Construção de Compiladores

Análise sintática

Professor: Luciano Ferreira Silva, Dr.

Árvores sintáticas

- **A árvore sintática é uma representação das derivações utilizadas no reconhecimento de uma sentença em uma estrutura de árvore;**
  - ✓ As folhas são os símbolos terminais;
  - ✓ O nó raiz é sempre o símbolo sentencial;
  - ✓ Os nós intermediários correspondem aos demais símbolos não-terminais de maneira a representar devidas produções.

Árvores sintáticas

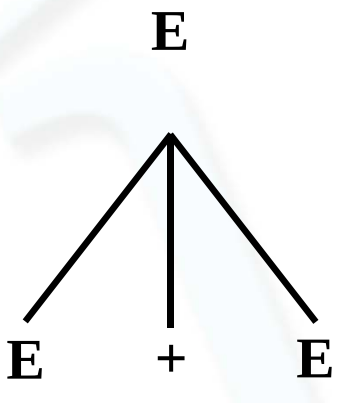
- Por exemplo: considere ainda a gramática G, apresentada abaixo;

Gramática G para expressões com soma e multiplicação

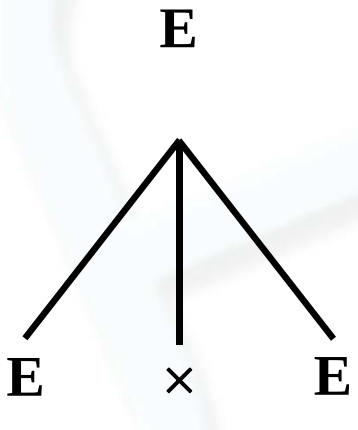
$E \rightarrow E + E$	Produção 1: contempla a soma
$E \rightarrow E \times E$	Produção 2: propicia a multiplicação
$E \rightarrow (E)$	Produção 3: possibilita o uso de parênteses
$E \rightarrow v$	Produção 4: terminal

Árvores sintáticas

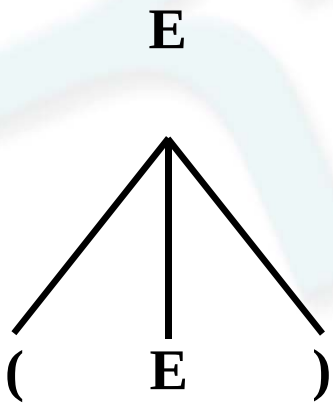
- Uma árvore sintática para uma sentença da gramática G só poderá conter subárvores na forma:



