Expressões Regulares

MCTA015-13 - Linguagens Formais e Autômata

Profa. Carla Negri Lintzmayer

carla.negri@ufabc.edu.br

www.professor.ufabc.edu.br/∼carla.negri

Centro de Matemática, Computação e Cognição – Universidade Federal do ABC

Expressões regulares

- São expressões que descrevem linguagens (descrições algébricas) usando as operações regulares (união, concatenação e estrela).
- O valor de uma expressão R é uma linguagem.
- Exemplo: $(\mathbf{0} \cup \mathbf{1})\mathbf{0}^*$ é a linguagem das cadeias que começam com 0 ou 1, seguidos por um número qualquer de 0's, ou $\{\omega \in \{0,1\}^* \mid \omega = 0^k, \text{ com } k \geq 1, \text{ ou } \omega = 10^j, \text{ com } j \geq 0\}$.

Expressões regulares

R é uma expressão regular sobre um alfabeto Σ que descreve a linguagem $L(R)$ se

- $R = \mathtt{a}$ para algum $\mathtt{a} \in \Sigma$, caso em que $L(R) = \{\mathtt{a}\}$;
- $R = \varepsilon$, caso em que $L(R) = \{\varepsilon\}$;
- $R = \emptyset$, caso em que $L(R) = \emptyset$;
- $R = R_1 \cup R_2$ onde R_i é expressão regular, $i = 1, 2$, caso em que $L(R) = L(R_1) \cup L(R_2)$;
- $R = R_1 R_2$ onde R_i é expressão regular, $i = 1, 2$, caso em que $L(R) = L(R_1)L(R_2)$;
- $R = R_1^*$ onde R_1 é expressão regular, caso em que $L(R) = L(R_1)^*$.

A ordem precedência das operações é: estrela, concatenação e união.

Precedência

- Parênteses não estão na definição, mas são usados para alterar precedência.
  - A linguagem de $\mathbf{0} \cup \mathbf{10}^*$ é diferente da linguagem de $(\mathbf{0} \cup \mathbf{10}^*)$.
- Assim, se $R = (R')$ e R' é uma expressão regular, então $L(R) = L(R')$.

Equivalência

Dizemos que duas expressões regulares R_1 e R_2 são equivalentes, denotando $R_1 \equiv R_2$, se $L(R_1) = L(R_2)$.

- $(\mathbf{0} \cup \mathbf{1})^* \equiv (\mathbf{0}^*\mathbf{1}^*)^*$.
- $(\mathbf{0} \cup \boldsymbol{\varepsilon})\mathbf{1} \equiv \mathbf{01} \cup \mathbf{1}$.

Açúcar sintático

- Se R é expressão regular, então $R^+ = RR^*$.
- Se $\Sigma = \{a_1, \ldots, a_k\}$, então $\mathbf{\Sigma} = a_1 \cup \cdots \cup a_k$.
- Para facilitar notação, vamos escrever

  $$(\mathbf{0} \cup \mathbf{1})\mathbf{0}^* = \{\omega \in \{0,1\}^* \; \omega = 0^k, \text{ com } k \geq 1, \text{ ou } \omega = 10^j, \text{ com } j \geq 0\}$$

  onde o correto seria

  $$L((\mathbf{0} \cup \mathbf{1})\mathbf{0}^*) = \{\omega \in \{0,1\}^* \; \omega = 0^k, \text{ com } k \geq 1, \text{ ou } \omega = 10^j, \text{ com } j \geq 0\}$$

Exercícios

Qual a linguagem descrita pelas seguintes expressões regulares?

1. $\mathbf{0}^*\mathbf{1}\mathbf{0}^*$
2. $\mathbf{\Sigma}^*\mathbf{0}\mathbf{0}\mathbf{1}\mathbf{\Sigma}^*$
3. $\mathbf{1}^*(\mathbf{0}\mathbf{1}^+)^*$
4. $\mathbf{0}\mathbf{1} \cup \mathbf{1}\mathbf{0}$
5. $\mathbf{0}\mathbf{\Sigma}^*\mathbf{0} \cup \mathbf{1}\mathbf{\Sigma}^*\mathbf{1} \cup \mathbf{0} \cup \mathbf{1}$

Exercícios

Forneça uma expressão regular para as linguagens a seguir. Considere $\Sigma = \{0, 1\}$:

1. $\{\omega \in \Sigma^* \mid \omega$ tem comprimento par$\}$
2. $\{\varepsilon, 0, 1, 01\}$
3. $\{\omega \in \Sigma^* \mid \omega$ não contém 00 nem 11$\}$
4. $\{\omega \in \Sigma^* \mid$ toda posição ímpar de ω é 1$\}$
5. $\{\omega \in \Sigma^* \mid \omega$ tem pelo menos um 0 e um 1$\}$
6. $\{\omega \in \Sigma^* \mid \omega$ tem no máximo uma ocorrência de 11$\}$
7. $\{\omega \in \Sigma^* \mid$ o número de 0's em ω é divisível por $3\}$
8. $\{\omega \in \Sigma^* \mid \omega$ não contém 110 como subcadeia$\}$

Propriedades

Sejam R, R_1, R_2 e R_3 expressões regulares.

Para a operação de união vale que:

- $R \cup \emptyset \equiv \emptyset \cup R \equiv R$ ($\emptyset$ é identidade)
- $R_1 \cup (R_2 \cup R_3) \equiv (R_1 \cup R_2) \cup R_3$ (associatividade)
- $R_1 \cup R_2 \equiv R_2 \cup R_1$ (comutatividade)
- $R_1 \cup R_1 \equiv R_1$ (idempotência)

Obs.: Nem sempre é verdade que $R \cup \varepsilon \equiv R$ (por quê?)

Propriedades

Sejam R, R_1, R_2 e R_3 expressões regulares.

Para a operação de concatenação vale que:

- $\boldsymbol{\varepsilon} R \equiv R\boldsymbol{\varepsilon} \equiv R$ (ε é identidade)
- $R_1(R_2R_3) \equiv (R_1R_2)R_3$ (associatividade)
- $R_1(R_2 \cup R_3) \equiv R_1R_2 \cup R_1R_3$ (distributividade à esquerda)
- $(R_1 \cup R_2)R_3 \equiv R_1R_3 \cup R_2R_3$ (distributividade à direita)
- $\boldsymbol{\emptyset} R \equiv R\boldsymbol{\emptyset} \equiv \boldsymbol{\emptyset}$ ($\emptyset$ é aniquilador)

Propriedades

Seja R uma expressão regular.

Para a operação estrela vale que:

- $R^* \equiv R^+ \cup \varepsilon$
- $\emptyset^* \equiv \varepsilon$
- $\varepsilon^* \equiv \varepsilon$

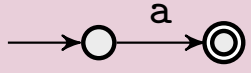
Teorema

A linguagem descrita por uma expressão regular é uma linguagem regular.

Demonstração.

Seja R uma expressão regular sobre um alfabeto Σ. Queremos mostrar que $L(R)$ é uma linguagem regular. Vamos provar isso por indução na quantidade de operações regulares de R.

Caso base: R não tem operações regulares. Então temos três casos:

1. $R = \mathbf{a}$, com $\mathrm{a} \in \Sigma$. Nesse caso, $L(R) = \{\mathrm{a}\}$. O AFN  reconhece $L(R)$.

2. $R = \varepsilon$. Nesse caso, $L(R) = \{\varepsilon\}$. O AFN  reconhece $L(R)$.
3. $R = \emptyset$. Nesse caso, $L(R) = \emptyset$. O AFN reconhece $L(R)$.

Então no caso base $L(R)$ é regular.

Teorema

A linguagem descrita por uma expressão regular é uma linguagem regular.

Demonstração (continuação).

Passo: R tem $n \geq 1$ operações regulares. Suponha que para qualquer expressão regular R' com k operações regulares, onde $0 \leq k < n$ vale que $L(R')$ é uma linguagem regular.
Como R contém ao menos uma operação, temos três casos:

1. $R = R_1 \cup R_2$. Como ambas R_1 e R_2 contêm menos de n operações, vale por hipótese que $L(R_1)$ e $L(R_2)$ são regulares. Como $L(R) = L(R_1) \cup L(R_2)$ e linguagens regulares são fechadas sob união, então $L(R)$ é regular.
2. $R = R_1 R_2$. Por hipótese, $L(R_1)$ e $L(R_2)$ são regulares. Como $L(R) = L(R_1)L(R_2)$ e linguagens regulares são fechadas sob concatenação, então $L(R)$ é regular.
3. $R = R_1^*$. Por hipótese, $L(R_1)$ é regular. Como $L(R) = L(R_1)^*$ e linguagens regulares são fechadas sob concatenação, então $L(R)$ é regular.

