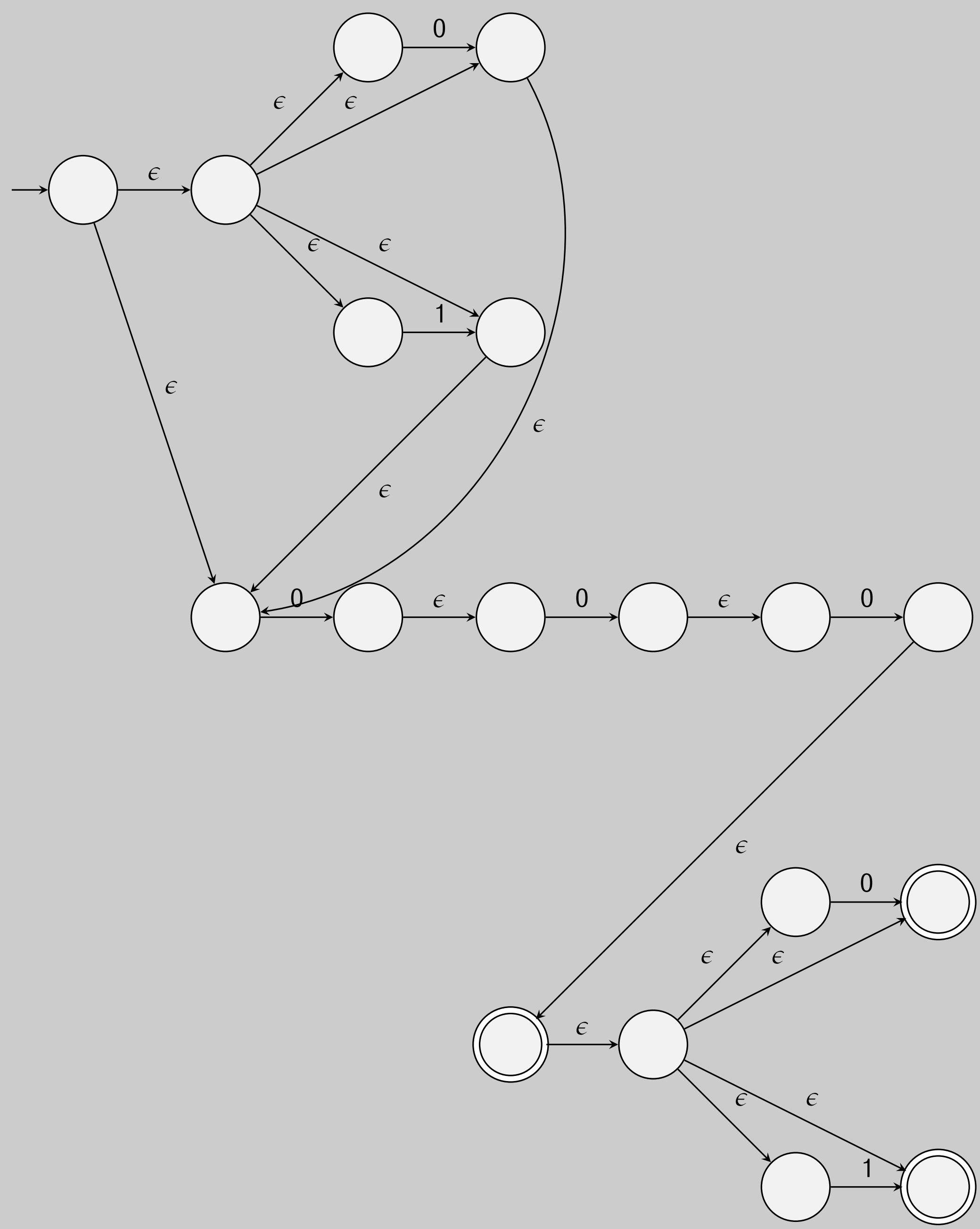
1.19 Use the procedure described in Lemma 1.55 to convert the following regular expressions to nondeterministic finite automata.

