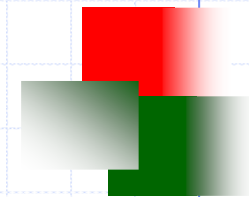




TCC 2000



ROTEIRO



SAIR

ANTERIOR

**PROTÓTIPO PARA TRANSFORMAÇÃO DE UMA
EXPRESSION REGULAR PARA UMA FUNÇÃO
EQUIVALENTE EM PASCAL, UTILIZANDO
DOIS ALGORITMOS BASEADOS NO TEOREMA
DE KLEENE**



RONALD GLATZ



ROTEIRO

ROTEIRO

 OBJETIVO DESTE TRABALHO

 ESPECIFICAÇÃO DOS PROTÓTIPOS

 ALGUNS CONCEITOS BÁSICOS

 APRESENTAÇÃO DOS PROTÓTIPOS

 ALGORITMO DE [HOP1979]

 [HOP1979]

 [SIL2000]

 ALGORITMO DE [SIL2000]

 COMPARAÇÃO DOS ALGORITMOS

SAIR

 TRANSFORMAÇÃO DE UM AFD EM PROGRAMA

ANTERIOR





ROTEIRO

ROTEIRO

 PROTÓTIPO DE
[SIL2000]

 CONCLUSÃO

 PROTÓTIPO DE
[HOP1979]

 ANEXOS

 EQUIVALÊNCIA DE
MOORE

 INTERPRETADOR
UNIVERSAL DE
CADEIAS



SAIR

ANTERIOR





OBJETIVO

ROTEIRO



SAIR

ANTERIOR

- ✓ APRESENTAR A TRANSFORMAÇÃO DE UMA EXPRESSÃO REGULAR EM UM AFD
- ✓ SERVIR DE FERRAMENTA DE ENSINO-APRENDIZAGEM
- ✓ APRESENTAR UMA COMPARAÇÃO ENTRE AS DIFERENTES FORMAS DE TRANSFORMAÇÃO





BÁSICO

ROTEIRO

✓ Aqui serão apresentados alguns conceitos mais importantes para este trabalho

✓ ALFABETO (Σ)

✓ $\{A, B, C, D, E, \dots, X, Z\}$

✓ $\{0, 1\}$

✓ $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

✓ PALAVRA SOBRE O NOSSO ALFABETO

✓ Abacate

✓ Livro

SAIR

ANTERIOR





BÁSICO

ROTEIRO

✓ EXPRESSÃO REGULAR (ER)