Então $L(R)$ é uma linguagem regular. □

Expressões regulares e autômatos

O teorema anterior nos garante três coisas:

- O conjunto das linguagens descritas por expressões regulares está contido no conjunto das linguagens regulares.
- Qualquer expressão regular pode ser descrita por um autômato finito não determinístico.
- Em conjunto com os teoremas de fechamento das linguagens regulares sob as operações regulares, a prova do teorema fornece um algoritmo para transformar uma expressão regular em um AFN.

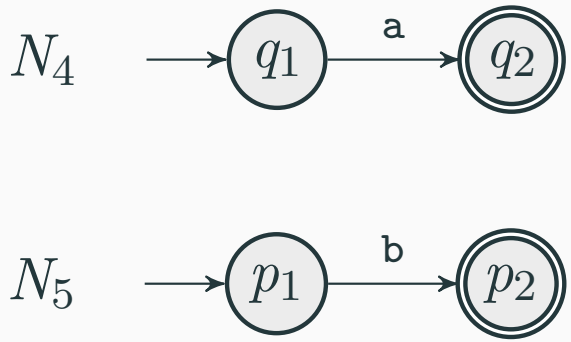
Expressões regulares e autômatos

Considere $R = (\mathbf{a} \cup \mathbf{b})^*\mathbf{ab}$.

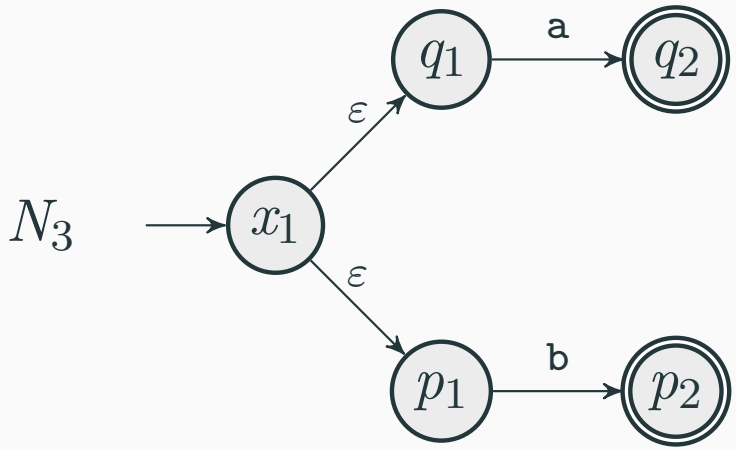
Note que

- $R = R_1 R_2$
- $R_1 = (\mathbf{a} \cup \mathbf{b})^*$
  - $R_1 = R_3^*$
  - $R_3 = \mathbf{a} \cup \mathbf{b}$
    - $R_3 = R_4 \cup R_5$
    - $R_4 = \mathbf{a}$
    - $R_5 = \mathbf{b}$
- $R_2 = \mathbf{ab}$
  - $R_2 = R_4 R_5$
  - $R_4 = \mathbf{a}$
  - $R_5 = \mathbf{b}$

As expressões R_4 e R_5 se encaixam no caso base do teorema, logo os seguintes AFNs as reconhecem:

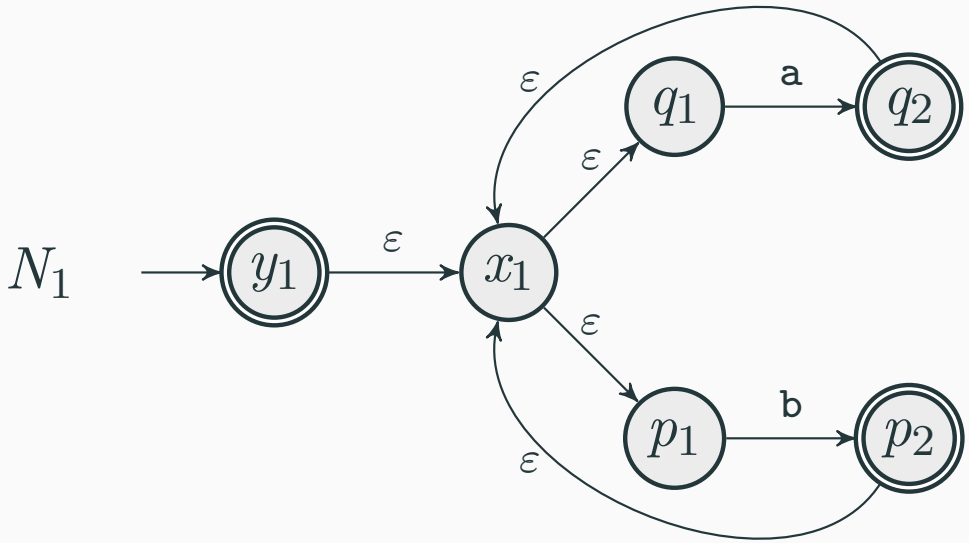


Um autômato para a expressão $R_3 = R_4 \cup R_5$ pode ser feito a partir dos autômatos N_4 e N_5:



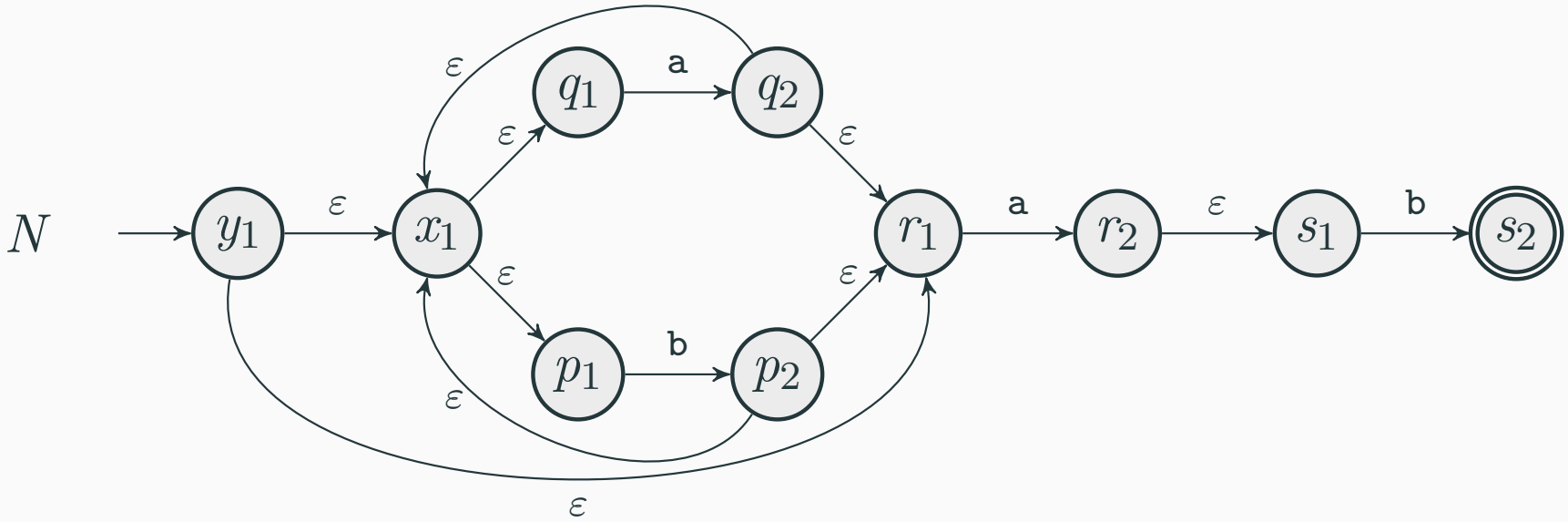
Um autômato para a expressão $R_2 = R_4 R_5$ pode ser feito a partir dos autômatos N_4 e N_5:

Um autômato para a expressão $R_1 = R_3^*$ pode ser feito a partir do autômato N_3:



Expressões regulares e autômatos

Finalmente, um autômato para a expressão $R = R_1 R_2$ pode ser feito a partir dos autômatos N_1 e N_2:



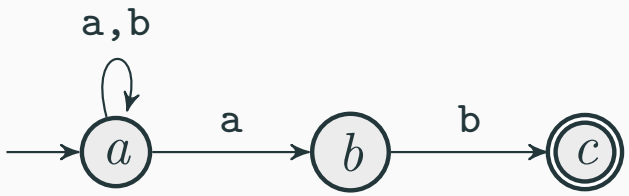
Cuidado!

