UniRuy & Área 1 | Wyden

PROGRAMA DE CIÊNCIA DA COMPUTAÇÃO

TEORIA DE COMPILADORES

JOÃO MARCELO TAVARES SOUZA MATOS

Propriedades de Linguagens Regulares

Salvador - Bahia - Brasil

2022

JOÃO MARCELO TAVARES SOUZA MATOS

Propriedades de Linguagens Regulares

Trabalho Acadêmico elaborado junto ao programa de Engenharia UniRuy & Área 1 | Wyden, como requisito para obtenção de nota parcial da AV2 na disciplina Teoria de Compiladores no curso de Graduação em Ciência da Computação, que tem como objetivo consolidar os tópicos do plano de ensino da disciplina.

Orientador: Prof. MSc. Heleno Cardoso

Salvador - Bahia - Brasil
2022

da Tal, Aluno Fulano

Teoria de Compiladores: Resenha / Mapa Mental / Perguntas

– Aluno Fulano de Tal. Salvador, 2022.
18 f. : il.

Trabalho Acadêmico apresentado ao Curso de Ciência da Computação, UniRuy & Área 1 | Wyden, como requisito para obtenção de aprovação na disciplina Teoria de Compiladores.

Prof. MSc. Heleno Cardoso da S. Filho.

1. Resenha
2. Mapa Mental
3. Perguntas/Respostas (Mínimo de 03 – Máximo de 05)
4. Conclusão

I. da Silva Filho, Heleno Cardoso II. UniRuy & Área 1 | Wyden. III. Trabalho Acadêmico

CDD:XXX

TERMO DE APROVAÇÃO

JOÃO MARCELO TAVARES SOUZA MATOS

PROPRIEDADES DE LINGUAGENS REGULARES

Trabalho Acadêmico aprovado como requisito para obtenção de nota parcial da AV1 na disciplina Teoria de Compiladores, UniRuy & Área 1 | Wyden, pela seguinte banca examinadora:

BANCA EXAMINADORA

Profº. MScº. Heleno Cardoso
Wyden

Salvador, 04 de Novembro de 2022

Dedico este trabalho acadêmico a todos que contribuíram direta ou indiretamente com minha formação acadêmica.

Agradecimentos

Primeiramente agradeço a Deus. Ele, sabe de todas as coisas, e através da sua infinita misericórdia, se fez presente em todos os momentos dessa trajetória, concedendo-me forças e saúde para continuar perseverante na minha caminhada.

E a todos aqueles que contribuíram direta ou indiretamente para a minha formação acadêmica.

"A educação tem raízes amargas, mas os seus frutos são doces".

Aristóteles.

Resumo

Sendo uma das linguagens formais, a linguagem regular é na ciencia da computação de extrema importancia para a produção de operações de concatenação, união e fecho de Kleene sobre os elementos alfabeticos.

Essas linguagens são comumente usadas no processo de compilação, sendo mais preciso na análise sintática.

Keywords: Computer science, language, syntactic analysis.

Abstract

Being one of the formal languages, a regular language is in computer science of extreme importance for the production of concatenation, union and Kleene closing operations on the alphabetic elements.

These languages are commonly used in the compilation process, being more precise in the parsing.

Keywords: Logic, Computing, Programming language, IoT, Machine Learning, algorithms.

Lista de figuras

Lista de abreviaturas e siglas

AFNs	- Automatos Finitos não deterministicos
ER	- Expressões regulares
AFD	- Automatos finitos deterministicos

Sumário

[1] De acordo com a Associação Brasileira de Normas Técnicas. NBR 6023.

1 Propriedades de Linguagens Regulares

1.1 Introdução

As propriedades da linguagem regular e suas teorias, permitiram definirmos o que é e o que não é uma linguagem regular, apresentando sempre teoremas para exemplificar e comprovar sua verdade.

Como representamos quase sempre uma linguagem regular através de autômatos finitos, podemos analisar pela máquina de estado essas comprovações.

1.1.1 Perguntas e Respostas - Mínimo de 2 e Máximo de 5

1° Pergunta: Mostre que existe um algoritmo para determinar se a intersecção e união de duas linguagens regulares é finita, vazia ou infinita.

2° Pergunta: Se a lingaugem é vazia, ela é regular?

1.2 Conclusão

Conforme os tópicos citados nesse artigo, existem diversos tipos de propriedades que são definidas com o intuito de produzir a linguagem regular e suas regras, juntamente com teorias para facilitar a sua identificação como o lema de bombeamento ou a Teorema de Myhill-Nerode.