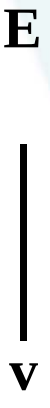
(a) $E \rightarrow E + E$



(b) $E \rightarrow E \times E$



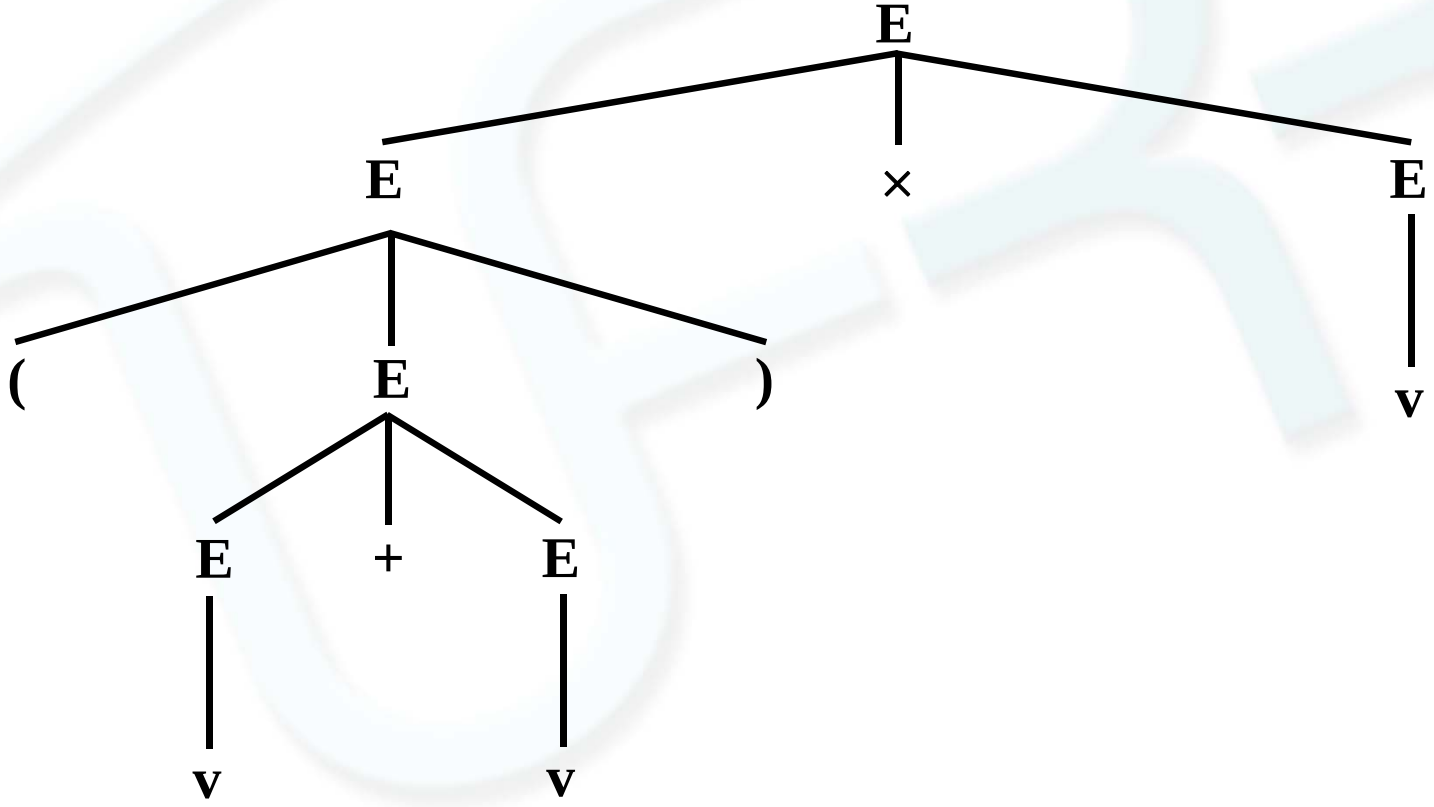
(c) $E \rightarrow (E)$



(d) $E \rightarrow v$

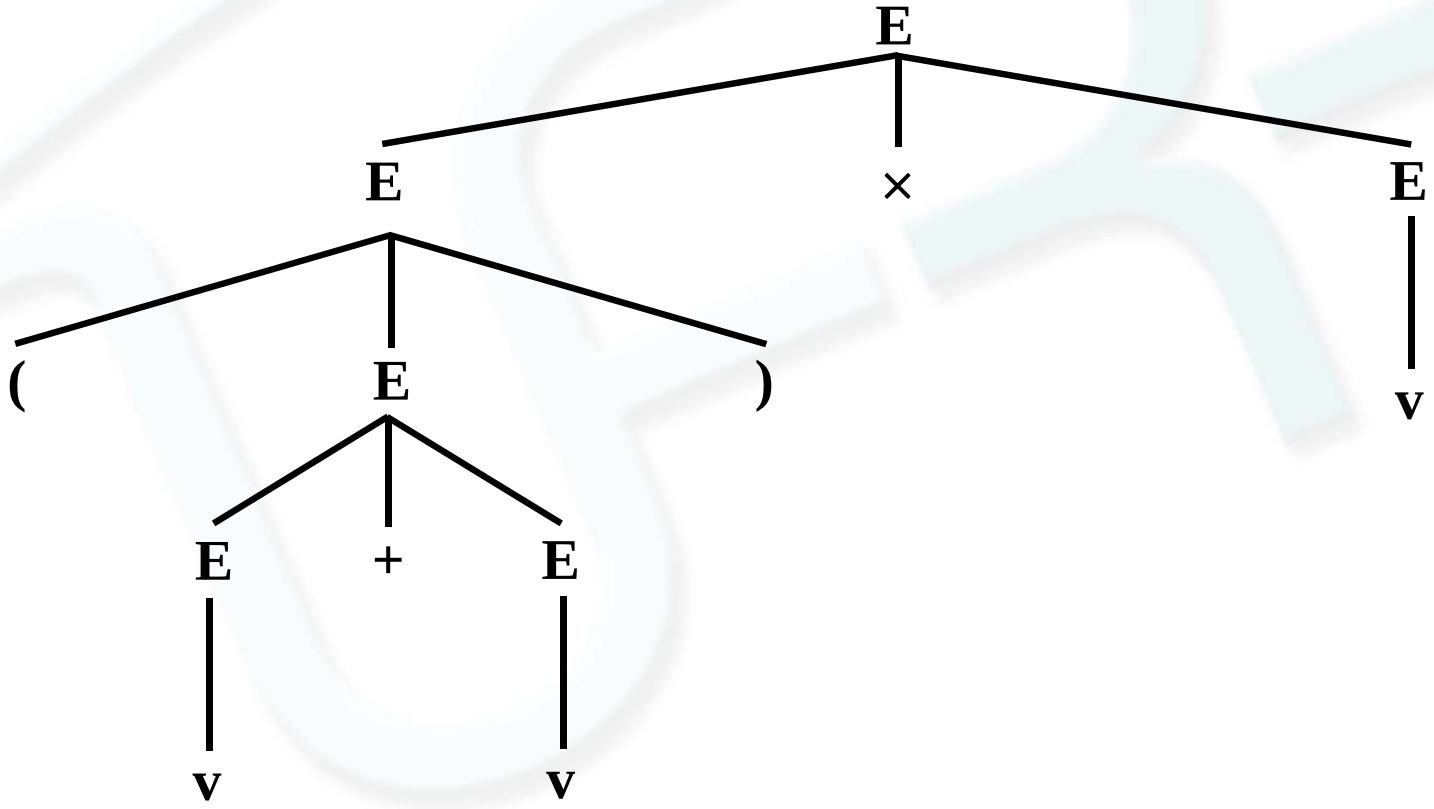
Árvores sintáticas

