# Análise LR(K)

- O método de análise **LR(K)** é mais geral e aproveita melhor as informações obtidas a partir dos símbolos já processados.

**<u>Idéia básica do método</u>:** ao invés de deslocar para a pilha símbolos da linguagen (terminais e não-terminais) serão empilhados outros símbolos, chamados *estados*.

A cada passo do algoritmo, o último estado empilhado será utilizado para tomar decisões sobre um eventual deslocamento ou redução. Além disso, o algoritmo poderá consultar um número k preestabelecido ($K \geq 0$) de símbolos seguintes da cadeia de entrada para tomar essas decisões. Esse tipo de algoritmo será denominado **analisador LR(K)**, indicando uma análise da esquerda para a direita, resultando a derivação direta, e consultando k símbolos seguintes de entrada.Discutiremos apenas os casos k=0 e k=1.

# Análise LR(K)

A tabela de análise é uma matriz retangular cujas linhas são indicadas pelos estados, e as colunas pelos símbolos do vocabulário da gramática.

Os elementos da matriz indicam as ações a serem tomadas pelo algoritmo e que podem ser (KOWALTOWSKI, 83):

1. Empilhar um estado $e_i$
2. Reduzir usando uma produção $A \rightarrow \alpha$
3. Aceitar
4. Rejeitar

A diferença entre LR(0) e LR(1) está na maneira de construir a tabela.

# marca o fim da cadeia

$e_i$ indicam as ações (empilhar o estado $e_i$)

$r_i$ (reduzir usando a i-ésima produção) *a* aceitar

# Análise LR(K)

Um *item* é uma produção na qual foi marcada uma posição na cadeia do lado direito, essa posição será indicada por meio do símbolo.

Itens são usados para construir estados.

A presença no topo da pilha de um estado contendo um item da forma $A \rightarrow \alpha\beta$ indica que já foi processado e deslocado para a pilha a parte incial $\alpha$ do retudendo $\alpha\beta$.

O símbolo $\bullet$ no fim de um item, isto é, $\beta=\lambda\bullet$, indica um redutendo completo. Dizemos, que o item é *completo*.

# Análise LR(K)

- Se um estado contém um item da forma $A \rightarrow \alpha \bullet B\beta$ com $A, B \in N$ e $\alpha, \beta \in V^*$. A presença desse estado no topo da pilha indicará que já foi empilhada a cadeia $\alpha$, sendo esperada a seguir a cadeia $B\beta$. Isto significa que a cadeia de entrada corrente (incluindo o eventual resultados da redução anterior) poderá ser da forma $\gamma w$, com $\gamma \in V^*$, $w \in T^*$ e $B\beta \Rightarrow^* \gamma$; desta maneira as reduções subseqüententes poderão transformar $\gamma$ em $B\beta$. Conseqüentemente, a presença do estado contendo $A \rightarrow \alpha \bullet B\beta$ no topo da pilha implica que o algoritmo deverá aceitar a seguir qualquer derivável de B, ou em outras palavras, esse estado deverá conter também todos os itens da forma $B ::= \bullet \xi$.

# Análise LR(K)

Diremos que um conjunto *K* de itens é *fechado* se para todo item de *K* da forma A::= α•Bβ todos os itens da forma B::=•ξ estão em *K*. Denoteremos por fecho (*K*) o menor conjunto fechado que contém *K*. Dado um conjunto *K*, fecho(*K*) pode ser calculado de uma maneira muito simples:

1. Adota-se o conjunto dado *K* como valor inicial de *H*.
2. Se existe um item A::=α•Bβ de *H* e uma produção B::=ξ tal que B::=•ξ não está em *H*, acrescenta-se B::= •ξ a *H.*
3. O passo 2 é repetido até que não se possam acrescentar mais itens a *H.*
4. *H* = fecho(*K*).

# Análise LR(K)

*Exemplo* (pag.43 do KOWALTOWSKI (83))

Consideremos os seguintes conjuntos *K*'s de itens de *G*:

1. E::=+EE     3. E::=a
2. E::=*EE     4. E::=b

$K_1$={E::=+•EE}

$K_2$={E::=+E•E / *•EE / •a}

$K_3$={E::=•b}

Então os seus fechos são

fecho($K_1$)={E::=+•EE/•+EE/•*EE/•a/•b}

fecho($K_2$)={E::=+E•E/*•EE/•a/ •+EE/•*EE/ •b}

fecho($K_3$)={E::=•b}

Esta discussão implica que um estado deve ser um conjunto fechado de itens. Cada item desse estado representa uma das possibilidades para continuar a análise.