Note que a expressão regular $R = (\mathbf{a} \cup \mathbf{b})^*\mathbf{ab}$ descreve a linguagem

$$\{\omega \in \{\mathbf{a}, \mathbf{b}\}^* \mid \omega \text{ termina em } \mathbf{ab}\}$$

O AFN N que construímos anteriormente reconhece essa linguagem pois foi construído diretamente a partir de R usando o algoritmo descrito na demonstração do teorema anterior.

Note ainda que o autômato a seguir também reconhece essa linguagem (e é muito mais "bonito") que o AFN N:



Assim, "transformar uma expressão regular em um autômato" é diferente de "descrever a linguagem da expressão e então criar um autômato para a mesma".

Autômatos e expressões regulares

- Agora que sabemos que qualquer expressão regular tem um AFN correspondente, será que é verdade que qualquer AFN tem uma expressão regular correspondente?
- Se isso for verdade, estaremos dizendo que o conjunto das linguagens reconhecidas por AFNs está contido no conjunto das linguagens descritas por expressões regulares.
- Como o teorema anterior disse que o conjunto das linguagens descritas por expressões regulares está contido no conjunto das linguagens regulares, então na verdade teremos que ambos conjuntos são idênticos!
- Antes de poder provar isso, precisamos de uma definição auxiliar.

Autômatos finitos não determinísticos generalizados

Um autômato finito não determinístico generalizado (AFNG) é uma 5-upla $(Q, \Sigma, \delta, q_i, q_f)$ em que

- Q é um conjunto finito de estados;
- Σ é um alfabeto;
- $\delta\colon (Q \setminus \{q_f\}) \times (Q \setminus \{q_i\}) \to \mathcal{R}$ é uma função de transição, em que $\mathcal{R}$ é o conjunto de todas as expressões regulares sobre Σ;
- $q_i \in Q$ é o estado inicial;
- $q_f \in Q$ é o estado final.

Autômatos finitos não determinísticos generalizados

- Um AFNG, em outras palavras, é um autômato no qual as transições possuem expressões regulares como rótulos.
- A definição de δ nos permite observar que:
  - Todo estado, exceto pelo final, possui transição para todo outro estado, exceto para o inicial.
  - Não há transições chegando no estado inicial q_i.
  - Não há transições saindo do estado final q_f.

Teorema

Todas linguagem regular pode ser descrita por uma expressão regular.

Demonstração.

Seja A uma linguagem regular qualquer. Então existe um AFN N que reconhece A. A ideia é transformar N em um AFNG G que seja equivalente (isto é, $L(N) = L(G)$), e então modificar o AFNG para que ele tenha apenas dois estados, o que significa que o rótulo da transição restante é a expressão regular equivalente.

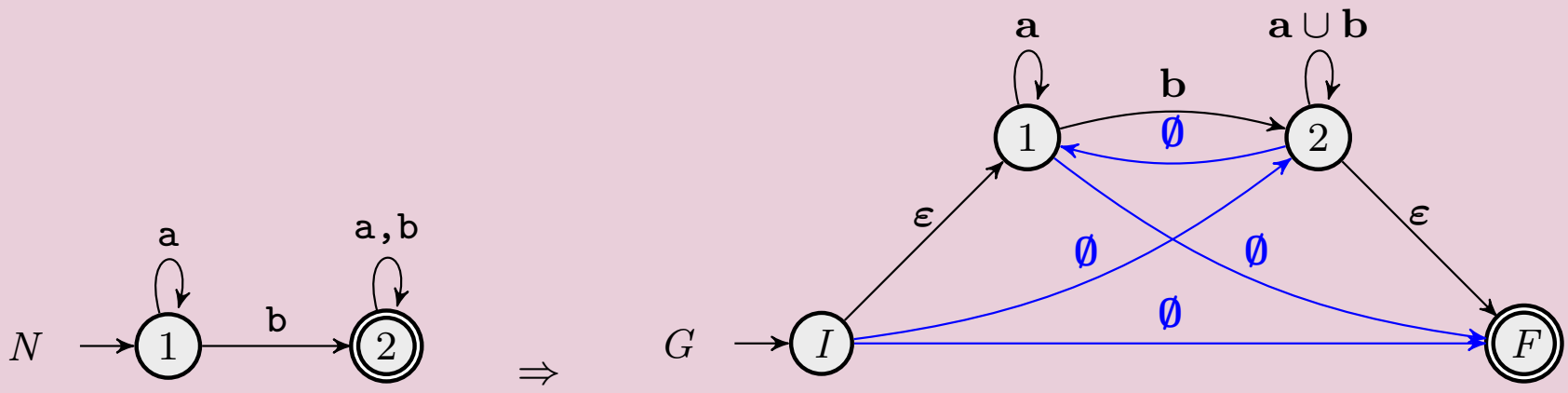
Teorema

Todas linguagem regular pode ser descrita por uma expressão regular.

Demonstração (continuação).

Para a transformação inicial, seja $N = (Q, \Sigma, \delta, q_0, F)$. Começamos por criar um novo estado inicial q_i com transição rotulada com ε para q_0. Criamos um novo estado final q_f com transições rotuladas com ε chegando de todos os estados de F. Se uma transição tem múltiplos rótulos, substituímos cada uma por uma única transição com rótulo igual à união dos rótulos anteriores. Por fim, coloca-se transições rotuladas com $\emptyset$ entre estados que não tenham transições.

■



▶

Teorema

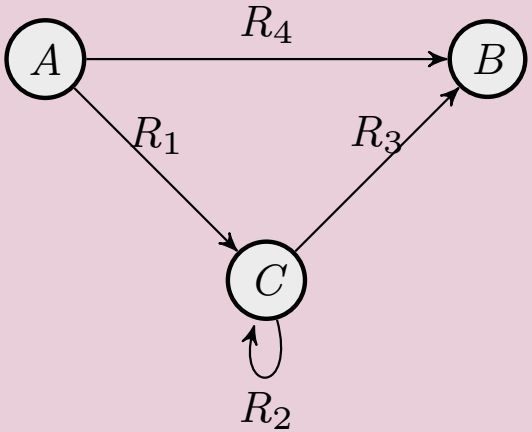
Todas linguagem regular pode ser descrita por uma expressão regular.

Demonstração (continuação).

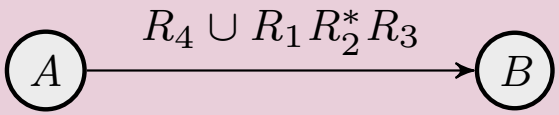
Com G construído, o passo crucial é reduzir seu número de estados: um a um, os estados originais de N serão removidos e a máquina restante será reorganizada para permanecer equivalente.

II

Ideia: se um certo estado C será removido, então qualquer outro par de estados A e B será afetado (pois num AFNG) todo estado tem transição para todo outro estado (exceto que não há transições chegando no inicial nem saindo do final).



$\Rightarrow$



▶

Teorema

Todas linguagem regular pode ser descrita por uma expressão regular.

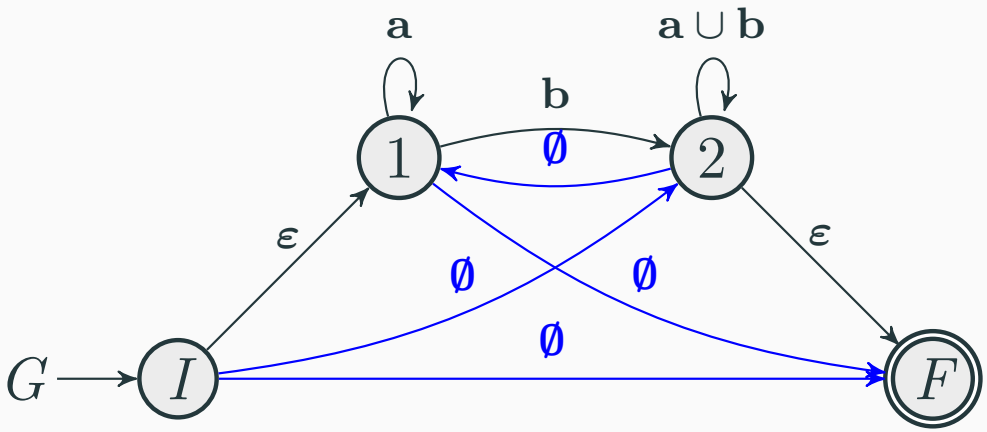
Demonstração (continuação).

