Problem Set 1

Instructions

Resources useful to solve the exercises in this problem set are the following:

Guille Godoy's video lectures

- Autómatas finitos deterministas
- Autómatas finitos indeterministas
- Notacions de DFAs i NFAs (1)
- Notacions de DFAs i NFAs (2)
- Operacions sobre Reg (1)
- Operacions sobre Reg (2)
- Operacions sobre Reg (3)
- Minimització de DFAs (1)
- Minimització de DFAs (2)
- Minimització de DFAs (3)

Books

- Exemples de construcció d'autòmats finits (Lluís Màrquez, Enrique Romero)
- (Cases and Màrquez 2003, § 4 and § 5)
- (Sipser 2013, § 1.1 and § 1.2)
- (Hopcroft, Motwani, and Ullman 2007, § 2.1, § 2.2 and § 4.2)

Cases, Rafel, and Lluís Màrquez. 2003. *Llenguatges, Gramàtiques i Autòmats : Curs Bàsic.* 2a ed. Edicions UPC.

Hopcroft, John E., Rajeev Motwani, and Jeffrey D. Ullman. 2007. *Introduction to Automata Theory, Languages, and Computation.* 3rd edition. Pearson Addison Wesley.

Sipser, Michael. 2013. *Introduction to the Theory of Computation.* 3rd edition. Cengage Learning.

All exercises

Exercise 1.1 (Union and intersection of regular languages – the product construction)**.** Given a word $w \in \{a, b\}^*$, let

$$L_w = \{xwy \mid x, y \in \{a, b\}^*\}.$$

In other words, L_w is the language of all words containing w as a subword.

(a) Show that for every word w, L_w is a regular language.

(b) Construct minimum DFAs recognizing the following languages. Show that the constructed DFAs are correct and have the smallest possible number of states.

- L_a
- L_{aa}
- L_{aaa}

(c) Using the cartesian product construction, construct DFAs recognizing the following languages and minimize the DFAs obtained.

- $L_{aa} \cup L_{bb}$
- $L_a \cup L_{bbb}$

What would change if we wanted the languages $L_{aa} \cap L_{bb}$ and $L_a \cap L_{bbb}$ instead?

(d) Given two DFAs A and B as input, what is the cost of computing a DFA for $L(A) \cup L(B)$ and $L(A) \cap L(B)$ using the cartesian product construction?

(e) Given minimum DFAs A and B, is the DFA recognizing $L(A) \cup L(B)$ obatained applying the product construction on A and B minimum? What about the DFA for $L(A) \cap L(B)$?

(f) What happens if we apply the cartesian product construction to NFAs instead of DFAs? Does the product construction on NFAs still give a good NFA for the union (resp. intersection)?

Exercise 1.2 (Complement of a regular language is regular)**.** Exchanging final and non-final states in a DFA A produces a new DFA that recognizes the complement of the language recognized by A.

(a) Show that $L = \{aax \mid x \in \{a,b\}^*\}$ is a regular language. Construct the minimum DFAs recognizing L and $\overline{L}$.

(b) Show that $L = \{w \in \{a,b\}^* \mid |w| \in 3\mathbb{N} + 1\}$ is a regular language. Construct the minimum DFAs recognizing L and $\overline{L}$.

(c) Show that $L = \{w \in \{0,1\}^* \mid \texttt{value}_2(w) \in 3\mathbb{N}\}$ is a regular language. Construct the minimum DFAs recognizing L and $\overline{L}$.

(d) Given as input a DFA A, what is the cost of constructing a DFA for $\overline{L(A)}$?

(e) If we started with a minimum DFA, is the obtained DFA for the complement minimum?

(f) Does exchanging final states and non-final states in an NFA A produce an NFA recognizing the complement of the language recognized by A?

Exercise 1.3 (Concatenation of regular languages is regular)**.**

(1) Construct the minimum DFA for the language $L_1 \cdot L_2$, where

(a) $L_1 = \{xaya \mid x, y \in \{a,b\}^*\}$ and $L_2 = \{bxby \mid x, y \in \{a,b\}^*\}$.
(b) $L_1 = \{xaay \mid x, y \in \{a,b\}^*\}$ and $L_2 = \{bxb \mid x \in \{a,b\}^*\}$.
(c) $L_1 = \{xaya \mid x, y \in \{a,b\}^*\}$ and $L_2 = \{bxb \mid x \in \{a,b\}^*\}$.

