em uma linguagem pequena
 - ◆ experimentá-lo com o uso da linguagem
 - ◆ definir a linguagem rigorosamente: sintaxe (gramática), verificações estáticas (sistema de tipos), comportamento de execução (semântica)
 - ◆ estudar as propriedades da linguagem
 - ◆ relacionar a linguagem com outras
- Além disso, a construção pode ser estudada no contexto de uma linguagem grande.

v1771

Semântica formal N, aula 1 – 24 / 35

Semântica - por quê?

Dado um linguagem de programação, tem questões naturais, como

- Dada uma implementação da linguagem: é correto?
- O que significa uma programa?
- Dado uma programa: é correto?
- Temos proposições sobre uma programa (por exemplo: “a programa para?”): são verdadeiras?

If you don't have a specification, it can't be wrong.
It can only be surprising.

Uma semântica é um modelo matemático de uma linguagem.

- Ela serve como especificação formal
- Porém, ela permite analisar programas e provar características delas
- Já a atividade de definir uma semântica ajuda em relevar problemas

v1771

Semântica formal N, aula 1 – 25 / 35

Exemplo 1 – Pascal

```
<with-statement>
  ::= with <record-variable-list> do <statement>
<record-variable-list>
  ::= <record-variable> { , <record-variable> }
<variable-identifier>
  ::= <field-identifier>
```

The occurrence of a <record-variable> in the <record-variable-list> is a defining occurrence of its <field-identifier>s as <variable-identifier>s for the <with-statement> in which the <record-variable-list> occurs.

v1771

Semântica formal N, aula 1 – 26 / 35

Exemplo 2 – C

O padrão de C++, secção 5.2.6 diz

The value obtained by applying a postfix ++ is the value that the operand had before applying the operator. [...] After the result is noted, the value of the object is modified by adding 1 to it [...].

Tentamos de entender o seguinte fragmento:

```
i = 0;
i = i + i++;

i = 0;
i = i++ + i;

i = 0;
i = i++ + i++;
```

v1771

Semântica formal N, aula 1 – 27 / 35

Observação

A declaração foi deixada fora.

v1771

Semântica formal N, aula 1 – note 1 of slide 27

Exemplo 3 – C

```
float q = 3.0/7.0;

if (q == 3.0/7.0)
    printf("Tudo bem.\n");
else
    printf("Tudo ruim.\n");
```

v1771

Semântica formal N, aula 1 – 28 / 35

Semântica - quem precisa?

Alguem que tem estas perguntas! Os que querem

- saber *exatamente* e *não-ambigua* o que uma programa significa
- explicar uma linguagem para outros (inventor de linguagem)
- programar ferramentas da análise (análise lexical, ...)
- programar ferramentas de transformação (compilador, interpretador, ...)
- provar que uma software está correto (controle de sistemas críticas)

v1771

Semântica formal N, aula 1 – 29 / 35

Exemplo de transformação correto com prova.

v1771

Semântica formal N, aula 1 – note 1 of slide 29

Semântica - quem quer?

- Semântica é precisada, mas também é “pesada” ?!
 - ◆ nem sempre é necessária ou nem sempre o benefício em relação aos custos vale a pena
- A maioria dos programadores se dá por satisfeito com conhecimento menos preciso a aprendem por tentativa e erro

v1771

Semântica formal N, aula 1 – 30 / 35

Visto geral

31 / 35

Semânticas

Semântica...

- operacional estrutural, operacional natural
- denotational
- axiomática
- algebráica
- translacional
- de ações

Por que tantas semânticas?

v1771

Semântica formal N, aula 1 – 32 / 35

Semânticas

Temos três linhas predominantes, com aplicações diferentes:

Semântica operacional O significado é dado em termos de execução em uma máquina abstrata. Uma semântica operacional ajuda a implementação.

Semântica denotational O significado é dado em termos de conceitos matemáticos abstratos. Uma semântica denotational é mais abstrata e ajuda um entendimento profundo.

Semântica axiomática O significado é dado com regras de provar propriedades. Uma semântica axiomática ajuda o desenvolvimento e a verificação.

v1771

Semântica formal N, aula 1 – 33 / 35

Exemplo: Semântica translacional

- Uma tradução para uma outra linguagem define a semântica
- Geralmente a linguagem destino é mais simples, por exemplo, uma linguagem de máquina
- Um método de tradução é uma gramática atributiva, que produz a tradução em uma atribuição sintética
- Vantagem: Abordagem simples
- Desvantagens:
 - ◆ A semântica depende da semântica da linguagem destino
 - ◆ A definição indireta complica o entendimento da semântica

v1771

Semântica formal N, aula 1 – 34 / 35

Bibliografia

- [1] John Warner Backus. Can programming be liberated from the von Neumann style? *Communications of the ACM*, 21(8), 1978.
- [2] C. A. R. Hoare. Hints on programming language design. Technical Report STAN-CS-73-403, Stanford artificial intelligence laboratory, 1973.
- [3] Peter J. Landin. The next 700 programming languages. *Communications of the ACM*, 9(3), 1966.