Tradução Dirigida à Sintaxe

Prof. Ricardo Santos

Tradução Dirigida à Sintaxe

1. Símbolos da gramática possuem **atributos** para manter informação das construções de LP que eles (os símbolos) representam

3. Os valores desses atributos são avaliados através de regras semânticas **associadas às regras de produção**

5. A avaliação dessas regras semânticas:
   - Pode gerar código intermediário
   - Pode inserir informação na TS
   - Pode realizar a checagem de tipos
   - Pode emitir mensagens de erro

Definições Dirigidas à Sintaxe e Esquemas de Tradução

1. Ao associar regras semânticas com produções pode-se adotar duas técnicas
   - **Definições Dirigidas à Sintaxe (Syntax-Directed Definitions/SDDs)**
   - **Esquemas de Tradução**

B. SDDs:

- Fornece especificações de alto nível para as traduções
- Esconde detalhes de implementação tais como: ordem de avaliação das ações semânticas
- Associamos regras de produção com um conjunto de ações semânticas

C. Esquemas de Tradução:

- Indicam a ordem de avaliação das ações semânticas associadas com uma regra de produção

Definições Dirigidas à Sintaxe

1. Uma SDD é uma generalização de uma gramática livre de contexto em que:
   - Cada símbolo da gramática está associado com atributo(s)
   - Esse(s) atributo(s) pode(m) ser:
     - **Sintetizados ou**
     - **Herdados**
   - Cada regra de produção possui um conjunto de regras semânticas

2. Regras semânticas definem dependências entre atributos. Tais dependências podem ser representadas por um grafo de dependências

4. O grafo de dependências determina a ordem de avaliação dessas regras semânticas

6. A avaliação de uma regra semântica define a avaliação de um atributo. No entanto, uma regra semântica também pode ter outros efeitos (Ex: imprimir um valor)

8. Uma árvore sintática que exibe os valores dos atributos em cada nó é chamada de árvore sintática anotada

Definição Dirigida à Sintaxe

Em uma SDD, cada produção $A \rightarrow \alpha$ está associada com um conjunto de regras semânticas na forma:

$$b=f(c_1,c_2,\ldots,c_n)$$

Onde *f* é uma função e *b* pode ser um dos seguintes:

➔ *b* é um atributo sintetizado de A e $c_1,c_2,\ldots,c_n$ são atributos dos símbolos da gramática na produção $A \rightarrow \alpha$.

OU

➔ *b* é um atributo herdado de um dos símbolos da gramática em α, e $c_1,c_2,\ldots,c_n$ são atributos dos símbolos da gramática na produção $A \rightarrow \alpha$.

Gramática de Atributos

- Assim, uma regra semântica $b=f(c_1,c_2,\ldots,c_n)$ indica que o atributo b *depende dos atributos* $c_1,c_2,\ldots,c_n$.

- Uma gramática de atributos **é uma definição dirigida à sintaxe em que as funções nas regras semânticas não podem ter outros efeitos além de avaliar os valores dos atributos**