Find the minimum DFAs recognizing L_1 and L_2. From those DFAs, construct a λ-NFA A recognizing the language $L_1 \cdot L_2$. Now, using the power-set construction, make A deterministic and minimize the DFA obtained.

(2) Given two DFAs A and B as input, what is the cost of constructing a DFA for $L(A) \cdot L(B)$?

Exercise 1.4 (Kleene star of a regular language is regular)**.**

(1) Explicitly construct the minimum DFA for the language L^*, where

  (a) $L = \{xay \in \{a,b\}^* \mid |y| = 1\}$.
  (b) $L = \{xaby \in \{a,b\}^* \mid |y| = 1\}$.
  (c) $L = \{axaby \in \{a,b\}^* \mid |y| = 1\}$.

  Construct the minimum DFA recognizing L. From that DFA, construct a λ-NFA A recognizing the language L^*. Using the power-set construction, make A deterministic and then minimize the DFA obtained.

(2) Given a DFA A as input, what is the cost of computing a DFA for $L(A)^*$?

Exercise 1.5 (Reverse of a regular language is regular)**.**

(1) Explicitly construct the minimum DFA for the language L^R, where

  (a) $L = \{w \in \{a,b\}^* \mid \forall w_1, w_2\ (w = w_1 a w_2 \Rightarrow |w_1|_b \in 2\mathbb{N})\}$.
  (b) $L = \{w \in \{a,b\}^* \mid \forall w_1, w_2\ (w = w_1 a w_2 \Rightarrow |w_1|_b \in 2\mathbb{N}+1)\}$.
  (c) $L = \{w \in \{a,b\}^* \mid \forall w_1, w_2\ (w = w_1 a w_2 \Rightarrow |w_1| \in 2\mathbb{N})\}$.

  Construct the minimum DFA recognizing L. From that DFA, construct an NFA A recognizing the language L^R. Now, using the power-set construction, make A deterministic and then minimize the DFA obtained.

(2) Given a DFA A as input, what is the cost of constructing a DFA for $L(A)^R$?

(3) Reversing the direction of transitions and exchanging initial and final states in a DFA A gives an NFA B for $L(A)^R$. If the obtained NFA B is actually a DFA and A was minimum, is B minimum too?

(4) An NFA is *uniquely accepting* if for every word there is a unique accepting execution. Show that, for a uniquely accepting NFA A, the NFA A^R is uniquely accepting.

Exercise 1.6 (Homomorphism of a regular language is regular)**.**

(1) Compute explicitly the minimum DFA for the language $\sigma(L)$, where

   (a) $L = \{axbya \mid x, y \in \{a, b\}^*\}$ and σ is the homomorphism defined by $\sigma(a) = aa$ and $\sigma(b) = ba$.
   (b) $L = \{axbyc \mid x, y \in \{a, b, c\}^*\}$ and σ is the homomorphism defined by $\sigma(a) = ab$, $\sigma(b) = b$, and $\sigma(c) = \lambda$.
   (c) $L = \{xbcya \mid x, y \in \{a, b, c\}^*\}$ and σ is the homomorphism defined by $\sigma(a) = bbb$, $\sigma(b) = a$,, and $\sigma(c) = \lambda$.

   Compute the minimum DFA recognizing L. From that DFA construct a λ-NFA A recognizing the language $\sigma(L)$. Now, using the power-set construction make A deterministic and minimize the DFA obtained.

   Recall that given a DFA A and a homomorphism σ, it is possible to construct an NFA recognizing the language $\sigma(L(A))$ by transforming each transition $\xrightarrow{a}$ in A into $\xrightarrow{\sigma(a)}$ in an *extended* automaton and converting each extended transition into a λ-NFA by adding new states.

(2) Given as input a DFA A, what is the cost of computing a DFA for $\sigma(L(A))$? Does the construction to obtain an NFA recognizing $\sigma(L(A))$ give a DFA?

(3) Does the construction to obtain an NFA recognizing $\sigma(L(A))$ still work if we started with an NFA A? In other words, if each transition $\xrightarrow{a}$ in an NFA A is transformed into the extended transition $\xrightarrow{\sigma(a)}$ and then converted into a λ-NFA by adding new states, does that give a correct λ-NFA for $\sigma(L(A))$?

