Análise léxica

O papel do analisador léxico

Prof. Edson Alves

Faculdade UnB Gama

Sumário

Fragmento de gramática que será utilizada nos exemplos

$$
\begin{array}{lcl}
cmd & \rightarrow & \textbf{if } expr \textbf{ then } cmd \\
 & | & \textbf{if } expr \textbf{ then } cmd \textbf{ else } cmd \\
 & | & \epsilon \\
 & & \\
expr & \rightarrow & termo \textbf{ relop } termo \\
 & | & termo \\
 & & \\
termo & \rightarrow & \textbf{id} \\
 & | & \textbf{num}
\end{array}
$$

Definições regulares dos tokens

$$
\begin{aligned}
\textbf{if} &\rightarrow \texttt{if} \\
\textbf{then} &\rightarrow \texttt{then} \\
\textbf{else} &\rightarrow \texttt{else} \\
\textbf{relop} &\rightarrow \texttt{<} \mid \texttt{<=} \mid \texttt{=} \mid \texttt{<>} \mid \texttt{>} \mid \texttt{>=} \\
\textbf{id} &\rightarrow \textbf{letra}\ (\textbf{letra} \mid \textbf{dígito})^{*} \\
\textbf{num} &\rightarrow \textbf{dígito}^{*}(\texttt{.}\ \textbf{dígito}^{+})?(\texttt{E}(\texttt{+}\mid\texttt{-})?\ \textbf{dígito}^{+})? \\
\textbf{letra} &\rightarrow \texttt{A} \mid \texttt{B} \mid \ldots \mid \texttt{Z} \mid \texttt{a} \mid \texttt{b} \mid \ldots \mid \texttt{z} \\
\textbf{dígito} &\rightarrow \texttt{0} \mid \texttt{1} \mid \texttt{2} \mid \ldots \mid \texttt{9}
\end{aligned}
$$

Tratamento de espaços em branco

- Assuma que os lexemas sejam separados por espaços em brancos

Tratamento de espaços em branco

- Assuma que os lexemas sejam separados por espaços em brancos
- São considerados espaços em branco: sequências de espaços em branco, tabulações e quebras de linha

Tratamento de espaços em branco

- Assuma que os lexemas sejam separados por espaços em brancos
- São considerados espaços em branco: sequências de espaços em branco, tabulações e quebras de linha
- O analisador léxico deve ignorar os espaços em branco

Tratamento de espaços em branco

- Assuma que os lexemas sejam separados por espaços em brancos
- São considerados espaços em branco: sequências de espaços em branco, tabulações e quebras de linha
- O analisador léxico deve ignorar os espaços em branco
- A definição regular **ws** identifica os espaços em branco:

$$\begin{aligned} \mathbf{delim} &\rightarrow \mathbf{branco} \mid \mathbf{tabulação} \mid \mathbf{quebradelinha} \\ \mathbf{ws} &\rightarrow \mathbf{delim}^{+} \end{aligned}$$

Tratamento de espaços em branco

- Assuma que os lexemas sejam separados por espaços em brancos
- São considerados espaços em branco: sequências de espaços em branco, tabulações e quebras de linha
- O analisador léxico deve ignorar os espaços em branco
- A definição regular **ws** identifica os espaços em branco:

$$\begin{aligned} \mathbf{delim} &\rightarrow \mathbf{branco} \mid \mathbf{tabulação} \mid \mathbf{quebradelinha} \\ \mathbf{ws} &\rightarrow \mathbf{delim}^{+} \end{aligned}$$

- Se o analisador léxico identificar o padrão **ws**, ele não irá gerar um token

Especificação dos tokens

Expressão regular	Token	Valor do atributo
ws	-	-
if	if	-
then	then	-
else	else	-
id	id	Lexema
num	num	Valor numérico do lexema
<	relop	LT
<=	relop	LE
=	relop	EQ
<>	relop	NE
>	relop	GT
>=	relop	GE

Diagramas de transição

- Um diagrama de transição é um fluxograma estilizado que delineia as ações a serem tomadas pelo analisador léxico a cada requisição de novo token por parte do parser

Diagramas de transição

- Um diagrama de transição é um fluxograma estilizado que delineia as ações a serem tomadas pelo analisador léxico a cada requisição de novo token por parte do parser
- Os estados são representados por círculos rotulados e identificam posições do diagrama

Diagramas de transição

- Um diagrama de transição é um fluxograma estilizado que delineia as ações a serem tomadas pelo analisador léxico a cada requisição de novo token por parte do parser
- Os estados são representados por círculos rotulados e identificam posições do diagrama
- As transições são representadas por arestas direcionadas, rotuladas por um caractere

