

– INF01147 –  
Compiladores

## Análise Sensível ao Contexto (Análise Semântica 1/2)

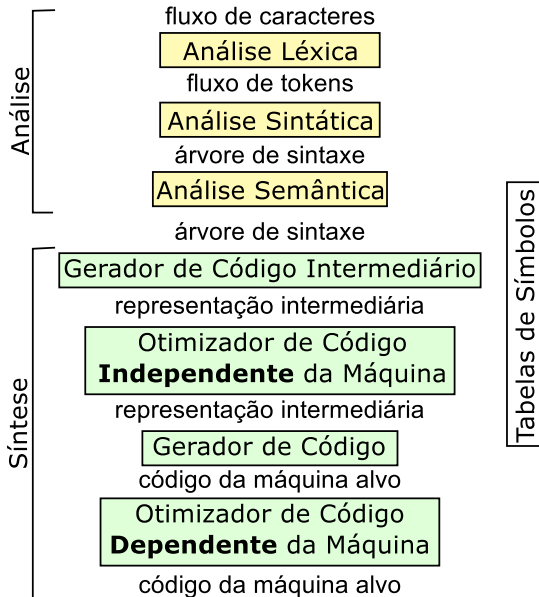
Prof. Lucas M. Schnorr  
– Universidade Federal do Rio Grande do Sul –



# Plano da Aula de Hoje

- ▶ Revisão LR(1)
- ▶ Contextualização da Análise Semântica
- ▶ Análise Sensível ao Contexto
- ▶ Ordem de Avaliação dos Atributos
- ▶ Lançamento da Etapa 4

# Estrutura de um **Compilador** em fases



# Análise Semântica – Motivação

- ▶ A análise sintática é o suficiente?
- ▶ Faltam uma série de informações
  - ▶ Quais são os valores representados
  - ▶ Onde eles residem dentro do programa
  - ▶ Como eles fluem de nome em nome
- ▶ Um compilador precisa entender a **estrutura do programa**
  
- ▶ Através de uma **Análise Sensível ao Contexto**
  - ▶ “Elaboração Semântica”
  - ▶ “Análise Semântica”
  - ▶ “Tradução Dirigida pela Sintaxe”

# Análise Semântica – Visão Geral

- ▶ Considerando um nome **x**, no código fonte
- ▶ Antes da geração de código, o compilador precisa saber
  - ▶ Qual o **tipo** do valor estocado em **x**?  
inteiro, caractere, booleano, ponteiro, conjunto, vetor, estrutura, cadeia, ...
  - ▶ Qual o **tamanho** de **x**?  
inteiro: 4 bytes, caractere: 1 byte, ponteiro: 4 bytes, cadeia: ?
  - ▶ Se **x** é uma função, quais são seus **argumentos/retorno**?  
tipo dos argumentos, tipo do retorno, onde o valor de retorno será guardado, ...
  - ▶ Por **quanto tempo** o valor de **x** tem que ser mantido?  
qual o escopo, qual o tempo de vida do nome
  - ▶ Quem é responsável por **alocar espaço** para **x**?  
vinculação de estocagem é estática ou dinâmica (explícita ou implícita)
- ▶ Respostas para estas questões
  - ▶ Devem ser derivadas a partir do código fonte
  - ▶ Obtidas das regras do projeto da linguagem de entrada
- ▶ Durante o processo de **Análise Sensível ao Contexto**

# Análise Semântica – Alguns usos

- ▶ Associa um **significado** ao código de entrada
- ▶ Algumas associações possíveis
  - ▶ Associar uma representação abstrata
    - ▶ Criar a AST
    - ▶ **Etapa 3 do Projeto de Compiladores**
  - ▶ Associar tipos
    - ▶ Permitir a verificação de tipos (de variáveis, de instruções)  
Em tempo de compilação (vinculação de tipos estática)  
Em tempo de execução (dinâmica; fornecendo suporte)
    - ▶ **Etapa 4 do Projeto de Compiladores**
  - ▶ Associar código executável à estrutura sintática
    - ▶ Etapa de Geração de Código  
(Será o assunto chave desta disciplina)
    - ▶ **Etapa 5 do Projeto de Compiladores**
- ▶ Tradução Dirigida pela Sintaxe
  - ▶ Código intermediário é gerado durante a análise sintática