✓ $a(b|c)^*ca^*|^{\wedge}$

✓ $a(aa|bb)^*$

✓ a^*

✓ a^{**}

✓ CONJUNTO REGULAR

✓ $\{0,1,01,001,011, \dots\}$: cadeias de $\Sigma=\{0,1\}$

SAIR

ANTERIOR





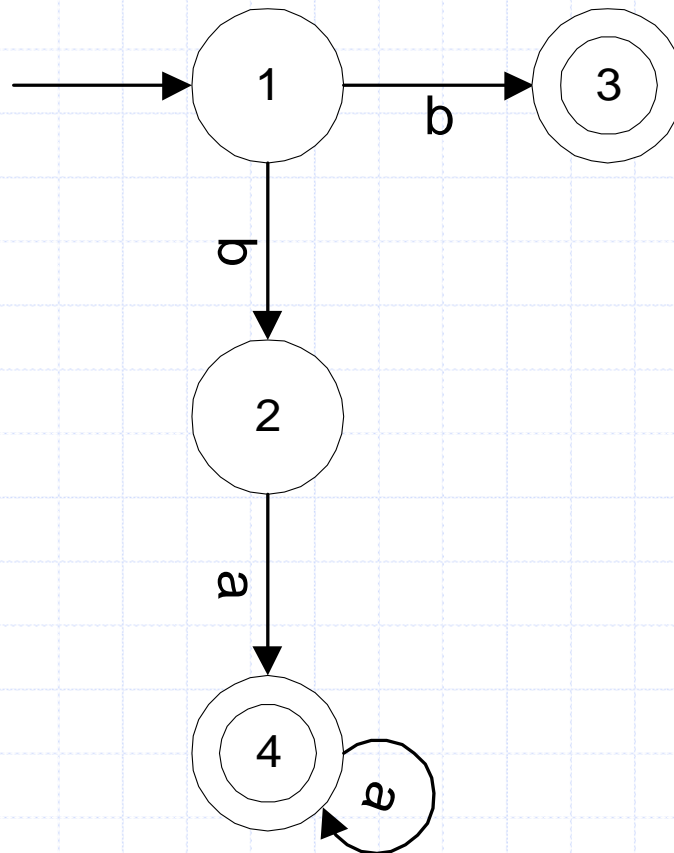
BÁSICO-AFND

ROTEIRO



SAIR

ANTERIOR





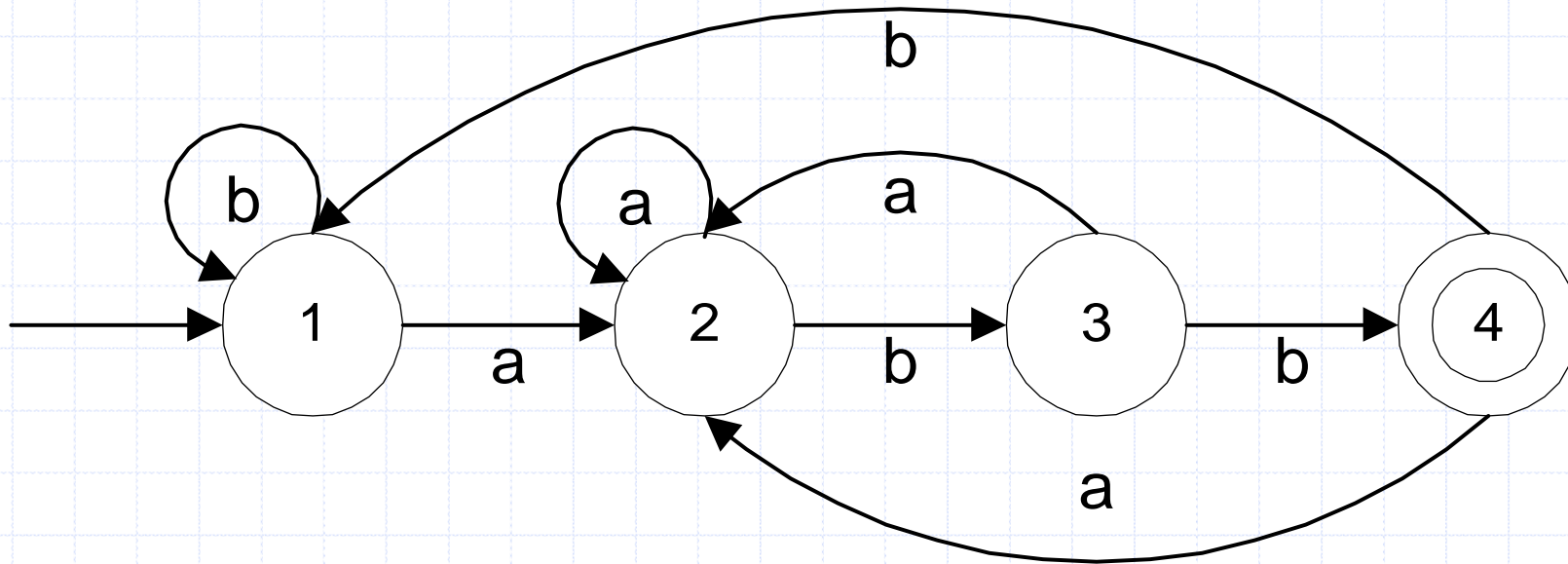
BÁSICO-AFD

ROTEIRO



SAIR

ANTERIOR





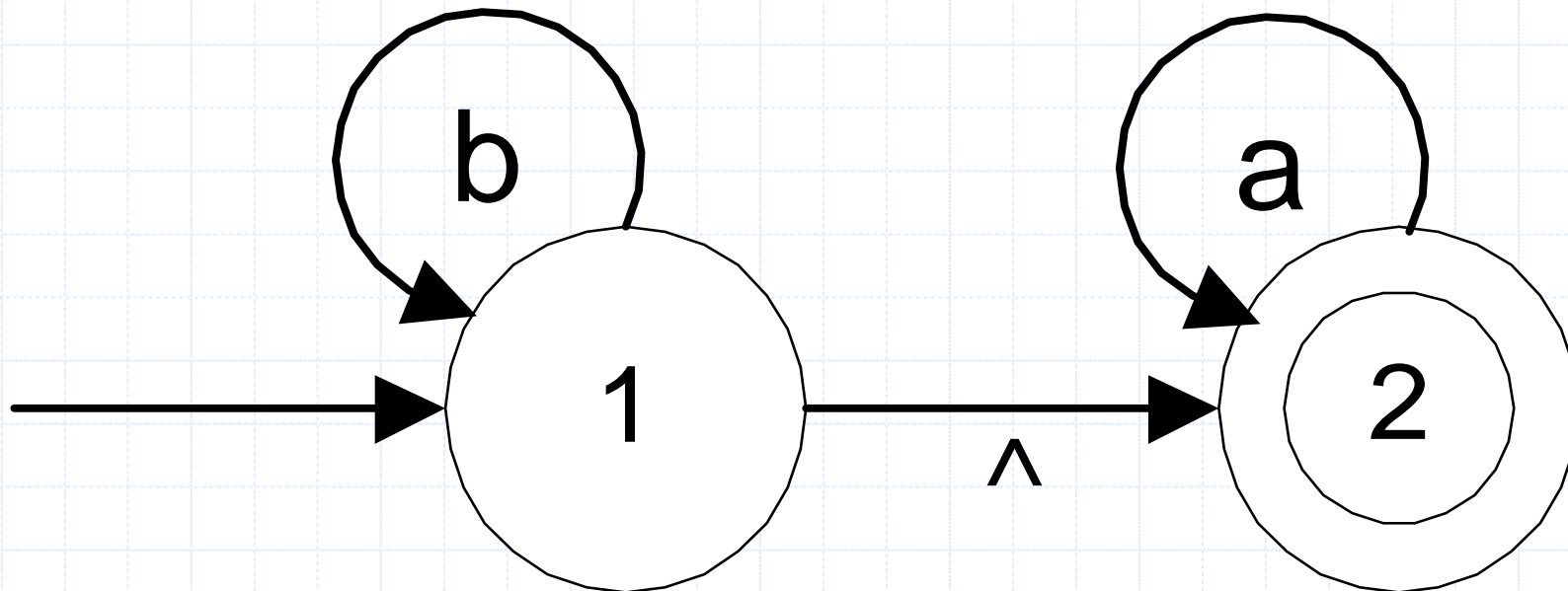
BÁSICO-AF ϵ

ROTEIRO



SAIR

ANTERIOR





BÁSICO-LLC

ROTEIRO

- ✓ LINGUAGEM PARA DEFINIR OUTRAS LINGUAGENS
- ✓ LINGUAGENS DE PROGRAMAÇÃO
- ✓ ANALISADORES LÉXICOS
- ✓ EDIÇÃO DE TEXTO



SAIR

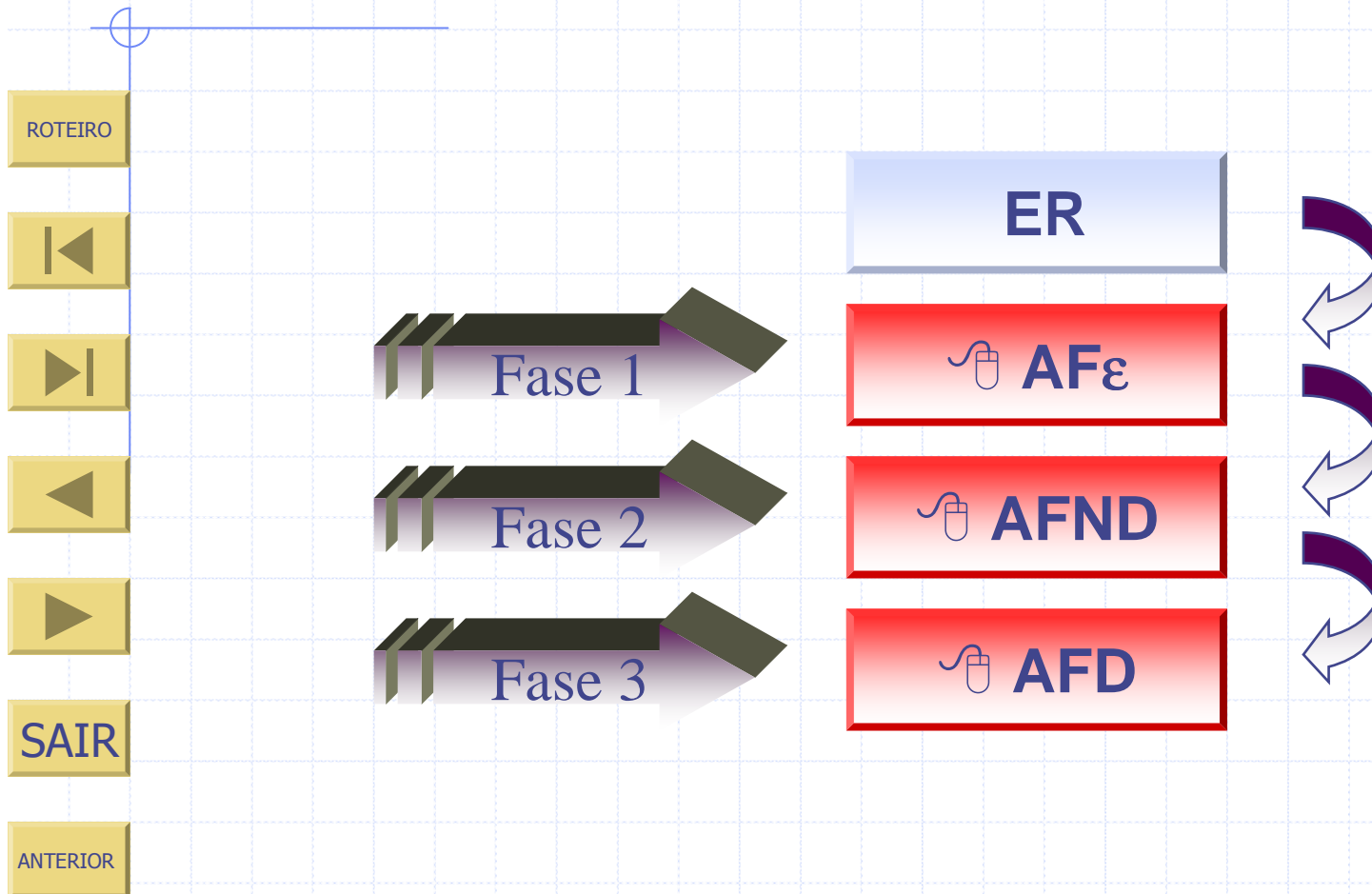
ANTERIOR





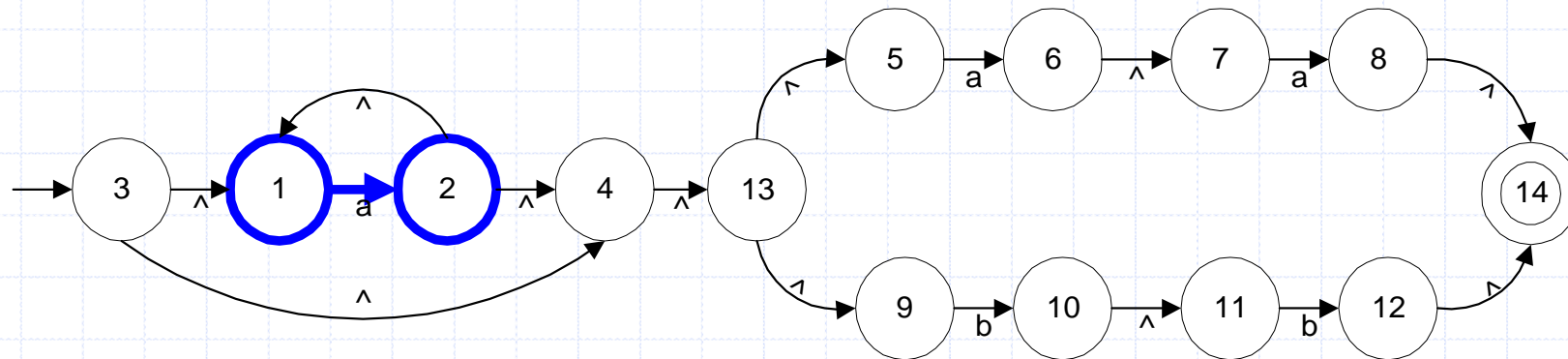
ALGORITMO DE [HOP1979]

FASES





GERAÇÃO DO AF ϵ



⇒ a^{*}(aa|bb)

ROTEIRO



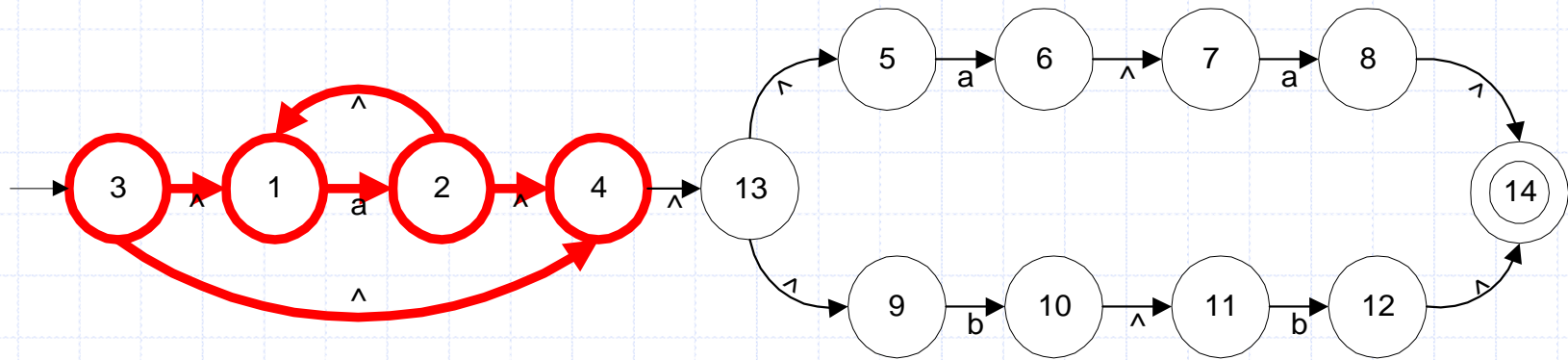
SAIR

ANTERIOR





GERAÇÃO DO AF ϵ



$\Rightarrow \underline{a^*}(aa|bb)$

- ROTEIRO
- ◀
- ▶
- ◀
- ▶
- SAIR
- ANTERIOR





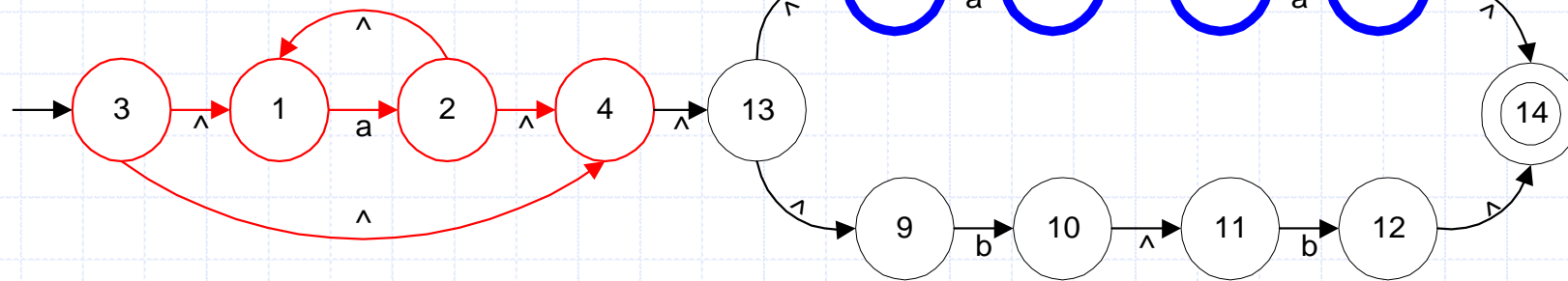
GERAÇÃO DO AF ϵ

ROTEIRO



SAIR

ANTERIOR

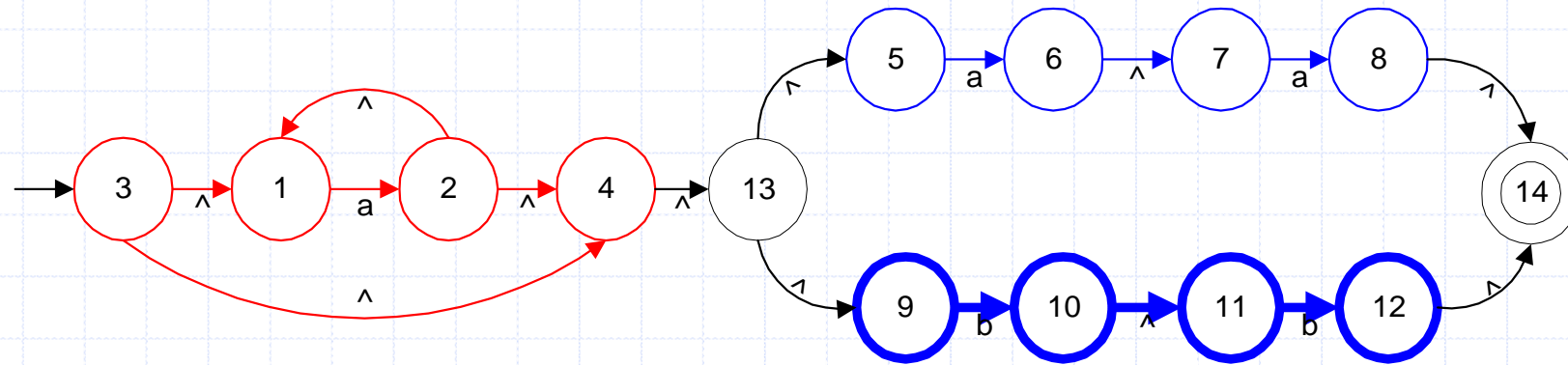


$\Rightarrow a^*(\underline{aa} | bb)$





GERAÇÃO DO AF ϵ



⇒ $a^*(aa|bb)$

ROTEIRO



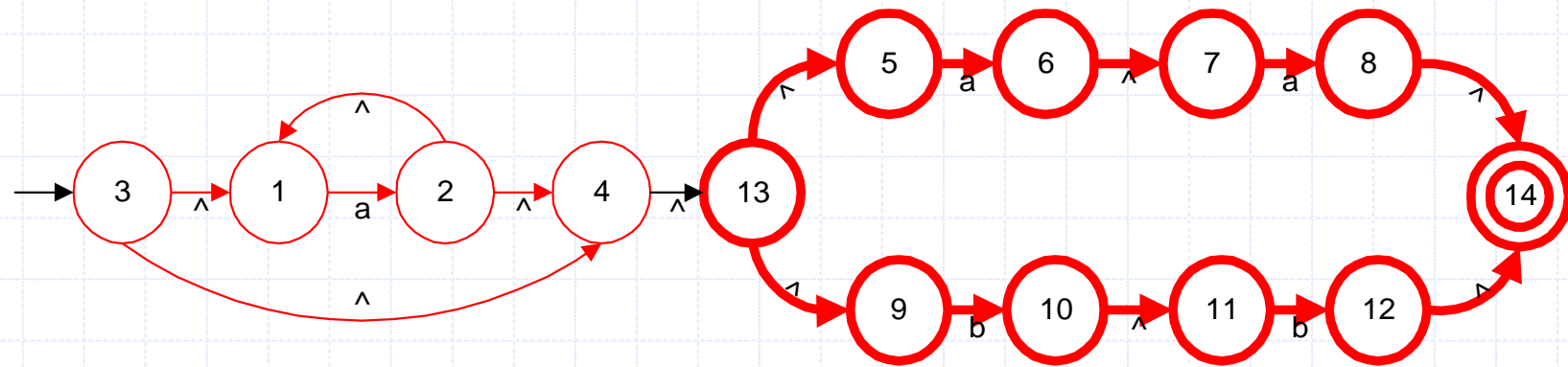
SAIR

ANTERIOR





GERAÇÃO DO AF ϵ



$\Rightarrow a^*(\underline{aa|bb})$

ROTEIRO



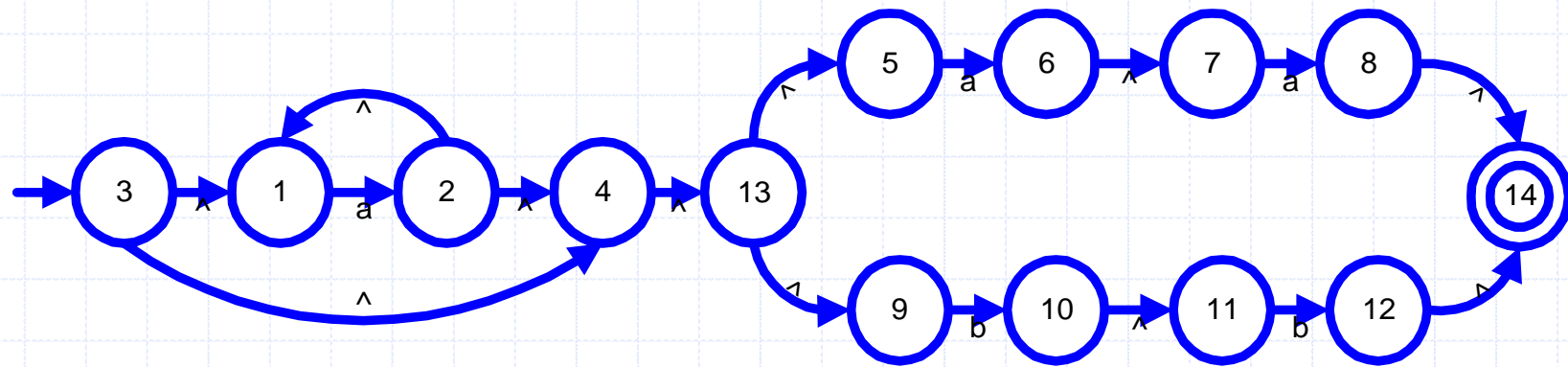
SAIR

ANTERIOR





GERAÇÃO DO AF ϵ



⇒ $a^*(aa|bb)$

ROTEIRO



SAIR

ANTERIOR





GERAÇÃO DO AFND A PARTIR DO AF ϵ

ROTEIRO

- ✓ Consiste em gerar um AFND equivalente a partir do AF ϵ da fase 1
- ✓ AFND $M = (\Sigma, Q, \delta', q_0, F)$
 - ✓ $\Sigma = \{a, b\}$
 - ✓ $Q = \{1, 2, 3, 4, \dots, 13, 14\}$
 - ✓ $\delta' = F_\epsilon(\delta''(\delta'''(q, \wedge), x))$
 - ✓ $F =$ Conjunto de Estados Finais



SAIR

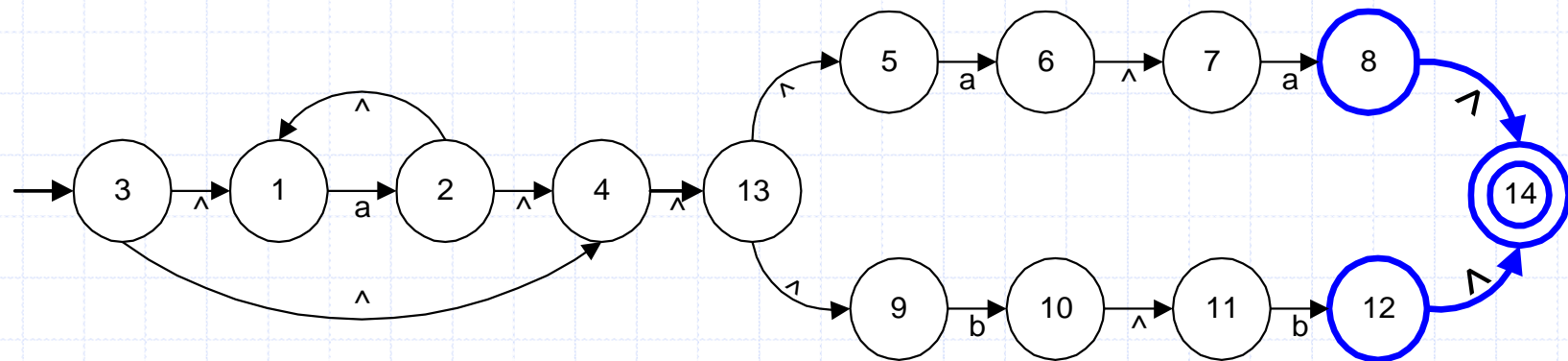
ANTERIOR





GERAÇÃO DO AFND A PARTIR DO AF ϵ

Conjunto de Estados Finais



✓ O conjunto de estados finais do AFND é formado pelo estado final do AF ϵ e todos os estados que atingem o estado final com o símbolo vazio (^) (direta ou indiretamente) no AF ϵ .