O processo formal de remoção de estados de G está descrito no seguinte algoritmo:

Função CONVERTE_AFNG_EM_ER($G = (Q, \Sigma, \delta, q_i, q_f)$)
  Se $|Q| == 2$ **então**
    Devolve $\delta(q_i, q_f)$
  seja $r \in Q \setminus \{q_i, q_f\}$ o estado a ser removido
  Para cada $q_a \in Q \setminus \{r, q_f\}$ **faça**
    Para cada $q_b \in Q \setminus \{r, q_i\}$ **faça**
      seja $R_4 = \delta(q_a, q_b)$
      seja $R_1 = \delta(q_a, r)$
      seja $R_2 = \delta(r, r)$
      seja $R_3 = \delta(r, q_b)$
      faça $\varphi(q_a, q_b) = R_1 R_2^* R_3 \cup R_4$
  construa $G' = (Q \setminus \{r\}, \Sigma, \varphi, q_i, q_f)$
  Devolve CONVERTE_AFNG_EM_ER(G')

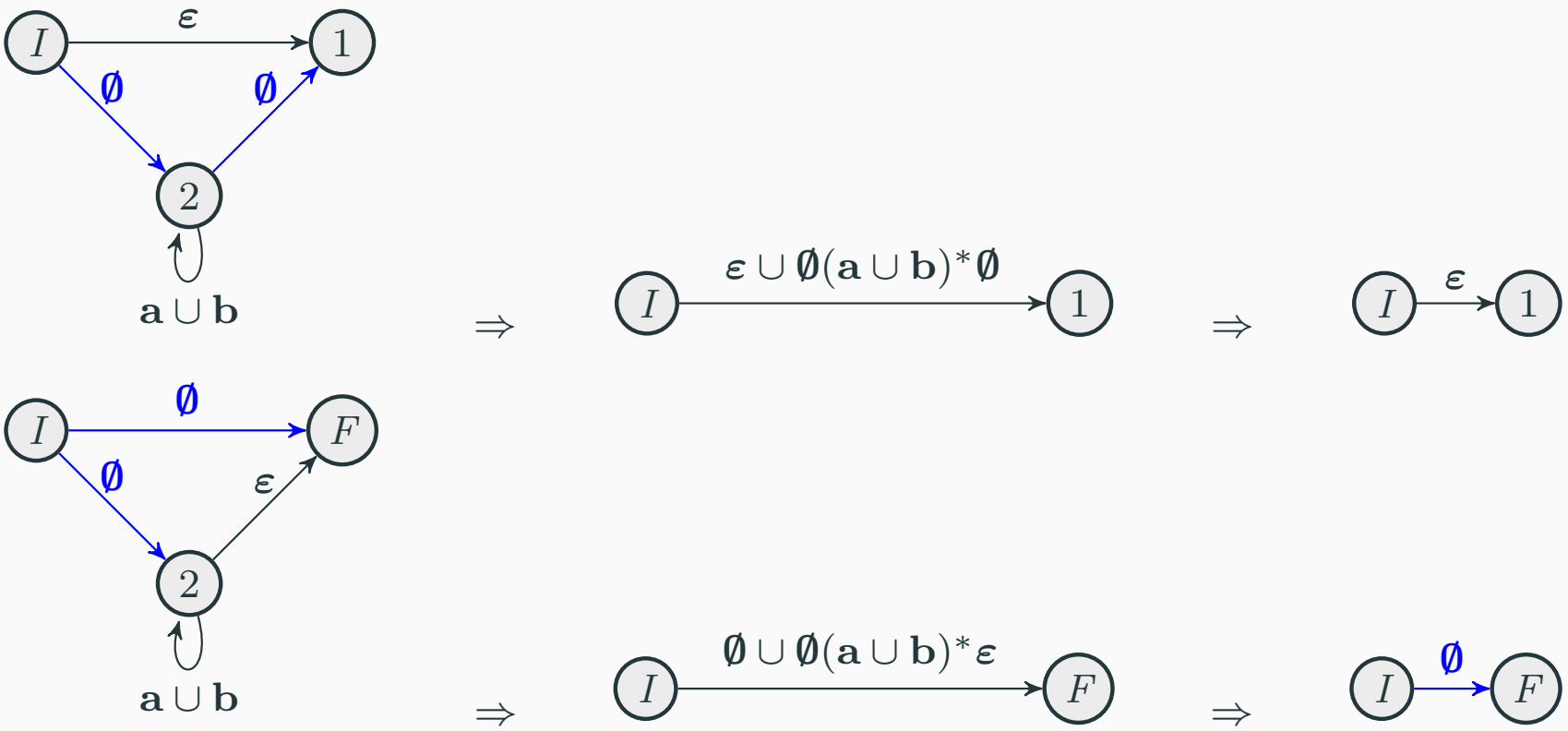
A expressão devolvida por CONVERTE_AFNG_EM_ER(G) é equivalente a G. □

Vamos executar o algoritmo $\textsc{Converte\_AFNG\_em\_ER}(G)$ sobre



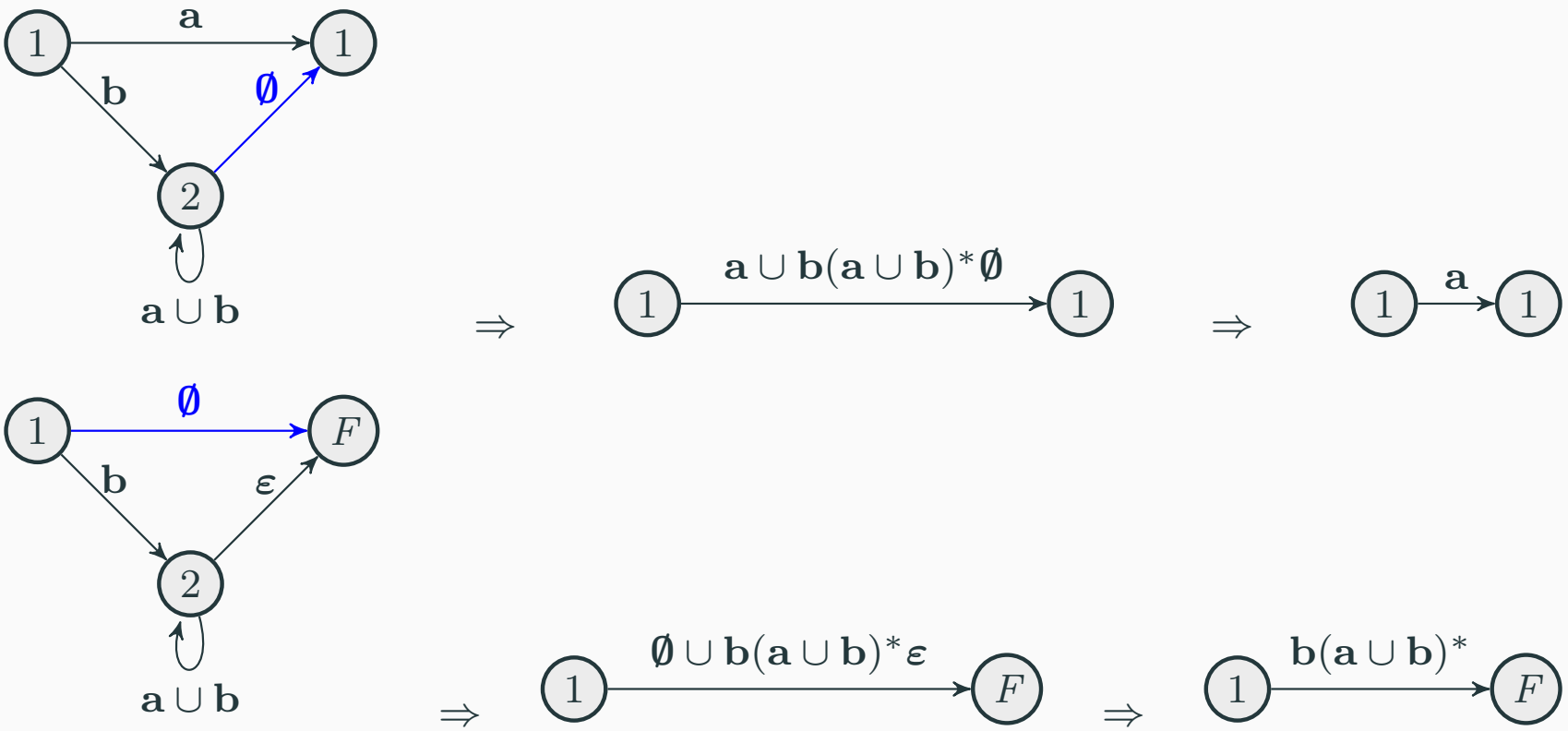
Como $|Q| = 4 > 2$, vamos remover o estado $r = 2$. Nesse caso, q_a pode assumir valores em $\{I, 1\}$ e q_b pode assumir valores em $\{1, F\}$.