# Análise LR(K)

Examinaremos agora o problema de determinar as da forma $e_j$ da tabela de análise. Suponhamos então que um estado $e_i$ contenha um item incompleto da forma $A::=\alpha\bullet X\beta$, $X\in V$. A presença desse estado $e_i$ no topo da pilha indica que se o próximo símbolo a ser consultado for X, então terá sido processada a parte $\alpha X$ do redutendo, devendo ser empilhado, portanto, um estado que contém o item $A::=\alpha X\bullet\beta$. Definiremos então a função *transfere(K,X)* como sendo o fecho do conjunto de todos os itens da forma $A::=\alpha X\bullet\beta$ tais que o item $A::= \alpha\bullet X\beta$ está em K.

# Análise LR(K)

*Exemplo* (pag.44 do KOWALTOWSKI (83)): Considere *G*:

1. E::=+EE    2. E::=*EE    3. E::=a    4. E::=b

e os conjuntos:

$K_1$= {E::=*•EE}

$k_2$ = fecho($K_1$)={*•EE/•+EE/•*EE/ •a/•b}

$k_3$ = {E::=•b}

Tem-se, então:

transfere($K_1$,*)=Ø

transfere($K_1$,E)={E::=*E•E/•+EE/•*EE/ •a/•b}

transfere($K_2$,+)={E::=+•EE/•+EE/•*EE/ •a/•b}

transfere($K_2$,a)={E::=a•}

transfere($K_3$,E)= Ø

transfere($K_3$,b)={E::=b•}

# Análise LR(K)

As funções *fecho* e *transfere* permitem a construção dos estados que serão identificados com conjunto de itens. A construção partirá de um estado inicial, e aplicando as funções *fecho* e *transfere* calculará novos estados, até que não seja possível prosseguir na construção.

A fim de introduzir o estado inicial $e_0$ convencionaremos que a toda gramática de raiz S acrescenta-se a produção S'::=S#, onde o símbolo # não está em T; o símbolo S' passa a ser a nova raiz. Convencionaremos, também, que toda cadeia de entrada será seguida do símbolo #. Fica claro, dessa maneira que o estado $e_0$ deve conter o item S'::=•S#. Por outro lado, se um estado que contém S'::=S•# estiver no topo da pilha quando o símbolo seguinte a ser consultado é #, então a cadeia de entrada ser aceita.

# Análise LR(K)

Para determinar os estados de uma gramática (*C* é uma coleção de estasos, ou seja, um conjunto de conjuntos de itens):

1. Adota-se o estado $e_0$={fecho({S'::=•S#})} como o valor inicial da coleção *C*.

2. Se existe um estado *e* de C, e um símbolo $X \in V$ tais que *e*'=transfere(*e*,*X*) não é vazio, e *e*' não está em *C*, então acrescenta-se *e*' à coleção C.

3. O passo 2 é repetido até que não se possam acrescentar mais estados à coleção *C*.

4. *C* é a coleção de estados tipo LR(0) da gramática.

# Análise LR(K)

Consideremos a gramática:

0.E'::=E# 1.E::=+EE 2.E::=*EE 3.E::=a 4.E::=b

Aplicando-se a construção acima, obtêm-se os seguintes estados (omitimos as chaves que indicam conjuntos):

$e_0$: E'::=•E# E::=•+EE/•*EE/ •a/•b **Item incompleto**

$e_1$: E'::=E•# **Item incompleto**

$e_2$: E::=+•EE/•+EE/•*EE/•a/•b **Item incompleto**

$e_3$: E::=*•EE/•+EE/•*EE/•a/•b **Item incompleto**

$e_4$: E::=a• **Item completo**

$e_5$: E::=b• **Item completo**

$e_6$: E:: +E•E/•+EE/•*EE/•a/•b **Item incompleto**

$e_7$: E:: *E•E/•+EE/•*EE/•a/•b **Item incompleto**

$e_8$: E:: +EE• **Item completo**

$e_9$: E:: *EE• **Item completo**

Como existem estados que contém tanto itens completos como incompletos a gramática **não é LR(0)**.

# Análise LR(K)

Os estados constituídos de itens completos indicam que os últimos estados empilhados correspondem a um redutendo, e que portanto deve haver redução, independentemente do próximo símbolo.

É essa propriedade que faz com que a gramática dada pertença à classe LR(0).

Se o estado que está no topo da pilha contém um único item da forma $A::=\alpha\bullet$, então deve-se aplicar uma redução utilizando a produção $A::=\alpha$. As entradas $r_j$ da tabela a seguir foram obtidas dessa maneira (KOWALTOWSKI, 83).

# Análise LR(K)

O símbolo E  pode ser seguido pelos símbolos 'a', 'b', '+', '*' ou '#'

$\Delta(E)=\{a, b, *, +, \#\}$