(a) $(0 \cup 1)^* 000 (0 \cup 1)^*$

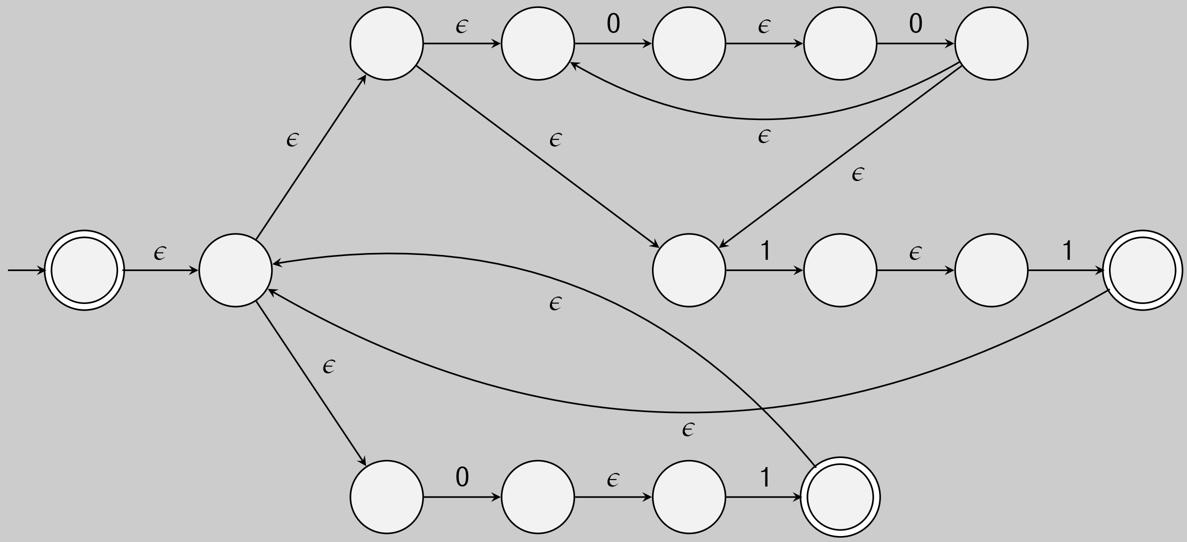
Solution.



■

(b) $(((00)^*(11)) \cup 01)^*$

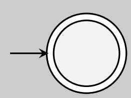
Solution.



■

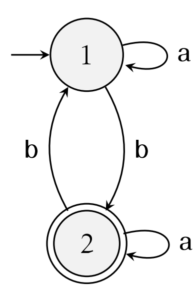
(c) $\emptyset^*$

Solution.



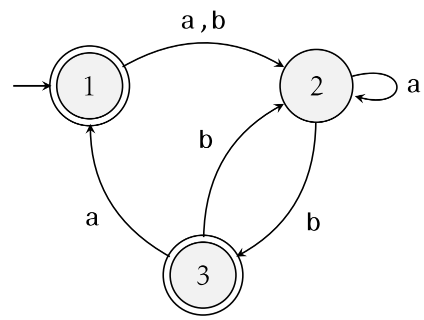
■

1.21 Use the procedure described in Lemma 1.60 to convert the following finite automata to regular expressions.



(a)