Assim, as novas transições serão

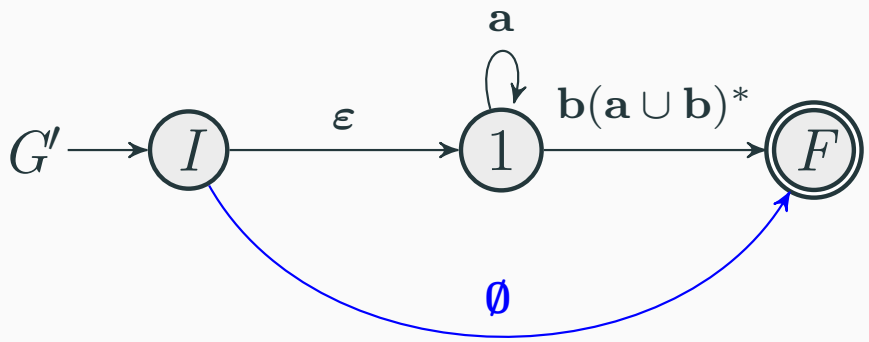


Como $|Q| = 4 > 2$, vamos remover o estado $r = 2$. Nesse caso, q_a pode assumir valores em $\{I, 1\}$ e q_b pode assumir valores em $\{1, F\}$.

Assim, as novas transições serão



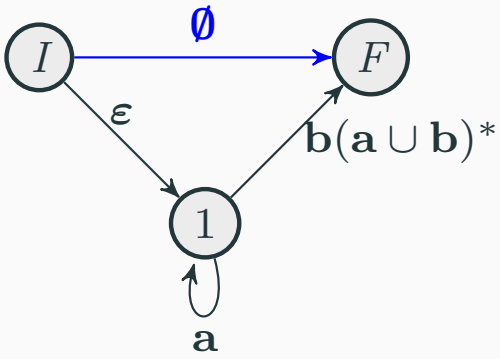
Agora temos



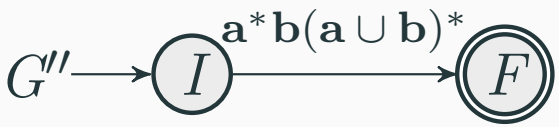
Vamos executar $\textsc{Converte\_AFNG\_em\_ER}(G')$.

Como $|Q| = 3 > 2$, vamos remover o estado $r = 1$. Nesse caso, q_a pode assumir valores em $\{I\}$ e q_b pode assumir valores em $\{F\}$.

Assim, as novas transições serão



Agora temos

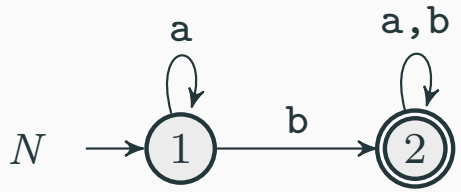


Ao executar $\textsc{Converte\_AFNG\_em\_ER}(G'')$, entramos no caso base, pois $|Q| == 2$.

O algoritmo então retorna $\mathbf{a}^*\mathbf{b}(\mathbf{a} \cup \mathbf{b})^*$.

Cuidado!

Note que o autômato



reconhece a linguagem $\{\omega \in \{\mathtt{a}, \mathtt{b}\}^* \mid \omega \text{ contém ao menos um } \mathtt{b}\}$.

A expressão $\mathbf{a}^*\mathbf{b}(\mathbf{a} \cup \mathbf{b})^*$ que construímos anteriormente descreve essa linguagem pois foi construída diretamente a partir de N usando o algoritmo descrito na demonstração do teorema anterior.

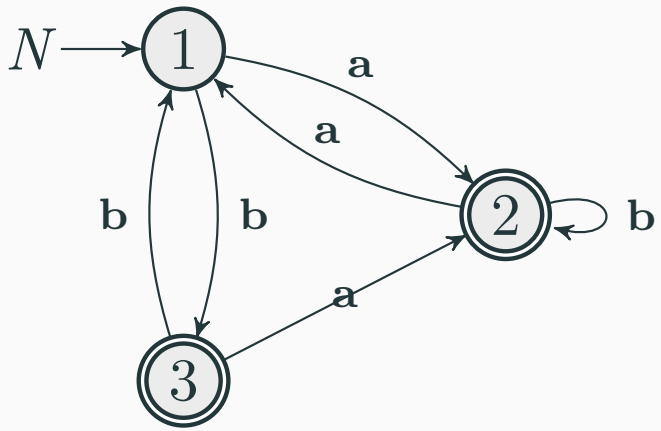
Note ainda que outras expressões descrevem essa linguagem:

- $(\mathbf{a} \cup \mathbf{b})^*\mathbf{b}(\mathbf{a} \cup \mathbf{b})^*$
- $\mathbf{a}^*\mathbf{b}^+(\mathbf{a}^*\mathbf{b}^*)^*$

Assim, "encontrar uma expressão regular equivalente a um autômato" é diferente de "encontrar a linguagem que o autômato reconhece e então criar uma expressão que a descreva".