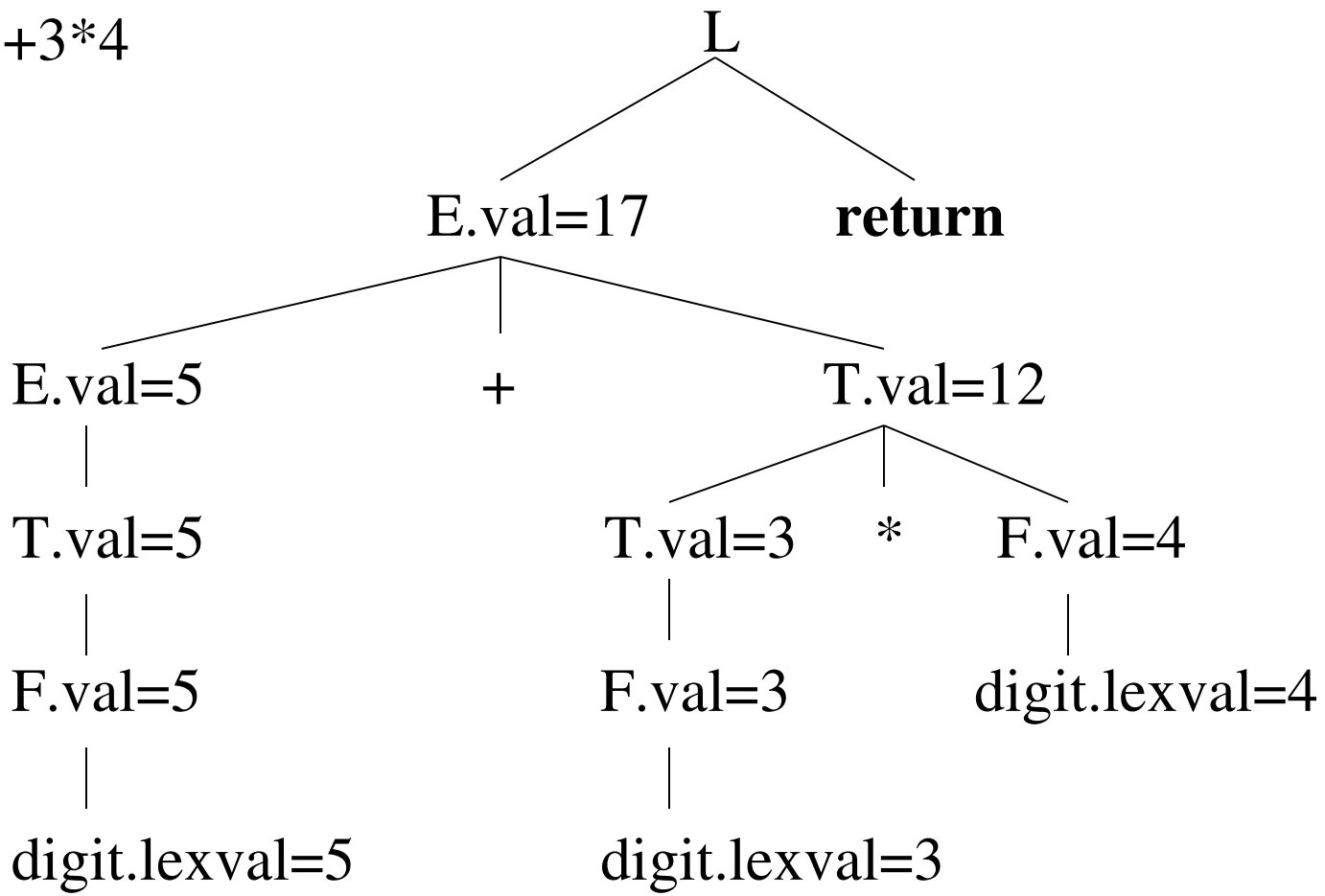
Definição Dirigida à Sintaxe -- Exemplo

Produção	**Regras Semânticas**
$L \rightarrow E$ **return**	print(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
$F \rightarrow$ **digit**	F.val = **digit**.lexval

10. Símbolos E, T e F estão associados com o atributo sintetizado *val*.
11. O token **digit** tem um atributo sintetizado *lexval*