✓ $F = \{8, 12, 14\}$

ROTEIRO



SAIR

ANTERIOR





GERAÇÃO DO AFND A PARTIR DO AF ϵ

Geração das Transições do AFND

ROTEIRO

- ✓ A geração das transições do AFND é realizada através da função de transição $\delta' = F_\epsilon(\delta''(\delta'''(q, \hat{}), x))$
- ✓ Esta função é dividida nas seguintes etapas:
 - ✓ $\delta'''(q, \hat{}) =$ gera conjuntos de cada estado do AFND de todos os estados atingidos pelo símbolo vazio ($\hat{}).$ O estado que gerou o conjunto também faz parte do conjunto – ETAPA 1
 - ✓ $\delta''(\delta'''(q, \hat{}), x) =$ para cada estado dos conjuntos da ETAPA 1 gera-se o conjunto de todos os estados atingidos pelos símbolos do alfabeto – ETAPA 2
 - ✓ $F_\epsilon(\delta''(\delta'''(q, \hat{}), x)) =$ para todos os estados atingidos na ETAPA 2 gera-se o conjunto de todos os estados atingidos (direta ou indiretamente) pelo símbolo vazio ($\hat{})$ – ETAPA 3



SAIR

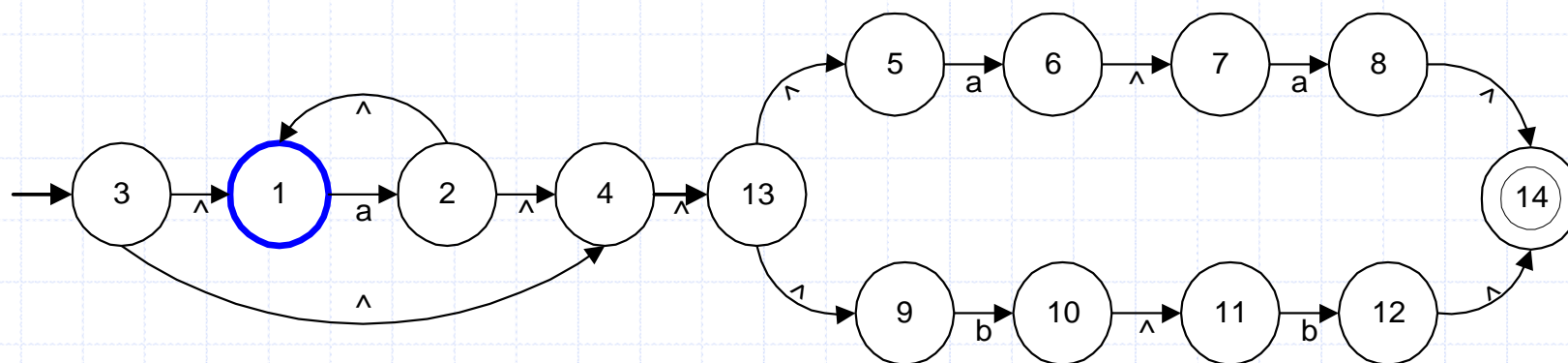
ANTERIOR





GERAÇÃO DO AFND A PARTIR DO AF ϵ

$$\delta' = F\epsilon(\delta''(\delta'''(q, \wedge), x))$$



✓ Sendo $q = 1$ aplicado sobre $\delta'''(q, \wedge)$

temos:

$$\checkmark \delta'''(1, \wedge) = \{1\}$$

ROTEIRO



SAIR

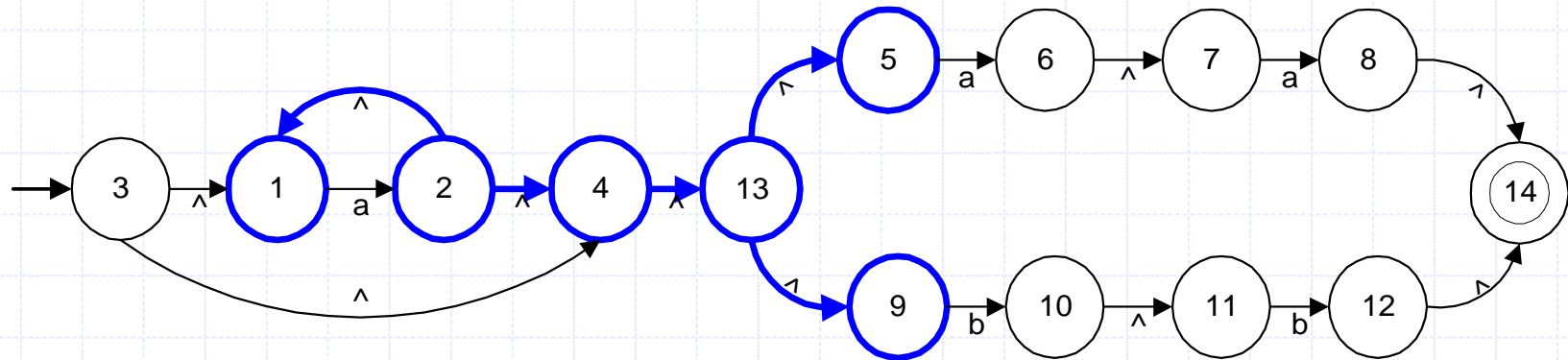
ANTERIOR





GERAÇÃO DO AFND A PARTIR DO AF ϵ

$$\delta' = F\epsilon(\delta''(\delta'''(q, \wedge), x))$$



✓ Sendo $q = 2$ aplicado sobre $\delta'''(q, \wedge)$
temos:

$$✓ \delta'''(2, \wedge) = \{1, 2, 4, 5, 9, 13\}$$

ROTEIRO



SAIR

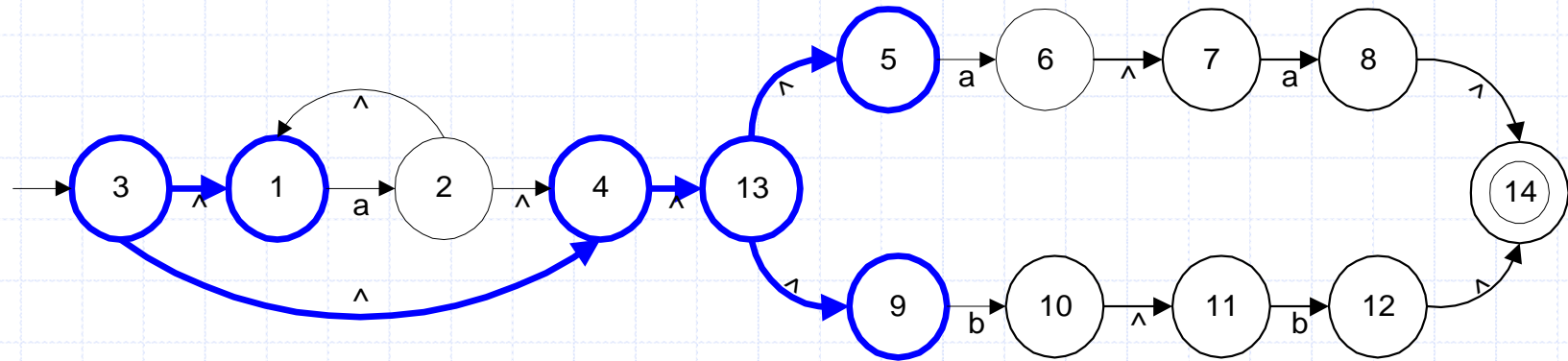
ANTERIOR





GERAÇÃO DO AFND A PARTIR DO AF ϵ

$$\delta' = F\epsilon(\delta''(\delta'''(q, \wedge), x))$$



✓ Sendo $q = 3$ aplicado sobre $\delta'''(q, \wedge)$
temos:

$$\checkmark \delta'''(3, \wedge) = \{1, 3, 4, 5, 9, 13\}$$

ROTEIRO



SAIR

ANTERIOR





GERAÇÃO DO AFND A PARTIR DO AF ϵ

$$\delta' = F\epsilon(\delta''(\delta'''(q, \wedge), x))$$

conjuntos obtidos:

ROTEIRO



✓ $\delta'''(1, \wedge) = \{1\}$

✓ $\delta'''(8, \wedge) = \{8, 14\}$

✓ $\delta'''(2, \wedge) = \{1, 2, 4, 5, 9, 13\}$

✓ $\delta'''(9, \wedge) = \{9\}$



✓ $\delta'''(3, \wedge) = \{1, 3, 4, 5, 9, 13\}$

✓ $\delta'''(10, \wedge) = \{10, 11\}$



✓ $\delta'''(4, \wedge) = \{4, 5, 9, 13\}$

✓ $\delta'''(11, \wedge) = \{11\}$



✓ $\delta'''(5, \wedge) = \{5\}$

✓ $\delta'''(12, \wedge) = \{12, 14\}$

SAIR

✓ $\delta'''(6, \wedge) = \{6, 7\}$

✓ $\delta'''(13, \wedge) = \{5, 9, 13\}$

ANTERIOR

✓ $\delta'''(7, \wedge) = \{7\}$

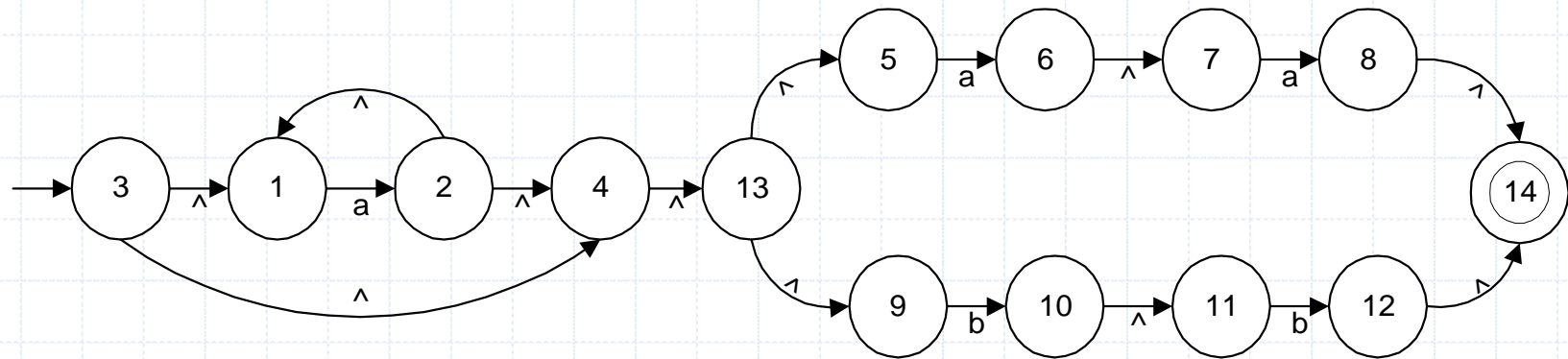
✓ $\delta'''(14, \wedge) = \{14\}$





GERAÇÃO DO AFND A PARTIR DO AF ϵ

$$\delta' = F\epsilon(\delta''(\delta'''(q, \wedge), x))$$



✓ Elementos da ETAPA 2:

- ✓ $\delta''(\delta'''(q, \wedge), x)$
- ✓ $\Sigma = \{a, b\}$
- ✓ $x =$ símbolo do Σ
- ✓ Conjunto de cada estado da ETAPA 1

ROTEIRO



SAIR

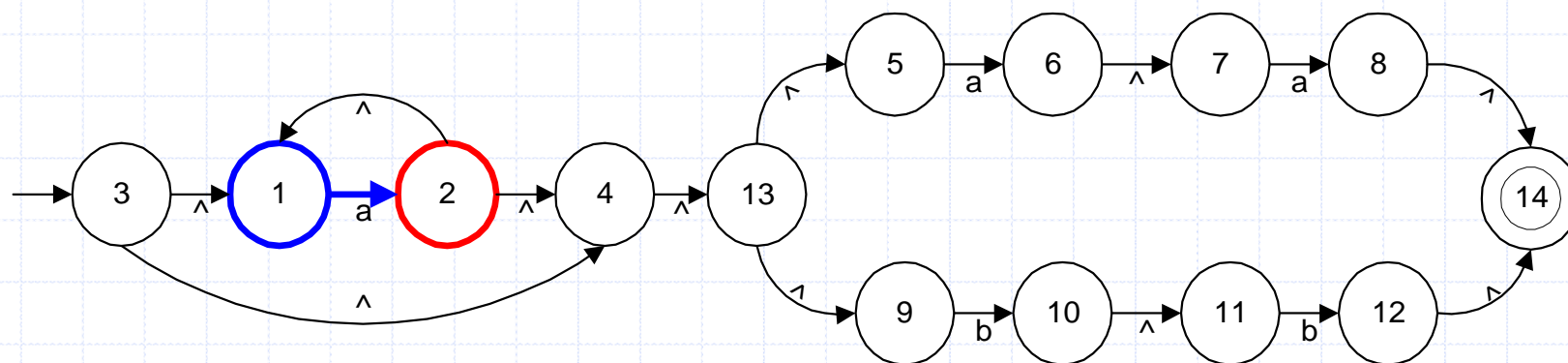
ANTERIOR





GERAÇÃO DO AFND A PARTIR DO AF ϵ

$$\delta' = F\epsilon(\delta''(\delta'''(q, \wedge), x))$$



- ✓ $\Sigma = \{a, b\}$
- ✓ $\delta'''(1, \wedge) = \{1\}$ - ETAPA 1
- ✓ $x = "a"$
- ✓ $\delta''(\{1}, "a") = \{2\}$
- ✓ $x = "b"$
- ✓ $\delta''(\{1}, "b") = \{\}$