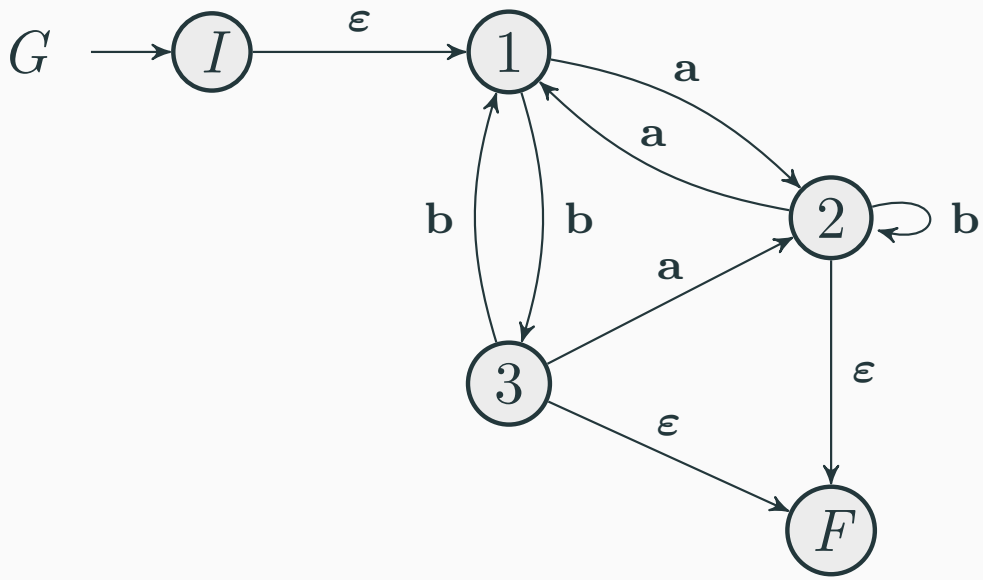
Outro exemplo

Vamos encontrar uma expressão regular equivalente ao autômato



Outro exemplo

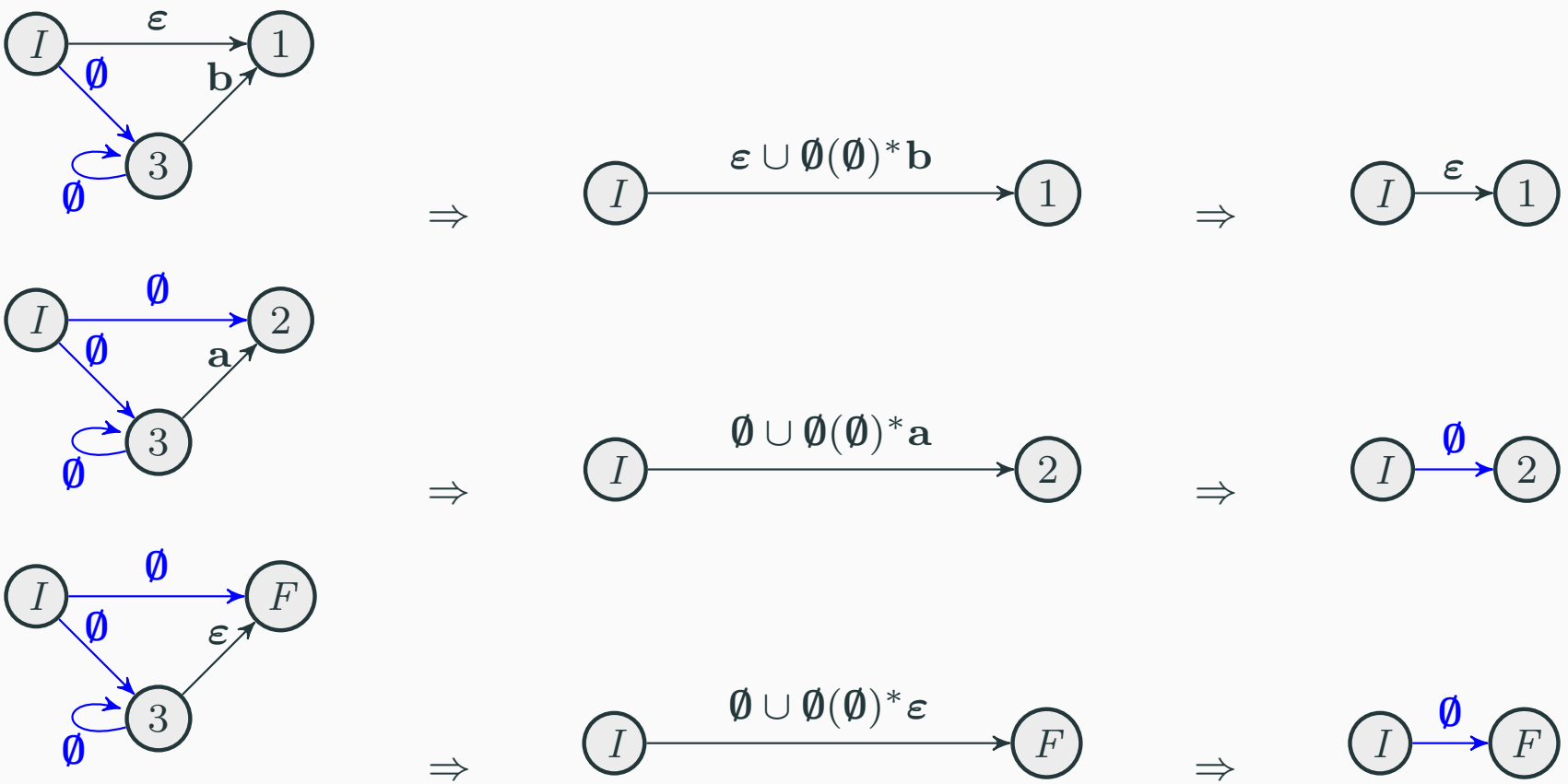
Primeiro o transformamos em um AFNG equivalente (transições rotuladas com $\emptyset$ estão omitidas para não poluir a imagem, mas lembre-se que elas existem!):



Vamos executar $\textsc{Converte\_AFNG\_em\_ER}(G)$.

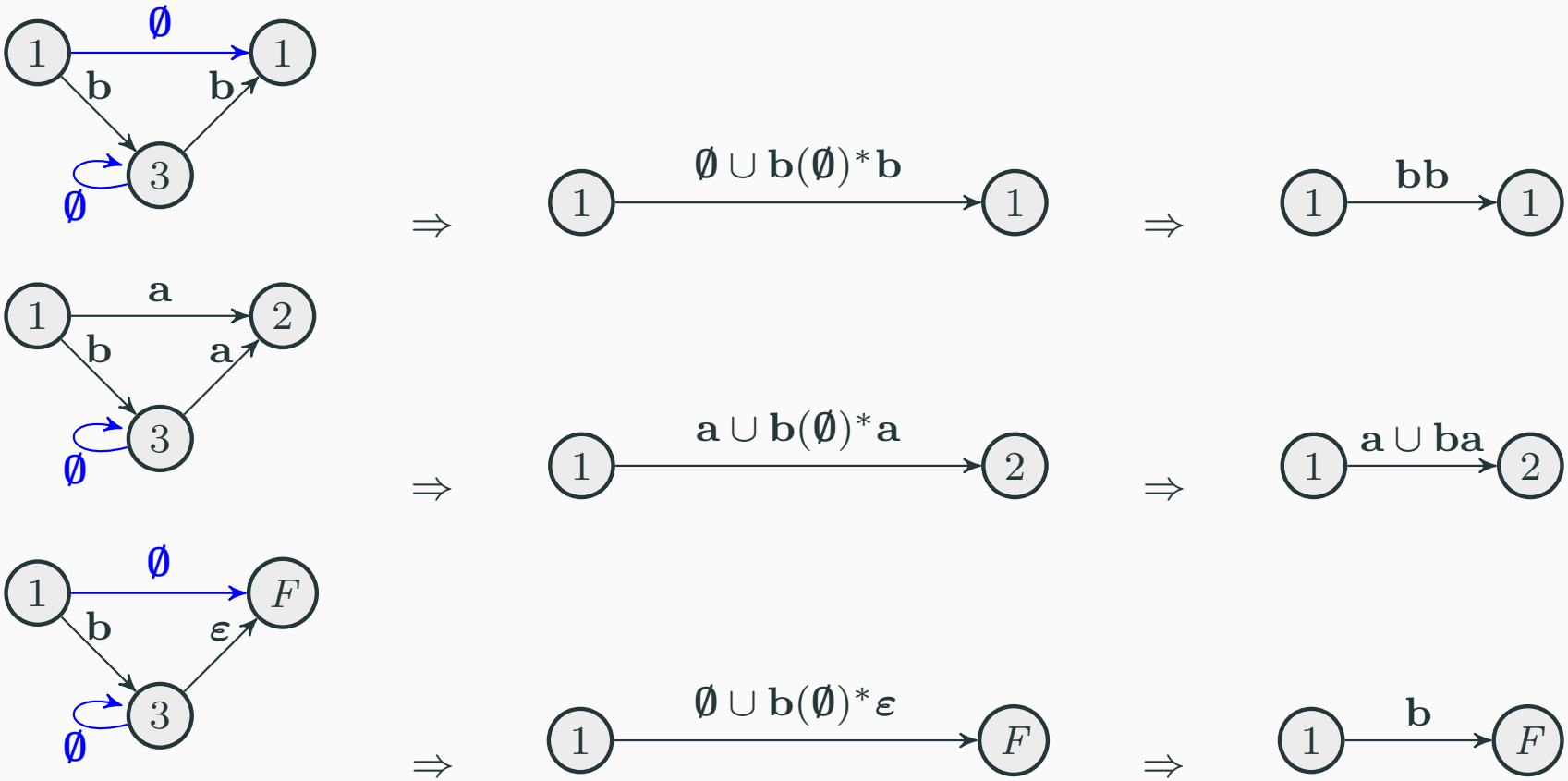
Como $|Q| = 5 > 2$, vamos remover o estado $r = 3$. Nesse caso, q_a pode assumir valores em $\{I, 1, 2\}$ e q_b pode assumir valores em $\{1, 2, F\}$.