- Construção descendente – usa-se a seqüência de reconhecimento mais à esquerda (2, 3, 1, 4, 4, 4 para (v + v) × v):



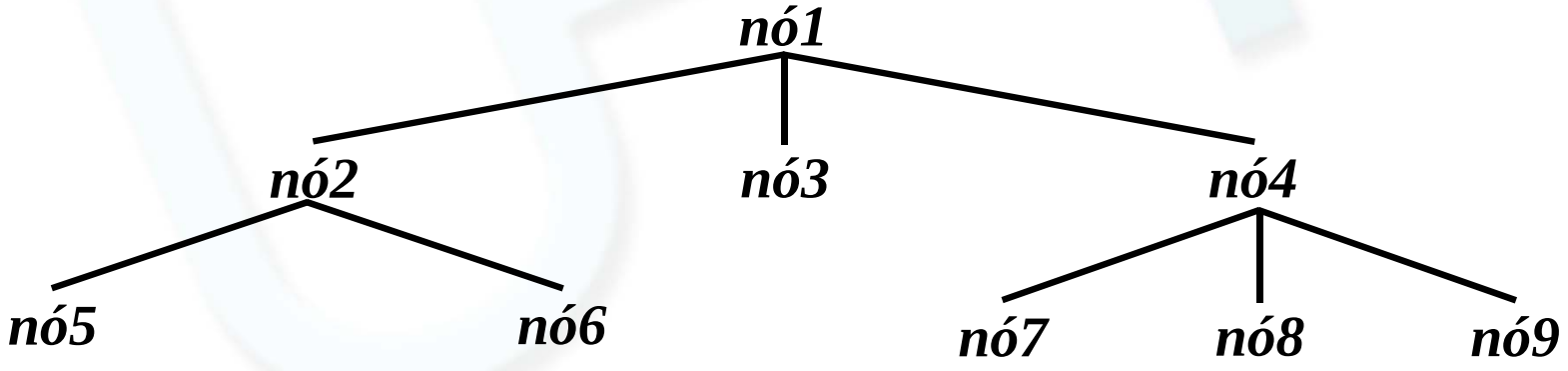
Árvores sintáticas

- Construção ascendente – usa-se a seqüência de reconhecimento mais à direita (4, 4, 1, 3, 4, 2 para (v + v) × v):



