Análise Sintáctica

Compiladores, Aula Nº 12

João M. P. Cardoso

Vocabulário gramatical

- Derivação à esquerda (Leftmost)
  - Expandir sempre o símbolo não terminal mais à esquerda que resta

$$NT_1\ T_1\ T_2\ T_3\ NT_2\ NT_3$$

Aplicar primeiro a produção aqui

E só depois aos outros símbolos não-terminais

- Derivação à direita (Rightmost)
  - Expandir sempre o símbolo não terminal mais à direita que resta

Ponto Inicial

- Assumir que a análise lexical produziu uma sequência de tokens (símbolos terminais)
  - Cada token tem um tipo e um valor
  - Tipos correspondem a símbolos terminais
  - Valores correspondem ao conteúdo do token lido
- Exemplo
  - INT(549) – token que identifica um inteiro de valor lido 549
  - IF – palavra chave “if” sem necessidade de valor
  - OP(+) – operador com valor +

Abordagem Básica

- Começar pelo símbolo Start ou pela primeira produção
- Construir uma derivação *leftmost*
  - Se o símbolo *leftmost* é não-terminal, seleccionar uma produção e aplicá-la
  - Se o símbolo *leftmost* é terminal, fazer corresponder com a entrada
  - Se todos os terminais foram correspondidos, foi encontrada uma derivação que aceita a String!
  - Chave: encontrar as produções correctas para os símbolos não-terminais

