E-fólio A | Folha de resolução para E-fólio

UNIDADE CURRICULAR: Linguagens e Computação

CÓDIGO: 21078

DOCENTES: Jorge Morais e Rúdi Gualter (tutor)

A preencher pelo estudante

NOME: Marcelo Dinis Bregieira

N.º DE ESTUDANTE: 2201083

CURSO: Licenciatura em Engenharia Informática

DATA DE ENTREGA: 29/11/2023

TRABALHO / RESOLUÇÃO:

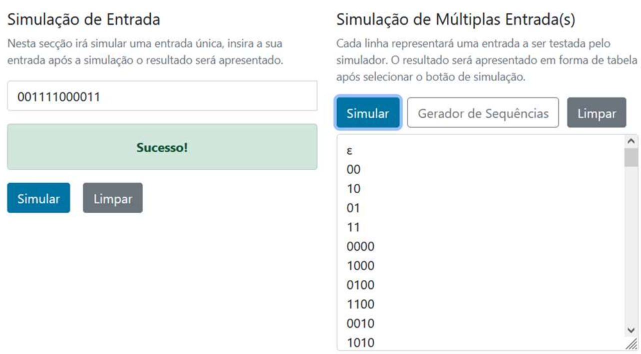
1. Para a construção desta expressão regular <u>usei **1 (um)** como sendo a retirada de uma **bola azul** e **0 (zero)** como sendo a retirada de uma **bola vermelha**</u>, assim sendo temos o alfabeto: Σ = {1, 0}.

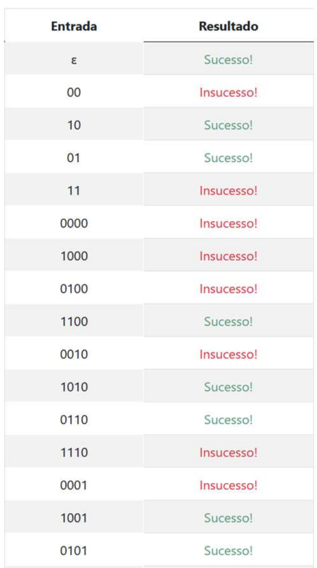
   Para este caso usei as seguintes condições:

   - Pode ser aceite string vazia, uma vez que ter zero bolas é igual a ter zero bolas azuis e zero vermelhas;
   - A string não pode começar por "000" ou "111";
   - Por cada "1" tem de haver um "0" correspondente, visto que não pode haver um diferencial de 3, o mesmo algarismo pode ocorrer no máximo duas vezes e depois terá que ter outros dois diferentes seguidos tipo 1100 ou 0011;
   - Mas pode se dar o caso de ser junto também com outra "quadra" válida, exemplo "1100" e "0011", ou seja, temos quatro 0's seguidos, mas a expressão é valida.

Assim chegou-se à seguinte expressão:

ε|1(10)*0|0(01)*1|(ε|1(10)*0|0(01)*1)(1(10)*0|0(01)*1)*(ε|1(10)*0|0(01)*1)

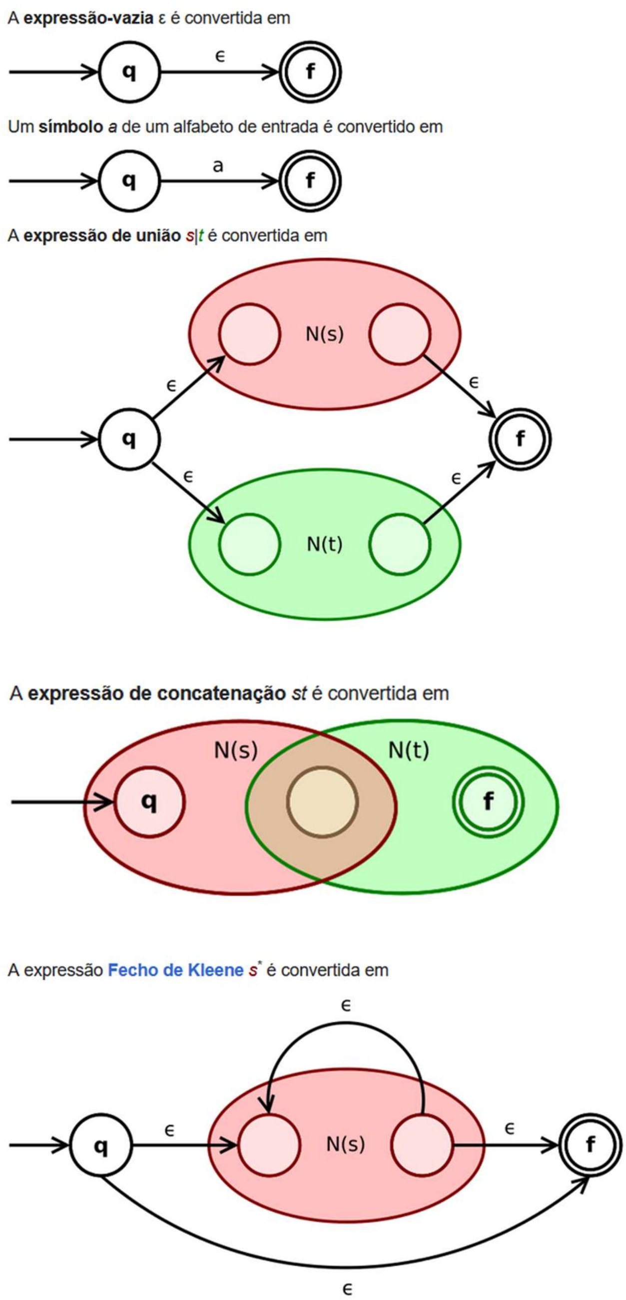




Entrada	Resultado
ε	Sucesso!
00	Insucesso!
10	Sucesso!
01	Sucesso!
11	Insucesso!
0000	Insucesso!
1000	Insucesso!
0100	Insucesso!
1100	Sucesso!
0010	Insucesso!
1010	Sucesso!
0110	Sucesso!
1110	Insucesso!
0001	Insucesso!
1001	Sucesso!
0101	Sucesso!