Exercise 1.7 (Inverse homomorphism of a regular language is regular)**.**

(1) Show that $L = \{w \in \{0,1\}^* \mid \texttt{value}_2(w) \in 3\mathbb{N}\}$ is a regular language. Compute explicitly the minimum DFA for the language $\sigma^{-1}(L)$, where $\sigma : \{a,b,c\} \to \{0,1\}$ is the homomorphism defined by

   (a) $\sigma(a) = 01$,
   $\sigma(b) = 0$, and
   $\sigma(c) = \lambda$.
   (b) $\sigma(a) = 10$,
   $\sigma(b) = 0$, and
   $\sigma(c) = \lambda$.
   (c) $\sigma(a) = 00$,
   $\sigma(b) = 11$, and
   $\sigma(c) = \lambda$.
   (d) $\sigma(a) = 001$,
   $\sigma(b) = 101$, and
   $\sigma(c) = 0$.

   Recall that, given a DFA A and a homomorphism σ, it is possible to construct a DFA A' recognizing the language $\sigma^{-1}(L(A))$ as follows: on input w, A' will run A on input $\sigma(w)$ and accept if A does.

(2) Given a DFA A and a morphism σ, what is the cost of constructing a DFA for the language $\sigma^{-1}(L(A))$?

(3) Does the construction used to obtain the DFA for $\sigma^{-1}(L(A))$ give us a minimum DFA if we started with a minimum DFA A?

(4) Does the construction used to obtain the DFA for $\sigma^{-1}(L(A))$ still work if we started with an NFA A?

Exercise 1.8 (On DFA minimization)**.**

(a) A DFA with unreachable states cannot be minimum. What is the cost of determining whether a DFA has unreachable states?

(b) What is the cost of Moore minimization algorithm (with a reasonable implementation)?

(c) The minimum DFA recognizing a given language is unique up to isomorphism. What about NFAs? In particular, is an NFA of minimum size unique for a given language?

Exercise 1.9 (NFAs can be exponentially more succinct than DFAs)**.** Given an $n \in \mathbb{N}$, consider the language $L_n = \{xay \mid x, y \in \{a,b\}^* \wedge |y| = n\}$.

(1) What is the cost of the determinization algorithm (as a function of the size of the input NFA)?

(2) Show that L_n is regular by constructing an NFA recognizing L_n and having $n+2$ states.

(3) Show that the minimum DFA recognizing L_n has 2^{n+1} states. In other words,

   (a) show that there is a DFA recognizing L_n having 2^{n+1} states and
   (b) show that no DFA with strictly less than 2^{n+1} states can recognize L_n.

Remember that the compact notation used in the definition of L_n corresponds to the language

$$\{w \in \{a,b\}^* \mid \exists x, y \ (w = xay \wedge |y| = n)\}.$$

Exercise 1.10 (DFAs – some decidable properties)**.** Show that the following computational problems are *decidable* by providing an algorithm (with reasonable cost) that solves them.

(a) Given a DFA A, is $L(A) = \emptyset$?
(b) Given a DFA A, is $L(A)$ an infinite set?
(c) Given two DFAs A and B, is $L(A) = L(B)$?
(d) Given two DFAs A and B, is $L(A) \subseteq L(B)$?

Exercise 1.11 (Finite languages are regular)**.**

(a) Show that for every word w, the language $\{w\}$ is regular.

(b) Show that every finite language, i.e. any language consisting only of a finite number of words, is regular.

(c) A language L is *co-finite* if its complement $\overline{L}$ is finite. Show that if a language is co-finite, then it is regular.

(d) Show that the language $L = \{0^n \mid \text{the decimal expansion of } \pi \text{ contains } n \text{ consecutive 0s}\}$ is regular.

> Hint
>
> No knowledge about any property of the decimal expansion of π is needed (apart from being infinite). Think instead of a nonconstructive proof by cases.

(e) A crucial part of the definition of DFAs is that they are only allowed to have a **finite** number of states. Show that the definition would become trivial if DFAs were allowed to have an infinite number of states, in the sense that every language (say over the alphabet $\{a, b\}$) could be recognized by a *DFA* if we allowed it to have an infinite number of states.

Exercise 1.12 (Membership in a regular language is decidable in polynomial time)**.** Consider the following decision problem:

$$\texttt{Membership}_{\texttt{Reg}}: \text{ given an input } x \in \{0,1\}^* \text{ and a DFA } A, \text{ determine whether } x \in L(A).$$

Show that $\texttt{Membership}_{\texttt{Reg}}$ can be decided in polynomial time w.r.t. $|x|$ and the size of A.

