ciência da computação

V EPAC - Encontro Paranaense de Computação ISSN:1981-8653
18 a 20 de setembro de 2013
Unioeste – Cascavel – Paraná

EXTLex: UM ANALISADOR LÉXICO CONFIGURÁVEL

Gustavo Henrique Paetzold[1]; Elder Schemberger[2]
[1,2] Universidade Estadual do Oeste do Paraná – Unioeste – Cascavel – Paraná
[1]**ghpaetzold@outlook.com,** [2]**elderes@gmail.com**

Resumo
Analisadores léxicos compõem a primeira fase de compiladores. No desenvolvimento de um compilador, a funcionalidade do analisador léxico, assim como a funcionalidade do restante dos componentes, comumente se limita apenas à linguagem de programação para qual o compilador foi desenvolvido. Com o objetivo de promover reusabilidade no âmbito de desenvolvimento de compiladores, este artigo apresenta um analisador léxico configurável. Por meio de dois arquivos de configuração, o analisador léxico desenvolvido para este trabalho realiza a análise lexical de múltiplas linguagens distintas, promovendo reusabilidade no âmbito de desenvolvimento de compiladores.
Palavras-chave: Analisador Léxico, Compiladores, Linguagens de Programação

1. Introdução
Sempre que uma nova linguagem de programação é proposta, se faz necessário a construção de um compilador/interpretador capaz de traduzir códigos representados em seus formalismos em códigos compreensíveis por máquinas. (ALFRED et al., 1995) define a compilação como um processo composto por duas etapas: a **análise**, que divide o programa fonte nas partes constituintes e cria uma representação intermediária do mesmo, e a **síntese**, que constrói o programa alvo desejado, a partir da representação intermediária do mesmo.

O analisador léxico é um dos componentes que constituem um sistema de compilação (TOSCANI e ALENCAR, 2008). A função do analisador léxico em um compilador, assim como descrito por (ALFRED et al., 1995), é "ler os caracteres do código e produzir uma sequência de *tokens* para ser posteriormente utilizada na análise sintática". Comumente, o analisador léxico deve ser desenvolvido individualmente para cada nova linguagem de programação. Para diminuir os custos no desenvolvimento de compiladores, este artigo apresenta o EXTLex, um analisador léxico que, por meio de ajustes em seus arquivos de configuração, é capaz de fazer a análise lexical de diferentes linguagens de programação.

2. Especificação e Modelagem do EXTLex
A função do analisador léxico configurável EXTLex é, após configurado com base na estrutura lexical de certa linguagem de programação, detectar erros em sua estrutura, e determinar quais os *tokens* que o compõe. Diferente dos trabalhos de (JOHNSON, 1975) e (LESK e SCHMIDT, 1990), o EXTLex não é um gerador de analisadores léxicos, mas sim um analisador léxico extensível que atua com base em arquivos externos de configuração.

O EXTLex é um aplicativo em Java acompanhado de dois arquivos de configuração que devem ser confeccionados individualmente de acordo com a estrutura da linguagem de programação a qual se deseja compilar. São eles:

- **Automato.EXT:** Arquivo de texto contendo o autômato finito referente à estruturação dos *tokens* da linguagem de programação. A Figura 1 ilustra um autômato que reconhece comentários em código envoltos em caracteres "#".

Figura 1 – Autômato finito de reconhecimento de comentários

- **Palavras_Reservadas.EXT:** Arquivo de texto contendo todas as palavras reservadas da linguagem de programação sendo compilada.

2.1. Estrutura do Arquivo Automato.EXT

Conforme supracitado, o EXTLex reconhece *tokens* e detecta erros por meio de um diagrama de estados, que deve ser descrito pelo usuário no arquivo Automato.EXT. Este arquivo deve seguir o formato ilustrado no exemplo da Tabela 1.

Descrição	Entradas do arquivo
Nome do estado inicial do autômato.	01: ESTADO_1
Token de retorno do ESTADO_1.	02: <token_resultante>
Valor de retorno do ESTADO_1.	03: ME
Primeira transição de estado: caractere em ASCII e estado de destino.	04: N1 ESTADO_2
Linha vazia indicando final da descrição do estado.	06:

Tabela 1 – Descrição dos componentes do formato do arquivo Automato.EXT do EXTLex

O formato do arquivo de descrição de autômato finito permite que sejam representados tanto estados finais, que retornam um *token* e seu valor correspondente, como também estados de erro. A Tabela 2 apresenta um exemplo de autômato de texto equivalente ao autômato ilustrado na Figura 1, que possui tanto estados finais, quanto estados de erro.

Entradas do arquivo Automato.EXT
0 <ERROR_VAZIO> Comentário vazio. 35 10 10 <ERROR_FECHAMENTO> Ausência do caractere "#" de fechamento do comentário. 32~126 10 35 11 11 <comment> LEXEMA

Tabela 2 – Exemplo de autômato com diferentes tipos de estados.