2. Para transformar a ER num autómato finito determinístico mínimo, temos primeiro transformar em NFA-ε depois em DFA e depois minimizar

Utilizaremos as regras do algoritmo Thompson, são elas:

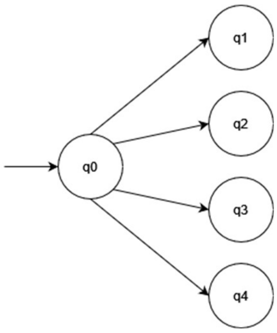


Para tal vamos dividir a expressão em partes e sub partes:

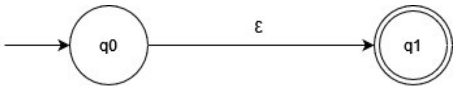
ε|1(10)*0|0(01)*1|(ε|1(10)*0|0(01)*1)(1(10)*0|0(01)*1)*(ε|1(10)*0|0(01)*1)

1. **ε**
2. **1(10)*0**
    a. (10)
    b. (10)*
    c. 1(10)*
    d. (10)*0
3. **0(01)*1**
    a. (01)
    b. (01)*
    c. 0(01)*
    d. (01)*1
4. **(ε|1(10)*0|0(01)*1)(1(10)*0|0(01)*1)*(ε|1(10)*0|0(01)*1)**
    a. **(ε|1(10)*0|0(01)*1)**
        i. ε – igual a 1.
        ii. 1(10)*0 – igual a 2.
        iii. 0(01)*1 – igual a 3.
    b. **(1(10)*0|0(01)*1)***
        i. 1(10)*0 – igual a 2.
        ii. 0(01)*1 – igual a 3.
        iii. (…)*
    c. **(ε|1(10)*0|0(01)*1)** – igual 4.a
        i. ε – igual a 1.
        ii. 1(10)*0 – igual a 2.
        iii. 0(01)*1 – igual a 3.

Como temos união de quatro expressões "principais" teremos quatro estados a "sair" do estado inicial e iniciaremos as expressões a partir daí



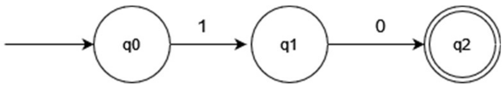
Expressão 1 – ε



Note que o “q0” corresponderá ao “q1” da imagem anterior.

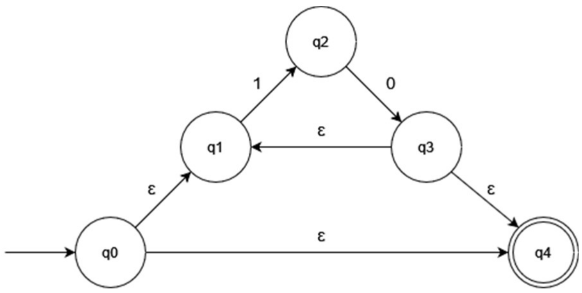
Expressão 2 – 1(10)*0

Subexpressão 2.a – (10)



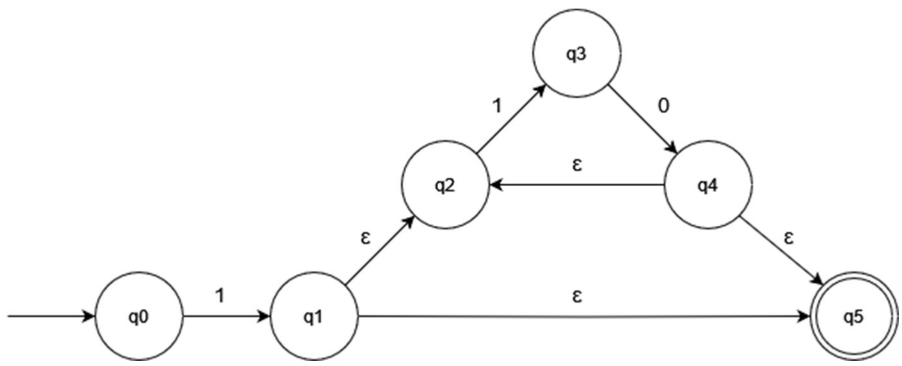
Subexpressão 2.b – (10)*

É apenas aplicado o fechamento de Kleene na expressão anterior, sendo que do estado inicial pode ter ε ou pode ter todas as combinações possíveis de 10.



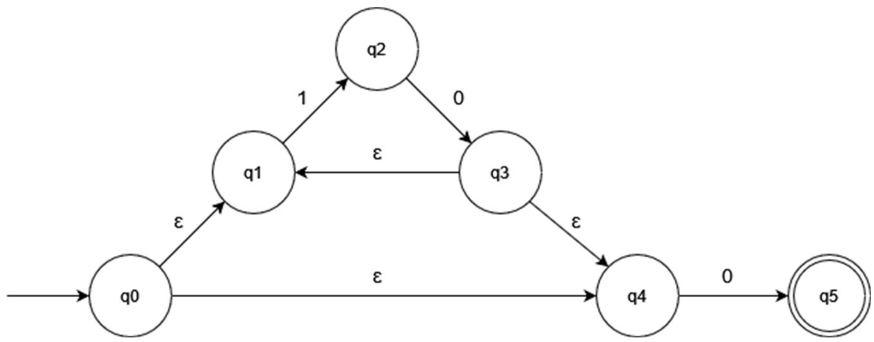
Subexpressão 2.c – 1(10)*

É aplicada a concatenação da expressão 1 com a expressão anterior

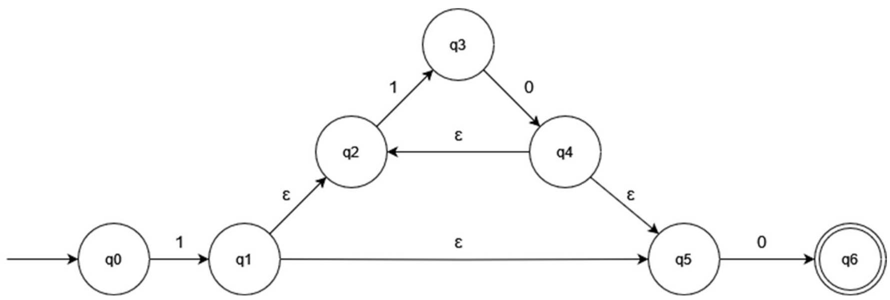


Subexpressão 2.d – (10)*0

É aplicada a concatenação da expressão 0 no final da expressão 2b "(10)*"