ROTEIRO



SAIR

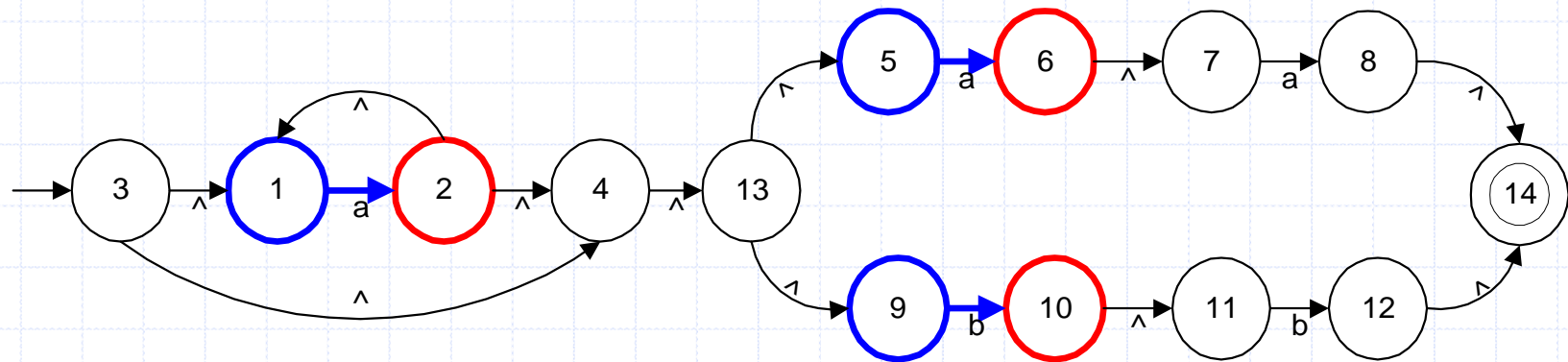
ANTERIOR





GERAÇÃO DO AFND A PARTIR DO AF ϵ

$$\delta' = F\epsilon(\delta''(\delta'''(q, \wedge), x))$$



- ✓ $\Sigma = \{a, b\}$
- ✓ $\delta'''(2, \wedge) = \{1, 2, 4, 5, 9, 13\}$ - ETAPA 1
- ✓ $x = "a"$
- ✓ $\delta''(\{1, 2, 4, 5, 9, 13\}, "a") = \{2, 6\}$
- ✓ $x = "b"$
- ✓ $\delta''(\{1, 2, 4, 5, 9, 13\}, "b") = \{10\}$

ROTEIRO

⏪

⏩

⏴

⏵

SAIR

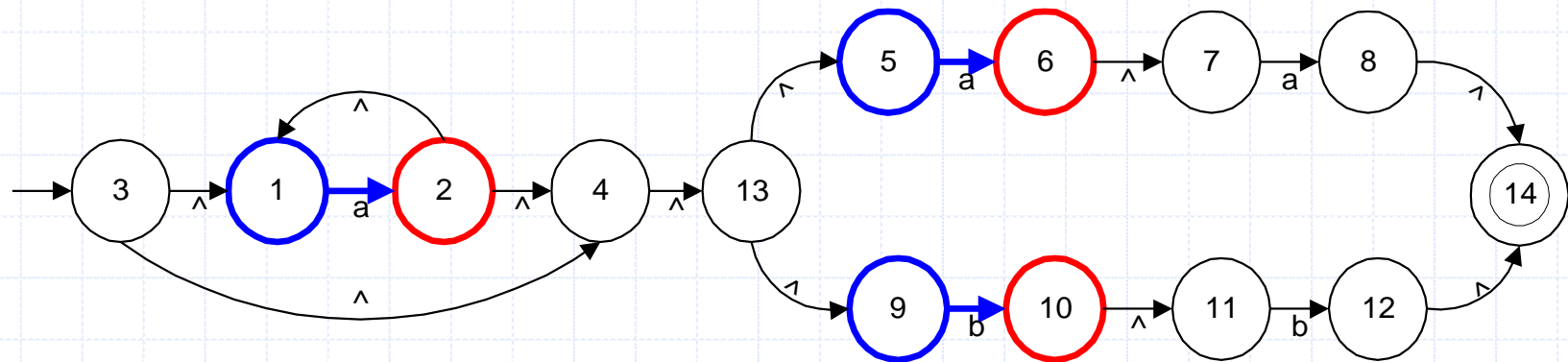
ANTERIOR





GERAÇÃO DO AFND A PARTIR DO AF ϵ

$$\delta' = F\epsilon(\delta''(\delta'''(q, \wedge), x))$$



- ✓ $\Sigma = \{a, b\}$
- ✓ $\delta'''(3, \wedge) = \{1, 3, 4, 5, 9, 13\}$ - ETAPA 1
- ✓ $x = "a"$
- ✓ $\delta''(\{1, 3, 4, 5, 9, 13\}, "a") = \{2, 6\}$
- ✓ $x = "b"$
- ✓ $\delta''(\{1, 3, 4, 5, 9, 13\}, "b") = \{10\}$

ROTEIRO

⏪

⏩

⏴

⏵

SAIR

ANTERIOR





GERAÇÃO DO AFND A PARTIR DO AF ϵ

$$\delta' = F\epsilon(\delta''(\delta'''(q, \wedge), x))$$

ROTEIRO

RESULTADOS DA ETAPA 2 PARA $\Sigma = \{a, b\}$

- ✓ $\delta''(\{1\}, "a") = \{2\}$ (Estado 1)
- ✓ $\delta''(\{1\}, "b") = \{\}$
- ✓ $\delta''(\{1, 2, 4, 5, 9, 13\}, "a") = \{2, 6\}$ (Estado 2)
- ✓ $\delta''(\{1, 2, 4, 5, 9, 13\}, "b") = \{10\}$
- ✓ $\delta''(\{1, 3, 4, 5, 9, 13\}, "a") = \{2, 6\}$ (Estado 3)
- ✓ $\delta''(\{1, 3, 4, 5, 9, 13\}, "b") = \{10\}$
- ✓ $\delta''(\{4, 5, 9, 13\}, "a") = \{6\}$ (Estado 4)
- ✓ $\delta''(\{4, 5, 9, 13\}, "b") = \{10\}$
- ...
- ✓ $\delta''(\{14\}, "a") = \{\}$ (Estado 14)
- ✓ $\delta''(\{14\}, "b") = \{\}$

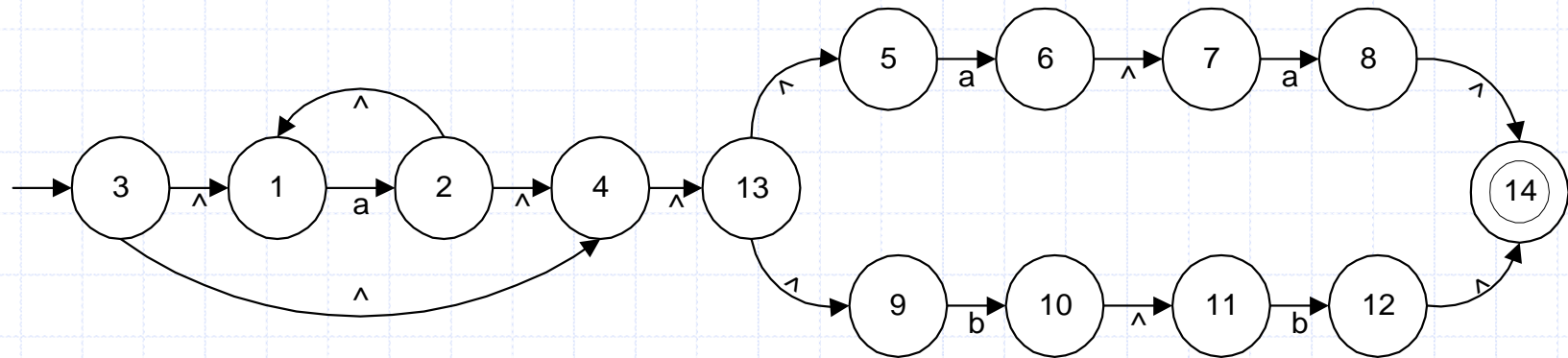
SAIR

ANTERIOR



GERAÇÃO DO AFND A PARTIR DO AF ϵ

$$\delta' = F\epsilon(\delta''(\delta'''(q, \wedge), x))$$



✓ Elementos da ETAPA 3:

- ✓ $\Sigma = \{a, b\}$
- ✓ $F\epsilon(\delta''(\delta'''(q, \wedge), x))$ onde $F\epsilon = \delta(q, \wedge)$
- ✓ Conjunto de cada estado da ETAPA 2

ROTEIRO



SAIR

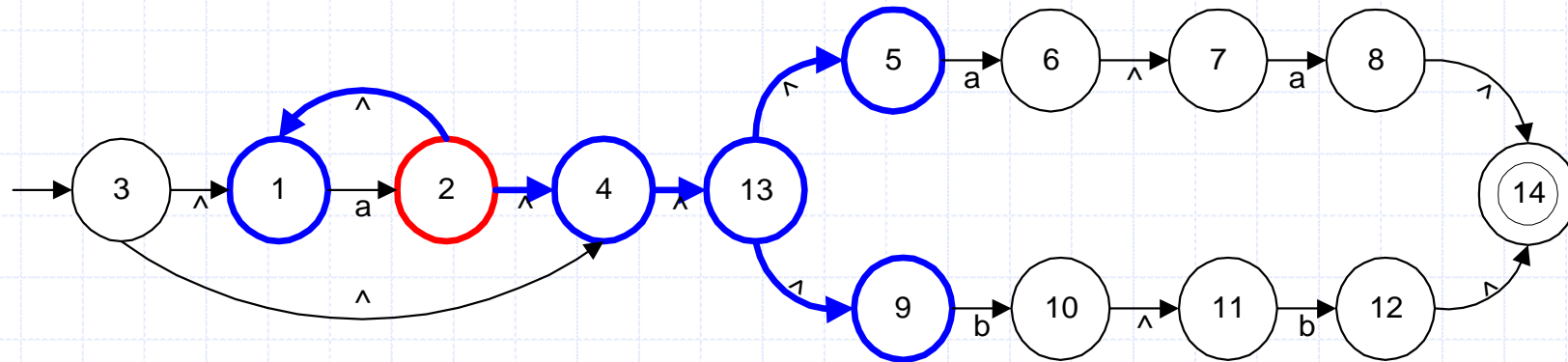
ANTERIOR





GERAÇÃO DO AFND A PARTIR DO AF ϵ

$$\delta' = F\epsilon(\delta''(\delta'''(q, \wedge), x))$$



TRANSIÇÕES DO ESTADO 1

- ✓ $\Sigma = \{a, b\}$
- ✓ $\delta''(\{1\}, "a") = \{2\}$ (Estado 1)
- ✓ $\delta''(\{1\}, "b") = \{\}$
- ✓ $F\epsilon(\{2\}) = \{1, 2, 4, 5, 9, 13\}$ (símbolo "a" do Σ)
- ✓ $F\epsilon(\{\}) = \{\}$ (símbolo "b" do Σ)

ROTEIRO

⏪

⏩

⏴

⏵

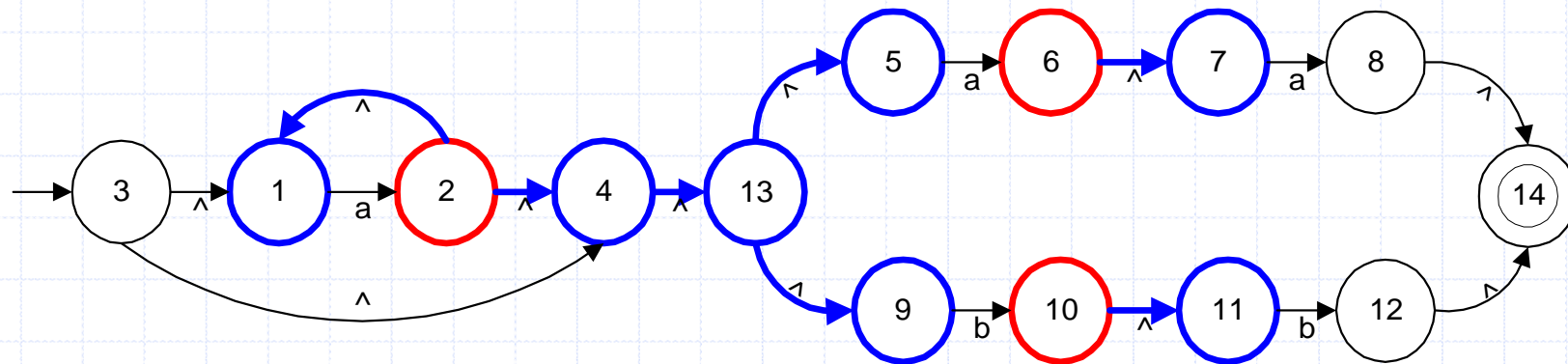
SAIR

ANTERIOR



GERAÇÃO DO AFND A PARTIR DO AF ϵ

$$\delta' = F\epsilon(\delta''(\delta'''(q, \wedge), x))$$



TRANSIÇÕES DO ESTADO 2

- ✓ $\Sigma = \{a, b\}$
- ✓ $\delta''(\{1, 2, 4, 5, 9, 13\}, "a") = \{2, 6\}$ (Estado 2)
- ✓ $\delta''(\{1, 2, 4, 5, 9, 13\}, "b") = \{10\}$
- ✓ $F\epsilon(\{2, 6\}) = \{1, 2, 4, 5, 7, 9, 13\}$ (símbolo "a" do Σ)
- ✓ $F\epsilon(\{10\}) = \{10, 11\}$ (símbolo "b" do Σ)