Exercise 1.13 (Arithmetic progressions are regular)**.** Let $a = (a_1, a_2, \dots)$ be an arithmetic progression, i.e. a sequence in which the difference between any two consecutive terms is a constant. The goal of this exercise is (somewhat informally) to show that regardless of the base chosen to represent the arithmetic progression, this gives rise to a regular language.

More precisely:

(a) Show that the language $\{1^m \mid m \in a\}$ is regular.

(b) Given $n \in \mathbb{N}$, how many states does the minum DFA recognizing $\{1^m \mid m \in n\mathbb{N}\}$ have?

(c) Show that the language $\{w \in \{0,1\}^* \mid \texttt{value}_2(w) \in a\}$ is regular.

(d) Given $n \in \mathbb{N}$, how many states does the minimum DFA recognizing $\{w \in \{0,1\}^* \mid \texttt{value}_2(w) \in n\mathbb{N}\}$ have when $n = 2^k$ for some $k \in \mathbb{N}$? What about when n is odd? What about when $n = m2^k$ for some $k \in \mathbb{N}$ and m is odd?

(e) Show that the language $\{w \in \{0,1,2\}^* \mid \texttt{value}_3(w) \in a\}$ is regular.

(f) Show that the language $\{w \in \Sigma^* \mid \texttt{value}_b(w) \in a\}$ is regular, where $b \geq 2$ and $\Sigma = \{0, 1, 2, \dots, b-1\}$ is an alphabet of digits.

Remember that, for any $b \geq 2$, $\texttt{value}_b(w)$ denotes the number obtained interpreting the string w as a number written using base b. For instance,

$$\texttt{value}_2(00101) = 1 \cdot 2^0 + 0 \cdot 2^1 + 1 \cdot 2^2 + 0 \cdot 2^3 + 0 \cdot 2^4 = 1 + 4 = 5,$$

$$\texttt{value}_3(0121) = 1 \cdot 3^0 + 2 \cdot 3^1 + 1 \cdot 3^2 + 0 \cdot 3^3 = 1 + 6 + 9 = 16.$$

Exercise 1.14 (At least k occurrences of every symbol is a regular language)**.** Given $k \in \mathbb{N}$, consider the language

$$L_k = \{w \in \{a, b, c\}^* \mid |w|_a \geq k \wedge |w|_b \geq k \wedge |w|_c \geq k\} \ .$$

(a) Show that for any k, L_k is a regular language.

(b) How many states (as a function of k) does the minimum DFA recognizing L_k have?

Exercise 1.15 (First half of regular is regular)**.** Given a language L, we define $\texttt{FirstHalf}(L)$ as the set of strings that constitute the *first half* of strings of even length in L, that is,

$$\texttt{FirstHalf}(L) = \{x \mid \exists y\,(|x| = |y| \,\wedge\, xy \in L)\}.$$

Show that if L is regular, then $\texttt{FirstHalf}(L)$ is regular.

Exercise 1.16 (IntercalAND of regulars is regular)**.** Given two languages $L_1, L_2 \subseteq \Sigma^*$, define

$\texttt{intercalAND}(L_1, L_2) = \{x_1y_1...x_ny_n \mid (n \geq 1) \wedge (x_1, \ldots, x_n, y_1, \ldots, y_n \in \Sigma) \wedge (x_1 \cdots x_n \in L_1) \wedge (y_1 \cdots y_n \in L_2)\}$.

Show that if L_1 and L_2 are regular, then $\texttt{intercalAND}(L_1, L_2)$ is also regular.

Exercise 1.17 (Prefixes and suffixes)**.** Given a language L, define

$$\texttt{Prefixes}(L) = \{w \mid \exists x \ (wx \in L)\}$$

and

$$\texttt{Suffixes}(L) = \{w \mid \exists x \ (xw \in L)\}.$$

(a) Given a DFA A, how can we construct a DFA to recognize the language $\texttt{Prefixes}(L(A))$?

(b) Given a DFA A, how can we construct a DFA to recognize the language $\texttt{Suffixes}(L(A))$?

Exercise 1.18 (Variations on $a^n b^n$)**.** We saw in class that $A = \{a^n b^n \mid n \in \mathbb{N}\}$ is not a regular language.

(1) Consider now a language $L \subseteq A$. Show that L is regular if and only if L is finite.