Diagramas de transição

- Um diagrama de transição é um fluxograma estilizado que delineia as ações a serem tomadas pelo analisador léxico a cada requisição de novo token por parte do parser
- Os estados são representados por círculos rotulados e identificam posições do diagrama
- As transições são representadas por arestas direcionadas, rotuladas por um caractere
- Uma transição do estado `X` para o estado `Y` cujo rótulo é o caractere c indica que, se a execução está no estado `X` e o próximo caractere lido é c, então a execução deve consumir c e seguir para o estado `Y`

Diagramas de transição

- Um diagrama de transição é um fluxograma estilizado que delineia as ações a serem tomadas pelo analisador léxico a cada requisição de novo token por parte do parser
- Os estados são representados por círculos rotulados e identificam posições do diagrama
- As transições são representadas por arestas direcionadas, rotuladas por um caractere
- Uma transição do estado `X` para o estado `Y` cujo rótulo é o caractere c indica que, se a execução está no estado `X` e o próximo caractere lido é c, então a execução deve consumir c e seguir para o estado `Y`
- Um diagrama de transição é determinístico se todas as transições que partem de um estado são rotuladas por caracteres distintos

Estados e ações

- Um estado deve ser rotulado como estado de partida, o qual marca o início da execução

Estados e ações

- Um estado deve ser rotulado como estado de partida, o qual marca o início da execução
- Se os rótulos dos estados são numéricos, a convenção é que o estado inicial seja o de número zero (ou um)

Estados e ações

- Um estado deve ser rotulado como estado de partida, o qual marca o início da execução
- Se os rótulos dos estados são numéricos, a convenção é que o estado inicial seja o de número zero (ou um)
- Alguns estados podem ter ações associadas, as quais são executadas quando a execução atinge tal estado

Estados e ações

- Um estado deve ser rotulado como estado de partida, o qual marca o início da execução
- Se os rótulos dos estados são numéricos, a convenção é que o estado inicial seja o de número zero (ou um)
- Alguns estados podem ter ações associadas, as quais são executadas quando a execução atinge tal estado
- Executada a ação, se existir, deve ser lido o próximo caractere c da entrada: se existir uma transição rotulada por c, a execução segue para o novo estado, indicado pela aresta; caso contrário, deve ser sinalizado um erro

Estados e ações

- Um estado deve ser rotulado como estado de partida, o qual marca o início da execução
- Se os rótulos dos estados são numéricos, a convenção é que o estado inicial seja o de número zero (ou um)
- Alguns estados podem ter ações associadas, as quais são executadas quando a execução atinge tal estado
- Executada a ação, se existir, deve ser lido o próximo caractere c da entrada: se existir uma transição rotulada por c, a execução segue para o novo estado, indicado pela aresta; caso contrário, deve ser sinalizado um erro
- Os estados de aceitação, que indicam que um token foi reconhecido, são marcados com um círculo duplo

Retrações e erros léxicos

- Um estado de aceitação que demande o retorno do último caractere lido para o buffer de entrada é marcado um símbolo $*$

Retrações e erros léxicos

- Um estado de aceitação que demande o retorno do último caractere lido para o buffer de entrada é marcado um símbolo $*$
- Isto ocorre, por exemplo, em casos em que um token é finalizado por um espaço ou por um caractere que inicia um novo token

Retrações e erros léxicos

- Um estado de aceitação que demande o retorno do último caractere lido para o buffer de entrada é marcado um símbolo $*$
- Isto ocorre, por exemplo, em casos em que um token é finalizado por um espaço ou por um caractere que inicia um novo token
- Um analisador léxico pode ter vários diagramas de transição