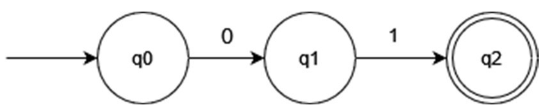


Expressão 2. final, juntando tudo ficamos com:



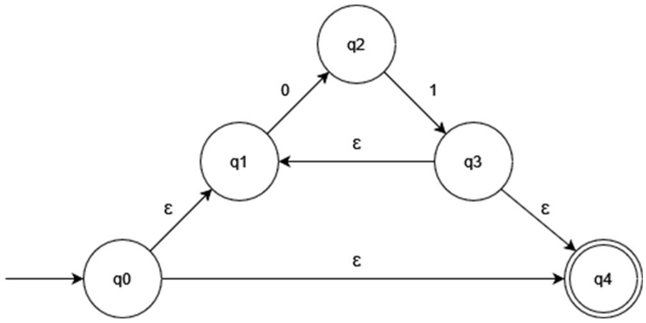
Expressão 3 – 0(01)*1

Subexpressão 3.a – (01)



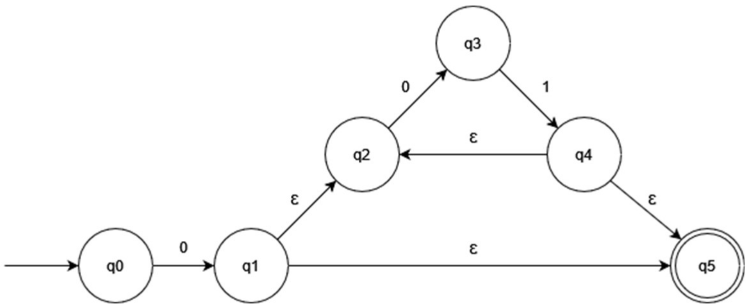
Subexpressão 3.b – (01)*

É apenas aplicado o fechamento de Kleene na expressão anterior, sendo que do estado inicial pode ter ε ou pode ter todas as combinações possíveis de 01.



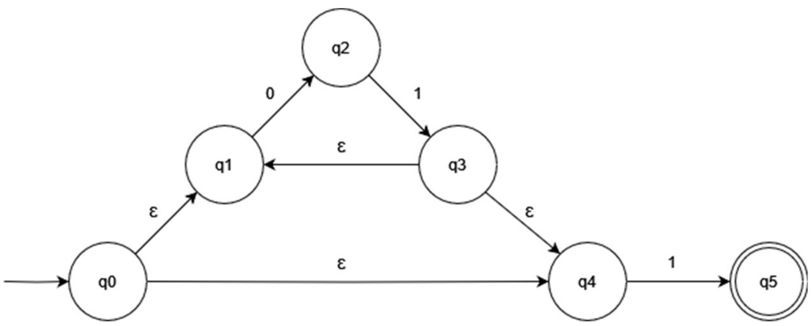
Subexpressão 3.c – 0(01)*

É aplicada a concatenação da expressão 0 com a expressão anterior

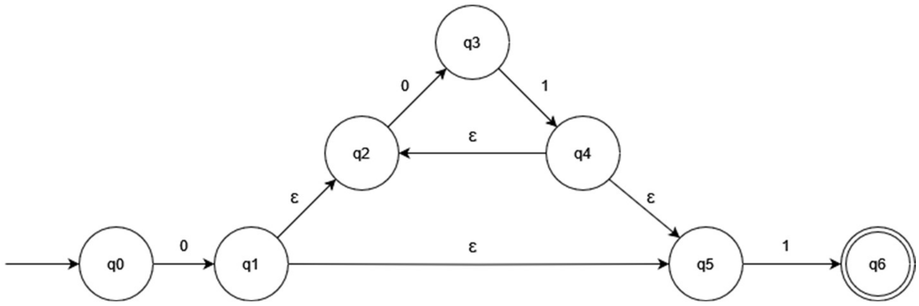


Subexpressão 3.d – (01)*1

É aplicada a concatenação da expressão 1 no final da expressão 3b "(01)*)"



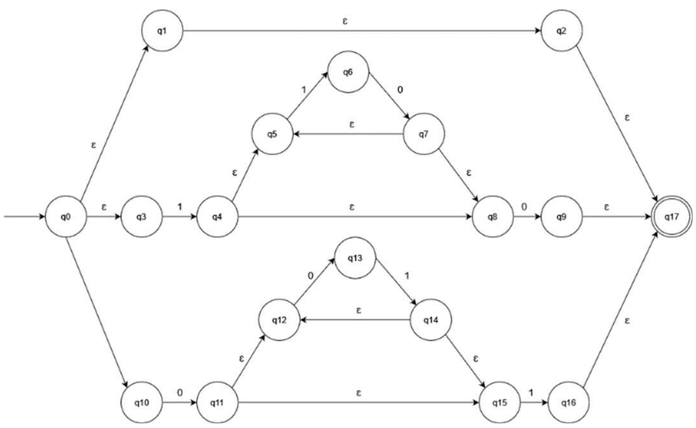
Expressão 3. final, juntando tudo ficamos com:



Expressão 4 – (ε|1(10)*0|0(01)*1)(1(10)*0|0(01)*1)*(ε|1(10)*0|0(01)*1)