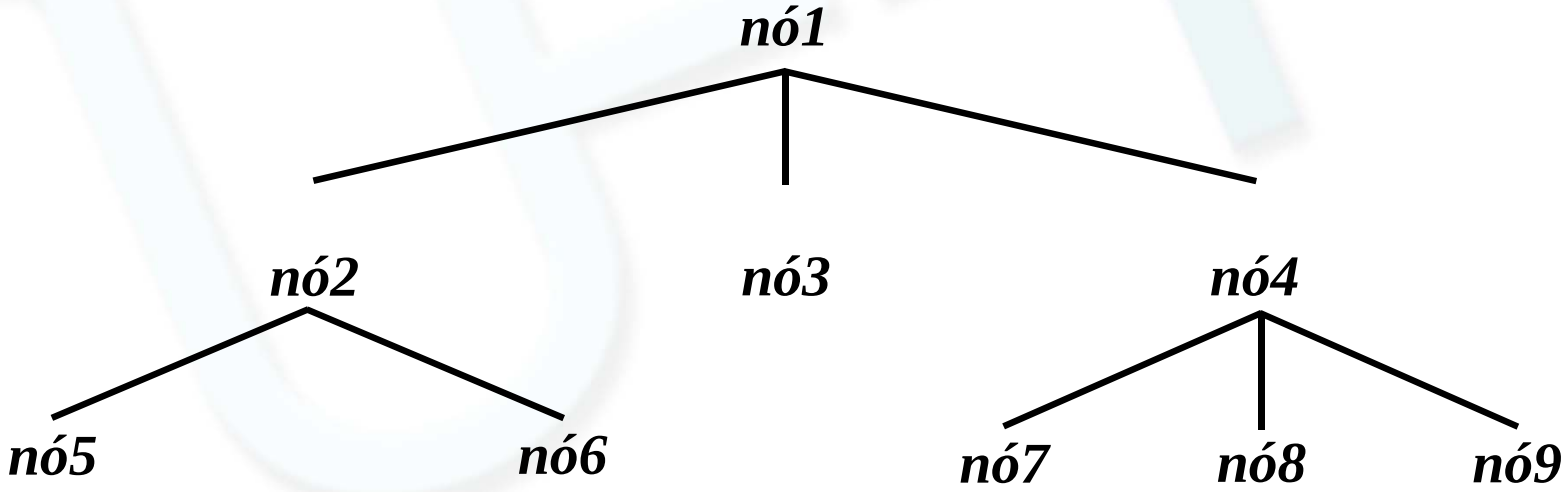
Árvores sintáticas

- **Estratégias de varredura:**
  - ✓ Pré-ordem:
    - o nó raiz é o primeiro;
    - os elementos são lidos da esquerda para direita;
    - cada subárvore é percorrida em pré-ordem;
    - Exemplo: *nó1 nó2 nó5 nó6 nó3 nó4 nó7 nó8 nó9*