ROTEIRO

⏪

⏩

⏴

⏵

SAIR

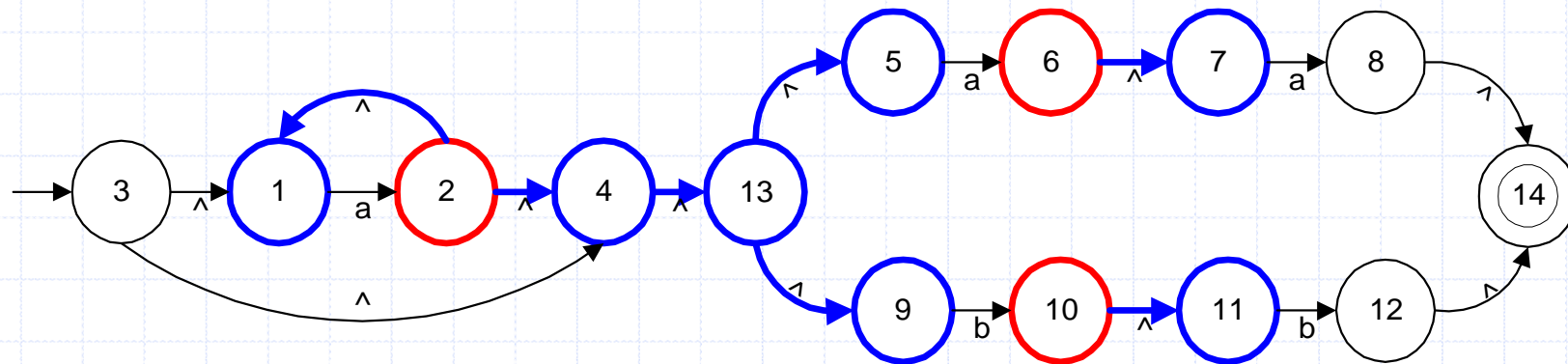
ANTERIOR





GERAÇÃO DO AFND A PARTIR DO AF ϵ

$$\delta' = F\epsilon(\delta''(\delta'''(q, \wedge), x))$$



TRANSIÇÕES DO ESTADO 3

- ✓ $\Sigma = \{a, b\}$
- ✓ $\delta''(\{1, 3, 4, 5, 9, 13\}, "a") = \{2, 6\}$ (Estado 3)
- ✓ $\delta''(\{1, 3, 4, 5, 9, 13\}, "b") = \{10\}$
- ✓ $F\epsilon(\{2, 6\}) = \{1, 2, 4, 5, 7, 9, 13\}$ (símbolo "a" do Σ)
- ✓ $F\epsilon(\{10\}) = \{10, 11\}$ (símbolo "b" do Σ)

ROTEIRO

⏪

⏩

⏴

⏵

SAIR

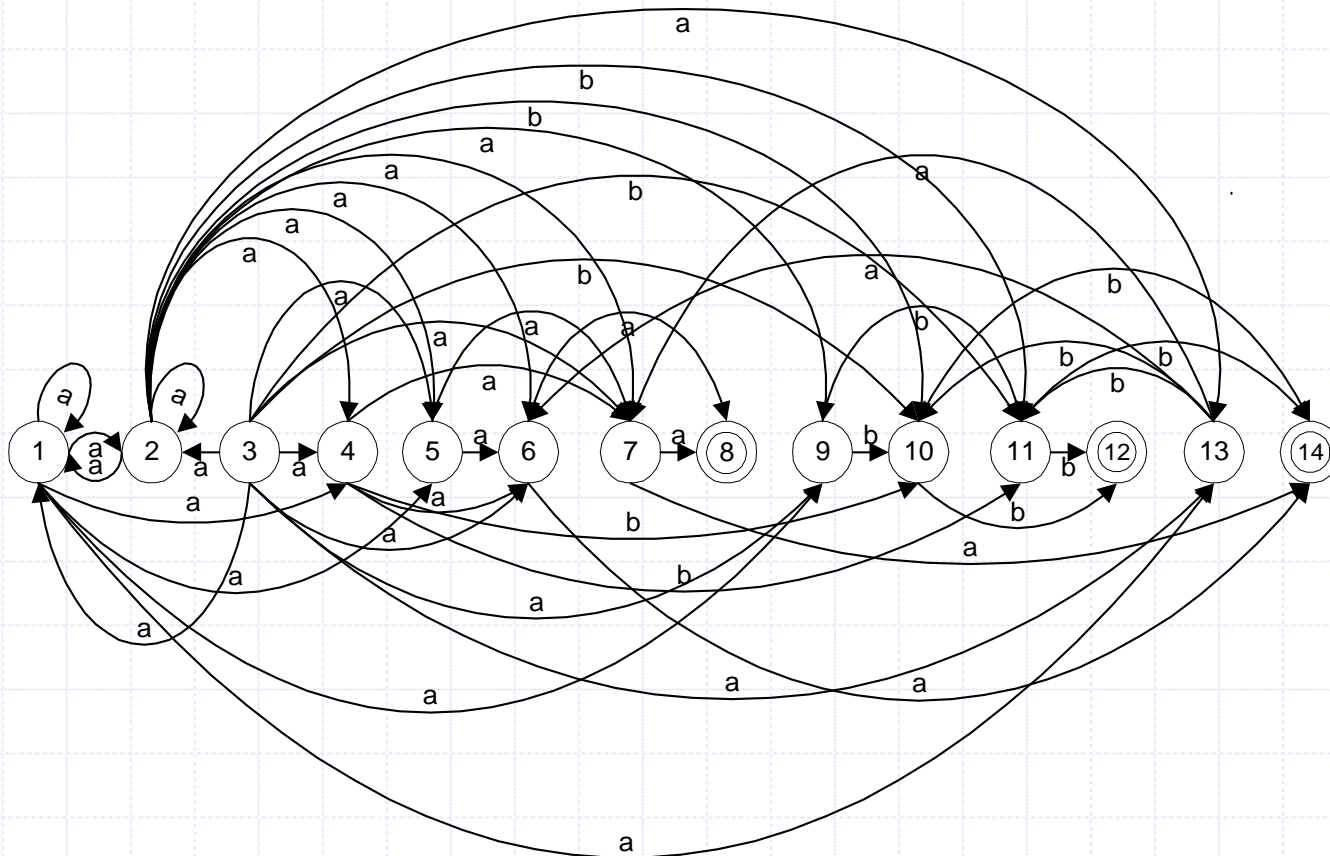
ANTERIOR





GERAÇÃO DO AFND A PARTIR DO AF ϵ

AFND RESULTANTE



ROTEIRO



SAIR

ANTERIOR





ALGORITMO DE [SIL2000]

FASES

ROTEIRO



SAIR

ANTERIOR





GERAÇÃO DA EXPRESSÃO PÓS-FIXADA

ROTEIRO

- ✓ A EXPRESSÃO PÓS-FIXADA É GERADA ATRAVÉS DE UMA DEFINIÇÃO LLC DESCRITA EM [SIL2000].

$a.(b+c)^*|c.a^*|^{\wedge}$

ER

$abc+*.ca^*.*+^{\wedge}+$

EXPRESSÃO PÓS-FIXADA

SAIR

ANTERIOR



EXPRESSÃO PÓS-FIXADA LLC DE [SIL2000]

ROTEIRO

\wedge → símbolo vazio
R → expressão regular
SR → simples expressão regular
RR → resto da expressão regular

R → SR RR;

RR → '+' SR | '.' SR | ^;

SR → T SR1;

SR1 → '+' T SR1 | ^; {EMITIR '+'}

T → F T1;

T1 → '.' F T1 | ^; {EMITIR '.'}

F → '(' R ')' F1 | #S F1; {EMITIR Símbolo S}

F1 → '*' F1 | ^; {EMITIR '*'}

SAIR

ANTERIOR





EXPRESSÃO PÓS-FIXADA

Geração pela LLC

ROTEIRO

✓ DADA A EXPRESSÃO REGULAR:

✓ $a.(b+c)^* + c.a^* + \wedge$

✓ PROCESSO: LER TODOS OS SÍMBOLOS DA EXPRESSÃO REGULAR DA ESQUERDA PARA A DIREITA E SUBMETÊ-LOS A DEFINIÇÃO EM LLC APRESENTADA POR [SIL2000]

SAIR

ANTERIOR

 **VEJA PROCESSO**



TABELA DO DESMONTE

ROTEIRO

- ✓ CONSISTE EM GERAR UMA TABELA A PARTIR DA EXPRESSÃO PÓS-FIXADA OBTIDA NA FASE 1
- ✓ A EXPRESSÃO PÓS-FIXADA É SUBMETIDA A UM CONJUNTO DE OPERAÇÕES QUE DARÃO ORIGEM A TABELA
- ✓ A EXPRESSÃO PÓS-FIXADA É PROCESSADA SÍMBOLO A SÍMBOLO DA DIREITA PARA A ESQUERDA. CADA SÍMBOLO É SUBMETIDO AO CONJUNTO DE PASSOS QUE FORMARÃO A TABELA DO DESMONTE.



SAIR

ANTERIOR





TABELA DO DESMONTE

ROTEIRO

✓ ESTADO INICIAL: 1

✓ ESTADO FINAL: 6

✓ $abc+*.ca*.*+^+$

✓ **1 PARA ESTADO INICIAL**

✓ **1 PARA ESTADO FINAL**

✓ **2 PARA O OPERADOR DE CONCATENAÇÃO**

✓ **2 PARA O OPERADOR DE FECHAMENTO**

SAIR

ANTERIOR





TABELA DO DESMONTE

abc+*.ca*.^+

ROTEIRO



SAIR

ANTERIOR

SAÍDA:	SÍMBOLO:	CHEGADA:
1	+	6
1	+	6
1	^	6
1	.	6
1	.	6
1	c	2
...

 **VEJA PROCESSO**

RONALD GLATZ



GRAFO DE TRANSIÇÕES (GT)

Geração a partir da FASE 2

ROTEIRO



SAIR

ANTERIOR

- ✓ É GERADO INICIALMENTE UMA NOVA TABELA A PARTIR DA TABELA DO DESMONTE
- ✓ NESTA NOVA TABELA ESTÃO ELIMINADAS TODAS AS LINHAS QUE CONTÉM OPERADORES DE UNIÃO, FECHAMENTO E CONCATENAÇÃO
- ✓ A NOVA TABELA RESULTA NO GRAFO DE TRANSIÇÕES.





GRAFO DE TRANSIÇÕES (GT)

Geração a partir da FASE 2

ROTEIRO



SAIR

ANTERIOR

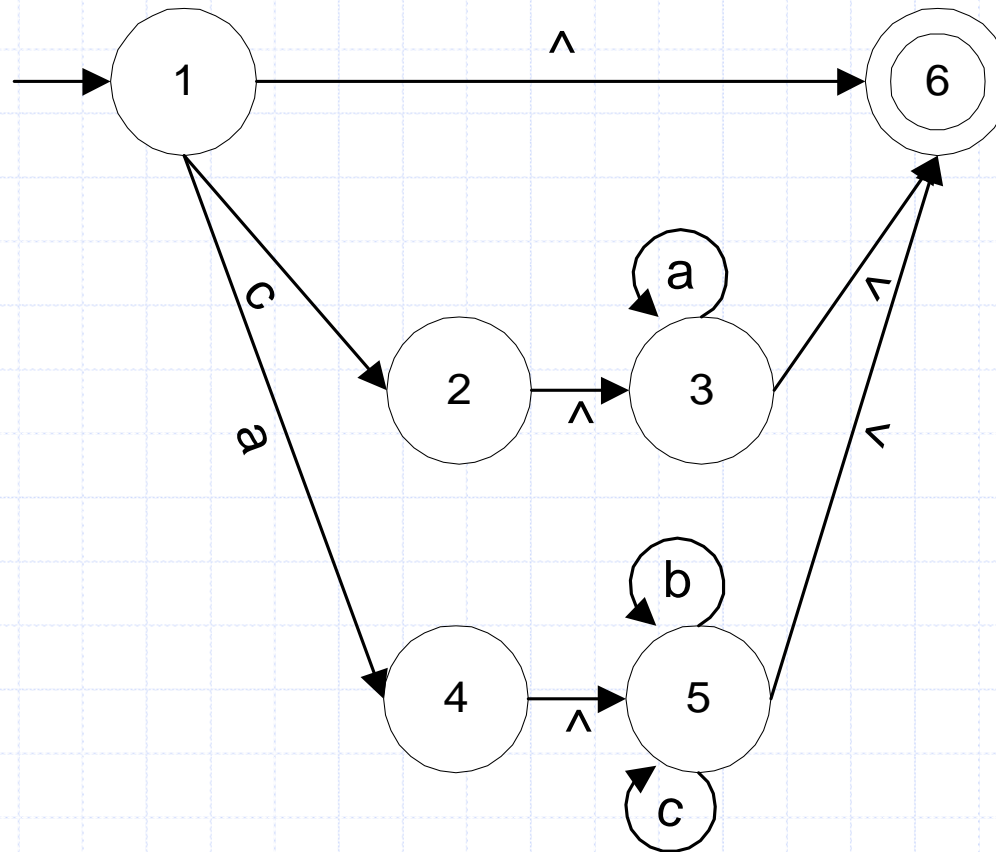
SAÍDA:	SÍMBOLO:	CHEGADA:
1	^	6
1	c	2
2	^	3
3	^	6
3	a	3
...