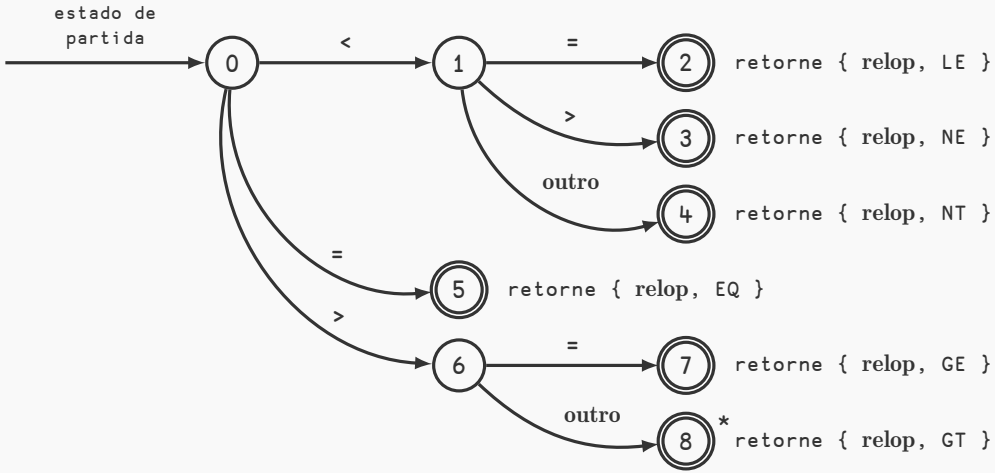
Retrações e erros léxicos

- Um estado de aceitação que demande o retorno do último caractere lido para o buffer de entrada é marcado um símbolo *
- Isto ocorre, por exemplo, em casos em que um token é finalizado por um espaço ou por um caractere que inicia um novo token
- Um analisador léxico pode ter vários diagramas de transição
- Se acontecer um erro no fluxo de execução de um diagrama, o ponteiro de leitura deve ser reposicionado ao ponto que estava no estado de partida e um novo diagrama deve ser seguido

Retrações e erros léxicos

- Um estado de aceitação que demande o retorno do último caractere lido para o buffer de entrada é marcado um símbolo *
- Isto ocorre, por exemplo, em casos em que um token é finalizado por um espaço ou por um caractere que inicia um novo token
- Um analisador léxico pode ter vários diagramas de transição
- Se acontecer um erro no fluxo de execução de um diagrama, o ponteiro de leitura deve ser reposicionado ao ponto que estava no estado de partida e um novo diagrama deve ser seguido
- Se ocorrem erros em todos os diagramas, então há um erro léxico no programa fonte

Diagrama de transição para operadores relacionais



Identificadores e palavras-chave

- Não é prático identificar as diferentes palavras-chave da linguagem por meio de diagramas de transição

Identificadores e palavras-chave

- Não é prático identificar as diferentes palavras-chave da linguagem por meio de diagramas de transição
- Na maioria das linguagens, as palavras-chave obedecem à mesma regra de construção dos identificadores

Identificadores e palavras-chave

- Não é prático identificar as diferentes palavras-chave da linguagem por meio de diagramas de transição
- Na maioria das linguagens, as palavras-chave obedecem à mesma regra de construção dos identificadores
- Uma abordagem mais geral e efetiva é construir o diagrama de transição dos identificadores e usá-los para reconhecer tanto os identificadores quanto as palavras-chave

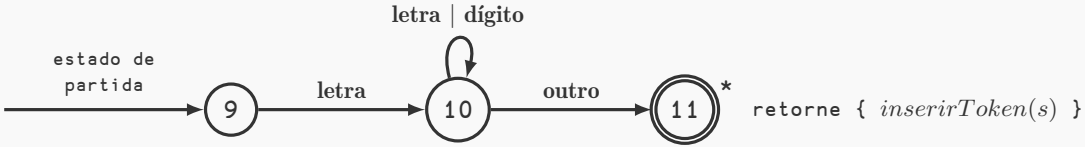
Identificadores e palavras-chave

- Não é prático identificar as diferentes palavras-chave da linguagem por meio de diagramas de transição
- Na maioria das linguagens, as palavras-chave obedecem à mesma regra de construção dos identificadores
- Uma abordagem mais geral e efetiva é construir o diagrama de transição dos identificadores e usá-los para reconhecer tanto os identificadores quanto as palavras-chave
- Para isto, os lexemas de todas as palavras-chave devem ser inseridos na tabela de símbolos, com seus respectivos tokens e atributos

Identificadores e palavras-chave

- Não é prático identificar as diferentes palavras-chave da linguagem por meio de diagramas de transição
- Na maioria das linguagens, as palavras-chave obedecem à mesma regra de construção dos identificadores
- Uma abordagem mais geral e efetiva é construir o diagrama de transição dos identificadores e usá-los para reconhecer tanto os identificadores quanto as palavras-chave
- Para isto, os lexemas de todas as palavras-chave devem ser inseridos na tabela de símbolos, com seus respectivos tokens e atributos
- A função $inserirToken(s)$ insere o lexema s na tabela de símbolos como um token **id**, caso s não esteja presente na tabela; caso contrário, a função retorna o token e os atributos associados a s na tabela

Diagrama de transição para identificadores e palavras-chave



Identificação de constantes numéricas

- A identificação de tokens deve ser gulosa

Identificação de constantes numéricas

- A identificação de tokens deve ser gulosa
- Por exemplo, se a entrada consiste em `12.3E4`, o analisador léxico não deve retornar a constante inteira `12` e nem mesmo a constante em ponto flutuante `12.3`: ele deve retornar a constante `12.3E4`