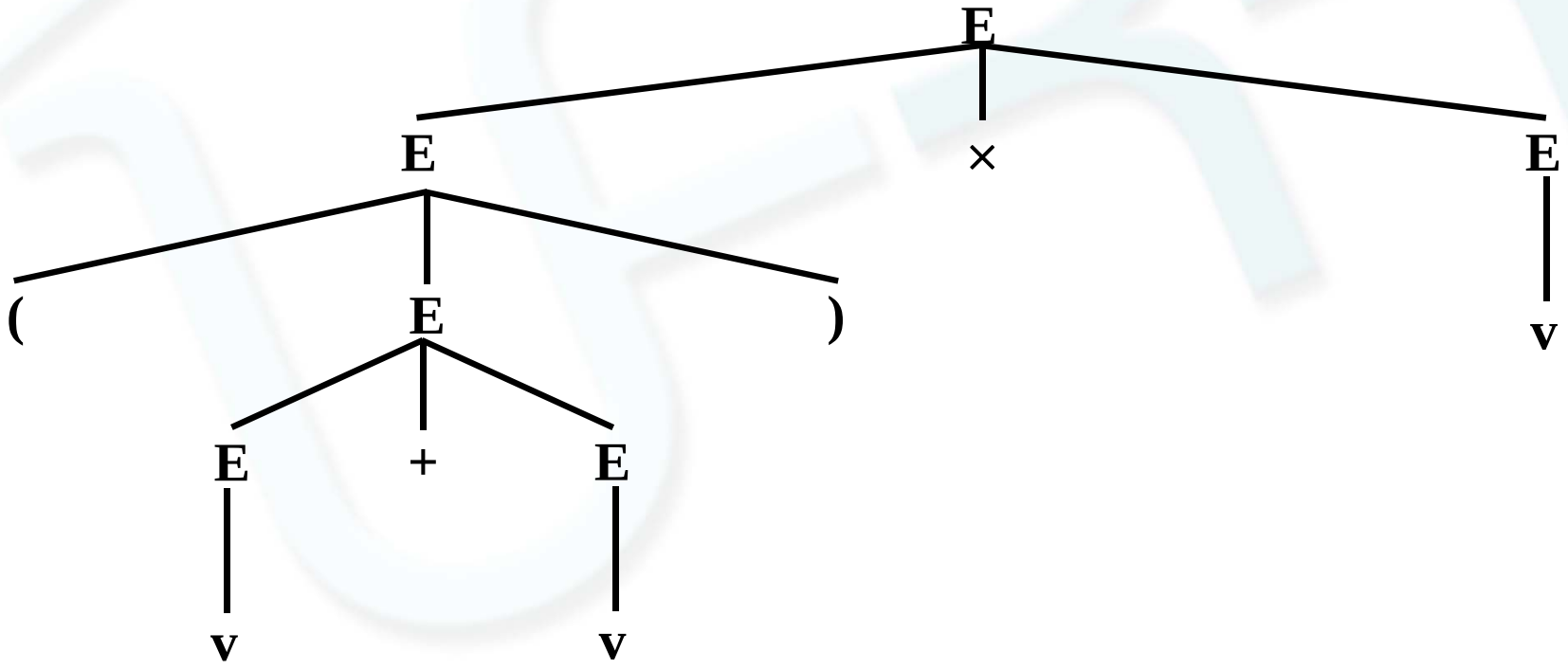
Árvores sintáticas

- Pós-ordem:
  - os primeiros elementos a serem lidos são os nós das subárvores, da esquerda para direita;
  - o nó raiz é o último elemento da seqüência;
  - Exemplo: *nó5 nó6 nó2 nó3 nó7 nó8 nó9 nó4 nó1*



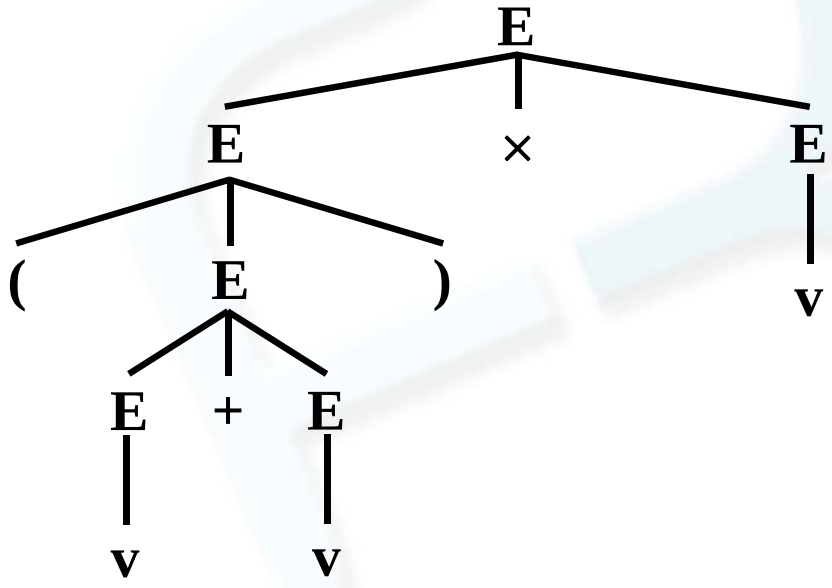
Árvores sintáticas

- Construção descendente – usa-se a estratégia de varredura pré-ordem: 2, 3, 1, 4, 4, 4 para (v + v) × v
- Construção ascendente – usa-se a estratégia de varredura pós-ordem: 4, 4, 1, 3, 4, 2 para (v + v) × v