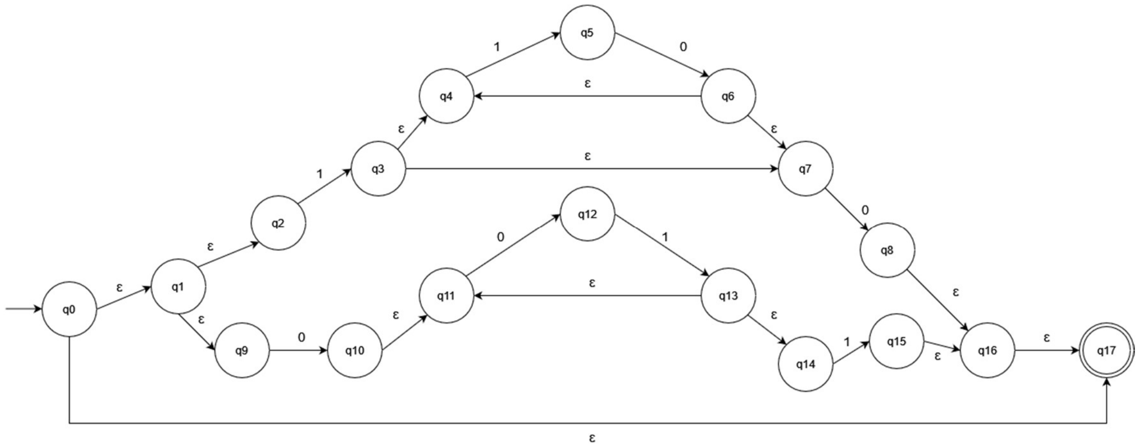
Subexpressão 4.a – (ε|1(10)*0|0(01)*1)

Esta subexpressão é gerada por uniões de subexpressões já feitas anteriormente pelo que apenas temos que "montar" a expressão final, o que fica:



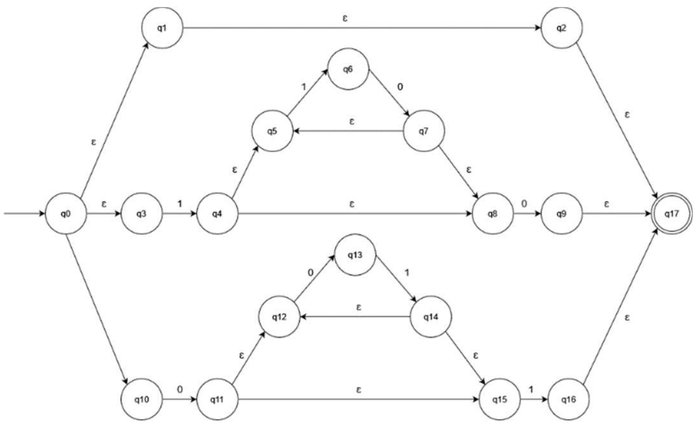
Subexpressão 4.b – (1(10)*0|0(01)*1)*

Nesta subexpressão não temos a união de ε, como na expressão anterior, e temos o fechamento de Kleene a toda a expressão o que fica:

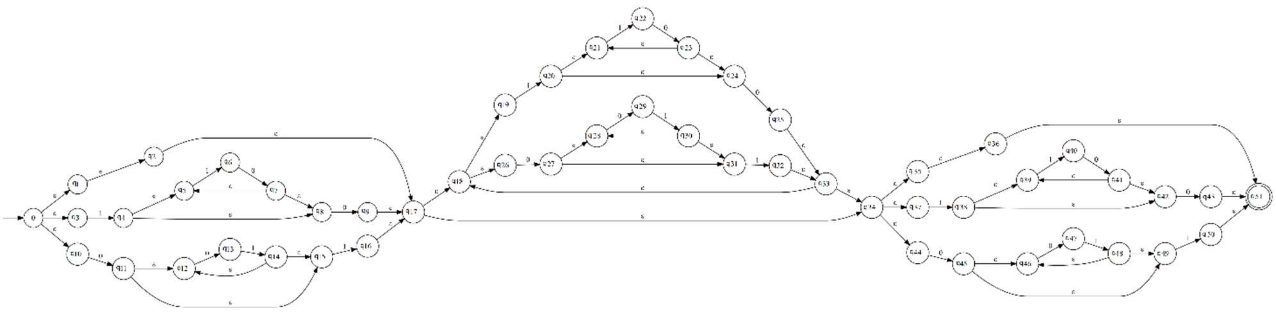


Subexpressão 4.c – (ε|1(10)*0|0(01)*1)

Esta subexpressão é igual a 4.a:

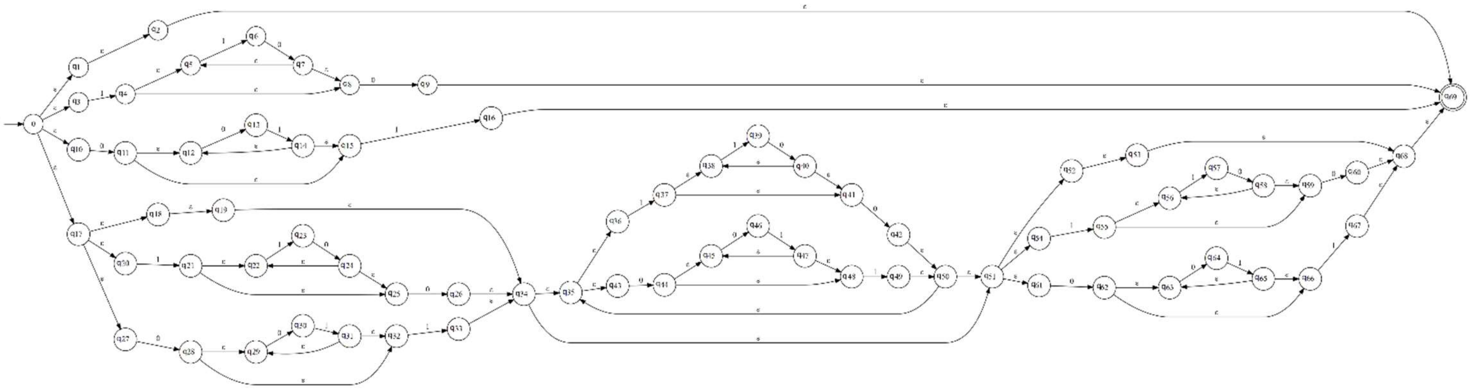


Expressão 4. final, juntando tudo ficamos com:



Expressão final – **ε|1(10)*0|0(01)*1|(ε|1(10)*0|0(01)*1)(1(10)*0|0(01)*1)*(ε|1(10)*0|0(01)*1)**

Para a expressão final temos a união das quatro expressões anteriores o que fica em:



Conversão do NFA-ε em DFA

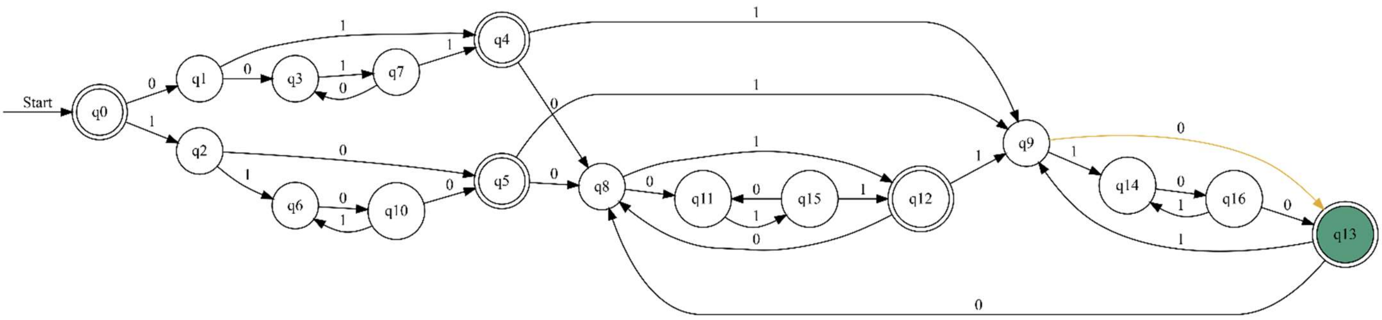
Por forma a converter o NFA-ε obtido em DFA, necessitamos de calcular o fecho-ε do estado inicial q0:

fecho-ε(q0) = {q0, q1, q2, q3, q10, q17, q18, q19, q20, q27, q34, q35, q36, q43, q51, q52, q53, q54, q61, q68, q69}

Designação	Estado (q)	0	1
→*q0	{0,1,2,3,10,17,18,19,20,27,34,35,36,43,51,52,53,54,61,68,69}	q1	q2
q1	{11,12,15,28,29,32,44,45,48,62,63,66}	q3	q4
q2	{4,5,8,21,22,25,37,38,41,55,56,59}	q5	q6
q3	{13,30,46,64}	∅	q7
*q4	{16,33,34,35,36,43,49,50,51,52,53,54,61,67,68,69}	q8	q9
*q5	{9,26,34,35,36,42,43,50,51,52,53,54,60,61,68,69}	q8	q9

q6	{6,23,39,57}	q10	∅
q7	{12,14,15,29,31,32,45,47,48,63,65,66}	q3	q4
q8	{44,45,48,62,63,66}	q11	q12
q9	{37,38,41,55,56,59}	q13	q14
q10	{5,7,8,22,24,25,38,40,41,56,58,59}	q5	q6
q11	{46,64}	∅	q15
*q12	{35,36,43,49,50,51,52,53,54,61,67,68,69}	q8	q9
*q13	{35,36,42,43,50,51,52,53,54,60,61,68,69}	q8	q9
q14	{39,57}	q16	∅
q15	{45,47,48,63,65,66}	q11	q12
q16	{38,40,41,56,58,59}	q13	q14

Ficamos assim com o seguinte DFA de 17 estados:



Vamos então minimizar o DFA anterior, o que é necessário primeiro separar os estados finais dos não-finais. Neste caso temos como estados finais q0, q4, q5, q12 e q13. Ora o q4, q5, q12 e q13 podem ser incluídos no mesmo já que são equivalentes, chamemos de estado X, temos:

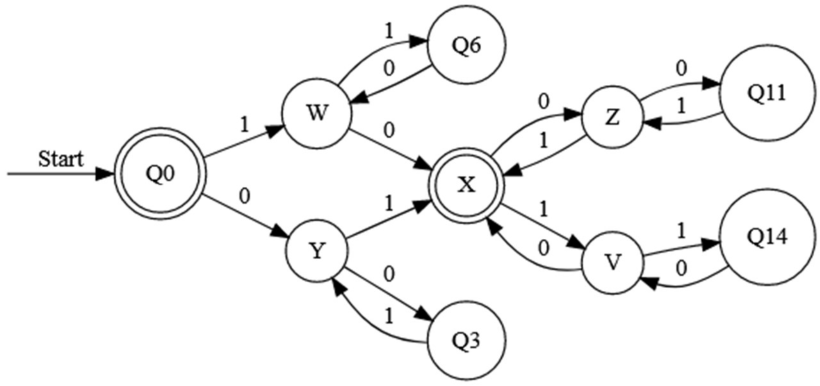
	0	**1**
***Q0**	Q1	Q2
Q1	Q3	X
Q2	X	Q6
Q3	∅	Q7
***X**	Q8	Q9
Q6	Q10	∅
Q7	Q3	X
Q8	Q11	X
Q9	X	Q14
Q10	X	Q6
Q11	∅	Q15
Q14	Q16	∅
Q15	Q11	X
Q16	X	Q14

Ficamos agora com os seguintes estados equivalentes: Q1=Q7 -> Y; Q8=Q15 -> Z; Q2=Q10 -> W; Q9=Q16 -> V.