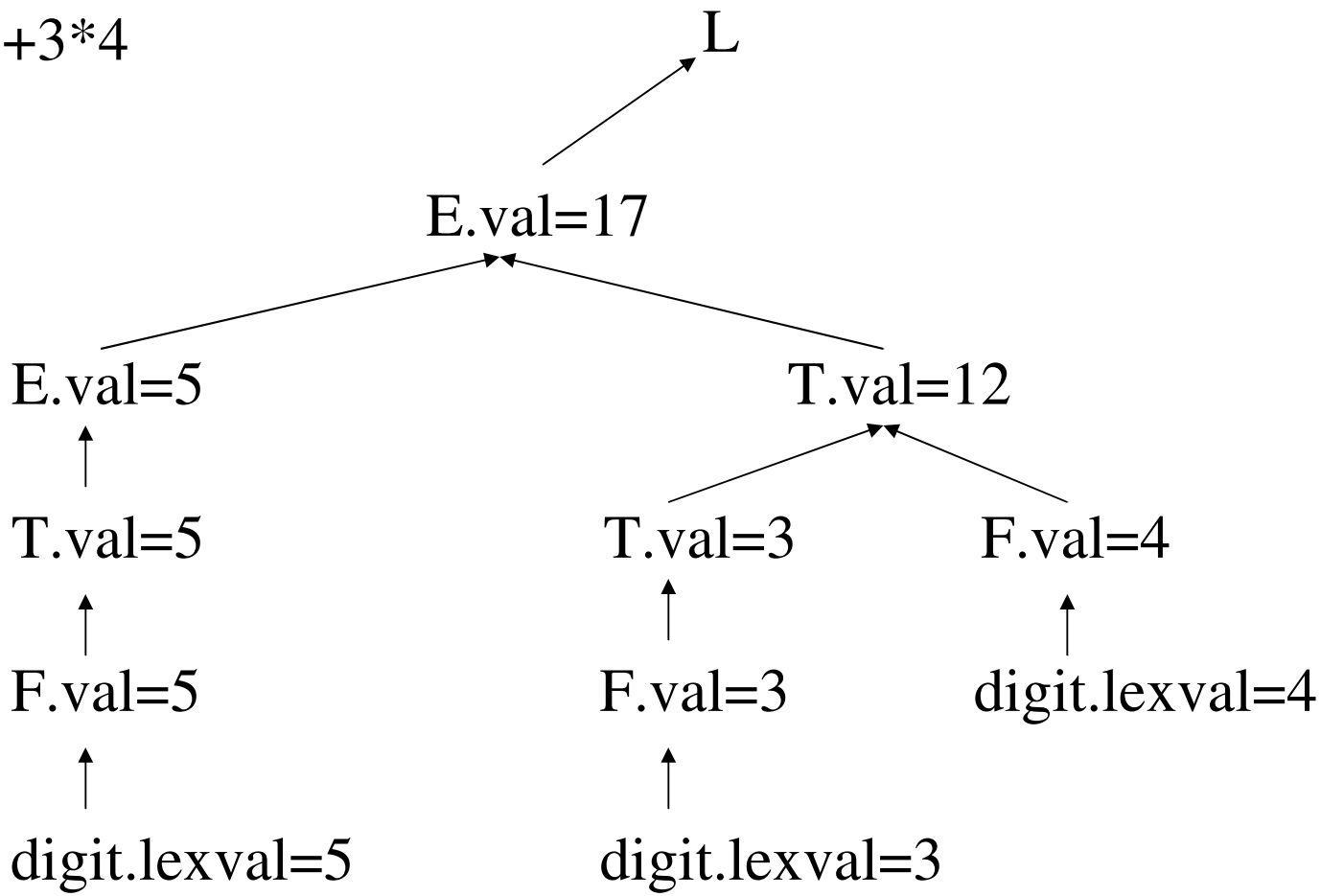
Árvore Sintática Anotada -- Exemplo

Entrada: 5+3*4



Grafo de Dependência

Entrada: 5+3*4



Definição Dirigida à Sintaxe – Exemplo2

Produções	**Regras Semânticas**
$E \rightarrow E_1 + T$	E.loc=newtemp(), E.code = E_1.code ‖ T.code ‖ add E_1.loc,T.loc,E.loc
$E \rightarrow T$	E.loc = T.loc, E.code=T.code
$T \rightarrow T_1 * F$	T.loc=newtemp(), T.code = T_1.code ‖ F.code ‖ mult T_1.loc,F.loc,T.loc
$T \rightarrow F$	T.loc = F.loc, T.code=F.code
$F \rightarrow (E)$	F.loc = E.loc, F.code=E.code
F → **id**	F.loc = **id**.name, F.code=“”

10. Símbolos E, T e F estão associados com os atributos sintetizados *loc* e *code*.
11. O token **id** tem um atributo sintetizado *name*.
12. Considere que o símbolo “||” é um operador de concatenação de strings.

Definição Dirigida à Sintaxe – Atributos Herdados

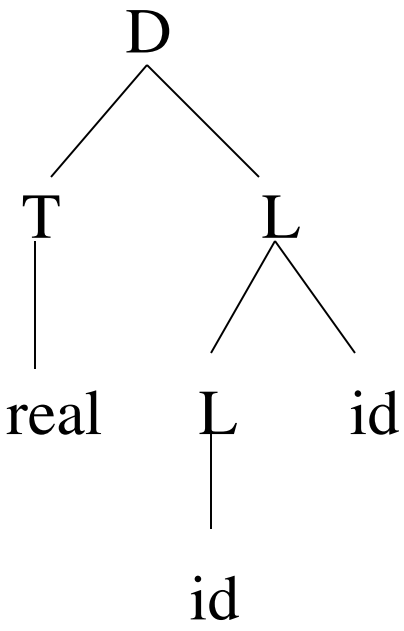
Produção	**Regras Semânticas**
$D \rightarrow T\ L$	L.in = T.type
$T \rightarrow$ **int**	T.type = integer
$T \rightarrow$ **real**	T.type = real
$L \rightarrow L_1$ **id**	L_1.in = L.in, addtype(**id**.entry,L.in)
$L \rightarrow$ **id**	addtype(**id**.entry,L.in)