2 Propriedade de fechamento das linguagens regulares

Propriedades de fechamento em linguagens regulares são definidas como certas operações em linguagem regular que são garantidas para produzir linguagem regular. Fechamento se refere a alguma operação em um idioma, resultando em um novo idioma que é do mesmo “tipo” originalmente operado, ou seja, regular. (LIMA, 2022)

1 - Kleen Closure: RS é uma expressão regular cuja linguagem é L, M. R * é uma expressão regular cuja linguagem é L *. (LIMA, 2022)

2 - Fechamento positivo: RS é uma expressão regular cuja linguagem é L, M.R+ é uma expressão regular cuja linguagem é L+.r (LIMA, 2022)

3 - Complemento: O complemento de uma linguagem L (em relação a um alfabeto E que E* contenha L) é E* –L. Como E* é certamente regular, o complemento de uma linguagem regular é sempre regular. (LIMA, 2022)

4 - Operador reverso: dado o idioma L, LR é o conjunto de strings cuja reversão está em L. Exemplo: L = 0, 01, 100; LR = 0, 10, 001. Prova: Seja E uma expressão regular para L. Mostramos como inverter E, para fornecer uma expressão regular ER para LR. (LIMA, 2022)

5 - Complemento: O complemento de uma linguagem L (em relação a um alfabeto que contenha L) é –L. Como é certamente regular, o complemento de uma linguagem regular é sempre regular. (LIMA, 2022)

6 - União: Sejam L e M as linguagens das expressões regulares R e S, respectivamente. Então R + S é uma expressão regular cuja linguagem é (LUM). (LIMA, 2022)

7 - Intersecção: Sejam L e M as linguagens das expressões regulares R e S, respectivamente, então é uma expressão regular cuja linguagem é L intersecção M. prova: Sejam A e B DFA’s cujas linguagens são L e M, respectivamente. Construa C, o autômato de produto de A e B faz com que os estados finais de C sejam os pares que consistem nos estados finais de A e B. Operador Set Difference: Se L e M são linguagens regulares, então L - M = strings em L, mas não M. Prova: sejam A e B DFAs cujos idiomas são L e M, respectivamente. Construa C, o autômato produto de A e B faz com que os estados finais de C sejam os pares, onde o estado A é final, mas o estado B não. (LIMA, 2022)

8 - Homomorfismo: um homomorfismo em um alfabeto é uma função que fornece uma string para cada símbolo nesse alfabeto. Exemplo: h (0) = ab; h (1) = E . Estenda para strings por h (a1 ... an) = h (a1) ... h (an). Exemplo: h (01010) = ababab.

Se L é uma linguagem regular eh é um homomorfismo em seu alfabeto, então h (L) = h (w) | w está em L também é uma linguagem regular. Prova: Seja E uma expressão regular para L. Aplique h a cada símbolo em E. Linguagem de R resultante, E é h (L). Homomorfismo inverso: seja h um homomorfismo e L uma língua cujo alfabeto é a língua de saída de h. h-1(L) = w | h (w) está em L. (LIMA, 2022)

3 Identificação de linguagens não regulares usando o lema de bombeamento

Para identificarmos uma linguagem não regular, precisamos entender antes, o que é de fato uma linguagem não regular:

Os autômatos finitos podem reconhecer muitas linguagens uteis. Porém, estas máquinas não podem reconhecer todas as linguagens. Ou seja, existem linguagens não - regulares, que estão além do poder de reconhecimento dos autômatos finitos. A intuição por detrás da limitação dos autômatos finitos é que eles tem memória muito limitada e, portanto, “se perdem em contagens complexas(SALVIM, 2022)

Sabendo como identificar uma linguagem não regular, veremos o lema:

Lema do Bombeamento (LB). Se A é uma linguagem regular, então existe um número p (o comprimento de bombeamento) tal que, se s é qualquer cadeia de A de comprimento no mínimo p, então s pode ser dividida em três subcadeias (SALVIM, 2022)

s = xyz

satisfazendo as seguintes condições:

Condição 1: para cada $i \geq 0$, $xy^iz \in A$,

Condição 2: $|y| > 0$, e

Condição 3: $|xy| \leq p$.

Figura 1 – Lema do Bombeamento: Condições

3.0.1 Demonstração

Seja $M = (Q, \Sigma, \delta, q_1, F)$ um AFD que reconhece A.

Figura 2 – Lema do Bombeamento: Demonstração

Atributos ao comprimento de bombeamento p o total |Q| de estados de M. Primeiro note que se a linguagem não tem nenhuma cadeia com mais do que p símbolos, o lema vale por vacuidade.