Gramática do Exemplo

INT = [0-9]+

Start → Expr

Expr → Expr “+” Term

Expr → Expr “-” Term

Expr → Term

Term → Term “*” INT

Term → Term “/” INT

Term → INT

- Conjunto de tokens (símbolos terminais):
  { +, -, *, /, INT }

Analisador Sintáctico para a gramática exemplo

Árvore Sintáctica

Start

Posição corrente na árvore sintáctica

Entrada que falta

2-2*2

Forma sentencial

Start

Analisador Sintáctico para a gramática exemplo

Árvore Sintáctica

Start

Expr

Posição corrente na árvore sintáctica

Entrada que falta

2-2*2

Forma sentencial

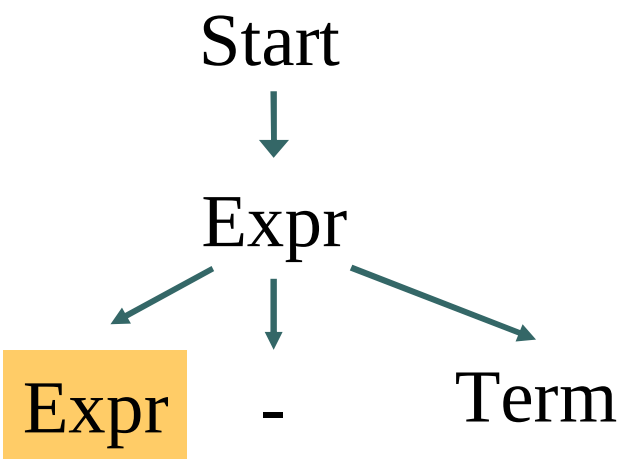
Expr

Produção Aplicada

Start → Expr

Analisador Sintáctico para a gramática exemplo

Árvore Sintáctica



Entrada a faltar

2-2*2

Forma sentencial