Assim, as novas transições serão



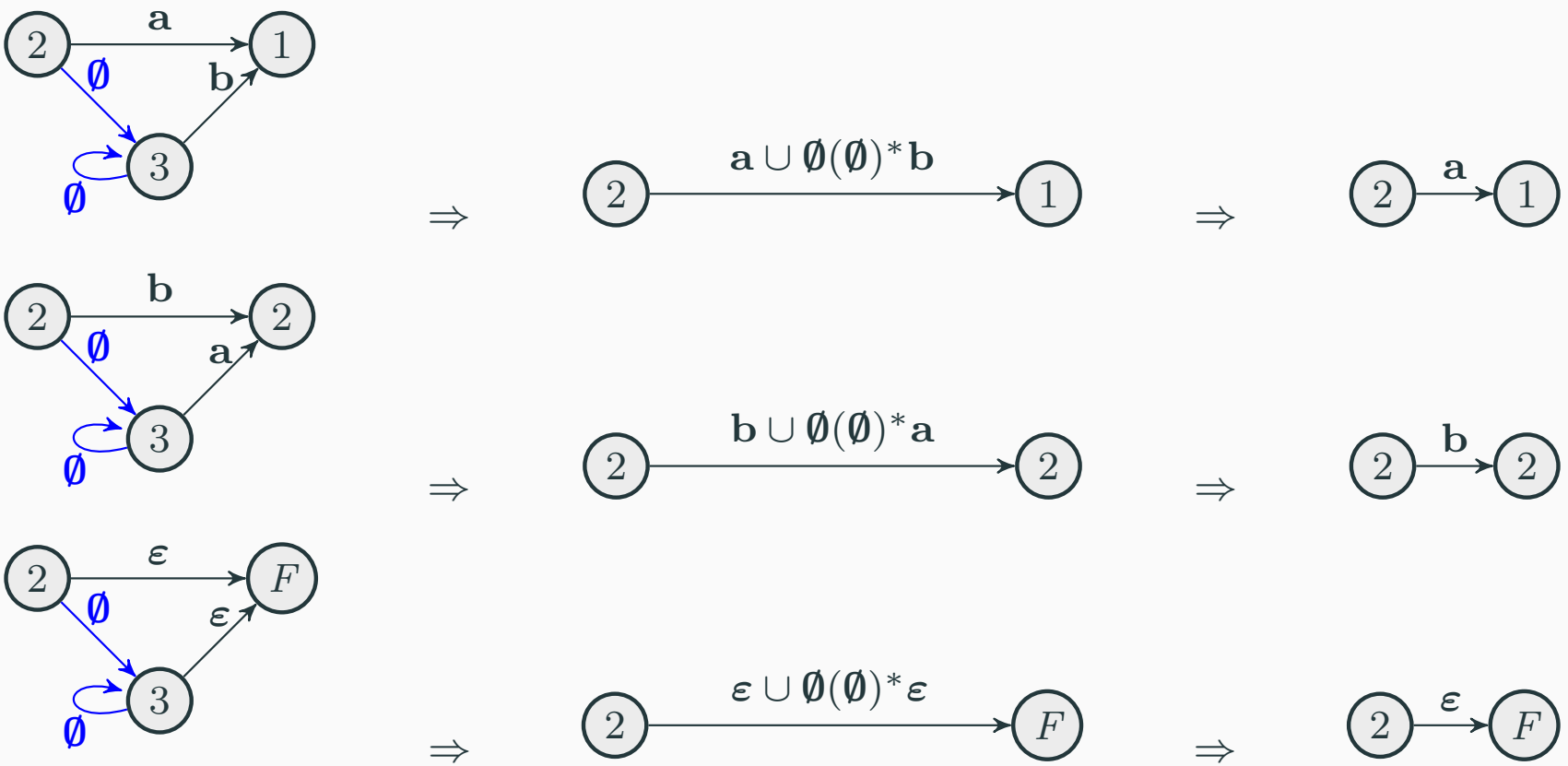
Como $|Q| = 5 > 2$, vamos remover o estado $r = 3$. Nesse caso, q_a pode assumir valores em $\{I, 1, 2\}$ e q_b pode assumir valores em $\{1, 2, F\}$.

Assim, as novas transições serão

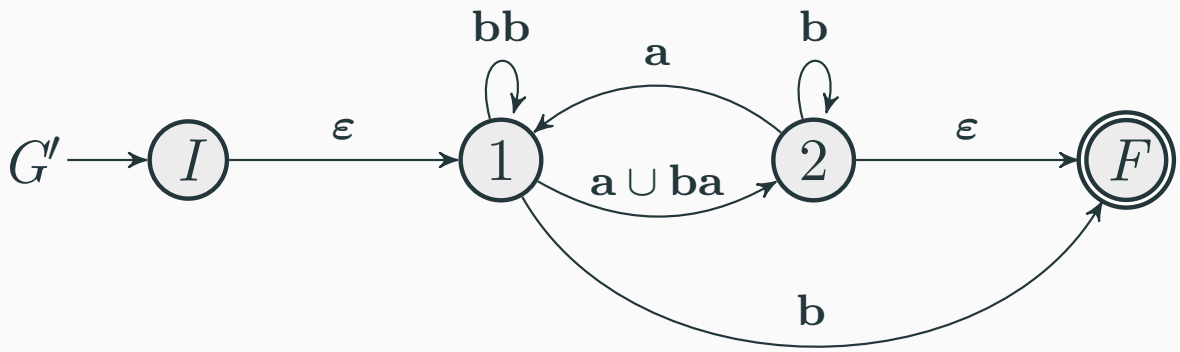


Como $|Q| = 5 > 2$, vamos remover o estado $r = 3$. Nesse caso, q_a pode assumir valores em $\{I, 1, 2\}$ e q_b pode assumir valores em $\{1, 2, F\}$.

Assim, as novas transições serão



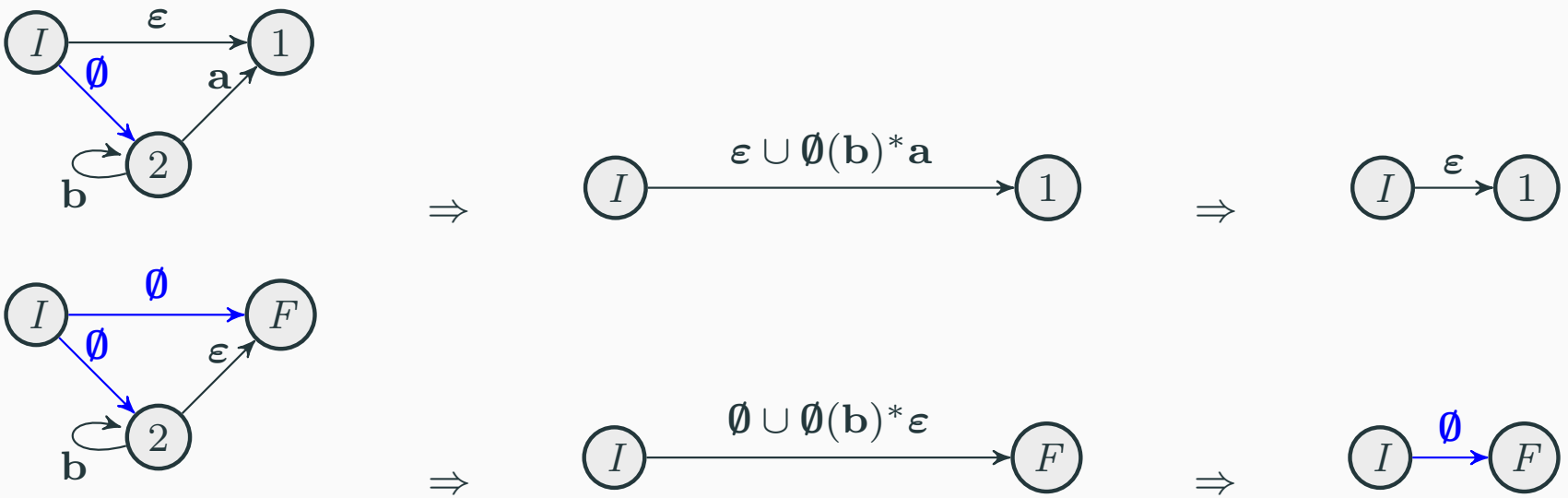
Agora temos (transições $\emptyset$ omitidas)



Vamos executar $\textsc{Converte\_AFNG\_em\_ER}(G')$.

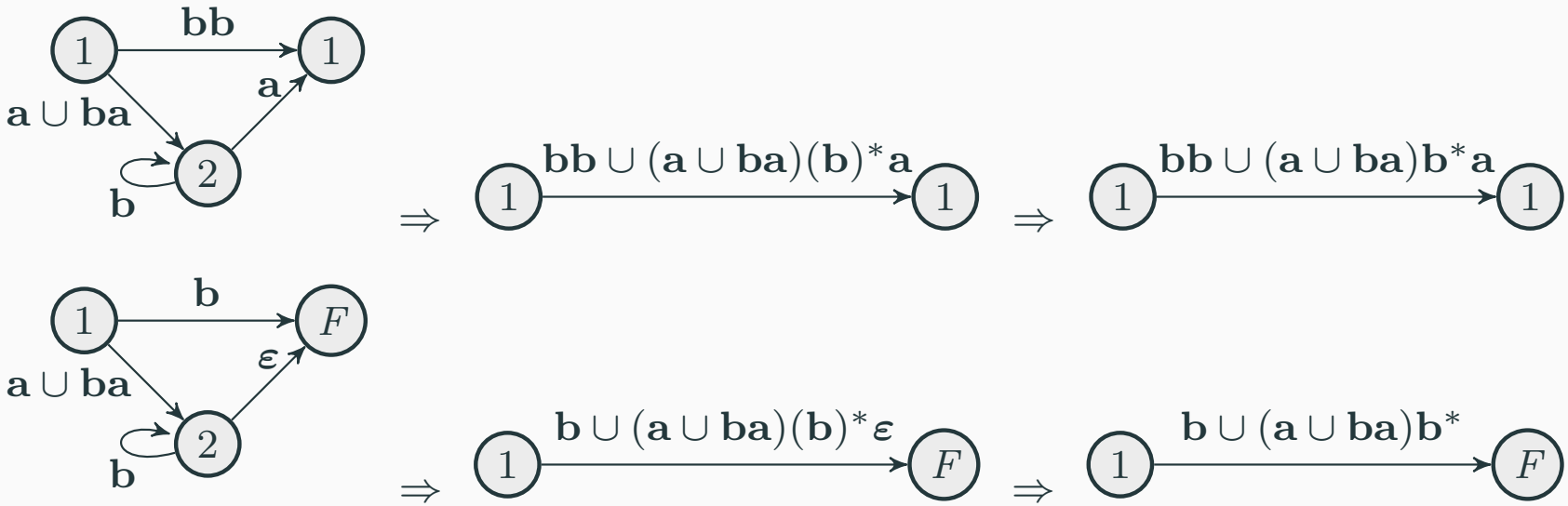
Como $|Q| = 4 > 2$, vamos remover o estado $r = 2$. Nesse caso, q_a pode assumir valores em $\{I, 1\}$ e q_b pode assumir valores em $\{1, F\}$.