No autômato da Tabela 2, é possível observar três estados:

1. **Estado 0:** Estado inicial da construção de um bloco de comentário.
   - **Token de retorno:** <ERROR_VAZIO> (indicador de erro de comentário vazio).
   - **Valor de retorno:** Mensagem de erro adequada às propriedades do estado.
   - **Transições**:
     - Caractere "#" de código ASCII 35 leva ao estado 10.

2. **Estado 10:** Estado intermediário na construção de um comentário.
   - **Token de retorno:** <ERROR_FECHAMENTO> (indicador de erro de comentário sem o caracter "#" de fechamento).
   - **Valor de retorno:** Mensagem de erro adequada às propriedades do estado.
   - **Transições**:
     - Caracteres de códigos ASCII de 32 a 126 levam ao estado 10.
     - Caractere "#" de código ASCII 35 leva ao estado 11.

3. **Estado 11:** Estado final na construção de um bloco de comentário.
   - **Token de retorno:** <comment> (*token* de comentário)
   - **Valor de retorno:** LEXEMA (buffer de caracteres do lexema em seu estado atual da análise de tokens).

Dependendo do tipo do *token* e de seu valor, o analisador léxico é capaz de reconhecer um token do código em um determinado estado do autômato, ou então acusar um erro e então registrar uma mensagem que descreve o problema.

2.2. Estrutura do Arquivo Palavras_Reservadas.EXT

No arquivo de configuração Palavras_Reservadas.EXT, o usuário do EXTLex deve armazenar todas as palavras reservadas da linguagem, uma por linha. A Tabela 3 ilustra a estrutura do arquivo Palavras_Reservadas.EXT para uma linguagem que contém as palavras reservadas: *for*, *while* e *int*.

Descrição	Entradas do arquivo
Primeira palavra reservada da linguagem de programação.	01: for
Segunda palavra reservada da linguagem de programação.	02: while
Terceira palavra reservada da linguagem de programação.	03: int
Linha vazia indicando final do arquivo.	14:

Tabela 3 – Exemplo do formato do arquivo Palavras_Reservadas.EXT

2.3. Fluxo de Execução da Análise Lexical do EXTLex

Como entrada de sua execução, o EXTLex recebe os dois arquivos de configuração, bem como um código a ser compilado, escrito na linguagem de programação especificada. Na análise lexical, o EXTLex lê cada caractere do código a ser compilado, e então navega pelos estados do autômato finito de acordo com as transições especificadas no arquivo Automato.EXT, armazenando cada caractere lido em um *buffer* de lexema. Se não existirem transições para o estado atual da análise, realiza-se um dos seguintes procedimentos:

- Se o estado atual for final, armazene o token, seu valor correspondente e o valor de linha atual na lista de tokens do código. Caso seja um identificador, armazene-o na tabela de símbolos.
- Se o estado atual não for final (estado de erro), armazene a mensagem de erro do estado e linha atual na lista de erros lexicais.

Ao final da análise, o EXTLex exibe em tela a estrutura de *tokens* do código lido, juntamente a todas as mensagem de erro aplicáveis ao código.

3. Experimentos e Resultados

No objetivo de validar o analisador léxico apresentado neste trabalho, foi desenvolvida a linguagem de programação EXTLanguage: uma linguagem procedural, que contém os principais recursos comumente encontrados em linguagens de programação populares.

Primeiramente, a estrutura da linguagem EXTLanguage foi especificada com base no formato dos arquivos Automato.EXT e Palavras_Reservadas.EXT. Em seguida, o EXTLex foi empregado na análise léxica de um conjunto com 25 códigos-fonte que resolviam diferentes exercícos de programação, e um conjunto com 25 versões destes códigos-fonte contendo diferentes erros de nível léxico (caracteres inválidos, strings mal formadas, etc).

O EXTLex foi capaz de confeccionar a estrutura léxica de todos os códigos-fonte, e também reconhecer os erros léxicos presentes em cada um.

4. Considerações Finais

Este artigo apresentou o EXTLex, um analisador léxico configurável, dependente apenas de dois arquivos de configuração, conforme destacado na Seção 2. Os arquivos de configuração permitem uma representação descomplicada da linguagem de programação, o que promove fácil usabilidade, permitindo que o EXTLex reduza o custo de desenvolvimento dos compiladores de múltiplas linguagens de programação distintas.

Como trabalhos futuros destaca-se a implementação dos analisadores sintático e semântico, seguindo a mesma ideologia de desenvolvimento apresentada neste ínterim.

5. Referências bibliográficas

ALFRED, V. A.; ULLMAN D. J.; SETHI, R. **Compiladores: Princípios, Técnicas e Ferramentas**. LTC, Rio de Janeiro. 1995.

JOHNSON, S. C. **Yacc: Yet Another Compiler Compiler.** Holt, Rinehart, and Winston, New York, NY, USA. 1979.

LESK, M. E.; SCHMIDT, E. **Lex: A Lexical Analyzer Generator.** W. B. Saunders Company, Philadelphia, PA, USA. 1990.

TOSCANI, S. S.; ALENCAR, A. M. **Implementação de Linguagens de Programação.** Artmed. Editora Sagra-Luzzatto. 2008