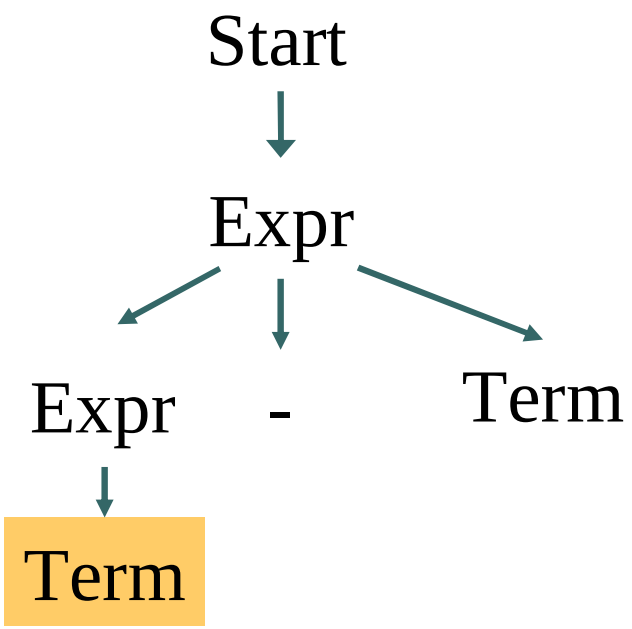
Expr - Term

Produção Aplicada

Expr → Expr - Term

Analisador Sintáctico para a gramática exemplo

Árvore Sintáctica



Entrada que falta

2-2*2

Forma sentencial

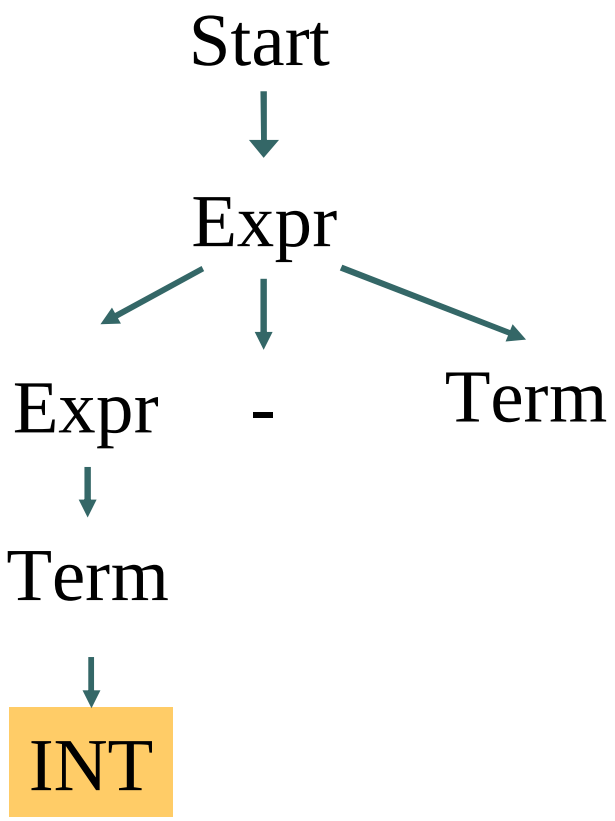
Term - Term

Produção Aplicada

Expr → Term

Analisador Sintáctico para a gramática exemplo

Árvore Sintáctica



Entrada que falta

2-2*2

Forma sentencial

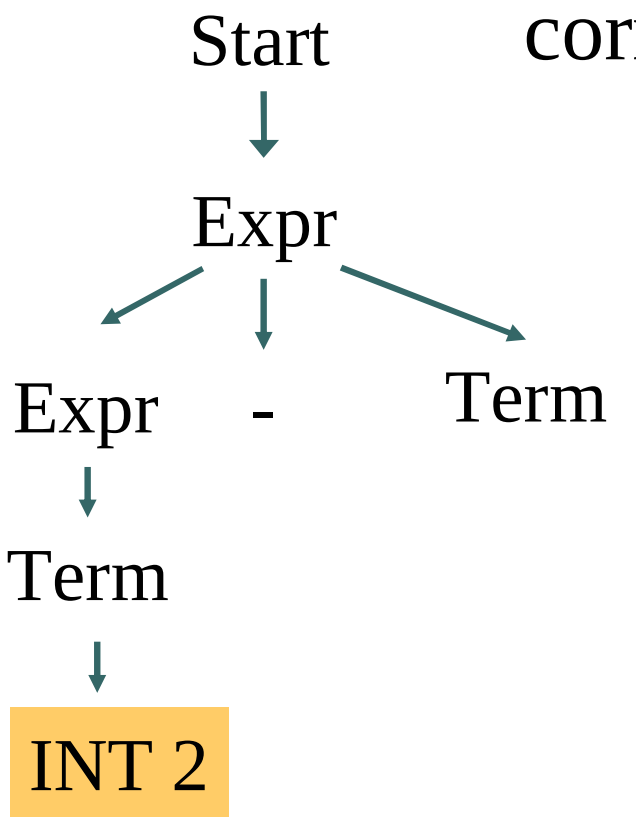
INT - Term

Produção Aplicada

Term → INT

Analisador Sintáctico para a gramática exemplo

Árvore Sintáctica



Token corresponde!

Entrada que falta

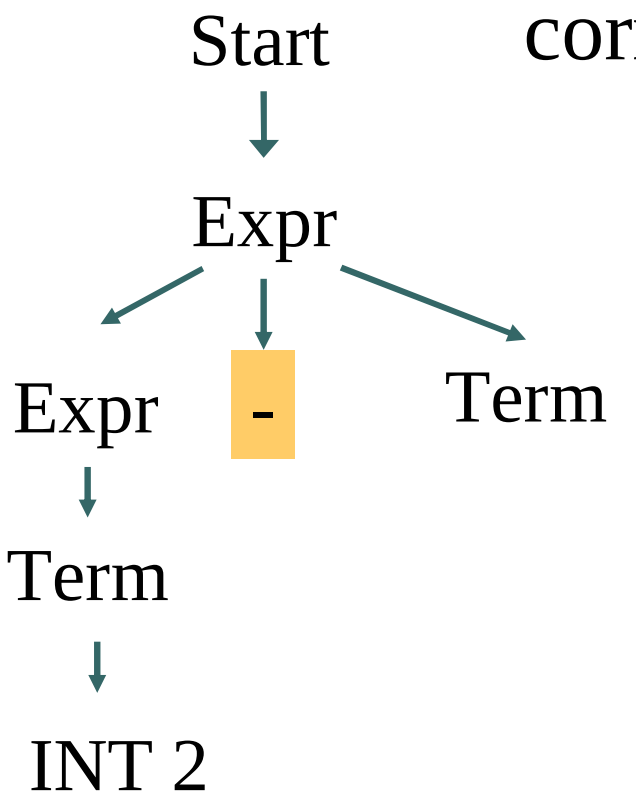
2-2*2

Forma sentencial

2 - Term

Analisador Sintáctico para a gramática exemplo

Árvore Sintáctica



Token corresponde!