Gramáticas ambíguas

- Uma gramática ambígua é aquela que permite que uma mesma sentença tenha mais de uma árvore sintática;
- Considere a árvore sintática da expressão (v + v) × v:



- Observe que a árvore indica claramente que a expressão entre parênteses é avaliada antes da multiplicação;

Gramáticas ambíguas

- **Considere agora a sentença v + v × v**

1. Em uma construção descendente, com derivação canônica mais a esquerda, a seqüência 1, 4, 2, 4, 4, leva o símbolo sentencial à sentença;

   1. $E \overset{1}{\Rightarrow} E + E$
   2. $E + E \overset{4}{\Rightarrow} v + E$
   3. $v + E \overset{2}{\Rightarrow} v + E \times E$
   4. $v + E \times E \overset{4}{\Rightarrow} v + v \times E$
   5. $v + v \times E \overset{4}{\Rightarrow} v + v \times v$

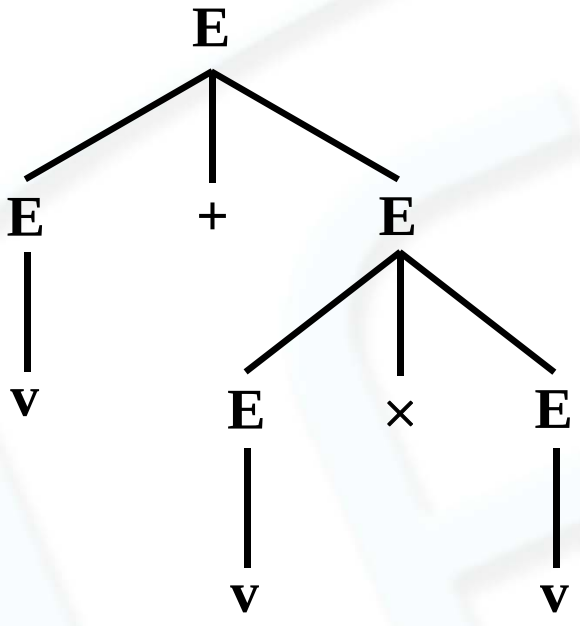
Gramáticas ambíguas

2. Para a mesma sentença v + v × v com uma construção descendente, com derivação canônica mais a esquerda, a seqüência 2, 1, 4, 4, 4, também leva o símbolo sentencial à sentença;

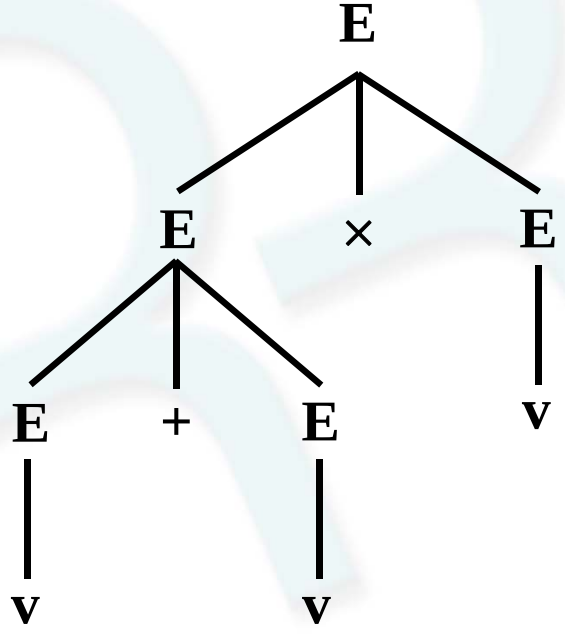
1. $E \overset{2}{\Rightarrow} E \times E$
2. $E \times E \overset{1}{\Rightarrow} E + E \times E$
3. $E + E \times E \overset{4}{\Rightarrow} v + E \times E$
4. $v + E \times E \overset{4}{\Rightarrow} v + v \times E$
5. $v + v \times E \overset{4}{\Rightarrow} v + v \times v$

Gramáticas ambíguas

- **Árvores sintáticas:**



1. A seqüência 1, 4, 2, 4, 4 equivale a v + (v × v);



2. A seqüência 2, 1, 4, 4, 4 equivale a (v + v) × v;

Gramáticas ambíguas

- A segunda seqüência de produções leva a processamento matematicamente errôneo da expressão;
- Gramáticas ambíguas não favorecem ao reconhecimento automático de sentenças;
- Dada uma gramática qualquer não há um procedimento que possa reconhecer se ela é ou não ambígua.
- Mas, uma vez que se observe que a gramática é ambígua, é possível reescrever suas produções de forma a eliminar a ambigüidade.