8. O símbolo T está associado com o atributo sintetizado *type*.

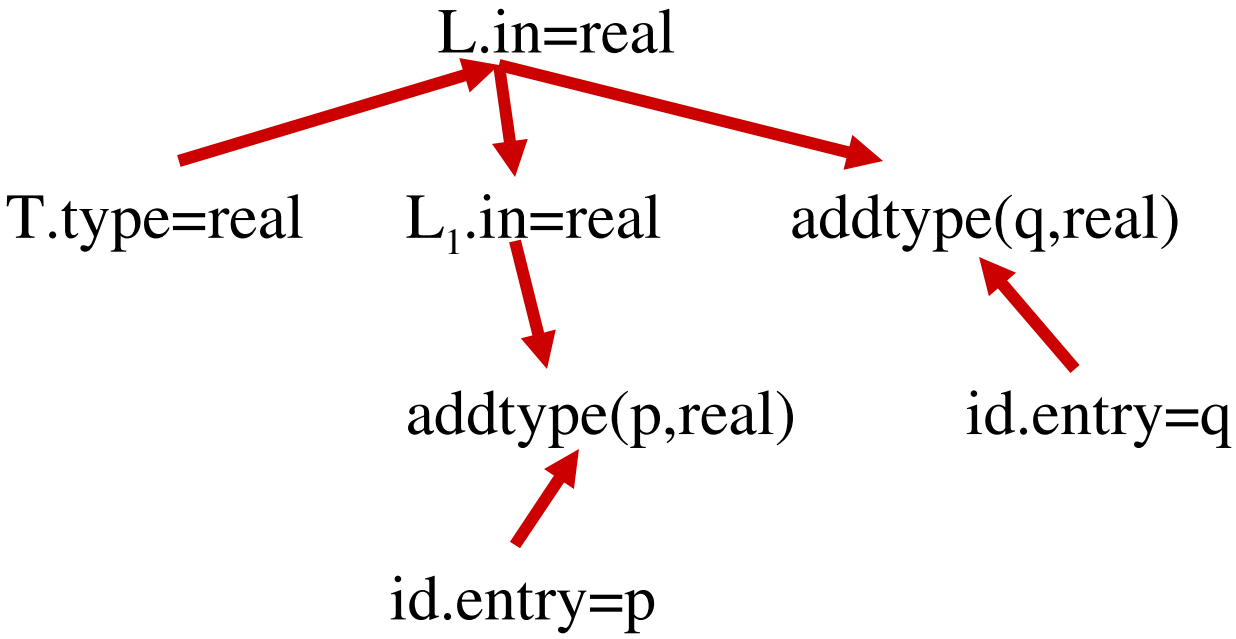
10. O símbolo L está associado com o atributo herdado *in.*

Grafo de Dependência – Atributos Herdados

Entrada: `real p q`



parse tree



dependency graph

Árvores Sintáticas

1. Separando tradução de árvores sintáticas
2. Árvore sintática: é uma representação intermediária da entrada (do programa fonte)
3. Procedimentos: *mknode*, *mkleaf*
4. Utilização do atributo sintetizado *nptr* (pointer)

PRODUÇÃO	**REGRA SEMÂNTICA**
$E \rightarrow E_1 + T$	$E.nptr = mknode(\text{“+”}, E_1.nptr, T.nptr)$
$E \rightarrow E_1 - T$	$E.nptr = mknode(\text{“-”}, E_1.nptr, T.nptr)$
$E \rightarrow T$	$E.nptr = T.nptr$
$T \rightarrow (E)$	$T.nptr = E.nptr$
$T \rightarrow id$	$T.nptr = mkleaf(id, id.lexval)$
$T \rightarrow num$	$T.nptr = mkleaf(num, num.val)$

Definições S-Atribuídas

1. SDDs são usadas para especificar traduções dirigidas à sintaxe

3. Criar um tradutor para uma SDD arbitrária pode ser complexo

5. É desejável fazer a análise semântica durante a análise sintática (isto é, em um único passo realiza a análise sintática e avalia as regras semânticas).