Assim, as novas transições serão

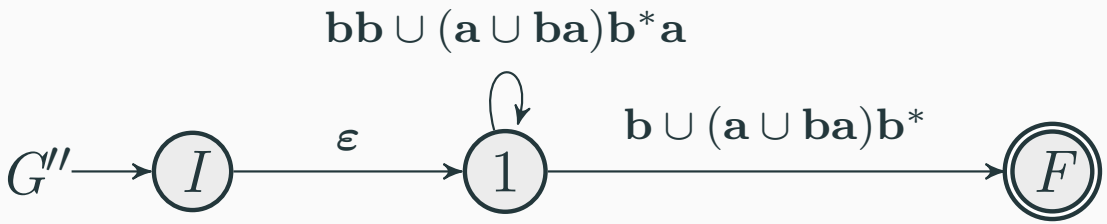


Como $|Q| = 4 > 2$, vamos remover o estado $r = 2$. Nesse caso, q_a pode assumir valores em $\{I, 1\}$ e q_b pode assumir valores em $\{1, F\}$.

Assim, as novas transições serão



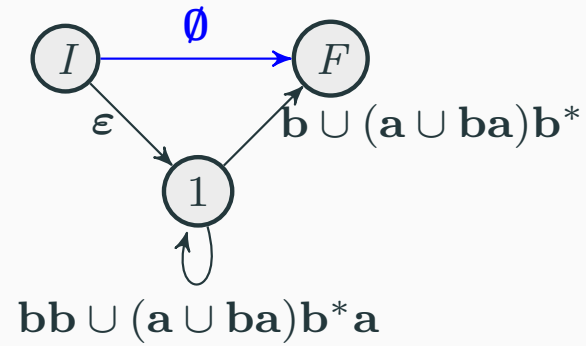
Agora temos (transições $\emptyset$ omitidas)



Vamos executar $\textsc{Converte\_AFNG\_em\_ER}(G'')$.

Como $|Q| = 3 > 2$, vamos remover o estado $r = 2$. Nesse caso, q_a pode assumir valores em $\{I\}$ e q_b pode assumir valores em $\{F\}$.

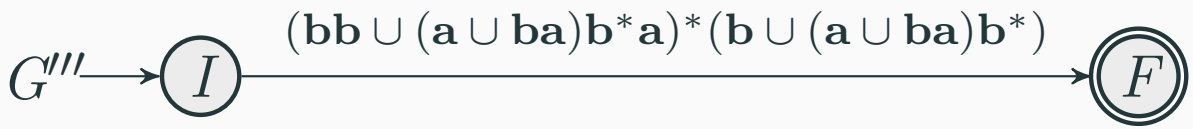
Assim, as novas transições serão



$$\Rightarrow \quad I \xrightarrow{\emptyset \cup \varepsilon(\mathbf{bb} \cup (\mathbf{a} \cup \mathbf{ba})\mathbf{b}^*\mathbf{a})^*(\mathbf{b} \cup (\mathbf{a} \cup \mathbf{ba})\mathbf{b}^*)} F$$

$$\Rightarrow \quad I \xrightarrow{(\mathbf{bb} \cup (\mathbf{a} \cup \mathbf{ba})\mathbf{b}^*\mathbf{a})^*(\mathbf{b} \cup (\mathbf{a} \cup \mathbf{ba})\mathbf{b}^*)} F$$

Agora temos



Ao executar $\textsc{Converte\_AFNG\_em\_ER}(G''')$, entramos no caso base, pois $|Q| == 2$.

O algoritmo então retorna $(\mathbf{bb} \cup (\mathbf{a} \cup \mathbf{ba})\mathbf{b}^*\mathbf{a})^*(\mathbf{b} \cup (\mathbf{a} \cup \mathbf{ba})\mathbf{b}^*)$.

Corolário

Uma linguagem é regular se e somente se alguma expressão regular a descreve.

Demonstração.

($\Rightarrow$) Se a linguagem é regular, então já vimos que um AFN a reconhece. Vimos também no teorema anterior que todo AFN pode ser convertido em uma expressão regular equivalente.
($\Leftarrow$) Se uma expressão regular descreve uma linguagem, então pelo primeiro teorema podemos construir um AFN equivalente. Logo, a linguagem é regular. $\square$

Corolário

Uma linguagem é regular se e somente se alguma expressão regular a descreve.

Demonstração.

($\Rightarrow$) Se a linguagem é regular, então já vimos que um AFN a reconhece. Vimos também no teorema anterior que todo AFN pode ser convertido em uma expressão regular equivalente.
($\Leftarrow$) Se uma expressão regular descreve uma linguagem, então pelo primeiro teorema podemos construir um AFN equivalente. Logo, a linguagem é regular. □

Resposta dos exercícios

Resposta dos exercícios

Por favor, não veja a resposta antes de tentar fazer os exercícios por conta própria!

Resposta dos exercícios

Sério mesmo!

Exercícios

Qual a linguagem descrita pelas seguintes expressões regulares?

1. $\mathbf{0^*10^*} = \{\omega \mid \omega = 0^k10^j \text{ com } k \geq 0 \text{ e } j \geq 0\}$
2. $\mathbf{\Sigma^*001\Sigma^*} = \{\omega \mid \omega \text{ contém a subcadeia } 001\}$
3. $\mathbf{1^*(01^+)^*} = \{\omega \mid \text{ todo 0 em } \omega \text{ é seguido de ao menos um 1}\}$
4. $\mathbf{01 \cup 10} = \{01, 10\}$
5. $\mathbf{0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1} = \{\omega \mid \omega \text{ começa e termina com o mesmo símbolo}\}$

Exercícios

Forneça uma expressão regular para as linguagens, para $\Sigma = \{0, 1\}$:

1. $\{\omega \in \Sigma^* \mid \omega$ tem comprimento par$\}$: $(\mathbf{\Sigma\Sigma})^*$
2. $\{\varepsilon, 0, 1, 01\}$: $(\mathbf{0} \cup \boldsymbol{\varepsilon})(\mathbf{1} \cup \boldsymbol{\varepsilon})$
3. $\{\omega \in \Sigma^* \mid \omega$ não contém 00 nem 11$\}$: $(\boldsymbol{\varepsilon} \cup \mathbf{1})(\mathbf{01})^* \cup (\boldsymbol{\varepsilon} \cup \mathbf{0})(\mathbf{10})^*$ $\equiv (\boldsymbol{\varepsilon} \cup \mathbf{1})(\mathbf{01})^*(\boldsymbol{\varepsilon} \cup \mathbf{0})$
4. $\{\omega \in \Sigma^* \mid$ toda posição ímpar de ω é $1\}$: $\mathbf{1}(\mathbf{\Sigma 1})^* \cup \boldsymbol{\varepsilon}$
5. $\{\omega \in \Sigma^* \mid \omega$ tem pelo menos um 0 e um $1\}$: $\mathbf{\Sigma^* 0 \Sigma^* 1 \Sigma^*} \cup \mathbf{\Sigma^* 1 \Sigma^* 0 \Sigma^*} \equiv \mathbf{0^+ 1 \Sigma^*} \cup \mathbf{1^+ 0 \Sigma^*}$
6. $\{\omega \in \Sigma^* \mid \omega$ tem no máximo uma ocorrência de $11\}$: $(\mathbf{0} \cup \mathbf{10})^*(\boldsymbol{\varepsilon} \cup \mathbf{1}) \cup (\mathbf{0} \cup \mathbf{10})^* \mathbf{11} (\mathbf{0^+ 1})^* (\boldsymbol{\varepsilon} \cup \mathbf{0^+})$
7. $\{\omega \in \Sigma^* \mid$ o número de 0's em ω é divisível por $3\}$: $(\mathbf{1^* 0 1^* 0 1^* 0 1^*})^*$
8. $\{\omega \in \Sigma^* \mid \omega$ não contém 110 como subcadeia$\}$: $(\mathbf{0} \cup \mathbf{10})^*(\boldsymbol{\varepsilon} \cup \mathbf{1^+})$