Solution. $a^*b\ (a \cup ba^*b)^*$ ■



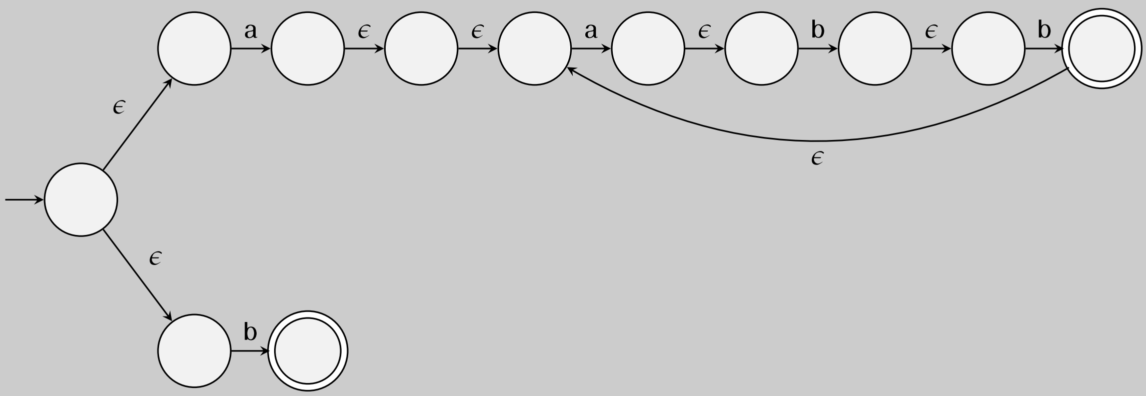
(b)

Solution. $\epsilon \cup ((a \cup b)a^*b)((a(a \cup b) \cup b)a^*b)^*(\epsilon \cup a)$ ■

1.28 Convert the following regular expressions to NFAs using the procedure given in Theorem 1.54. In all parts, $\sum = \{a, b\}$.

(a) $a(abb)^* \cup b$

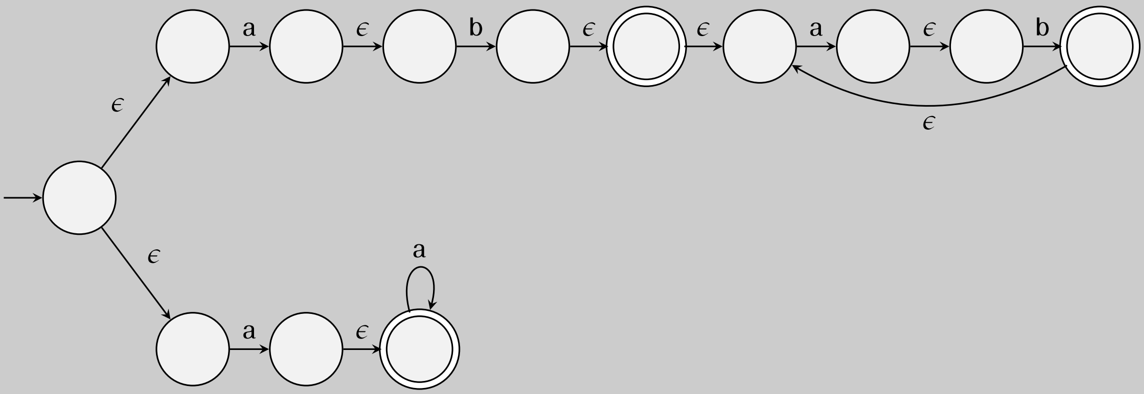
Solution.



■

(b) $a^+ \cup (ab)^+$

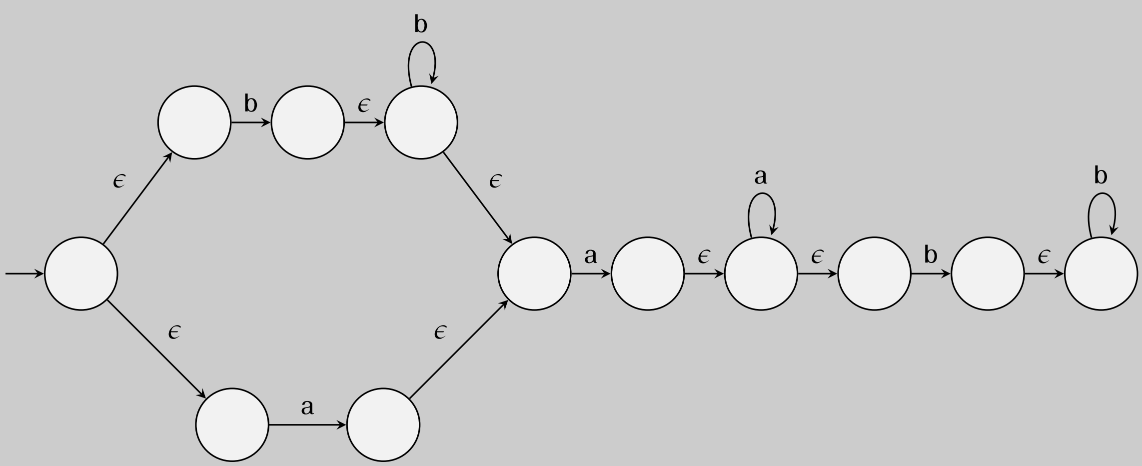
Solution.