> Hint
>
> Before proving the general result it might be easier to prove the special cases to get ideas on how to proceed in general.
>
> - How would you show that $\{a^n b^n \mid n \in 2\mathbb{N}\}$ is not regular?
> - How would you show it with $\{a^n b^n \mid n \in 3\mathbb{N}\}$?

(2) Show that the following languages are not regular.

(a) $\{a^n b^m \mid n, m \in \mathbb{N} \land n \leq m\}$
(b) $\{a^n b^m \mid n, m \in \mathbb{N} \land n \geq m\}$
(c) $\{a^n b^m \mid n, m \in \mathbb{N} \land n \neq m\}$
(d) $\{a^{2n} b^n \mid n \in 2\mathbb{N}\}$

> Hint
>
> Using the pumping lemma directly is doable but a bit tricky. It is simpler to use first closure properties of regular languages.

(3) Show that the following languages over $\{a, b, c\}$ are not regular.

(a) $\{c^m a^n b^n \mid n, m \in \mathbb{N}\}$
(b) $\{a^n c^m b^n \mid n, m \in \mathbb{N}\}$
(c) $\{a^n b^n c^m \mid n, m \in \mathbb{N}\}$
(d) $\{a, b\}^* \cup \{c^m a^n b^n \mid n, m \in \mathbb{N}\}$

(4) Show that the *Dyck language* is not regular, that is the language of all well-balanced parentheses is not regular. More precisely, given the alphabet $\Sigma = \{(,)\}$, show that the language

$$\{w \in \Sigma^* \mid |w|_(= |w|_) \land \text{for every prefix } u \text{ of } w \;\; |u|_(\geq |u|_)\}$$

is not regular.

Exercise 1.19 (Counting as and bs is (in general) not regular)**.** Show that the following languages are not regular.

(a) $\{w \in \{a,b\}^* \mid |w|_a = |w|_b\}$
(b) $\{w \in \{a,b\}^* \mid |w|_a \geq |w|_b\}$
(c) $\{w \in \{a,b\}^* \mid |w|_a \leq |w|_b\}$
(d) $\{w \in \{a,b\}^* \mid |w|_a \neq |w|_b\}$
(e) $\{w \in \{a,b,c\}^* \mid |w|_a \geq |w|_b \vee |w|_b \geq |w|_c\}$
(f) $\{w \in \{a,b\}^* \mid |w| \in 3\mathbb{N} \Rightarrow |w|_a = |w|_b\}$

Exercise 1.20 (On as in the first part and bs in the second part)**.** Given $k \in \mathbb{N}$, show that $L_k = \{xy \in \{a,b\}^* \mid |x|_a = k|y|_b\}$ is regular if and only if $k = 1$ or $k = 0$.

Hint

The hard part is to show that for $k > 1$, the language L_k is not regular. Think of $k = 2$ first. The argument for a generic k is a simple generalization of this case.

More hints

Consider the reverse of L_2.

Exercise 1.21 (Palindromes and partial palindromes)**.**

(a) Show that the language $\{w \in \{a,b\}^* \mid w = w^R\}$ is not regular.

(b) Show that the language $\{ww^R \mid w \in \{a,b\}^*\}$ is not regular.

(c) Show that the language $\{ww \mid w \in \{a,b\}^*\}$ is not regular.

(d) (**hard**) Show that the language $\{ww^R x \mid w, x \in \{a,b\}^+\}$ is not regular.

Caution

For this last point, the pumping lemma cannot be used to show non-regularity. Why?

Hint

Suppose, towards a contradiction, that the language is regular and there is a DFA with N states recognizing it. Test the behaviour of the automata against all the words $w_i = aba^2b^2 \cdots a^i b^i$, for $i = 1, 2, \dots, N+1$.

Exercise 1.22 (Checking basic arithmetic is not regular)**.** Show that the following languages over alphabet $\{0, 1, \#\}$ are not regular.