Identificação de constantes numéricas

- A identificação de tokens deve ser gulosa
- Por exemplo, se a entrada consiste em `12.3E4`, o analisador léxico não deve retornar a constante inteira `12` e nem mesmo a constante em ponto flutuante `12.3`: ele deve retornar a constante `12.3E4`
- Assim, o token deve ser o maior lexema aceito por um diagrama de transição

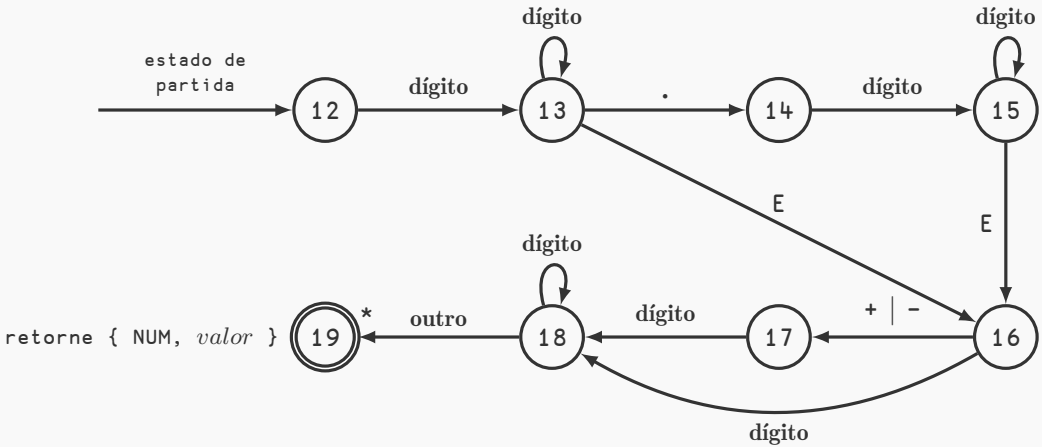
Identificação de constantes numéricas

- A identificação de tokens deve ser gulosa
- Por exemplo, se a entrada consiste em `12.3E4`, o analisador léxico não deve retornar a constante inteira `12` e nem mesmo a constante em ponto flutuante `12.3`: ele deve retornar a constante `12.3E4`
- Assim, o token deve ser o maior lexema aceito por um diagrama de transição
- Uma forma de implementar a abordagem gulosa é tratar os casos mais longos antes dos mais curtos

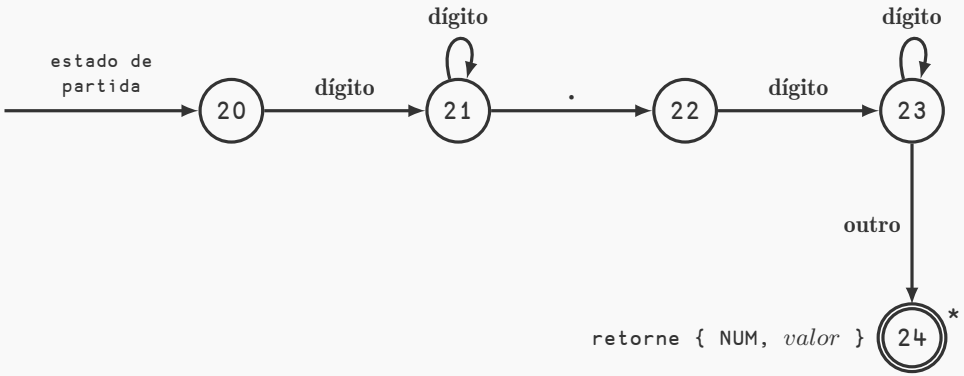
Identificação de constantes numéricas

- A identificação de tokens deve ser gulosa
- Por exemplo, se a entrada consiste em `12.3E4`, o analisador léxico não deve retornar a constante inteira `12` e nem mesmo a constante em ponto flutuante `12.3`: ele deve retornar a constante `12.3E4`
- Assim, o token deve ser o maior lexema aceito por um diagrama de transição
- Uma forma de implementar a abordagem gulosa é tratar os casos mais longos antes dos mais curtos
- Isto pode ser feito assumindo a convenção de que os estados de partida com menores rótulos devem ser testados antes dos estados com maiores rótulos e escrevendo os diagramas apropriadamente

Diagramas de transição para constantes numéricas



Diagramas de transição para constantes numéricas



Diagramas de transição para constantes numéricas