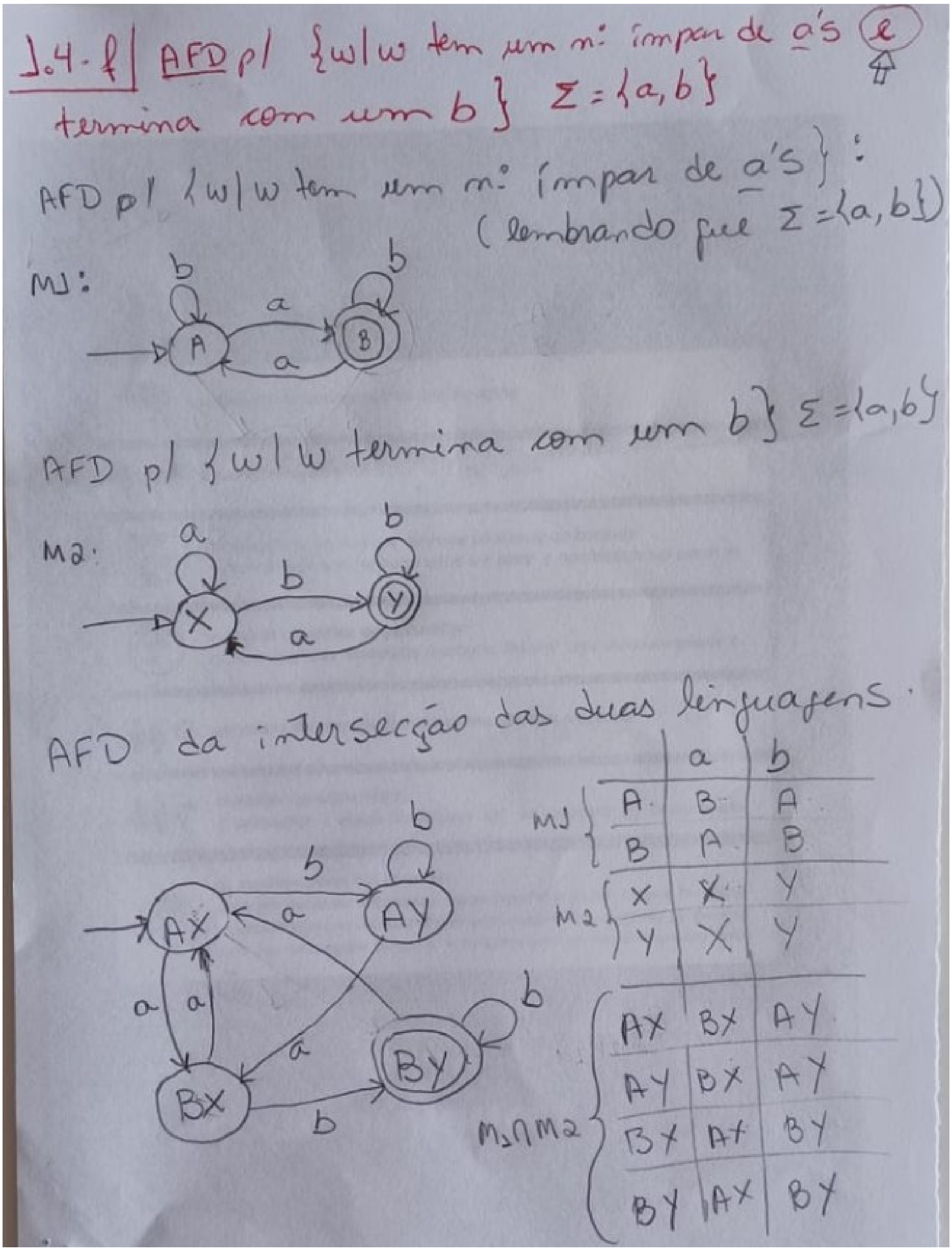
Alguns exercícios do livro Sipser resolvidos ou comentados – Capítulo 1

Dica 1: para todos os exercícios envolvendo a construção de autômatos, vocês podem testar o autômato de vocês no JFlap para vocês saberem se fizeram certo.

Dica 2: para todos os exercícios envolvendo a construção de autômatos que sejam a intersecção ou união de outras linguagens, primeiro criem e testem no JFlap os autômatos de cada linguagem, e então criem e testem no JFlap o autômato resultante.

A seguir a resolução do 1.4f, que é baseado em intersecção:



Dica sobre os exercícios 1.5: Esses são sobre o complemento de outras linguagens. Além da dica de testar o autômato final, fica uma dica especial para o item **c)** {w | w não contém nem a subcadeia ab , nem ba}. Aqui essa linguagem L é o complemento de uma união de outras duas. Ou seja, L1 = {w | w contém a subcadeia ab}, L2 = {w | w contém a subcadeia ba}, L = complemento (L1 U L2).