■

(c) $(a \cup b^+)a^+b^+$

Solution.



■

1.29 Use the pumping lemma to show that the following languages are not regular.

(a) $A_1 = \{0^n, 1^n, 2^n \mid n \geq 0\}$

Solution. Suppose that A_1 is a regular language, then A_1 must satisfy the three conditions of the pumping lemma. Consider the condition, $xy^iz \in A_1$ for each i ≥ 0. Assume that i = 2, then $xy^iz =$ 0112, which is not in A_1, hence the condition is not satisfied. Thus A_1 is not a regular language. ■

(b) $A_2 = \{www \mid w \in \{a,b\}^*\}$

Solution. Suppose that A_2 is a regular language, let p be the pumping length of the Pumping Lemma. Consider the string s = ababab. Then we can split the string s into 3 parts s = xyz such that x = ab, y = a, and z = bab satisfying the conditions of the Pumping Lemma. Consider the condition, $xy^iz \in A_2$ for each i ≥ 0. Assume that i = 2, then xy^iz = abaabab which is not in A_2, hence the condition is not satisfied. Thus A_2 is not a regular language. ■

(c) $A_3 = \{a^{2^n} \mid n \geq 0\}$ (Here, a a^{2^n} means a string of 2^n in a's.

Solution. Suppose that A_3 is a regular language, let p be the pumping length of the Pumping Lemma. Consider the string s = aaaa. Then we can split the string s into 3 parts s = xyz, such that x = a, y = a, z = aa satisfying the conditions of the Pumping Lemma. Consider the condition $xy^iz \in A_3$ for each i ≥ 0. Assume that i = 2, then $xy^iz = aa^2$aa = aaaaa which is not in A_3, hence the condition is not satisfied. Thus A_3 is not a regular language. ■

1.31 For any string $w = w_1, w_2, \ldots, w_n$, the reverse of w, written w^R, is the string w in reverse order, $w_n, \ldots, w_2, w_1$. For any language A, let $A^R = \{w^R \mid w \subset A\}$. Show that if A is regular, so is A^R.

Solution. Let Z = (Q, $\sum$, δ, q_0, F) be the DFA that recognizes A. Building a NFA M' for A^R by reversing all the arrows of M and designating the start for M as the only accept state q'accept for M'. Add a new start state q'0 for M', and from q'0, add e-transitions to each state of M' corresponding to the accept states of M. For any w ∈ $\sum$, there is a path following w from the start state to an accept state in Z if and only if there is a path following w^R from q'0 to q'accept in M'. ■

1.39 The construction in Theorem 1.54 shows that every GNFA is equivalent to a GNFA with only two states. We can show that an opposite phenomenon occurs for DFAs. Prove that every k > 1, a language $A_k \subseteq \{0,1\}^*$ exists that is recognized by a DFA with k states but not by one with only k - 1 states.