Ficamos com:

	0	**1**
***Q0**	Y	W
Y	Q3	X
W	X	Q6
Q3	∅	Y
***X**	Z	V
Q6	W	∅
Z	Q11	X
V	X	Q14
Q11	∅	Z
Q14	V	∅

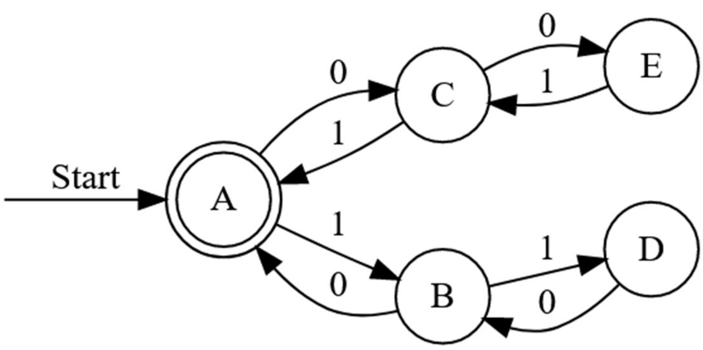
O que ficamos com o seguinte DFA:



Olhando para o diagrama consegue-se ver claramente que se pode minimizar estados que são equivalentes, nomeadamente Q0=X -> A, W=V -> B, Y=Z -> C, Q6=Q14 -> D, Q3=Q11 -> E, o que fica:

	0	**1**
***A**	C	B
B	A	D
C	E	A
D	B	∅
E	∅	C

O que resulta do min-DFA:



3. Para a construção desta expressão regular usei novamente **1 (um)** como sendo a retirada de uma **bola azul** e **0 (zero)** como sendo a retirada de uma **bola vermelha**, assim sendo temos o alfabeto: Σ = {1, 0}.

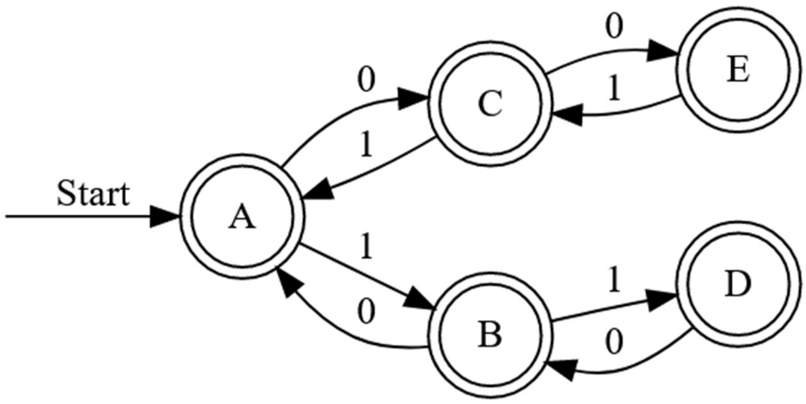
   Neste caso não há a restrição de ser número iguais, então já há uma combinação maior de string's aceitas, tal que aumentou significativamente a expressão.

Assim chegou-se à seguinte expressão:

(1|(0|1(10|01)*00)(01|1(0|1(10|01)*00))*11)(10|01)*1|(0|1(10|01)*00)(01|1(0|1(10|01)*00))*(0|ε|01|1(0|1(10|01)*(0(0|1)|0|ε|10)|1)|1)|ε|1(10|01)*(0|ε|10|0(1|0))|1|0

4. Dado que na questão 2 chegou se a um min-DFA funcional apenas terá que ser adaptado para cumprir agora com os novos requisitos. Isto é, na questão 2 apenas tem um estado de aceitação, que é quando o número de bolas azuis (1's) são iguais ao número de bolas vermelhas (0's), no entanto neste novo min-DFA teremos cinco estados de aceitação:
   - Número de 1's e 0's são iguais;
   - Mais um 0 que 1's;
   - Mais um 1 que 0's;
   - Mais dois 0's que 1's;
   - Mais dois 1's que 0's.

Isto traduz-se em que todos os estados do min-DFA da questão 2 passem agora a ser estados de aceitação ficando:



5.

Insira cada uma das ER e cada um dos DFA, e grave os respetivos modelos.

Modelo da ER da questão 1, ficheiro "1.json";

Modelo da ER da questão 3, ficheiro "3.json";

Modelo dos DFA da questão 2, ficheiro: "2_minDFA.json";

Modelo do minDFA da questão 4, ficheiro: "4_minDFA.json";

Teste todas as sequências possíveis do alfabeto de tamanho inferior a 7, em cada uma das ER, e apresente as que tiveram sucesso no ponto 3 e insucesso no ponto 1.

Questão 1		Questão 3	
Entrada	**Resultado**	**Entrada**	**Resultado**
0	Insucesso!	0	Sucesso!
1	Insucesso!	1	Sucesso!
00	Insucesso!	00	Sucesso!
11	Insucesso!	11	Sucesso!

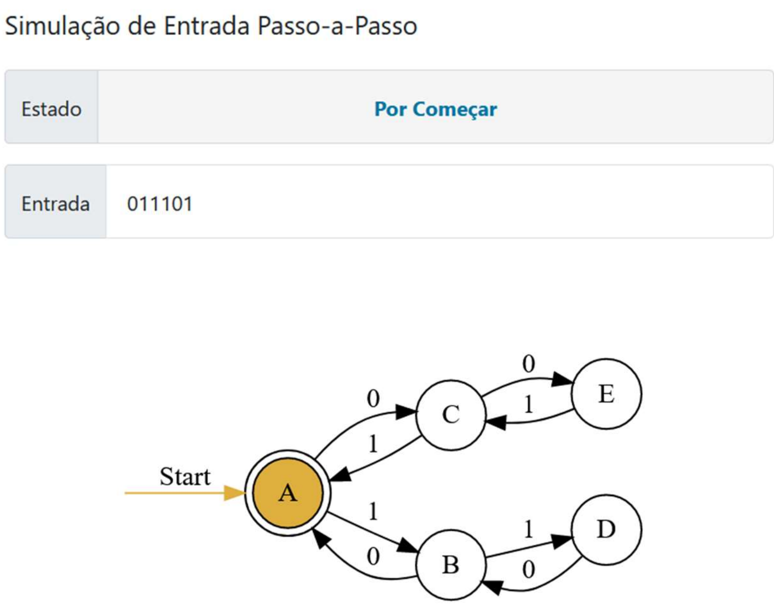
100	Insucesso!	100	Sucesso!
010	Insucesso!	010	Sucesso!
110	Insucesso!	110	Sucesso!
001	Insucesso!	001	Sucesso!
101	Insucesso!	101	Sucesso!
011	Insucesso!	011	Sucesso!
1000	Insucesso!	1000	Sucesso!
0100	Insucesso!	0100	Sucesso!
0010	Insucesso!	0010	Sucesso!
1101	Insucesso!	1101	Sucesso!
1011	Insucesso!	1011	Sucesso!
0111	Insucesso!	0111	Sucesso!
11000	Insucesso!	11000	Sucesso!
10100	Insucesso!	10100	Sucesso!
01100	Insucesso!	01100	Sucesso!
10010	Insucesso!	10010	Sucesso!
01010	Insucesso!	01010	Sucesso!
11010	Insucesso!	11010	Sucesso!
00110	Insucesso!	00110	Sucesso!
10110	Insucesso!	10110	Sucesso!