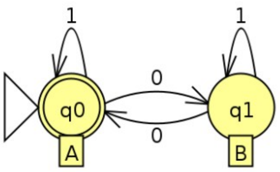
Exercício 1.6 l): Esse eu fiz em slides para mostrar como fazer baseando-se na tabela de transições:

1.6 Give state diagrams of AFDs recognizing the following languages. In all parts the alphabet is {0,1}

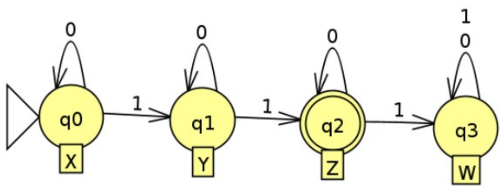
l. $\{w|\ w$ contains an even number of 0s, or contains exactly two 1s$\}$

Vamos construir um AFD M1 para a linguagem sublinhada em vermelho, um AFD M2 para a linguagem sublinhada em azul, e deles construir o AFD da união

M1:



M2:



Pode ser mais fácil (para não se confundir) construir as transições olhando as tabelas de transição.

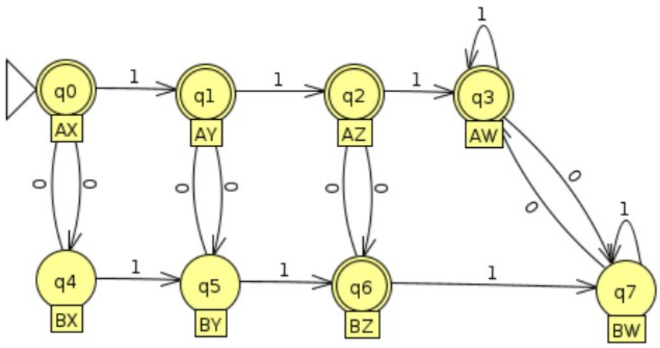
Note o highlight em amarelo as células usadas para a construção da transição δ(AX,0):

		0	1
M1	A	B	A
	B	A	B
M2	X	X	Y
	Y	Y	Z
	Z	Z	W
	W	W	W
M1 U M2	AX	BX	
	AY		
	AZ		
	AW		
	BX		
	BY		
	BZ		
	BW		

Tabela completa:

		0	1
M1	A	B	A
	B	A	B
M2	X	X	Y
	Y	Y	Z
	Z	Z	W
	W	W	W
M1 U M2	AX	BX	AY
	AY	BY	AZ
	AZ	BZ	AW
	AW	BW	AW
	BX	AX	BY
	BY	AY	BZ
	BZ	AZ	BW
	BW	AW	BW

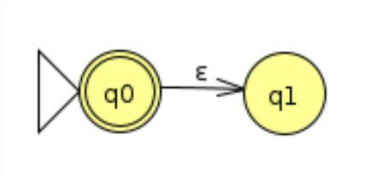
Diagrama de estados:



1.10 c)

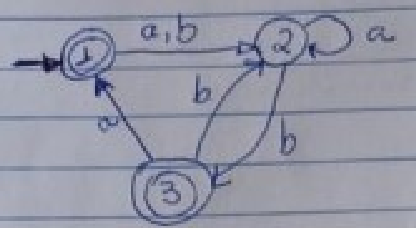
1.10 Use the construction given in the proof of Theorem 1.49 to give the state diagrams of AFNs recognizing the star of the language described in

c. Exercise 1.6m. → **m.** The empty set

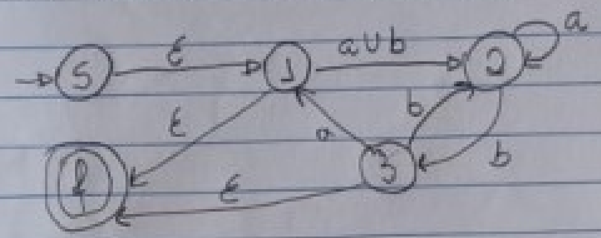


1.21 b) Agora um passo a passo (à mão) de conversão AF → ER (próximas páginas). Tudo bem se a expressão ficar grande (não precisa simplificar).