Entrada que falta

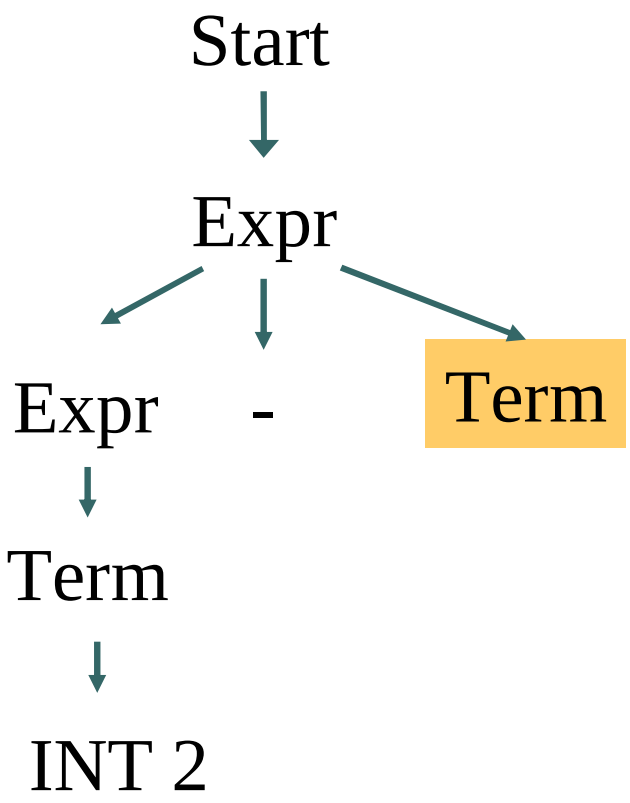
-2*2

Forma sentencial

2 - Term

Analisador Sintáctico para a gramática exemplo

Árvore Sintáctica



Entrada que falta

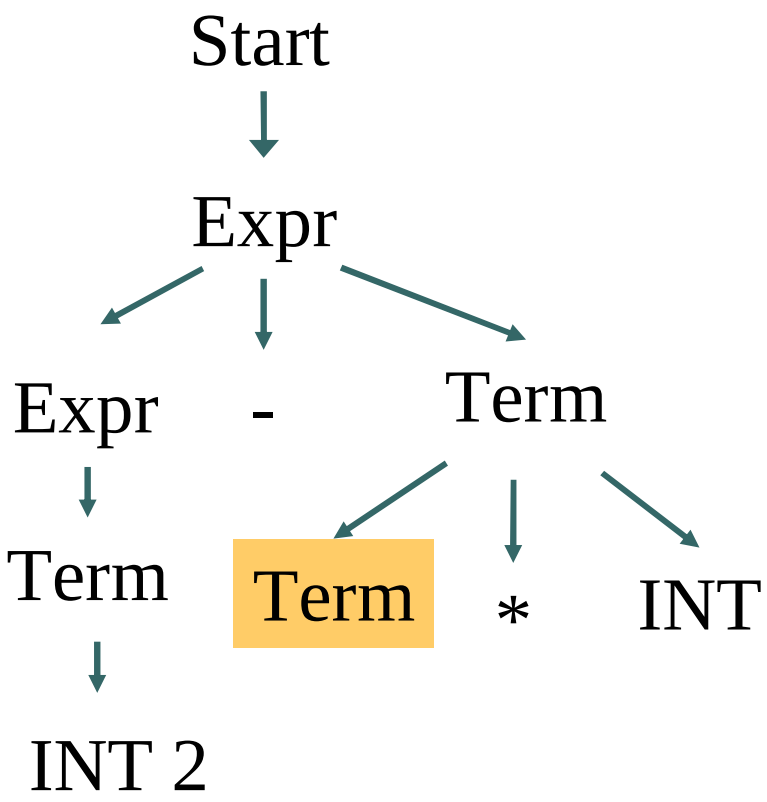
2*2

Forma sentencial

2 - Term

Analisador Sintáctico para a gramática exemplo

Árvore Sintáctica



Entrada que falta

2*2

Forma sentencial