Gramáticas ambíguas

- Uma nova gramática (G`), equivalente a G, que remove a sua ambigüidade, encontra-se abaixo (com E sentencial).

Gramática G`, equivalente à gramática G mas sem ambiguidade

E → E + M	Produção 1: adição com recursividade
E → M	Produção 2: marca o fim da recursividade da adição
M → M x P	Produção 3: multiplicação com recursividade, contempla a associatividade
M → P	Produção 4: marca o fim recursividade da multiplicação
P → (E)	Produção 5: possibilita o uso de parênteses
P → v	Produção 6: terminal

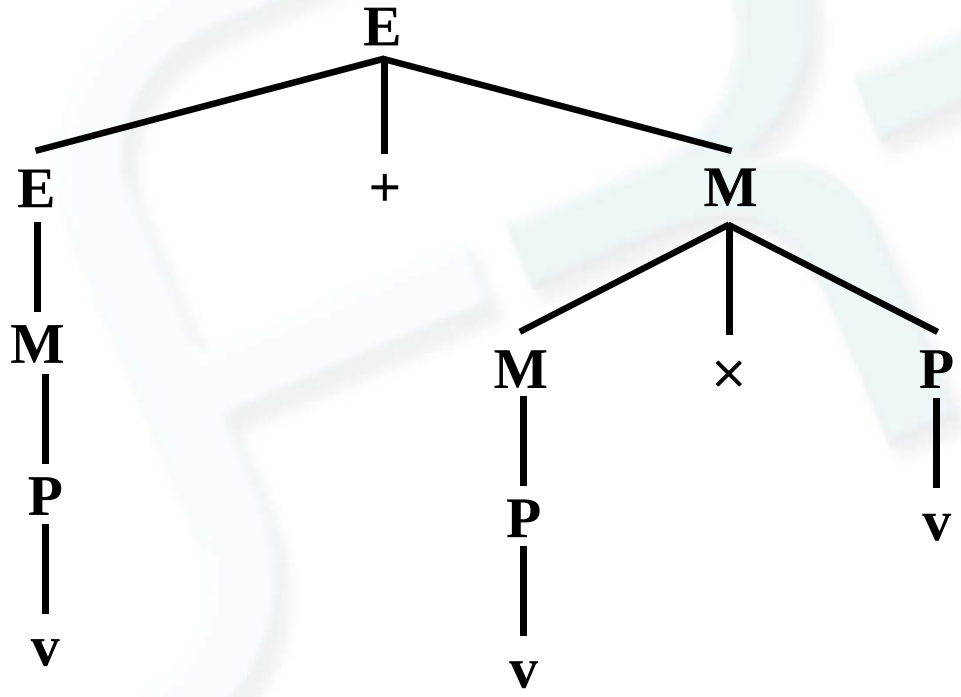
Gramáticas ambíguas

- **Para a sentença v + v × v com uma construção descendente, com derivação canônica mais a esquerda, sobre G`, a seqüência seria 1, 2, 4, 6, 3, 4, 6, 6;**

1. $E \overset{1}{\Rightarrow} E + M$
2. $E + M \overset{2}{\Rightarrow} M + M$
3. $M + M \overset{4}{\Rightarrow} P + M$
4. $P + M \overset{6}{\Rightarrow} v + M$
5. $v + M \overset{3}{\Rightarrow} v + M \times P$
6. $v + M \times P \overset{4}{\Rightarrow} v + P \times P$
7. $v + P \times P \overset{6}{\Rightarrow} v + v \times P$
8. $v + v \times P \overset{6}{\Rightarrow} v + v \times v$

Gramáticas ambíguas

- A árvore sintática (única) associada a seqüência de produções anterior é:



Gramáticas ambíguas

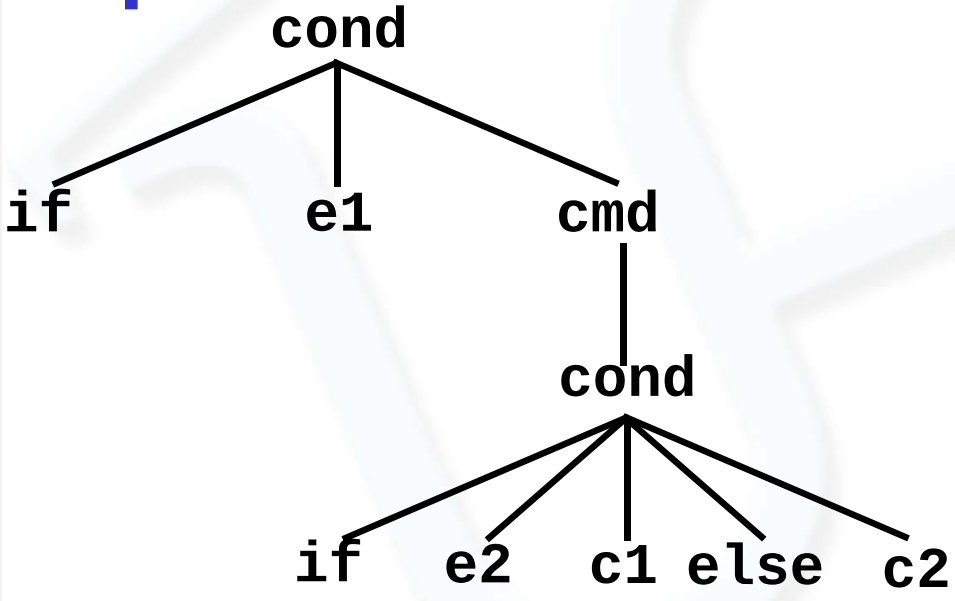
- Considere a gramática H que permite verificar a sintaxe de comandos condicionais.

Gramática H, representa comandos condicionais em liguagens de programação

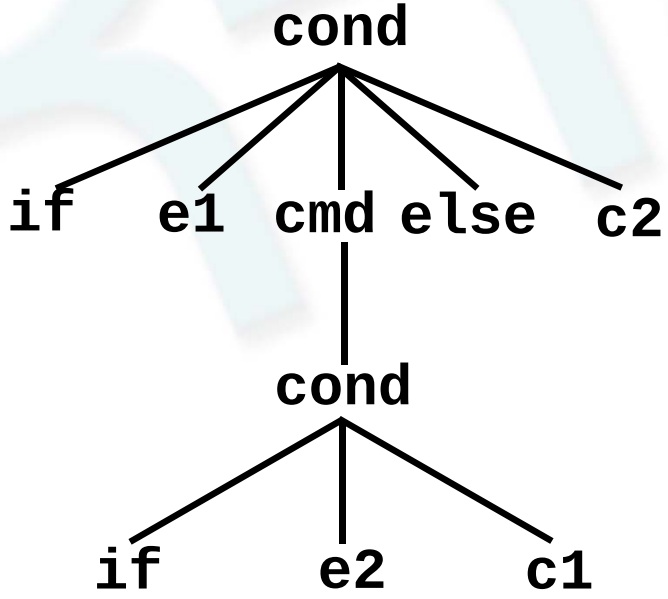
cond → `if` (expr) cmd	Produção 1: usando `if` sem o `else`
cond → `if` (expr) cmd `else` cmd	Produção 2: usando `if` com o `else`
cmd → cond	Produção 3: usando comandos codicionais dentro do `if` e `else`

Gramáticas ambíguas

- **Considere o trecho de código:**
  - ✓ `if (e1) if (e2) c1 else c2`
- **São possíveis duas árvores sintáticas para este comando:**



1. **Interpretação usual**



2. **Outra interpretação possível**

Gramáticas ambíguas

- **Na construção de analisadores sintáticos, nem sempre é necessário ter produções não-ambíguas;**
  - ✓ É possível definir alternativas para o tratamento de ambigüidades sem que seja necessário ter a gramática construída sem produções ambíguas;
    - Atribuindo de uma ordem preferencial para a aplicação de produções;
    - Definindo precedências distintas para operadores.