1.21 b Use o lema 1.60 para converter os seguintes autômatos em expressões regulares

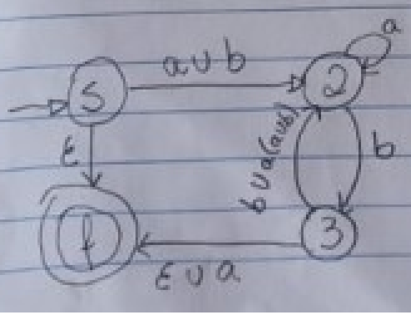
b) 

Primeiro vamos transformar esse AFD em AFNG:
(vou omitir arestas com ϕ)



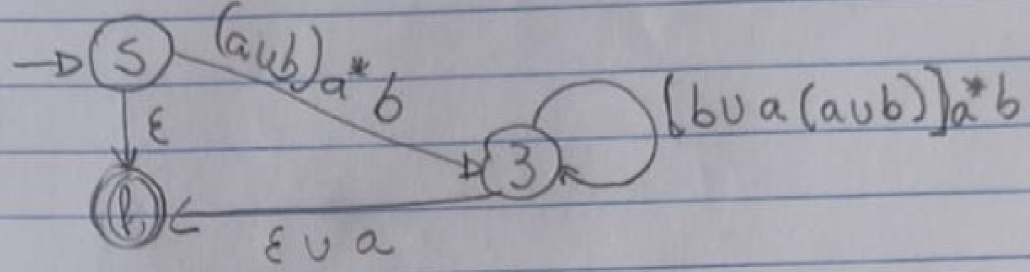
Vou remover o estado 1.
Vou fazer a tabela de cada par de estados para não esquecer nenhum par (lembrando que "S" não pode ser destino e "f" não pode ser origem)

Origem \ Destino	2	3	f
S	a ∪ b	ϕ	ε
2	a	b	ϕ
3	b ∪ a(a∪b)	ϕ	ε ∪ a



Removendo o estado 2:

Origem \ Destino	3	f
S	$(a \cup b)a^*b$	ε
3	$[b \cup a(a \cup b)]a^*b$	$\varepsilon \cup a$



Finalmente removendo o estado 3:

$$\rightarrow S \xrightarrow{\varepsilon \cup (a \cup b)a^*b\{[b \cup a(a \cup b)]a^*b\}^*(\varepsilon \cup a)} f$$

e está é a expressão regular

1.54 Considere a linguagem $F = \{a^i b^j c^k \mid i, j, k \geq 0 \text{ e se } i = 1 \text{ então } j = k\}$.

a. Mostre que F não é regular.

b. Mostre que F funciona como uma linguagem regular no lema do bombeamento. Em outras palavras, dê um comprimento de bombeamento p e demonstre que F satisfaz as três condições do lema do bombeamento para esse valor de p.

c. Explique porque os itens (a) e (b) não contradizem o lema do bombeamento.

a) Sejam L e B as linguagens L = $\{a\,b^j c^k \mid j,k >= 0 \text{ e } j=k\}$ e B = {ab*c*}. L é estritamente livre de contexto, e B é regular.
L = B ∩ F.
Vamos assumir que F seja regular.
Como B é regular, se F fosse regular L também seria, pois a intersecção de duas linguagens regulares necessariamente é regular também (pois as linguagens regulares são fechadas com relação à intersecção). Sabemos que isso é absurdo, pois L não é regular. Então F também não é.

b) F = L U R1 U R2, sendo L a linguagem estritamente livre de contexto definida no item a (L = $\{a\,b^j c^k \mid j,k >= 0 \text{ e } j=k\}$), R1 = {b*c*} e R2 = $\{aa^+b^*c^*\}$, sendo R1 e R2 regulares.
Como R1 e R2 são regulares, elas repeitam o lema do bombeamento para um dado p1 e p2, seus respetivos comprimentos de bombeamento. A linguagem R1 U R2 é regular, e portanto segue o lema do bombeamento para p = max(p1, p2).

Para esse valor de p, F respeita o lema do bombeamento:
Como F = L U R1 U R2, se vc tomar uma cadeia de R1 U R2 maior ou igual a p, como é regular, bombeá-la acaba caindo em R1 U R2 de novo (ou seja, pertencerá a R1 U R2), e portanto pertencerá a F.
Se vc tomar uma cadeia de L, NÃO DÁ PARA BOMBEÁ-LA DE FORMA QUE ELA FIQUE EM **L**, mas uma subdivisão possível é fazer y = a, e tirar o a ou adicionar outros a's fará com que a cadeia resultante pertença a R1 U R2, e portanto pertencerá em F.

c) Não há contradição porque **o lema do bomb. NÃO é um se e somente se.** Isto é, SE A é regular => vale o lema do bombeamento (a volta não necessariamente é verdadeira). E no item b) explicamos como isto foi possível.