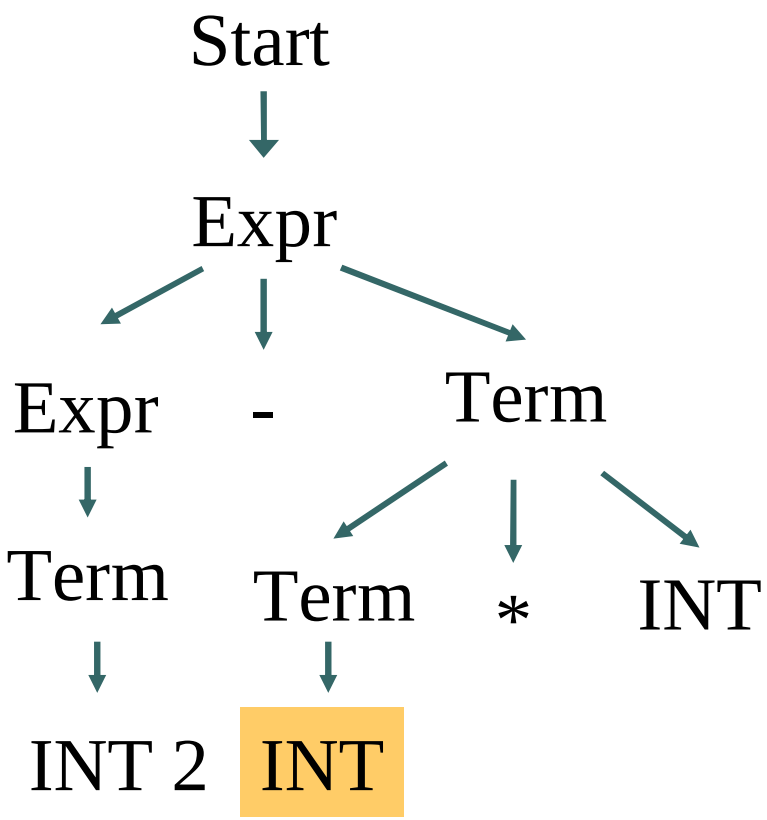
2 – Term*INT

Produção Aplicada

Term → Term * INT

Analisador Sintáctico para a gramática exemplo

Árvore Sintáctica



Entrada que falta

2*2

Forma sentencial

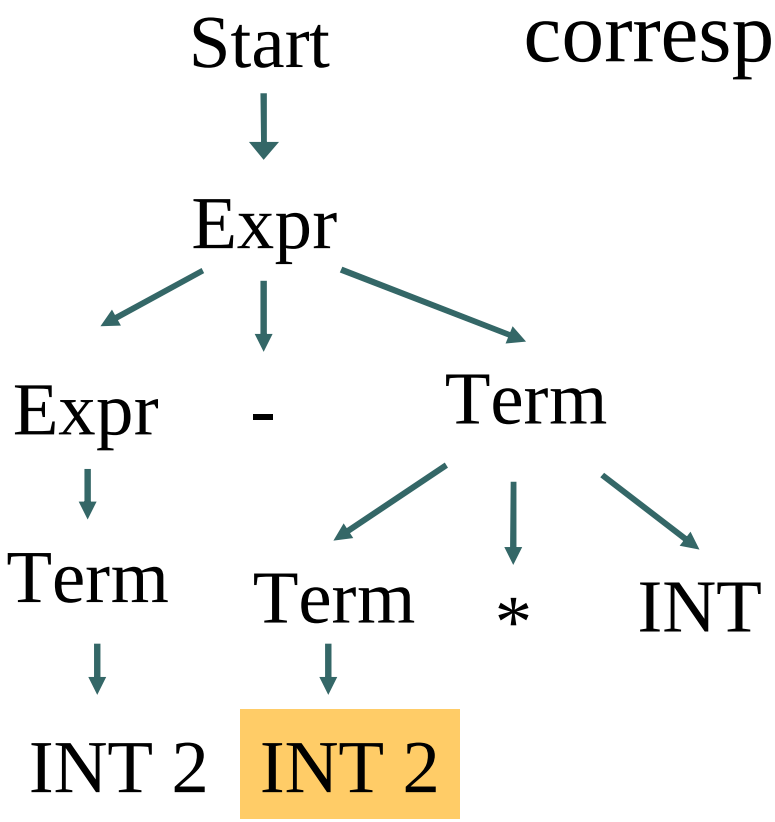
2 – INT*INT

Produção Aplicada

Term → INT

Analisador Sintáctico para a gramática exemplo

Árvore Sintáctica



Token corresponde!

