

Nome: OBENSON MAURICE E RENAN BABINSKY

Data : 08 de Abril de 2022

Disciplina : LFA

Universidade : Universidade Federal da Fronteira Sul (UFFS)



Relatório de Linguagens Formais e Autômatos

Projeto Final da Disciplina de Linguagens Formais e Autômatos

Objetivo:

Este trabalho é o resultado do processo de aprendizagem da Disciplina de Linguagens Formais e Autômatos. Construímos um projeto prático dessa disciplina em que aprendemos e fixamos alguns conceitos chaves apresentados no curso da Ciência da Computação da Universidade Federal da Fronteira Sul.

Ilustradamente, alguns conceitos estudados são: entender o que é uma gramática que em breve que trataremos do decorrer deste trabalho, os tipos de gramática, os elementos e as características distintas entre as gramáticas e outros assuntos.

O nosso objetivo consiste em construir este projeto com a finalidade de entender as transformações dos estados autômatos determinísticos finitos em autômatos não determinísticos, bem como estudar o processo de minimização dos estados.

Assim, para cumprir o escopo deste trabalho, abordaremos alguns temas principais tais como: autômatos finitos determinísticos e autômatos finitos não determinísticos.

Autômatos Finitos Determinísticos (AFD)

Um autômato finito **determinístico** — também chamado máquina de estados finita **determinística** (AFD) — é uma Máquina de estados finita que aceita ou rejeita cadeias de símbolos gerando um único ramo de computação para cada cadeia de entrada.

Na teoria da computação, uma **máquina de estados finita não-determinística** ou um **autômato finito não-determinístico** (AFND) é uma máquina de estado finita em que para cada par de estado e símbolo de entrada pode haver vários próximos estados possíveis. Isso o distingue do autômato finito determinístico (AFD), onde o próximo estado possível é univocamente determinado. Embora AFD e AFND possuam definições distintas, pode ser mostrado na teoria formal que eles são equivalentes e, deste modo, para qualquer AFND dado, pode-se construir um AFD equivalente e vice-versa: essa é a construção do conjunto das partes. Ambos os tipos de autômatos reconhecem apenas linguagens regulares.

Cabe esclarecer que máquinas de estados finitas não-determinísticas são às vezes estudadas com o nome de “Subshifts” de tipo finito. Essas máquinas são generalizadas pelo autômato probabilístico, o qual atribui uma probabilidade para cada transição de estado, e por várias outras maneiras, tais como autômato com pilha e AFNs com transições ϵ .

Autômatos finitos não-determinísticos foram introduzidos em 1959 por Michael O Rabin e Dana Scott que também mostraram sua equivalência com autômatos finitos determinísticos. Comparando esses dois termos, pode-se constatar que os AFDs e os AFNDs representam a mesma classe de linguagens, ou seja, as linguagens regulares.

Vale mencionar que há algumas vantagens e desvantagens entre si, visto que o AFD possui uma vantagem que é a implementação trivial e a sua desvantagem é não representar claramente algumas L. R (linguagens regulares) e portanto a AFND possui sua vantagem que é a explicitude, quer dizer que representa mais claramente e a sua desvantagem é a Implementação complexa.

Para uma compreensão mais acurada de AFD e AFND, cumpre analisarmos o que é um *token*. Um **token** ou também chamado **componente léxico** é uma cadeia de caracteres que tem um significado coerente em verdadeira linguagem de programação. Exemplos de tókenes poderiam ser palavras-chave (if, else, while, int, ...), identificadores, números, sinais, ou um operador de vários caracteres, (por exemplo, := "" "+").

por fim, São os elementos mais básicos sobre os quais se desenvolve toda a tradução de um programa, surgem na primeira fase, chamada análise léxica, no entanto se seguem utilizando nas seguintes fases (análises sintáticas e análises semânticas) antes de perder na fase de síntese.

Projeto Prático da Disciplina

Construção da AF a partir de token ou GR

VAMOS DECLARAR OS TOKENS:

Nesse trabalho usaremos os três tokens seguintes porém é possível usar qualquer token desejado, Vale mencionar que a inserção dos tokens fica ao critério do usuário.

Tokens:

else
end
se

S::= eA
A::= lB
B::= sC
C::= eD
D::= ϵ

S::= iA
A::= fB
B::= eC
D::= ϵ

se
S::=sA
A::=eB
B::= ϵ

cada vez que saiu de um estado daí temos que gerar um novo estado assim fizermos o mapeamento do projeto manualmente.

δ	e	l	s	n
$\rightarrow S$	A,E,G			
A		B		
B			C,H	
C	D			
*D	-	-	-	-
E				F
F				
G				
*H				

δ	e	l	s	n	d
$\rightarrow S$	[AE G]	X	X	X	X
[AE G]	X	B	X	F	X
B	X	X	[CH]	X	X
F	X	X	X	X	H
C	D	X	X	X	X
*H	X	X	X	X	X
*D	X	X	X	X	X
*X	X	X	X	X	X

Cada vez que achamos uma coluna sem mapeamento nós preenchemos com X que representa o estado de erro.

Aqui temos a imagem da tela de execução do trabalho com a passagem de Automatos finitos não determinístico(AFND) para automatos finitos determinísticos (AFD) que achamos importante adicionar também.

AFND									
∅	s	e	n	t	a	o	i	u	
->S'	A,H	C,M			M	M	M	M	
A		B							
B*									
C			D						
D				E					
E					F				
F						G			
G*									
H		I							
I			J						
J					K				
K						L			
L*									
M		M			M	M	M	M	

AFD									
∅	s	e	n	t	a	o	i	u	
->S'	[A,H]	[C,M]			M	M	M	M	
[A,H]		[B,I]							
[C,M]		M	[D,]		M	M	M	M	
M		M			M	M	M	M	
[B,I]*			J						
[D,]				E					
J					K				
E					F				
K						L			
F						G			
L*									
G*									

(.env) anonymous@CTNDELLB9HCJH3:~/LFA\$

Referências

https://pt.wikipedia.org/wiki/M%C3%A1quina_de_estados_finitos_n%C3%A3o_determin%C3%ADstica

<http://www.fsg.rnu.tn/imgsite/cours/Chapitre4.pdf>

<http://gallium.inria.fr/~maranget/X/421/poly/automate.html>