# Análise Semântica – Funcionamento

- ▶ Ações semânticas são associadas às regras de produção
  - ▶ Executadas no momento da derivação ou redução
  - ▶ Ações possíveis
    - ▶ Gerar código
    - ▶ Armazenar informações na tabela de símbolos
    - ▶ Emitir mensagens de erro
- ▶ Ação Principal: **Associar variáveis** aos símbolos de G
  - ▶ Armazenam valores durante a tradução
  - ▶ São perenes durante a análise
- ▶ Definindo **atributos** para um símbolo

$T \rightarrow \text{int} \{ T.\text{tipo} = \text{inteiro} \}$

$D \rightarrow T \{ L.\text{in} = T.\text{tipo} \} L$

# Análise Semântica – Exemplo

- ▶ Gramática para declarar uma variável (mais ações semânticas)

$D \rightarrow T \text{ Id } \{ \text{adTabSimb} (\text{Id.nome}, T.\text{tipo}); \}$

$T \rightarrow \text{float } \{ T.\text{tipo} = \text{TipoFlutuante}; \}$

$T \rightarrow \text{int } \{ T.\text{tipo} = \text{TipoInteiro}; \}$

- ▶ Com a entrada **float pi** e a entrada **int iterador**
- ▶ **Anotação da árvore de derivação**
  - ▶ Propagar os atributos que definem a semântica
    - ▶ Das folhas (tokens) para cima
  - ▶ Exemplos
    - ▶ Cada não-terminal tem um ou mais atributos associados
    - ▶ Cada terminal (token) tem um atributo definido pelo léxico
  - ▶ Podem ser comunicados durante a criação da árvore  
(ou depois, se a análise semântica for independente)

# Exercício

- Considerando a gramática com ações semânticas

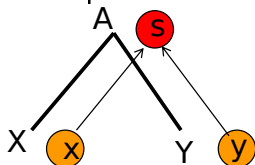
$L$	$\rightarrow$	$E\ n$	$\{ L.val = E.val \}$
$E$	$\rightarrow$	$E_1 + T$	$\{ E.val = E_1.val + T.val \}$
$E$	$\rightarrow$	$T$	$\{ E.val = T.val \}$
$T$	$\rightarrow$	$T_1 * F$	$\{ T.val = T_1.val * F.val \}$
$T$	$\rightarrow$	$F$	$\{ T.val = F.val \}$
$F$	$\rightarrow$	$( E )$	$\{ F.val = E.val \}$
$F$	$\rightarrow$	<b>digit</b>	$\{ F.val = digit.lexval \}$

- Crie a árvore de derivação anotada para
  - $(3 + 4) * (5 + 6)n$
- O que faz este compilador?

# Análise Semântica – Tipos de Atributos

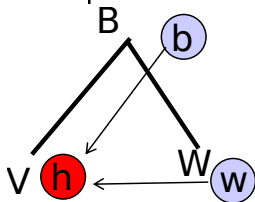
- ▶ Atributos Sintetizados

→ Depende somente de atributos dos filhos



- ▶ Atributos Herdados

→ Depende de atributos do pai ou dos irmãos



# Análise Semântica – Atributos herdados

- Considerando a gramática com ações semânticas

$$\begin{array}{lll} T & \rightarrow & FT' \quad T'.h = F.val \\ & & T.val = T'.s \\ T' & \rightarrow & *FT'_1 \quad T'_1.h = T'.h * F.val \\ & & T'.s = T'_1.s \\ T' & \rightarrow & \epsilon \quad T'.s = T'.h \\ F & \rightarrow & \text{digit} \quad F.val = \text{digit.lexval} \end{array}$$

- Crie a árvore de derivação anotada para
  - $3 * 5$

Ordem de Avaliação dos Atributos

# Análise Semântica – Ordem de Avaliação

- ▶ Cálculo de Atributos
  - ▶ Calculam atributos considerando outros atributos
  - ▶ Estabelece uma dependência entre as regras semânticas
- ▶ Grafo de dependências
  - ▶ Representa o fluxo de informações entre os atributos
  - ▶ Somente atributos sintetizados
    - ▶ O grafo é igual a árvore sintática
  - ▶ Atributos sintetizados e herdados
    - ▶ O grafo é parecido com a árvore sintática
  - ▶ Outra configuração de cálculo de atributo
    - ▶ Obriga uma análise semântica independente da sintática
- ▶ Esquemas S-atribuídos e L-atribuídos
  - ▶ Têm restrições para criar o grafo de dependência