Entrada que falta

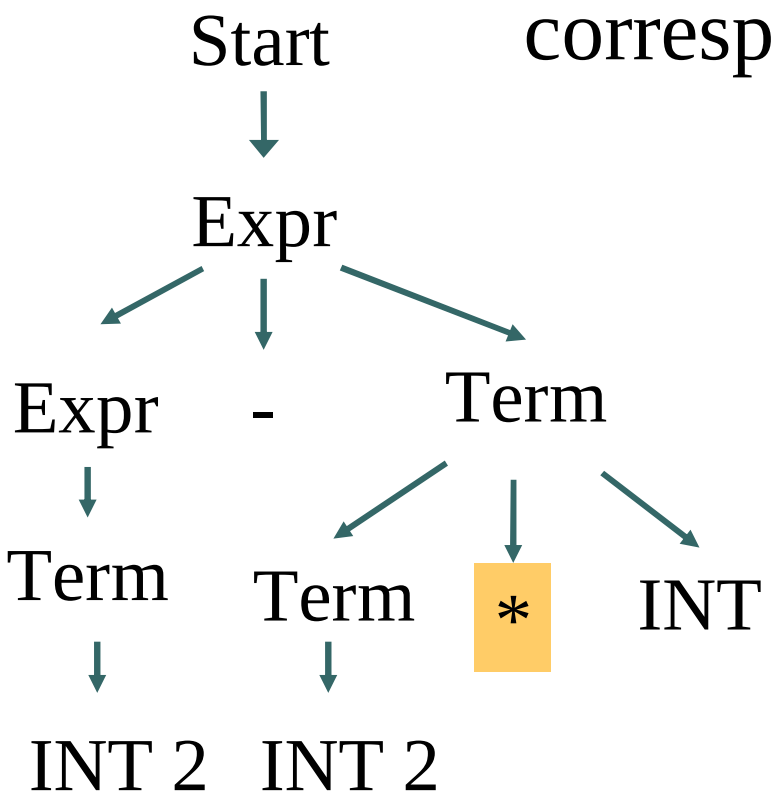
2*2

Forma sentencial

2 – 2*INT

Analisador Sintáctico para a gramática exemplo

Árvore Sintáctica



Token corresponde!