GRAFO DE TRANSIÇÕES (GT)

Geração a partir da FASE 2



ROTEIRO



SAIR

ANTERIOR





TABELA DE TRANSIÇÕES DO AFD

ROTEIRO

- ✓ CONSISTE NA GERAÇÃO DE UMA TABELA DE TRANSIÇÕES A PARTIR DO GT
- ✓ ESTADO INICIAL: ESTADOS INICIAIS DO GT E TODOS OS ESTADOS ATINGIDOS A PARTIR DOS ESTADOS INICIAIS PELO SÍMBOLO VAZIO (^)
- ✓ O PROCESSO DE GERAÇÃO DA TABELA DE TRANSIÇÕES A PARTIR DO GT É IDÊNTICO AO PROCESSO DE GERAÇÃO DA TABELA DE TRANSIÇÕES A PARTIR DO AFND NO ALGORITMO DE [HOP1979]
- ✓ O GT É UMA REPRESENTAÇÃO GRÁFICA DO MODELO MATEMÁTICO AUTÔMATO FINITO



SAIR

ANTERIOR





TABELA DE TRANSIÇÕES DO AFD

ROTEIRO



SAIR

ANTERIOR

- ✓ ESTADOS FINAIS: CONJUNTO DE ESTADOS FORMADOS A PARTIR DO GT E QUE CONTÉM ALGUM DOS ESTADOS FINAIS DO GT
- ✓ PROCESSAMENTO DOS SÍMBOLOS ALFABETO COM OS ESTADOS DO GT E SUAS TRANSIÇÕES DE ACORDO COM O ALGORITMO APRESENTADO POR [MAN1974]
- ✓ UMA EXPLICAÇÃO DETALHADA DOS PASSOS DO ALGORITMO DESCRITO POR [MAN1974] PODE SER ENCONTRADO NA MONOGRAFIA





TABELA DE TRANSIÇÕES DO AFD

ROTEIRO



SAIR

ANTERIOR

ESTADOS DO AFD	Mi	Mj		
		a	b	c
1 (-/+)	{1,6}	{4,5,6}	{}	{2,3,6}
2 (+)	{4,5,6}	{}	{5,6}	{5,6}
3 (+)	{2,3,6}	{3,6}	{}	{}
4 (+)	{5,6}	{}	{5,6}	{5,6}
5 (+)	{3,6}	{3,6}	{}	{}





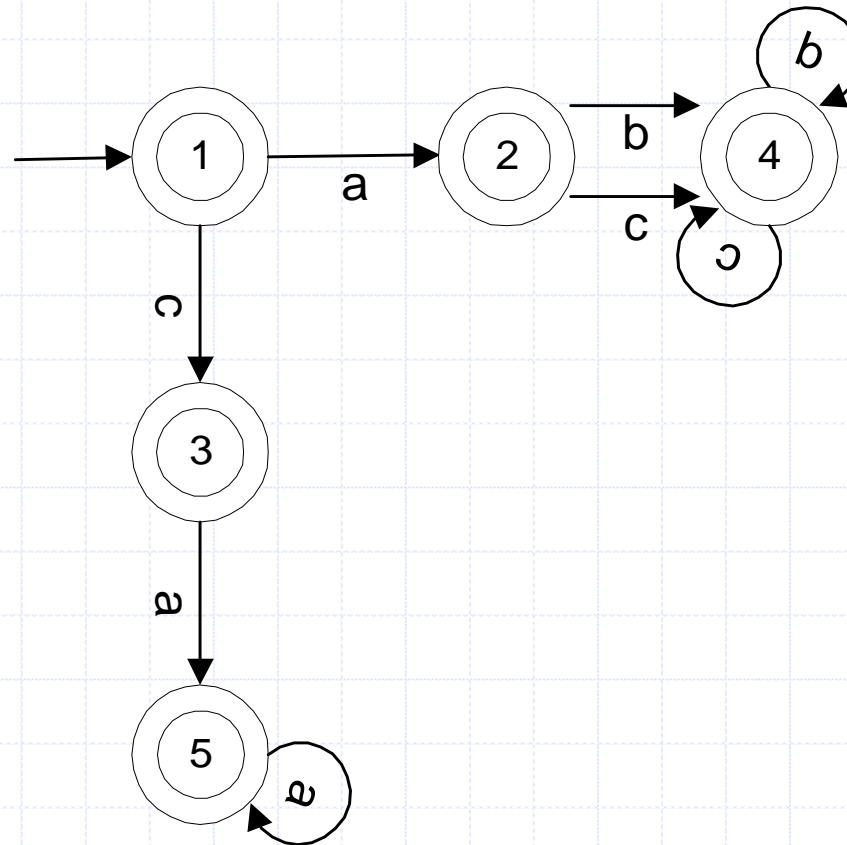
AFD OBTIDO A PARTIR DA TABELA DE TRANSIÇÕES

ROTEIRO



SAIR

ANTERIOR





TRANSFORMAÇÃO DE UM AFD EM PROGRAMA

ROTEIRO



SAIR

ANTERIOR

- ✓ CONSISTE EM TRANSFORMAR UM AFD EM UM PROGRAMA
- ✓ A IMPLEMENTAÇÃO DE UM AFD EM UM COMPUTADOR SEGUE DUAS LINHAS:
 - ✓ IMPLEMENTAÇÃO POR CÓDIGO DIRETO
 - ✓ CONTROLE DE TABELA DE TRANSIÇÕES





TRANSFORMAÇÃO DE UM AFD EM PROGRAMA

ROTEIRO



SAIR

ANTERIOR

- ✓ IMPLEMENTAÇÃO POR CÓDIGO DIRETO: cada estado e suas transições são implementadas diretamente no programa
- ✓ CONTROLE DE TABELA DE TRANSIÇÕES: um interpretador universal interpreta a Tabela de Transições do AFD

 VEJA O INTERPRETADOR UNIVERSAL DE TABELAS



TRANSFORMAÇÃO DE UM AFD EM PROGRAMA

ROTEIRO



SAIR

ANTERIOR

- ✓ Os algoritmos vistos neste trabalho realizam a transformação de uma ER em um AFD. Como parte complementar, foi implementada a transformação do AFD resultante em uma função na linguagem de programação Pascal. Esta transformação é equivalente nos dois protótipos que implementam os algoritmos.
- ✓ A função é gerada para uma *unit* da linguagem de programação Pascal e esta *unit* poderá ser utilizada em um programa desta linguagem

 **EXEMPLO DE PROGRAMA QUE DECLARA FUNÇÃO**



ESPECIFICAÇÃO DOS PROTÓTIPOS

ROTEIRO

 PROTÓTIPO DE [HOP1979]

 PROTÓTIPO DE [SIL2000]

✓ OS ALGORITMOS FORAM
IMPLEMENTADOS SEPARADAMENTE
COM O OBJETIVO DE SIMPLIFICAR O
ENTENDIMENTO DE CADA UM.

SAIR

ANTERIOR





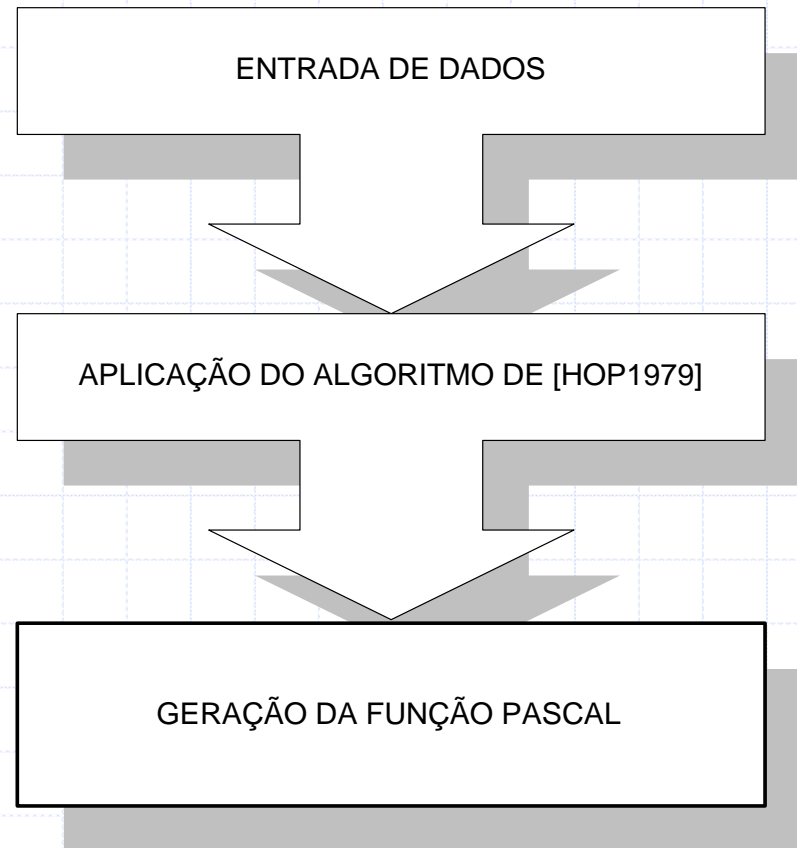
ESPECIFICAÇÃO DOS PROTÓTIPOS – HOP1979

ROTEIRO



SAIR

ANTERIOR





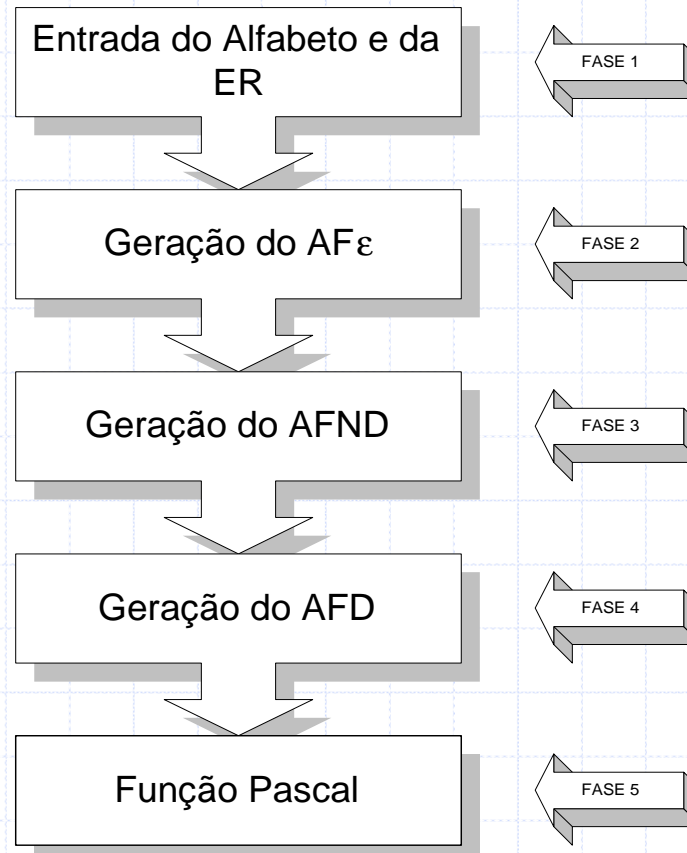
ESPECIFICAÇÃO DOS PROTÓTIPOS – HOP1979

ROTEIRO



SAIR

ANTERIOR





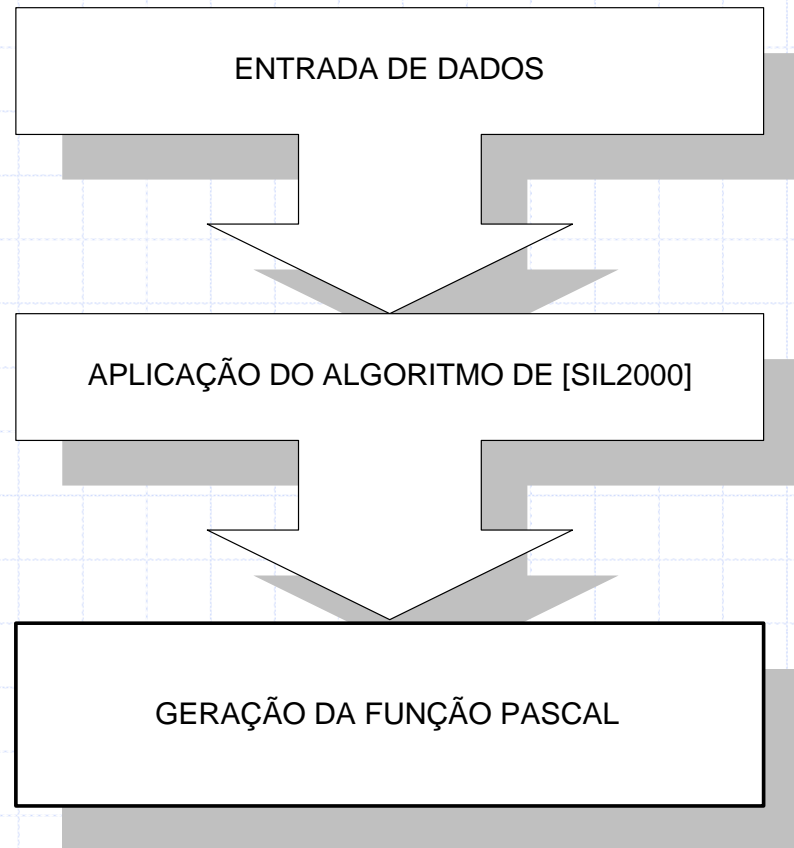
ESPECIFICAÇÃO DOS PROTÓTIPOS – SIL2000

ROTEIRO



SAIR

ANTERIOR





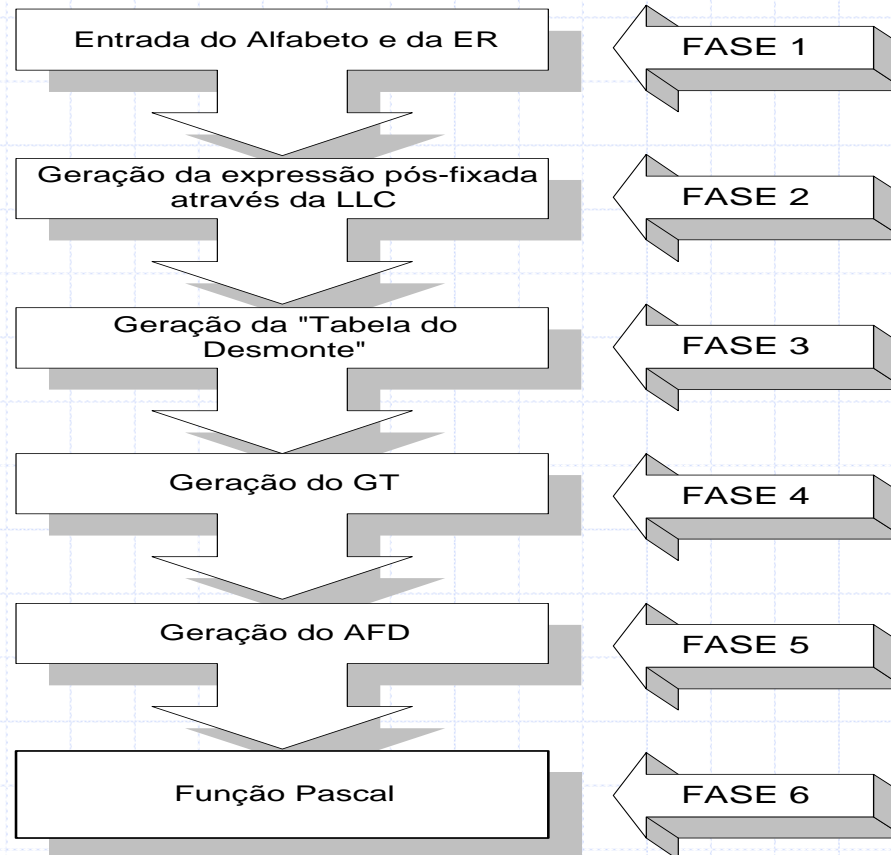
ESPECIFICAÇÃO DOS PROTÓTIPOS SIL2000

ROTEIRO



SAIR

ANTERIOR





APRESENTAÇÃO DOS PROTÓTIPOS TELA PRINCIPAL ([HOP1979])

ROTEIRO



SAIR

ANTERIOR

Hopcroft

Alfabeto:

Expressão Regular:

Processa Expressão Regular

Visualizar Função Pascal

Salvar AFD

Autômato Finito Não-Determinístico:

Transições:	Saída:	Símbolo:	Chegada:

Estado Inicial: Estados Finais:

Autômato Finito com Movimentos Vazios:

Transições:	Saída:	Símbolo:	Chegada:

Estado Inicial: Estado Final:

Autômato Finito Determinístico:

Estados:

Estado Inicial: Estados Finais:





APRESENTAÇÃO DOS PROTÓTIPOS ENTRADA DE DADOS

ROTEIRO

◀

▶

◀

▶

SAIR

ANTERIOR

Hopcroft

Alfabeto:

Expressão Regular:

Processa Expressão Regular

Visualizar Função Pascal

Salvar AFD

Autômato Finito Não-Determinístico:

Transições:	Saída:	Símbolo:	Chegada:

Estado Inicial: Estados Finais:

Autômato Finito com Movimentos Vazios:

Transições:	Saída:	Símbolo:	Chegada:

Estado Inicial: Estado Final:

Autômato Finito Determinístico:

Estados:

Estado Inicial: Estados Finais:

Dê um clique-duplo para ver o processo de obtenção do AFND !