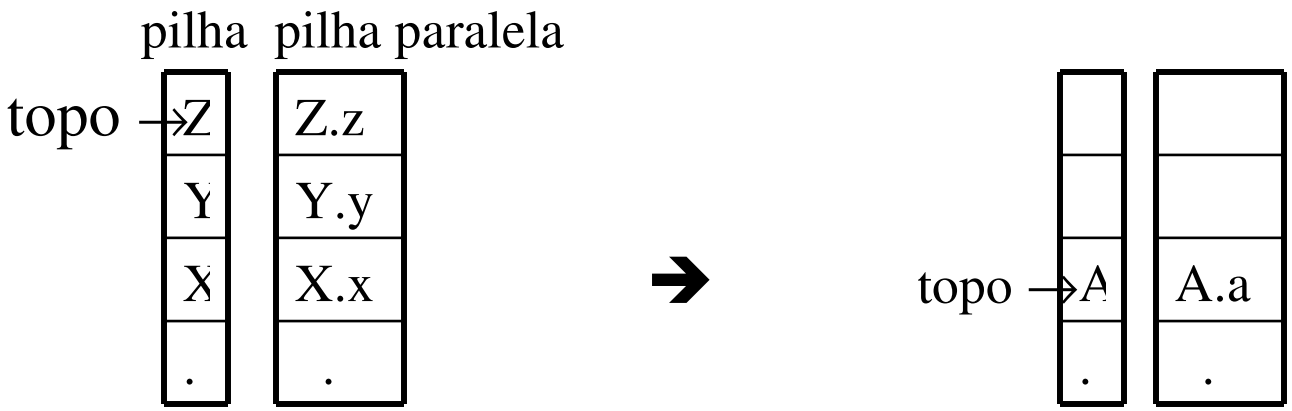
7. Duas sub-classes de SDDs:
   - **Definições S-atribuídas**: apenas atributos sintetizados são usados em SDDs.
   - **Definições L-atribuídas**: além dos atributos sintetizados, pode-se também usar atributos herdados

8. Para implementar definições S-atribuídas e L-atribuídas pode-se avaliar regras semânticas em um único passo

Avaliação Bottom-Up de Definições S-Atribuídas

- Coloca-se valores dos atributos sintetizados dos símbolos da gramática em uma pilha paralela
  - Quando uma entrada da pilha da sintaxe possui o símbolo X (terminal ou não-terminal), a entrada correspondente na pilha paralela manterá o atributo sintetizado do símbolo X
- Avalia-se os valores dos atributos durante a redução.

$A \rightarrow XYZ$ A.a=f(X.x,Y.y,Z.z) onde todos os atributos são sintetizados



Avaliação Bottom-Up de Definições S-Atribuídas (cont.)

Produção	**Regras Semânticas**
L → E **return**	print(val[top-1])
$E \rightarrow E_1 + T$	val[ntop] = val[top-2] + val[top]
E → T	
$T \rightarrow T_1 * F$	val[ntop] = val[top-2] * val[top]
T → F	
F → (E)	val[ntop] = val[top-1]
F → **digit**	

11. Em cada shift de **digit**, colocamos **digit.lexval** em *val-stack*.
12. Nos demais shifts, nada é inserido em *val-stack* pois outros terminais não possuem atributos (mas o ponteiro da pilha val-stack é incrementado).

    ntop = top – r + 1

    r = no. de símbolos no lado direito da produção

Avaliação Bottom-Up -- Exemplo

- Em cada shift, inserimos **digit.lexval** em *val-stack*.

stack	val-stack	input	action	semantic rule
0		5+3*4r	s6	d.lexval(5) em val-stack
0d6	5	+3*4r	F→d	F.val=d.lexval
0F4	5	+3*4r	T→F	T.val=F.val
0T3	5	+3*4r	E→T	E.val=T.val
0E2	5	+3*4r	s8	insere slot vazio em val-stack
0E2+8	5-	3*4r	s6	d.lexval(3) em val-stack
0E2+8d6	5-3	*4r	F→d	F.val=d.lexval
0E2+8F4	5-3	*4r	T→F	T.val=F.val
0E2+8T11	5-3	*4r	s9	insere slot vazio em val-stack
0E2+8T11*9	5-3-	4r	s6	d.lexval(4) em val-stack
0E2+8T11*9d6	5-3-4	r	F→d	F.val=d.lexval
0E2+8T11*9F12	5-3-4	r	T→T*F	$T.val=T_1.val*F.val$
0E2+8T11	5-12	r	E→E+T	$E.val=E_1.val*T.val$
0E2	17	r	s7	insere slot vazio em val-stack
0E2r7	17-	$	L→Er	print(17), retira slot vazio de val-stack
0L1	17	$	acc	