Entrada que falta

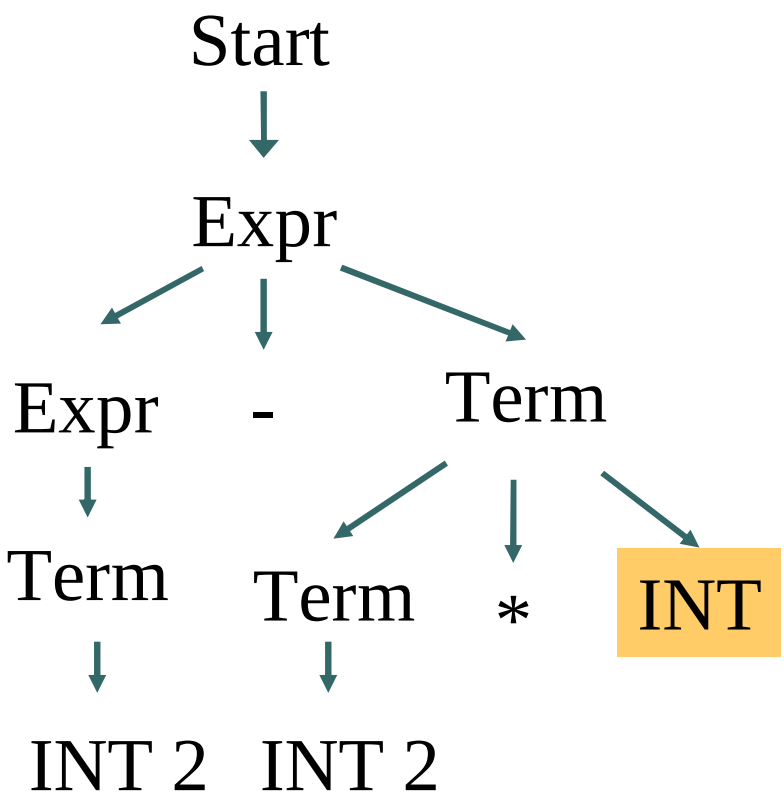
*2

Forma sentencial

2 – 2*INT

Analisador Sintáctico para a gramática exemplo

Árvore Sintáctica



Entrada que falta

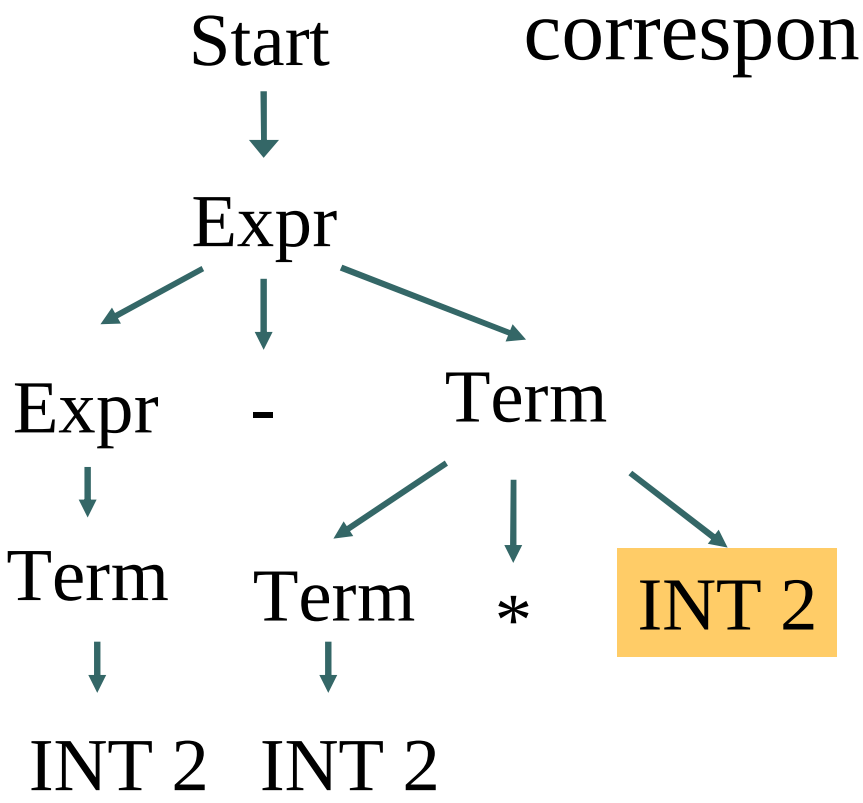
2

Forma sentencial

2 – 2*INT

Analisador Sintáctico para a gramática exemplo

Árvore Sintáctica



Token corresponde!