(a) $\{u\#v \mid u, v \in \{0,1\}^* \ \wedge \ v \text{ is a subword of } u\}$
(b) $\{u\#v \mid u, v \in \{0,1\}^* \ \wedge \ (|u| < |v| \vee |u| \in 2\mathbb{N})\}$
(c) $\{u\#v \mid u, v \in \{0,1\}^* \ \wedge \ \texttt{value}_2(u) = \texttt{value}_2(v)\}$
(d) $\{u\#v \mid u, v \in \{0,1\}^* \ \wedge \ \texttt{value}_2(u) = \texttt{value}_2(v) + 1\}$
(e) $\{u\#v\#z \mid u, v \in \{0,1\}^* \ \wedge \ \texttt{value}_2(u) + \texttt{value}_2(v) = \texttt{value}_2(z)\}$

Exercise 1.23 (Approximations of real numbers)**.** Given a real number $r \in [0, 1)$, let $L_r \subseteq \{0, 1\}^*$ be the language consisting of non-empty words w, where w coincides with the first $|w|$ digits of the binary expansion of r. For instance,

- $1/2$ in binary is $.1$, and hence $L_{1/2} = \{1, 10, 100, 1000, \ldots\}$
- $1/3$ in binary is $.01010101\ldots$, and hence $L_{1/3} = \{0, 01, 010, 0101, 01010, \ldots\}$

(a) Show that L_r is a regular language if and only if r is a rational number.
(b) Argue why for almost all real numbers $r \in [0, 1)$, L_r is not a regular language.

Exercise 1.24 (Unary sequences with arbitrarily large gaps)**.** Let $a = (a_1, a_2, a_3, \dots)$ be an ordered sequence of natural numbers. We say that the sequence a has arbitrarily large gaps if for any $n \in \mathbb{N}$ there is an index i such that $a_{i+1} > a_i + n$.

(1) Show that a has arbitrarily large gaps when

  (a) $a_i = 2^i$
  (b) $a_i = i^2$
  (c) a_i is the i-th Fibonacci number
  (d) a_i is the i-th prime number

(2) Given a sequence a with arbitrarily large gaps, show that the language $L_a = \{1^k \mid k \in a\}$ is not regular.

> Hint
>
> Try to prove that L_a is not regular in the following two cases *before* proving the general result: $a_i = 2^i$ and $a_i = i^2$. This might give you ideas on how to proceed in the general case.

(3) Using the ideas from the previous questions, show that the following languages are not regular:

  (a) $\{0101^201^3 \cdots 01^n \mid n \in \mathbb{N}\}$
  (b) $\{1^n \mid n \text{ is even or prime}\}$

Exercise 1.25 (What operations preserve non-regularity?)**.** Let A and B be two **non-regular** languages and σ a homomorphism. Which of the following languages can we always assure is non-regular? Justify your answer (or give a counter-example if this is not true).

(a) $\bar{A}$.
(b) $A \cup B$.
(c) $A \cap B$.
(d) $A \cdot B$.
(e) A^R.
(f) A^*.
(g) $S(A)$ (recall that with $S(A)$ we denote the *shift* of the language A, see Problem Set 1).
(h) $\sigma(A)$.
(i) $\sigma^{-1}(A)$.

Exercise 1.26 (Arden's Lemma and applications)**.**

(1) **Arden's Lemma**. Given languages $A, B \subseteq \Sigma^*$ and the equation

$$X = AX \cup B, \tag{1}$$

show that

a. $X = A^*B$ is a solution of (Equation 1), that is $A^*B = AA^*B \cup B$;
b. if L is a solution of (Equation 1) then $L \supseteq A^*B$;
c. if $\lambda \notin A$ then A^*B is the unique solution of (Equation 1).

(2) **Symmetric version of Arden's Lemma**. Given $A, B \subseteq \Sigma^*$ and the equation

$$Y = YA \cup B, \tag{2}$$

show that

a. $Y = BA^*$ is a solution of (Equation 2);
b. if L is a solution of (Equation 2) then $L \supseteq BA^*$;
c. if $\lambda \notin A$ then BA^* is the unique solution of (Equation 2).

(3) For each of the following languages L, give a DFA A_L recognizing L and two regular expressions representing L. Obtain the regular expressions using Arden's lemma and the symmetric version of Arden's lemma on A_L where

(a) L is the language of words on $\{a, b\}$ with an even number of as;
(b) L is the language of words on $\{a, b\}$ with either an even number of as or an even number of bs;
(c) L is the language of words on $\{a, b\}$ ending with $ababa$;
(d) L is the language of words on $\{a, b\}$ not containing the subword aba;
(e) L is the language of words on $\{a, b, c\}$ such that between every two as there is at least one b;
(f) L is the language of words on $\{0, 1\}$ with at least two consecutive 0s;
(g) $L = \{w \in \{0, 1\}^* \mid \texttt{value}_2(w) \in 3\mathbb{N}\}$.