APRESENTAÇÃO DOS PROTÓTIPOS PROCESSAR EXPRESSÃO REGULAR

ROTEIRO



SAIR

ANTERIOR

The screenshot shows the Hopcroft software interface. The main window is titled 'Hopcroft' and contains several sections:

- Alfabeto:** Input field containing 'abc'.
- Expressão Regular:** Input field containing 'a(blc)*|ca*|^'.
- Buttons:** 'Processa Expressão Regular' (circled in red), 'Visualizar Função Pascal', and 'Salvar AFD'.
- Autômato Finito Não-Determinístico:** A table with columns 'Transições:', 'Saída:', 'Símbolo:', and 'Chegada:'. The table contains 4 rows of data. Below the table are input fields for 'Estado Inicial:' (21) and 'Estados Finais:' ({22,2,4,6,8,9,10,12,14,1}).
- Autômato Finito com Movimentos Vazios:** A table with columns 'Transições:', 'Saída:', 'Símbolo:', and 'Chegada:'. The table contains 4 rows of data. Below the table are input fields for 'Estado Inicial:' (21) and 'Estado Final:' ({22}).
- Autômato Finito Determinístico:** A table with columns 'Estados:', 'a', 'b', and 'c'. The table contains 4 rows of data. Below the table are input fields for 'Estado Inicial:' (1) and 'Estados Finais:' ({1,2,3,4,5,6}).



APRESENTAÇÃO DOS PROTÓTIPOS VISUALIZAR FUNÇÃO

ROTEIRO



SAIR

ANTERIOR

Hopcroft

Alfabeto: abc
Expressão Regular: a(blc)*lca*l^

Processa Expressão Regular

Visualizar Função Pascal

Salvar AFD

Autômato Finito com Movimentos Vazios:

Transições:	Saída:	Símbolo:	Chegada:
1	1	a	2
2	3	b	4
3	5	c	6
4	7	^	5

Estado Inicial: 21 Estado Final: {22}

Autômato Finito Não-Determinístico:

Transições:	Saída:	Símbolo:	Chegada:
1	1	a	2
2	1	a	9
3	1	a	7
4	1	a	5

Estado Inicial: 21 Estados Finais: {22,2,4,6,8,9,10,12,14,1}

Autômato Finito Determinístico:

Estados:	a	b	c
1 (Final)	2	0	3
2 (Final)	0	4	5
3 (Final)	6	0	0
4 (Final)	0	4	5

Estado Inicial: 1 Estados Finais: {1,2,3,4,5,6}





APRESENTAÇÃO DOS PROTÓTIPOS FUNÇÃO PASCAL

ROTEIRO



SAIR

ANTERIOR

```
Função Pascal
[Salvar] Numerar Linhas?

{ TCC - RONALD GLATZ }
unit TCC;

interface

function ValidaER(Cadeia : string) : boolean;

implementation

function ValidaER(Cadeia : string) : boolean;
{ Estados do AFD }
label S0,S1,S2,S3,S4,S5,S6,S7,S8,S9;
var
  I : integer;
begin
  S0: I := 0;
  S1: inc(I);
  if length(Cadeia) < I then
    goto S7;
  if Cadeia[I] = 'a' then
    goto S2;
  if Cadeia[I] = 'c' then
```





APRESENTAÇÃO DOS PROTÓTIPOS

AF ϵ

ROTEIRO

⏪

⏩

⏴

⏵

SAIR

⏴

Hopcroft

Alfabeto: abc

Expressão Regular: a(blc)*lca*l^

Processa Expressão Regular

Visualizar Função Pascal

Salvar AFD

Autômato Finito Não-Determinístico:

Transições:	Saída:	Símbolo:	Chegada:
1	1	a	2
2	1	a	9
3	1	a	7
4	1	a	5

Estado Inicial: 21

Estados Finais: {22,2,4,6,8,9,10,12,14,1}

Autômato Finito com Movimentos Vazios:

Transições:	Saída:	Símbolo:	Chegada:
1	1	a	2
2	3	b	4
3	5	c	6
4	7	^	5

Estado Inicial: 21

Estado Final: {22}

Autômato Finito Determinístico:

Estados:	a	b	c
1 (Final)	2	0	3
2 (Final)	0	4	5
3 (Final)	6	0	0
4 (Final)	0	4	5

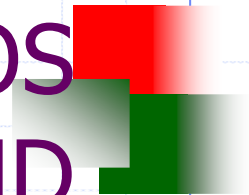
Estado Inicial: 1

Estados Finais: {1,2,3,4,5,6}





APRESENTAÇÃO DOS PROTÓTIPOS AFND



ROTEIRO

⏪

⏩

⏴

SAIR

ANTERIOR

Hopcroft

Alfabeto: abc

Expressão Regular: a(b|c)*|ca*|^

Processa Expressão Regular

Visualizar Função Pascal

Salvar AFD

Autômato Finito com Movimentos Vazios:

Transições:	Saída:	Símbolo:	Chegada:
1	1	a	2
2	3	b	4
3	5	c	6
4	7	^	5

Estado Inicial: 21 Estado Final: {22}

Autômato Finito Não-Determinístico:

Transições:	Saída:	Símbolo:	Chegada:
1	1	a	2
2	1	a	9
3	1	a	7
4	1	a	5

Estado Inicial: 21 Estados Finais: {22,2,4,6,8,9,10,12,14,1}

Autômato Finito Determinístico:

Estados:	a	b	c
1 (Final)	2	0	3
2 (Final)	0	4	5
3 (Final)	6	0	0
4 (Final)	0	4	5

Estado Inicial: 1 Estados Finais: {1,2,3,4,5,6}

Dê um clique-duplo para ver o processo de obtenção do AFND !





APRESENTAÇÃO DOS PROTÓTIPOS AFND – FASES DA δ

ROTEIRO



SAIR

ANTERIOR

Automato Finito com Movimentos Vazios para AFND

Estados Atingidos com a Palavra Vazia (^) (FASE 1)

Estado atinge com a Palavra Vaz
1	{1}
2	{2,3}
3	{3}
4	{4,5}
5	{5}
6	{6}

Estados Atingidos com um Símbolo do Alfabeto (FASE 2)

Saida:	Símbolo	Chegada
1	A	2
2	B	4
3	B	4
4	C	6
5	C	6

Alfabeto: Expressão Regular:





APRESENTAÇÃO DOS PROTÓTIPOS

AFD

ROTEIRO

⏪

⏩

⏴

⏵

SAIR

ANTERIOR

Hopcroft

Alfabeto: ABC

Expressão Regular: ABC

Processa Expressão Regular

Visualizar Função Pascal

Salvar AFD

Autômato Finito Não-Determinístico:

Transições:	Saída:	Símbolo:	Chegada:
1	1	A	2
2	1	A	3
3	2	B	4
4	2	B	5

Estado Inicial: 1 Estados Finais: {6}

Autômato Finito com Movimentos Vazios:

Transições:	Saída:	Símbolo:	Chegada:
1	1	A	2
2	3	B	4
3	2	^	3
4	5	C	6

Estado Inicial: 1 Estado Final: {6}

Autômato Finito Determinístico:

Estados:	A	B	C
1	2	0	0
2	0	3	0
3	0	0	4
4 (Final)	0	0	0

Estado Inicial: 1 Estados Finais: {4}

Dê um clique-duplo para ver a Tabela de Transições!





APRESENTAÇÃO DOS PROTÓTIPOS TABELA DE TRANSIÇÕES DO AFD

ROTEIRO



SAIR

ANTERIOR

Coluna Mi:	A	B	C
{1}	{2,3}	-	-
{2,3}	-	{4,5}	-
{4,5}	-	-	{6}
{6}	-	-	-





APRESENTAÇÃO DOS PROTÓTIPOS TELA PRINCIPAL ([SIL2000])

ROTEIRO



SAIR

ANTERIOR

Protótipo do Algoritmo de SIL2000

Alfabeto:

Expressão Regular:

Expressão Pós-Fixada:

Processa Expressão Regular

Visualizar Função Pascal

Salvar AFD

Tabela do "Desmonte"

Grafo de Transições:

Estado Inicial: Estado Final:

Autômato Finito Determinístico:

Estado Inicial: Estados Finais:





APRESENTAÇÃO DOS PROTÓTIPOS ENTRADA DE DADOS

ROTEIRO



SAIR

ANTERIOR

Protótipo do Algoritmo de SIL2000

Alfabeto:

Expressão Regular:

Expressão Pós-Fixada:

Tabela do "Desmonte"

Grafo de Transições:

Estado Inicial:

Estado Final:

Autômato Finito Determinístico:

Estado Inicial:

Estados Finais:





APRESENTAÇÃO DOS PROTÓTIPOS PROCESSAR EXPRESSÃO

ROTEIRO



SAIR

ANTERIOR

Protótipo do Algoritmo de SIL2000

Alfabeto:

Expressão Regular:

Expressão Pós-Fixada:

Processa Expressão Regular

Visualizar Função Pascal

Salvar AFD

Grafo de Transições:

Transições:	Saída:	Simbolo:	Chegada:
1	2	c	4
2	1	a	3
3	3	b	2

Estado Inicial: Estado Final:

Tabela do "Desmonte"

Transições:	Saída:	Simbolo:	Chegada:
1	1	&	4
2	1	&	2
3	2	c	4
4	1	a	3
5	3	b	2

Autômato Finito Determinístico:

Estados:	a	b	c
1	2	0	0
2	0	3	0
3	0	0	4

Estado Inicial: Estados Finais:





APRESENTAÇÃO DOS PROTÓTIPOS EXPRESSÃO PÓS-FIXADA

ROTEIRO



SAIR

ANTERIOR

Protótipo do Algoritmo de SIL2000

Alfabeto:

Expressão Regular:

Expressão Pós-Fixada:

Grafo de Transições:

Transições:	Saída:	Simbolo:	Chegada:
1	2	c	4
2	1	a	3
3	3	b	2

Estado Inicial: Estado Final:

Tabela do "Desmonte"

Transições:	Saída:	Simbolo:	Chegada:
1	1	&	4
2	1	&	2
3	2	c	4
4	1	a	3
5	3	b	2

Autômato Finito Determinístico:

Estados:	a	b	c
1	2	0	0
2	0	3	0
3	0	0	4

Estado Inicial: Estados Finais:





APRESENTAÇÃO DOS PROTÓTIPOS VISUALIZAR FUNÇÃO GERADA

ROTEIRO



SAIR

ANTERIOR

Protótipo do Algoritmo de SIL2000

Alfabeto:

Expressão Regular:

Expressão Pós-Fixada:

Grafo de Transições:

Transições:	Saida:	Simbolo:	Chegada:
1	2	c	4
2	1	a	3
3	3	b	2

Estado Inicial: Estado Final:

Tabela do "Desmonte"

Transições:	Saída:	Simbolo:	Chegada:
1	1	&	4
2	1	&	2
3	2	c	4
4	1	a	3
5	3	b	2

Autômato Finito Determinístico:

Estados:	a	b	c
1	2	0	0
2	0	3	0
3	0	0	4

Estado Inicial: Estados Finais:





APRESENTAÇÃO DOS PROTÓTIPOS FUNÇÃO PASCAL

ROTEIRO



SAIR

ANTERIOR

```
Função Pascal
[Salvar] Numerar Linhas?
{ TCC - RONALD GLATZ }
unit TCC;

interface

function ValidaER(Cadeia : string) : boolean;

implementation
I

function ValidaER(Cadeia : string) : boolean;
{ Estados do AFD }
label S0,S1,S2,S3,S4,S5,S6,S7;
var
I : integer;
begin
S0: I := 0;
S1: inc(I);
if length(Cadeia) < I then
goto S6;
if Cadeia[I] = 'a' then
goto S2;
goto S6;
```





APRESENTAÇÃO DOS PROTÓTIPOS

TABELA DO DESMONTE

ROTEIRO



SAIR

ANTERIOR

Protótipo do Algoritmo de SIL2000

Alfabeto:

Expressão Regular:

Expressão Pós-Fixada:

Processa Expressão Regular

Visualizar Função Pascal

Salvar AFD

Tabela do "Desmonte"

Transições:	Saída:	Simbolo:	Chegada:
1	1	&	4
2	1	&	2
3	2	c	4
4	1	a	3
5	3	b	2

Grafo de Transições:

Transições:	Saída:	Simbolo:	Chegada:
1	2	c	4
2	1	a	3
3	3	b	2

Estado Inicial: Estado Final:

Autômato Finito Determinístico:

Estados:	a	b	c
1	2	0	0
2	0	3	0
3	0	0	4

Estado Inicial: Estados Finais:





APRESENTAÇÃO DOS PROTÓTIPOS GT

ROTEIRO



SAIR

ANTERIOR

Protótipo do Algoritmo de SIL2000

Alfabeto:

Expressão Regular:

Expressão Pós-Fixada:

Processa Expressão Regular

Visualizar Função Pascal

Salvar AFD

Tabela do "Desmonte"

Transições:	Saída:	Simbolo:	Chegada:
1	1	&	4
2	1	&	2
3	2	c	4
4	1	a	3
5	3	b	2

Grafo de Transições:

Transições:	Saída:	Simbolo:	Chegada:
1	2	c	4
2	1	a	3
3	3	b	2

Estado Inicial: Estado Final:

Autômato Finito Determinístico:

Estados:	a	b	c
1	2	0	0
2	0	3	0
3	0	0	4

Estado Inicial: Estados Finais:





APRESENTAÇÃO DOS PROTÓTIPOS

AFD

ROTEIRO

⏪

⏩

⏴

⏵

SAIR

ANTERIOR

Protótipo do Algoritmo de SIL2000

Alfabeto:

Expressão Regular:

Expressão Pós-Fixada:

Processa Expressão Regular

Visualizar Função Pascal

Salvar AFD

Tabela do "Desmonte"

Transições:	Saída:	Simbolo:	Chegada:
1	1	&	4
2	1	&	2
3	2	c	4
4	1	a	3
5	3	b	2

Grafo de Transições:

Transições:	Saída:	Simbolo:	Chegada:
1	2	c	4
2	1	a	3
3	3	b	2

Estado Inicial: Estado Final:

Autômato Finito Determinístico:

Estados:	a	b	c
1	2	0	0
2	3	0	0
3	0	0	4

Estado Inicial: Estados Finais:

Dê um clique-duplo para ver a Tabela de Transi...