# Análise Semântica – Exemplo

- Considerando a gramática com ações semânticas

$$\begin{array}{ll} T \rightarrow FT' & \begin{array}{l} T'.h = F.val \\ T.val = T'.s \end{array} \\ T' \rightarrow *FT'_1 & \begin{array}{l} T'_1.h = T'.h * F.val \\ T'.s = T'_1.s \end{array} \\ T' \rightarrow \epsilon & T'.s = T'.h \\ F \rightarrow \text{digit} & F.val = \text{digit.lexval} \end{array}$$

- Calcular o grafo de dependências para  $3 * 5$

# Análise Semântica – S-Atribuído

- Requisitos

- Todos os atributos são sintetizados
- Ações semânticas dispostas à direita das produções

- Exemplo

$L$	$\rightarrow$	$E \mathbf{n}$	$\{ L.val = E.val \}$
$E$	$\rightarrow$	$E_1 + T$	$\{ E.val = E_1.val + T.val \}$
$E$	$\rightarrow$	$T$	$\{ E.val = T.val \}$
$T$	$\rightarrow$	$T_1 * F$	$\{ T.val = T_1.val * F.val \}$
$T$	$\rightarrow$	$F$	$\{ T.val = F.val \}$
$F$	$\rightarrow$	$( E )$	$\{ F.val = E.val \}$
$F$	$\rightarrow$	<b>digit</b>	$\{ F.val = digit.lexval \}$

- Especialmente útil em analisadores ascendentes (LR)

- Não precisamos criar a árvore explicitamente
- Possibilita geração de código durante a análise sintática

# Análise Semântica – L-Atribuído

## ► Requisitos

- Atributos podem ser sintetizados ou
- Atributos podem ser herdados se
  - A dependência é da esquerda para a direita  
(e jamais da direita para a esquerda)

## ► Formalmente, considerando

- $A \rightarrow X_1 X_2 \dots X_n$  e atributo herdado  $X_i.a$   
 $X_i.a$  só pode depender de
  - atributos herdados associados à cabeça  $A$
  - atributos herdados/sintetizados às ocorrências dos símbolos  $X_1, X_2, \dots, X_{i-1}$  localizados à esquerda de  $X_i$

## ► Exemplo

$$D \rightarrow T \{ L.in = T.tipo + D.val \} L$$

# Exercícios

- Para cada  $f$  e cada regra, diga se é S-Atribuída, L-Atribuída, ou nenhuma delas?

$A \rightarrow BC$      $B.h = f_1(A.h)$   
                   $C.h = f_2(B.s)$   
                   $A.s = f_3(C.s)$

$A \rightarrow DE$      $E.h = f_4(A.h)$   
                   $D.h = f_5(E.s)$   
                   $A.s = f_6(D.s)$

- Considerando esta G, crie o grafo para *float*  $id_1$ ,  $id_2$ ,  $id_3$

$D \rightarrow TL$      $L.h = T.tipo$   
 $T \rightarrow \text{int}$      $T.tipo = \text{inteiro}$   
 $T \rightarrow \text{float}$      $T.tipo = \text{flutuante}$   
 $L \rightarrow L_1, \text{id}$      $L_1.h = L.h$   
                                   $\text{adicionaTipo}(\text{id.key}, L.h)$   
 $L \rightarrow \text{id}$      $\text{adicionaTipo}(\text{id.key}, L.h)$

# Projeto de Compilador

## Lançamento da Etapa 4

# Conclusão

- ▶ Leituras Recomendadas
  - ▶ Livro do Dragão
    - ▶ Capítulo 5 até 5.2 inclusive
  - ▶ Série Didática
    - ▶ Capítulo 4 até 4.3 inclusive
  - ▶ Keith Cooper
    - ▶ Capítulo 4, seções 4.1 e 4.3
- ▶ Próxima Aula

Análise Semântica 2/2