Análise LALR

Construção de compiladores I

Objetivos

Objetivos

- Apresentar o algoritmo de análise sintática LALR.

Introdução

Introdução

- Nas aulas anteriores, vimos o algorithm LR(1).
- Este algoritmo consegue realizar o parsing de construções de linguagens de programação.

Introdução

- Problema: o algoritmo LR(1) gera um número muito grande de estados.
- **Solução**: combinar conjuntos de itens que diferem apenas pelos **lookaheads**.
  - Isso é o chamado analisador LALR.

Introdução

- Para combinar conjuntos de itens, é útil considerar o conceito de **núcleo**.
- O **núcleo** de um conjunto de itens é um subconjunto de itens utilizado para criar o conjunto.

Introdução

- Após criar os conjuntos LR(1), combinamos os conjuntos com o mesmo núcleo em um único.

Construção dos itens LALR

Construção dos itens LALR

- Fechamento de conjunto de itens I.
  - $I \subseteq closure(I)$.
  - Para cada item $[A \to \alpha.B\beta, a]$ em I
    - Para cada regra $B \to \gamma$ em G'
      - Para cada $b \in first(\beta a)$
      - Adicione $[B \to .\gamma, b]$ em I
- Repita enquanto houver alterações em I.

Construção dos itens LALR

- Função de $goto(I, X)$
  - Inicialize J como $\emptyset$.
  - Para cada item $[A \to \alpha.X\beta, a]$ em I
    - Adicione o item $[A \to \alpha X.\beta, a]$ ao conjunto J.
  - retorne $closure(J)$

Construção dos itens LALR

- Função de construção de itens G'
  - inicializa C como closure({[S $\to$ .S, \$]})
    - Para cada conjunto $I \in C$
      - Para cada símbolo X de G'
      - se $goto(I, X) \neq \emptyset \wedge goto(I, X) \notin C$
      - Adicione $goto(I, X)$ em C
  - repetir enquanto houver alterações em C.

Construção da tabela LALR

Construção da tabela LALR

- Se $[A \to \alpha.a\beta, b] \in I_i$ e $goto(I_i, a) = I_j$,
  - A[i,a] = shift j.

Construção da tabela LALR

- Se $[A \to \alpha., a] \in I_i$ e $A \neq S'$
  - A[i,a] = reduce A $\to \alpha$

Construção da tabela LALR

- Se [S' $\to$ S., \$] $\in$ I_i
  - A[i,\$] = accept

Construção da tabela LALR

- Seja J = $I_1 \cup \ldots \cup I_n$.
  - Núcleo de cada I_i é o mesmo.
- Seja K a união de todos os itens de goto($I_{1,X}$).
  - Fazemos G[J,X] = K

Exemplo

Exemplo

- Construção da tabela LALR para a gramática

$$\begin{array}{lcl} S & \to & (L) \mid x \\ L & \to & L, S \mid S \end{array}$$

Construção eficiente da tabela

Construção eficiente da tabela

- O algoritmo LALR melhora o consumo de memória.
  - Combinar o número de itens.
- Porém, ainda precisamos computar o conjunto completo de itens.

Construção eficiente da tabela

- Ao invés de construir os itens LR(1), podemos construir apenas os núcleos de itens LR(0) e calcular os lookaheads.
- A partir do núcleo dos itens LR(1), calculamos a tabela.

Construção eficiente da tabela

- Determinando lookaheads para um núcleo K e um símbolo X.
- Repita os passos seguintes para cada item
  - A $\rightarrow \alpha$. $\beta \in$ K.

Construção eficiente da tabela

- $J \leftarrow closure(\{[A \rightarrow \alpha.\beta, \#]\})$
- Se $[B \rightarrow \gamma.X\delta, a] \in J \wedge a \neq \#$
  - a é gerado espontaneamente para goto(I,X).
- Se $[B \rightarrow \gamma.X\delta, \#] \in J$
  - Propague lookaheads de $A \rightarrow \alpha.\beta \in I$ para $B \rightarrow \gamma X.\delta$ em goto(I,X).

Exemplo

Exemplo

- Construção eficiente da tabela LALR para a gramática

$$\begin{array}{lcl} S & \rightarrow & (L) \mid x \\ L & \rightarrow & L, S \mid S \end{array}$$

Concluindo

Concluindo

- Nesta aula apresentamos dois algoritmos para a construção de tabelas LALR.
- Próxima aula: Geradores de analisadores LALR.

Exercícios

Exercícios

- Determine se a seguinte gramática possui conflitos, utilizando o algoritmo de construção de tabelas LALR.

$$\begin{array}{lcl} E & \rightarrow & T{+}E \mid T \\ T & \rightarrow & \mathbf{x} \end{array}$$