Solution. Assume A_k be the set of words length at least k - 1. Therefore it can be said that A_k has at least k equivalence classes of words length 0, 1, 2, ..., k - 2, and k - 1 or more. So it is clear from this that A_k requires a DFA with k states. For any DFA fewer than k states, by the Pigeon Hole Principle, two of the k strings cause the machine to loop in the same state results in a rejection from the DFA. ■

1.53 Let $\sum = \{0, 1, +, =\}$ and ADD = $\{x = y + z | x, y, z$ are binary integers, and x is the sum of y and $z\}$. Show that ADD is not regular.

Solution. Suppose that ADD is a regular language, let p be the pumping length of the Pumping Lemma. Consider the string s = 1^p = 0 +1^p, s ∈ ADD. Then we can split the string s into 3 parts s = xyz. Consider the condition $xy^iz \in$ A_3 for each i ≥ 0. Assume that i = 0, then xy^iz is not in A_1 so the language ADD is not regular. ∎

1.63 (a) Let A be an infinite regular language. Prove that A can be split into two infinite disjoint regular subsets.

Solution. Let there be a string s ∈ A and s = xyz, since S belongs to the language of A, and the language of A is regular, xyz must belong to A, where i ≥ 0. Let A_1 be a language such that A_1 = $\{xy^{2i}z, where i \geq 0\}$. Since all the strings of the form xy^iz belong to A, the strings of the form $xy^{2i}z$ must also belong to A. Hence, the language A_1 is a subset of the language A. Since in the expression A_1 there is not upper limit of i, the language A_1 is infinite. Since a regular language is closed under the operation of the complement. the complement of A_1 is also a regular language. Let A_2 be the language such that, A_2 = complement of A_1 and A. A_1 and A_2 are two disjoint sets thus the language A can be split into two infinite disjoint regular subsets. ∎

(b) Let B and D be two languages. Write B ⋐ D if B ⊆ D and D contains infinitely many strings that are not in B. Show that if B and D are two regular languages where B ⋐ D, then we can find a regular language C where B ⋐ C ⋐ D.

Solution. Divide the regular language D into two regular disjoint subsets and let one of these subsets be B. Let the other subset be A such that A = D - B. Since D contains infinitely many strings that are not in B, A also contains infinitely many strings not in B. Further dividing the language A into two disjoint subsets, creating a set C such that C = A_1 ∪ B. B will be a subset of C and C will be a subset of D and since B ⋐ C and C ⋐ D, B ⋐ C ⋐ D is true. ∎

1.66 A homomorphism is a function f : $\sum \rightarrow \Gamma$ from one alphabet to strings over another alphabet. We can extend f to operate on strings by defining $f(w) = f(w_1), f(w_2), ..., f(w_n)$, where $w = w_1, w_2, ..., w_n$ and each $w_i \in \sum$. We further extend f to operate on languages by defining $f(\text{A}) = \{f(w) \mid w \in \text{A}\}$, for any language A.

(a) Show, by giving a formal construction, that the class of regular languages is closed under homomorphism. In other words, given a DFA M that recognizes B and a homomorphism f, construct a finite automaton M' that recognizes f(B). Consider the machine M' that you constructed. Is it a DFA in every case?

Solution. Let $\sum(a, b)$ represent the input alphabets and we define a regular language L1 = {x | x belongs to (a, b) and contains only a's and no b's} Clearly, we can see this string belongs to the language = {a, a, aaa, ...} which accepts infinitely many strings. The definition of homomorphism is a substitution h that replaces each symbol a in the input alphabet with another symbol say B. The newly generated language = {b, bb, bbb, ...} which is a language which accepts all strings of b's and does not contain any a's. The automaton M' is DFA in every case and homomorphism of any language is also closed under regular language. ■

(b) Show, by giving an example, that the class of non-regular languages is not closed under homomorphism.

Solution. Consider a non-regular language L1 = {x | x belongs to input alphabets (a, b) such that x contains equal amounts of a's and b's where |X| > 1}. This language is not regular because FSMs have limited memory which can't store counts so it cannot compare the count of A and B. Thus, it is not regular. ■