Tip

Given a DFA A, we can associate to each of its states q two variables:

$$X_q = \text{the language of the words that in } A \text{ bring us from } q \text{ to an accepting state}$$
$$Y_q = \text{the language of the words that in } A \text{ bring us from the initial state to } q$$

Using the variables above, we can then set-up two systems of equations and solve them using Arden's lemma (and its symmetric version). The system that uses the variables X_q can be resolved using Arden's lemma, while the one using Y_q can be solved using the symmetric version of Arden's lemma.

Exercise 1.27 (Regular expressions and closure properties of regular languages)**.** We know that regular expressions represent exactly the regular languages and we know that regular languages are closed under several operations. This exercise is about finding regular expressions for the languages after the application of one of such operations.

Given as input regular expressions r_1 and r_2, representing respectively languages L_1 and L_2, by construction $(r_1) + (r_2)$, $(r_1)(r_2)$, and $(r_1)^*$ represent respectively the languages $L_1 \cup L_2$, $L_1 L_2$, and L_1^*. *What about the other operations that preserve regularity?*

(a) *(complement)* Given as input a regular expression r, representing the language L, give an algorithm to find the regular expression representing the language $\overline{L}$. What is the asymptotic cost of the algorithm proposed as a function of the number of symbols in r?

(b) *(intersection)* Given as input regular expressions r_1 and r_2, representing respectively languages L_1 and L_2, give an algorithm to find a regular expression representing the language $L_1 \cap L_2$. What is the asymptotic cost of the algorithm proposed as a function of the number of symbols in r_1 and r_2?

(c) *(reverse)* Given as input a regular expression r, representing the language L, give an algorithm to find a regular expression representing the language L^R. What is the asymptotic cost of the algorithm proposed as a function of the number of symbols in r?

(d) *(homomorphism)* Given as input a regular expression r, representing the language L, and a homomorphism σ, give an algorithm to find a regular expression representing the language $\sigma(L)$.

(e) *(inverse homomorphism)* Given as input a regular expression r, representing the language L, and a homomorphism σ, give an algorithm to find a regular expression representing the language $\sigma^{-1}(L)$.

Exercise 1.28 (On some decidable properties of regular expressions)**.** Let L_r be the (regular) language represented by the regular expression r. Given as input regular expressions r and s, describe an algorithm to decide whether

(a) $L_r = L_s$.
(b) $L_r \subseteq L_s$.
(c) $L_r = \emptyset$.
(d) L_r is infinite.
(e) $L_r \cap L_s = \emptyset$.
(f) $L_r \cap L_s$ is infinite.

What is the asymptotic cost of the algorithm proposed as a function of the number of symbols in r and s?

Exercise 1.29 (Transformation of regular expressions)**.** We say that two regular expressions p, q are equivalent ($p \equiv q$) if the languages represented by p and q (resp. $L(p)$ and $L(q)$) are the same. To check whether two regular expressions p, q are equivalent one could always construct the associated DFAs and check whether they accept the same language. This is computationally expensive.[1]

This exercise is about checking the equivalence of two regular expressions using simple algebraic manipulations. Show that for all regular expressions p, q, and r:

(a) $(p+q)^* \equiv p^*(qp^*)^*$.
(b) $p(qp)^* \equiv (pq)^*p$.
(c) $(p+q^*)^* \equiv (p+q)^*$.
(d) If $p \equiv q$, then $pr \equiv qr$ and $rp \equiv rq$.
(e) If $L(q) \subseteq L(p)$, then $p^*q^* \equiv q^*p^* \equiv p^*$.
(f) $p^* \equiv (\lambda+p)^* \equiv (\lambda+p)(p^*pp)^*$.
(g) $p^*pp+\lambda \equiv (p^*pp)^* \equiv (pp+ppp)^*$.
(h) $p^*(q+rp^*)^* \equiv (p+q^*r)^*q^*$.
(i) $(qq+qp+p)^*qpp^* \equiv p^*q(pp^*q+qp^*q)^*pp^*$.
(j) $(\lambda+b)a^*(b+bba^*)^* \equiv b^*(a+bb+bbb)^*b^*$.

Tip

To prove some of the items (especially, the last three), it is useful to apply previous equivalences.

[1]Deciding whether two regular expressions are equivalent is **PSPACE**-complete, i.e., informally, it is the hardest among the decision problems solvable using a polynomial amount of memory, but no limitations on the running time.