Fatoração à esquerda.

* Útil para analisadores sintáticos top-down.

* Situação: Temos duas ou mais alternativas de produção para A.

$$A \to \alpha \mid \beta$$

Qual produção escolher?

Podemos reescrever as produções para adiar a escolha até que seja tokens suficiente é lido e isso permite fazer a escolha correta.

Exemplo:

stmt → if expr then (stmt) else stmt |
if expr then (stmt)

Importante: Ao ler token if não sabemos qual produção escolher até a substituição do símbolo não-terminal stmt.

Regra: Se $A \to \alpha\beta_1 \mid \alpha\beta_2$ são duas produções A, e a entrada começa com mesmo prefixo α ($\neq \varepsilon$), então adia a decisão entre as duas produções fazendo:

$$A \to \alpha A'$$
$$A' \to \beta_1 \mid \beta_2$$

Algoritmo para deixar uma gramática fatorada à esquerda

1. para cada não-terminal A
2. encontre o prefixo mais longo α comum a duas ou mais produções A ($A \to \alpha\beta_1 \mid \alpha\beta_2 \mid \ldots \mid \alpha\beta_n \mid \gamma$. γ representa todas as outras produções A sem prefixo α)
3. substitua tais produções por:
$$A \to \alpha A' \mid \gamma$$
$$A' \to \beta_1 \mid \beta_2 \mid \ldots \mid \beta_n$$ (A' é novo não-terminal)
4. volte ao passo 1 até que não existe mais n.-terminal A com produções com prefixo comum.

Exemplo: $S \to$ if E then S | if E then S else S | other

$E \to b$

S possui prefixo comum $\alpha =$ if E then S

$\beta_1 = \varepsilon$ e $\beta_2 =$ else S

$S \to$ if E then S S' | other
$S' \to \varepsilon$ | else S
$E \to b$

gramática fatorada à esquerda.

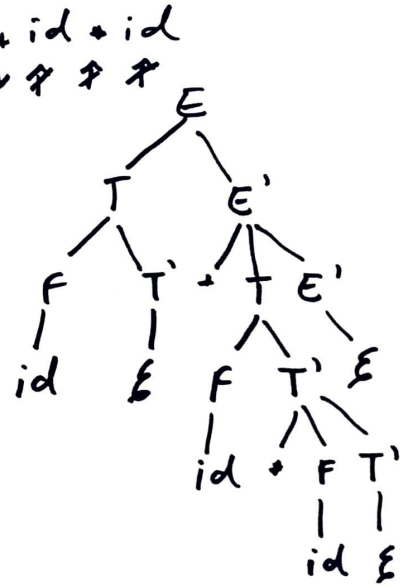
Exercício: Mostre que a gramática acima é ambígua.

Analisador sintático top-down

* Constroem (explícito/ou implícito) uma árvore de derivação para a entrada.
* Essa árvore é construída de cima para baixo em pré-ordem (raiz - esq. - direita).
* Implementa uma derivação mais à esq.

Exemplo: sentença: id + id * id

$E \to TE'$
$E' \to +TE' \mid \varepsilon$
$T \to FT'$
$T' \to *FT' \mid \varepsilon$
$F \to (E) \mid id$



Problema chave:

* Estando numa, e considerando o não-terminal A mais à esq., qual produção A devemos aplicar?

Analisadores sintáticos preditivos são analisadores top-down que escolhem a produção A correta olhando para um número fixo de tokens da entrada. → recursivo-descendente.

* Recursivos ou não.

Definição: Gramáticas LL(K).

Dada uma gramática G, se é possível construir um analisador sintático preditivo que escolha uma produção olhando para k tokens de entrada, então $G \in LL(K)$.

Analisador recursivo descendente

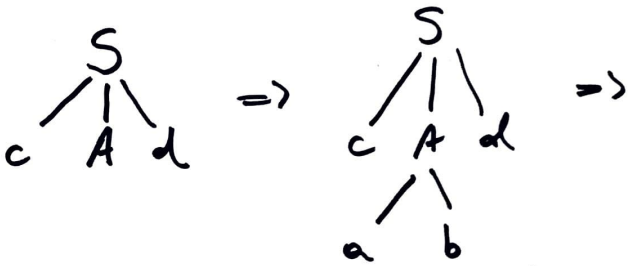
Possui uma função para cada símbolo não-terminal A. Uma função A tenta encontrar uma produção A que consome tokens da entrada. Começando de uma função S (onde S é símbolo inicial da gramática), se a entrada é bem formada, então alguma produção S consome todos os tokens de entrada. Se a entrada é mau-formada, então todas as produções S irão consumir todos os tokens da entrada.

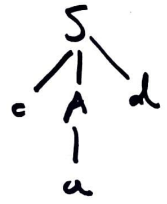
Exemplo:

G: $S \rightarrow cAd$

$A \rightarrow ab \mid a$.

Entrada: c a d





Importante: Uma gramática recursiva à esq. pode fazer um analisador recursivo descendente não parar.

$A \rightarrow A\alpha \mid \gamma$

Conjuntos First e Follows

* São conjuntos de terminais de uma gramática. São utilizados para construir analisadores sintáticos top-down e bottom-up.

Definição: First(α), para α sendo uma cadeia de símbolos de uma gramática, é o conjunto de terminais que iniciam cadeias derivadas por α. Se $\alpha \Rightarrow^{*} \varepsilon$ então $\varepsilon \in$ First(α).

Exemplo:

$E \to TE'$
$E' \to +TE' \mid \varepsilon$
$T \to FT'$
$T' \to *FT' \mid \varepsilon$
$F \to (E) \mid id$

First para todos os símbolos da gramática.

First(+) = {+}, First(*) = {*}, First(( ) = {(},
First( )) = {)}, First(id) = {id}
First(E) = {(, id}, First(T) = {(, id},
First(F) = {(, id}, First(E') = {+, ε},
First(T') = {*, ε}

Regras: ① Se X é um terminal, então First(X) = {X}.

② Se X é não-terminal e $X \to X_1 X_2 \dots X_k$, então First(X_1) $\subseteq$ First(X); e se $\varepsilon \in$ First(X_1) então First(X_2) $\subseteq$ First(X); e ... ; se $\varepsilon \in$ First(X_i) $\forall i = 1, \dots, k$, então $\varepsilon \in$ First(X).

③ First($X_1 X_2 \dots X_n$) contém First(X_1), contém First(X_2) se $\varepsilon \in$ F(X_1)

Definição: Follow(A), para um não-terminal A, é o conjunto de terminais a que aparecem à direita de A em alguma forma sentencial.

Importante: Se A pode ser símbolo mais à direita de alguma forma sentencial, então $ ∈ Follow(A) ($ representa marcador de final de entrada).

Exemplo: Follows dos não-terminais.

Follow(E) = {$,)}
Follow(E') = {$,)}
Follow(T) = {+, $,)}
Follow(T') = {+, $,)}
Follow(F) = {*, +, $,)}

Regras: Para cada não terminal B,

① Follow(B) contém $ se B é símbolo inicial da gramática

② Se $A \to \alpha B \beta$, então First(β) $\subseteq$ Follow(B) exceto por ε.

③ Se $A \to \alpha B$ é produção ou $A \to \alpha B \beta$ com $\varepsilon \in$ First(β), então Follow(A) $\subseteq$ Follow(B).