Entrada que falta

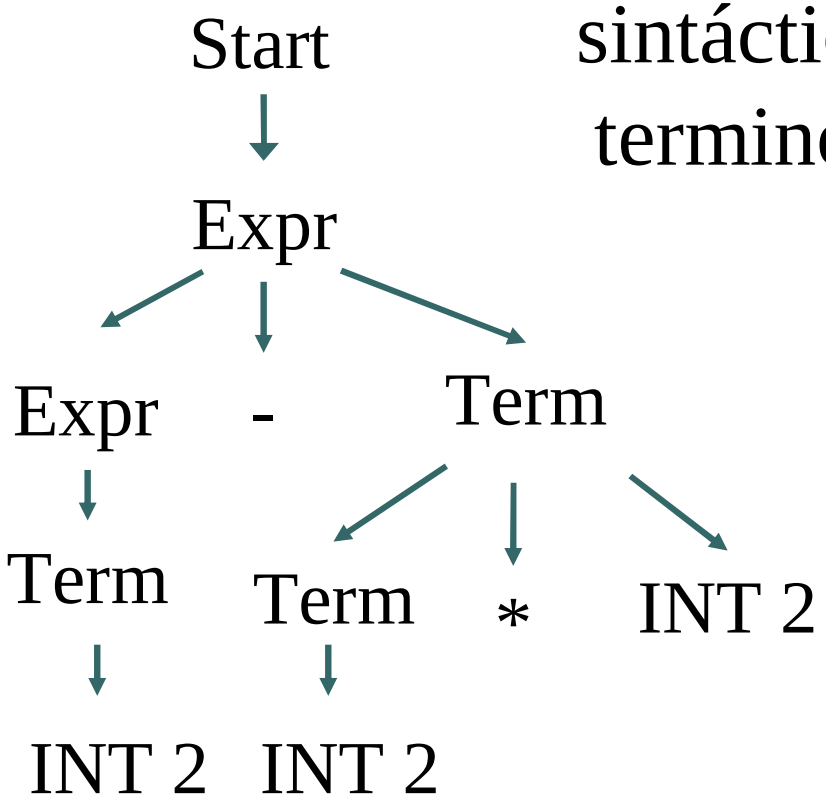
2

Forma sentencial

2 – 2*2

Analisador Sintáctico para a gramática exemplo

Árvore Sintáctica



Análise sintáctica terminou

Entrada que falta

Forma sentencial

2 – 2*2

Sumário

- Três acções (mecanismos)
  - Aplicar produção para expandir o símbolo não-terminal corrente na árvore sintáctica
  - Casar o símbolo terminal corrente
  - Aceitar a análise sintáctica como correcta
- Qual a produção a utilizar por cada símbolo não-terminal?
- Uma abordagem: *Backtracking*
  - Tenta uma alternativa
  - Quando for claro que a alternativa falhou tentar outra alternativa

Analisador Sintáctico Preditivo

- Alternativa ao *backtracking*
- Muito útil para linguagens de programação, que podem ser desenhadas para facilitar a análise
- Ideia básica
  - Ver à frente na sequência de tokens
  - Decisão de qual a produção a aplicar baseada nos tokens seguintes
  - Utilizaremos a profundidade de um token no mecanismo de ver à frente

Gramática Exemplo

Start $\rightarrow$ Expr

Expr $\rightarrow$ Term Expr'

Expr' $\rightarrow$ "+" Term Expr'

Expr' $\rightarrow$ "-" Term Expr'

Expr' $\rightarrow$ ε

Term $\rightarrow$ INT Term'

Term' $\rightarrow$ "*" INT Term'

Term' $\rightarrow$ "/" INT Term'

Term' $\rightarrow$ ε

INT = [0-9]+

- Conjunto de tokens (símbolos terminais):
  { +, -, *, /, INT }

Pontos de Escolha

- Assumir que Term' é a posição corrente na árvore sintáctica
- 3 produções diferentes a aplicar

  Term' $\rightarrow$ "*" INT Term'

  Term' $\rightarrow$ "/" INT Term'

  Term' $\rightarrow \varepsilon$
- Utilizar o próximo token para decidir
  - Se o próximo token for *, aplicar Term' $\rightarrow$ * Int Term'
  - Se o próximo token for /, aplicar Term' $\rightarrow$ / Int Term'
  - De outro modo, aplicar Term' $\rightarrow \varepsilon$

Múltiplas Produções com o mesmo prefixo no RHS

- Gramática Exemplo

  Nt $\rightarrow$ IF THEN

  Nt $\rightarrow$ IF THEN ELSE

- Assumir que Nt é a posição corrente na árvore sintáctica e IF é o próximo token
- Qual a produção a aplicar?

Solução: factorizar a gramática à esquerda

- Nova gramática factoriza o prefixo comum numa única produção

  $Nt \rightarrow$ IF THEN Nt'

  $Nt' \rightarrow$ ELSE

  $Nt' \rightarrow \varepsilon$
- Nenhuma escolha quando o próximo token é um IF
- Todas as alternativas foram unificadas numa única produção

Símbolos não-terminais

- E as produções com símbolos não-terminais?

  $Nt \rightarrow Nt_1\ \alpha_1$

  $Nt \rightarrow Nt_2\ \alpha_2$

- Tem de se escolher com base nos primeiros terminais possíveis que Nt_1 e Nt_2 podem gerar
- E se Nt_1 ou Nt_2 podem gerar ε?
  - Tem de se escolher com base em α_1 e α_2