01110	Insucesso!	01110	Sucesso!
10001	Insucesso!	10001	Sucesso!
01001	Insucesso!	01001	Sucesso!
11001	Insucesso!	11001	Sucesso!
00101	Insucesso!	00101	Sucesso!
10101	Insucesso!	10101	Sucesso!
01101	Insucesso!	01101	Sucesso!
10011	Insucesso!	10011	Sucesso!
01011	Insucesso!	01011	Sucesso!
00111	Insucesso!	00111	Sucesso!
110000	Insucesso!	110000	Sucesso!
101000	Insucesso!	101000	Sucesso!
011000	Insucesso!	011000	Sucesso!
100100	Insucesso!	100100	Sucesso!
010100	Insucesso!	010100	Sucesso!
001100	Insucesso!	001100	Sucesso!
100010	Insucesso!	100010	Sucesso!
010010	Insucesso!	010010	Sucesso!
001010	Insucesso!	001010	Sucesso!
110101	Insucesso!	110101	Sucesso!

101101	Insucesso!	101101	Sucesso!
011101	Insucesso!	011101	Sucesso!
110011	Insucesso!	110011	Sucesso!
101011	Insucesso!	101011	Sucesso!
011011	Insucesso!	011011	Sucesso!
100111	Insucesso!	100111	Sucesso!
010111	Insucesso!	010111	Sucesso!
001111	Insucesso!	001111	Sucesso!

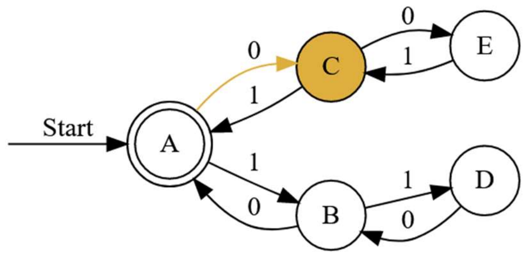
Nos DFA, faça passo a passo o teste da sequência {vermelha, azul, azul, azul, vermelha, azul} (**011101**), e apresente as imagens de cada passo.

DFA questão 1:



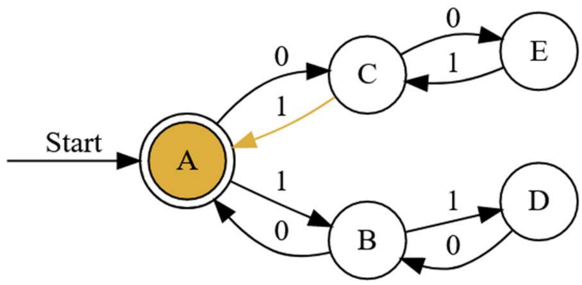
Simulação de Entrada Passo-a-Passo

Estado	**Em simulação!**
Entrada	**0**11101



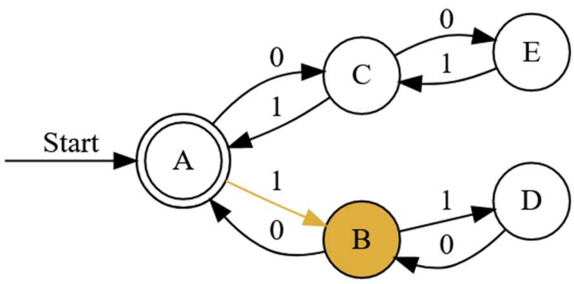
Simulação de Entrada Passo-a-Passo

Estado	**Em simulação!**
Entrada	0**1**1101



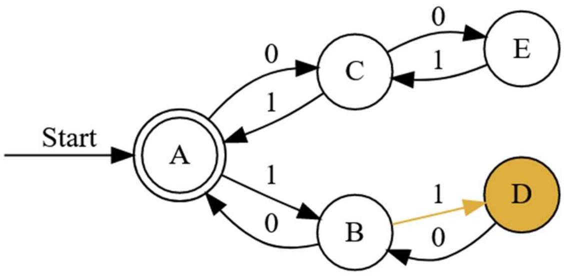
Simulação de Entrada Passo-a-Passo

Estado	**Em simulação!**
Entrada	01**1**101



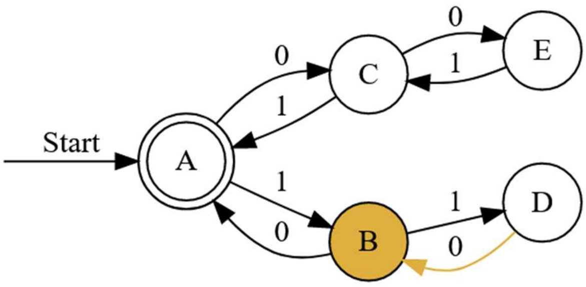
Simulação de Entrada Passo-a-Passo

Estado	**Em simulação!**
Entrada	011**1**01



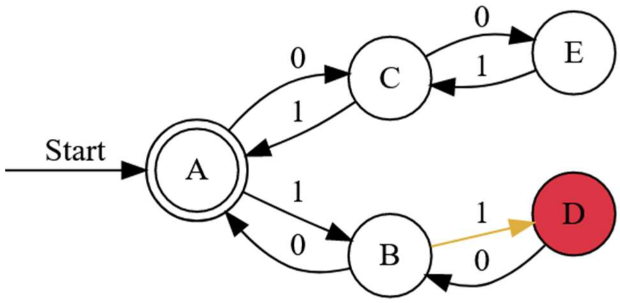
Simulação de Entrada Passo-a-Passo

Estado	**Em simulação!**
Entrada	0111**0**1



Simulação de Entrada Passo-a-Passo

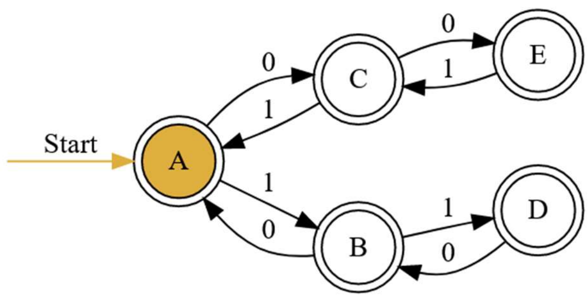
Estado	Insucesso!
Entrada	011101



DFA questão 3:

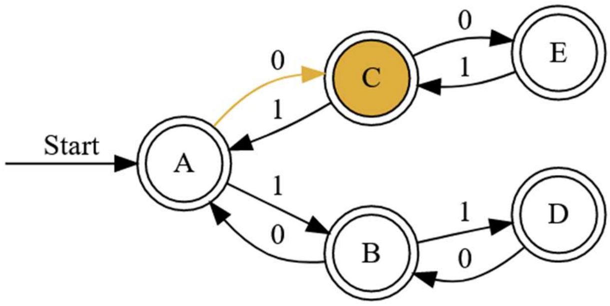
Simulação de Entrada Passo-a-Passo

Estado	Por Começar
Entrada	011101



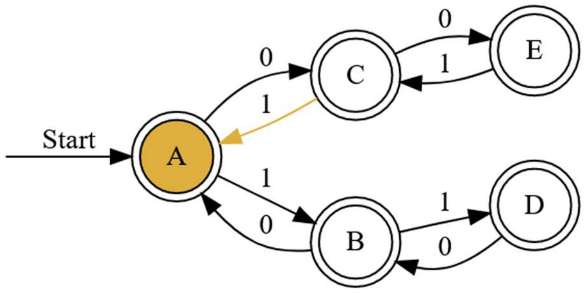
Simulação de Entrada Passo-a-Passo

Estado	Em simulação!
Entrada	011101



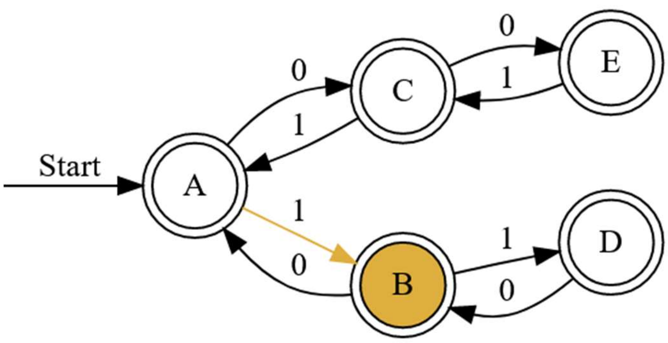
Simulação de Entrada Passo-a-Passo

Estado	**Em simulação!**
Entrada	0**1**1101



Simulação de Entrada Passo-a-Passo

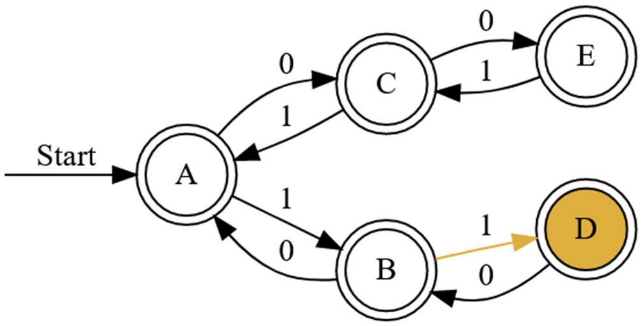
Estado	**Em simulação!**
Entrada	01**1**101



Simulação de Entrada Passo-a-Passo

Estado	**Em simulação!**

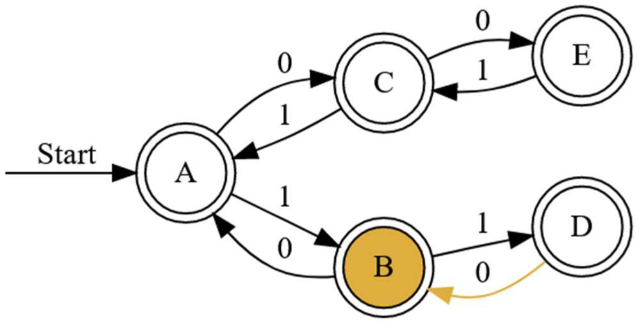
Entrada	011**1**01



Simulação de Entrada Passo-a-Passo

Estado	**Em simulação!**

Entrada	0111**0**1

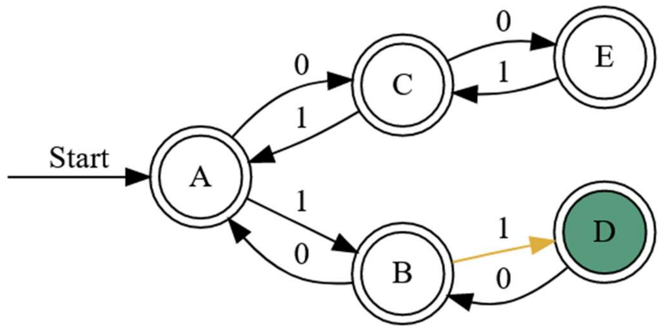


Simulação de Entrada Passo-a-Passo

O Automato aceitou a entrada! Recomece ou teste uma nova entrada ×

Estado	**Sucesso!**

Entrada	01110**1**



Start
A
B
C
D
E
0
1
1
0
0
1
1
0