Caso contrário, para qualquer cadeia s A com mais que p símbolos, pelo menos um mesmo estado do autômato tem que ser visitado mais de uma vez. Isto acontece porque não AFD a cada símbolo lido um estado é visitado, e se há mais símbolos que estados o Princípio da Casa dos Pombos garante que pelo menos um estado é visitado mais de uma vez. Para ilustrar essa ideia, note que na figura abaixo o estado q9 é visitado duas vezes ao se reconhecer a cadeia s que tem mais que p = |Q| símbolos. (SALVIM, 2022)

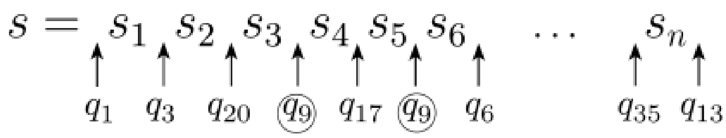

Figura 3 – Lema do Bombeamento: Demonstração - 2

Isso quer dizer que parte da cadeia s é reconhecida em um loop no AFD: a subcadeia visitada entre os estados que se repetem. No exemplo acima, os símbolos s4 e s5 são consumidos dentro do loop.

Isso quer dizer que a cadeia s pode ser quebrada em três partes s = xyz, em que y representa uma subcadeia não-vazia que é garantidamente consumida em um loop.

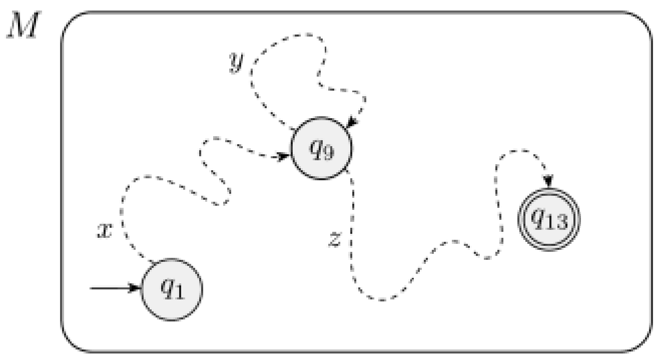

Figura 4 – Lema do Bombeamento: Demonstração - 3

Note que o fato de y ser não-vazia se reflete na Condição (2) do lema:

$|y| > 0.$

(Mas note que já quanto às partes x e z não temos nenhuma garantia: elas podem ou não ser vazias, e podem ou não ter loops também.)

Note também que durante o reconhecimento da cadeia s, o loop y deve ser atingido e percorrido antes de ser ultrapassado o n´umero de estados do autômato.

Isto se reflete na Condição (3) do lema:

$|xy| <= p.$

Logo, se o AFD M reconhece s = xyz, em que y é consumida em um loop, qualquer cadeia em que o mesmo loop de y seja usado um número qualquer de vezes também deve ser reconhecida por M (e, portanto, também pertence à linguagem A).

Isto se reflete na Condição (1) do lema:

Ai >= 0 : xyi z A.

Assim mostramos que para toda linguagem regular existe p tal que as Condições (1), (2) e (3) são satisfeitas para cadeias maiores que p. (SALVIM, 2022)

3.0.2 Intuição

Intuitivamente, o Lema do Bombeamento diz que AFDs “se perdem em contagens longas” porque não podem controlar quantas vezes um loop é utilizado:

1. Para qualquer cadeia grande o suficiente, o único jeito de o AFD aceitar a cadeia é fazendo pelo menos loop.

2. Se um AFD reconhece uma cadeia passando por um loop uma vez, qualquer cadeia que passe pelo loop um número qualquer de vezes (0, 1, 2, 3, . . .) e termine no mesmo estado de aceitação também tem que ser aceita pelo AFD.

Essa propriedade dos AFDs diz, intuitivamente, que linguagens regulares não podem ter cadeias que necessitem de contagem complexa de símbolos. Isto quer dizer que o LB pode ser usado para mostrar que uma linguagem não é regular. (SALVIM, 2022)

Referências[1]

LIMA, A. *PROPRIEDADES DE FECHAMENTO DE LINGUAGENS REGULARES*. 2022. Citado 2 vezes nas páginas 13 e 14.

SALVIM. *Linguagens Não-Regulares*. 2022. Citado 3 vezes nas páginas 15, 16 e 17.

[1] De acordo com a Associação Brasileira de Normas Técnicas. NBR 6023.