APRESENTAÇÃO DOS PROTÓTIPOS TABELA DE TRANSIÇÕES DO AFD

ROTEIRO



SAIR

ANTERIOR

Coluna Mi:	a	b	c
{1}	{3}	-	-
{3}	-	{2}	-
{2}	-	-	{4}
{4}	-	-	-





COMPARAÇÃO DOS ALGORITMOS

ROTEIRO

✓ CRITÉRIOS UTILIZADOS:

- ✓ TEMPO DE EXECUÇÃO (% DE EFICIÊNCIA DE UM ALGORITMO EM RELAÇÃO AO OUTRO)
- ✓ UTILIZAÇÃO DE MEMÓRIA (% DE EFICIÊNCIA DE UM ALGORITMO EM RELAÇÃO AO OUTRO)
- ✓ OTIMIZAÇÃO DE RESULTADOS



SAIR

ANTERIOR





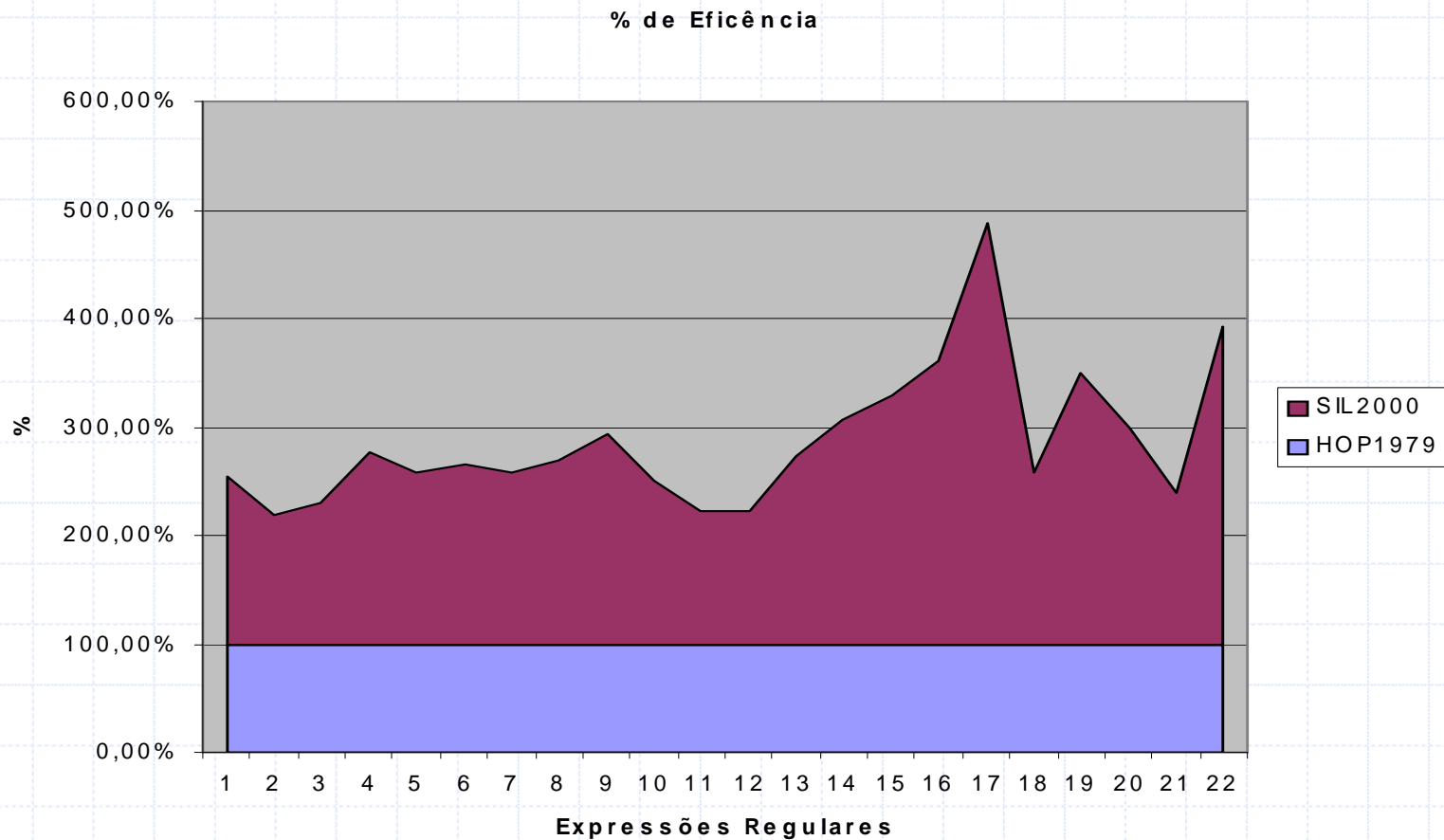
COMPARAÇÃO DOS ALGORITMOS TEMPO DE EXECUÇÃO

ROTEIRO



SAIR

ANTERIOR





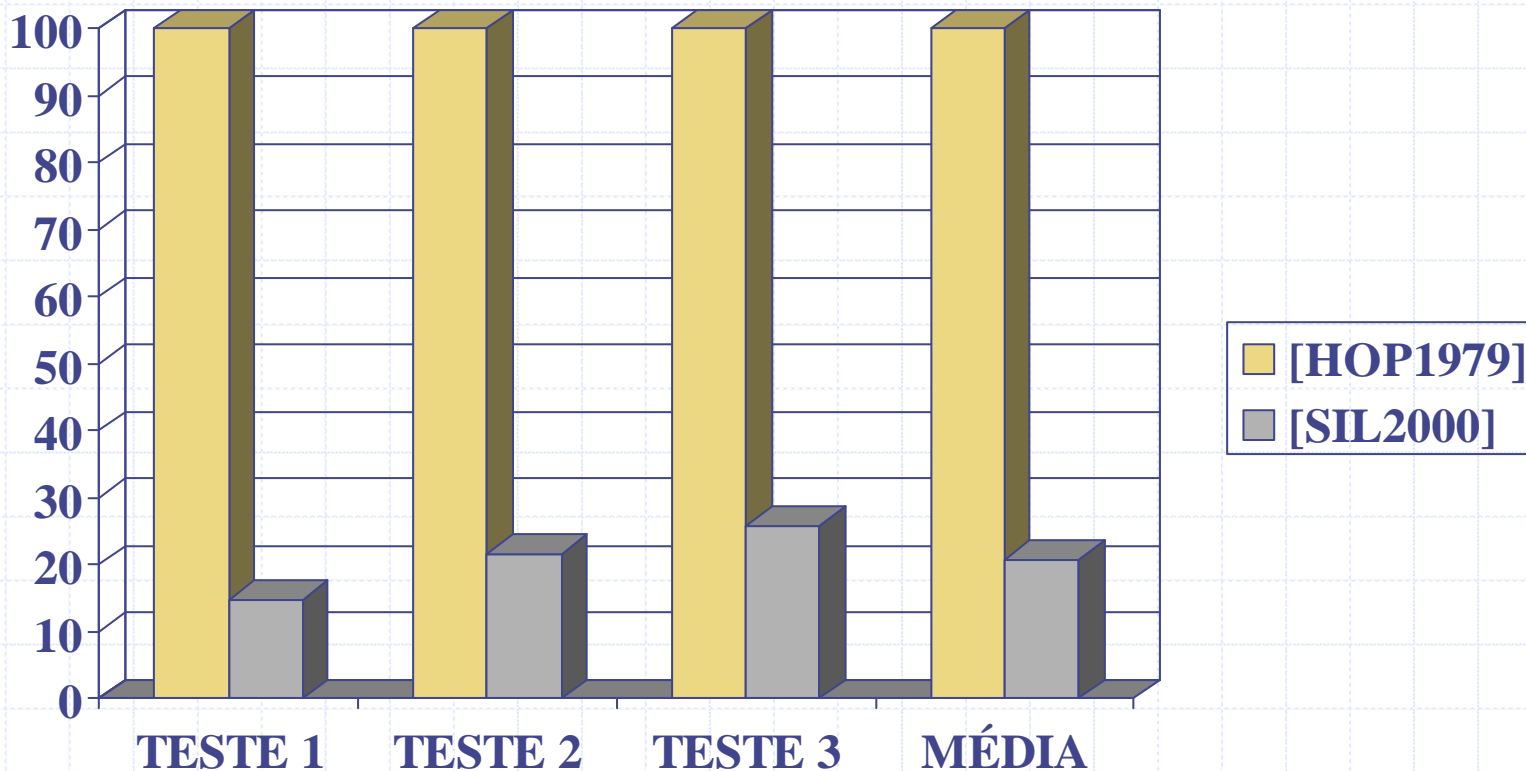
COMPARAÇÃO DOS ALGORITMOS UTILIZAÇÃO DE MEMÓRIA

ROTEIRO



SAIR

ANTERIOR





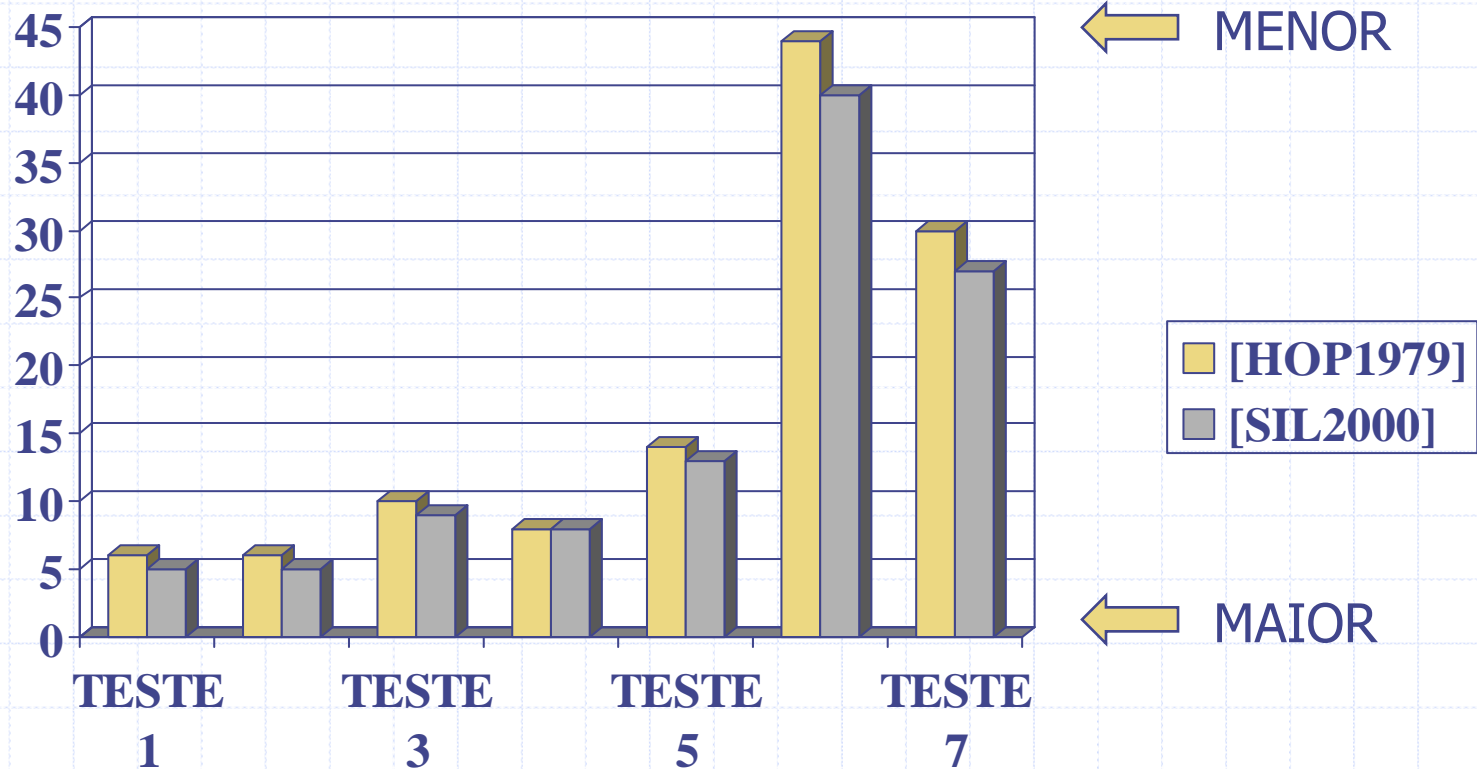
COMPARAÇÃO DOS ALGORITMOS OTIMIZAÇÃO DE RESULTADOS

ROTEIRO



SAIR

ANTERIOR





COMPARAÇÃO DOS ALGORITMOS

ROTEIRO



SAIR

ANTERIOR

- ✓ Para o critério TEMPO DE EXECUÇÃO, o algoritmo de [SIL2000] mostrou-se amplamente superior ao algoritmo de [HOP1979]
- ✓ Idem para o critério UTILIZAÇÃO DE MEMÓRIA
- ✓ No critério OTIMIZAÇÃO DE RESULTADOS foi utilizada a contagem dos estados do AFD resultante. Na quase totalidade dos testes o algoritmo de [SIL2000] se mostrou superior.





CONCLUSÃO

ROTEIRO



SAIR

ANTERIOR

- ✓ Durante este trabalho foram vistos alguns conceitos e técnicas para transformação de uma ER em AFD
- ✓ Também foram vistos os algoritmos de [HOP1979] e [SIL2000]
- ✓ Foram feitas considerações sobre cada algoritmo e comparações entre eles

