An algebraic theory for state complexity

Edwin Hamel-de Le Court

Normandie Université

Thèse

Pour obtenir le diplôme de doctorat

Spécialité Informatique

Préparée au sein de l'Université de Rouen

An Algebraic Theory for State Complexity

Une Théorie Algébrique pour la Complexité en États

Présentée et soutenue par:

Edwin HAMEL-DE LE COURT

Composition du jury		
Pascal Caron	Professeur des Universités Université de Rouen	Directeur de Thèse
Jean-Gabriel Luque	Professeur des Universités Université de Rouen	Co-directeur de Thèse
Galina Jirásková	Professor, Slovak Academy of Sciences	Rapporteur
Sylvain Lombardy	Professeur des Universités Institut Polytechnique de Bordeaux	Rapporteur
Jeffrey O. Shallit	Professor University of Waterloo	Rapporteur
Émilie Charlier	Chargée de cours Université de Liège	Examinateur
Anca Muscholl	Professeur des Universités Université de Bordeaux	Examinateur
Samuele Giraudo	Maître de conférences Université Gustave Eiffel	Examinateur

UNIVERSITÉ DE ROUEN NORMANDIE

litis

Contents

Chapter 1

Introduction

In the middle of the twentieth century, with the advent of the first computers, mathematicians became increasingly interested in classifying the different ways with which a machine computes. Automata are one of the simplest and most useful models that arose. Since then, a large theory was constructed around this concept. This theory found many practical applications like text processing, lexical analysis, or hardware engineering. In addition, many connections have been built between automata and other areas of mathematical research, like logic or algebra.

Many different kinds of automata have been defined and used in computer science. However, in this thesis, we are only interested in complete, deterministic and finite automata. We explain in the following the idea behind each of these terms. Intuitively, finite automata can be seen as machines that are able to enter a finite number of states. One may go from one state to the other by "reading" a certain letter chosen in a finite set, called an alphabet. We represent an automaton by drawing a circle for each state, with its name inside, and by drawing arrows going from state to state, labelled by letters. For example, in the automaton of Figure 1.1, we can go from state 0 to state 1 by reading either a or b, we can go from state 2 to state 1 by reading either a, b, or c, we can go from state 2 to state 2 by reading b, etc.

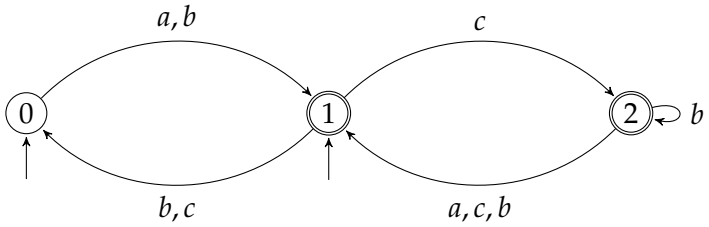


Figure 1.1: A finite automaton.

Thus, we may go from one state to another by "reading" a sequence of letters, called a word. For example, in the automaton of Figure 1.1, we may go from the state 0 to the state 2 by reading the word acb: we go to state 1 by reading a, then we go to state 2 by reading c, and then we stay in state 2 by reading b. Finite automata are designed to accept

or reject words, using this mechanism. Indeed, notice that some states of the automaton of Figure 1.1 bear some markings. The states at the end of an arrow that does not come from any other states (*i.e.*, the states 0 and 1) are called the initial states. The states with two circles around them (*i.e.*, states 1 and 2) are called the final states. A word is accepted by an automaton if we can go from an initial state to a final state by reading this word. For example, the word bca is accepted by the automaton of Figure 1.1, because we can go from 0 to 1 by reading b, then from 1 to 2 by reading c, and finally from 2 to 1 by reading a. However, the word ab, for example, is not accepted by the automaton, because there is no way to go from either 0 or 1 to either 1 or 2 by reading ab. Indeed, in order to read a, the only initial state we can begin with is the state 0. Furthermore, reading a from state 0 leads us to state 1, and then reading b from state 1 leads us to state 0, which is not final. A language is a set of words, and the set of all the words accepted by a finite automaton A is called the language recognized by A. A language L recognized by an automaton is called regular.

To be deterministic, a finite automaton must satisfy two conditions. The first one is to have exactly one initial state. The second one is that, by reading a letter from any state, there should be at most one state to which we can go. In other words, for every state q and for every letter a, there should be at most one arrow labelled by a starting from state q. For example, the automaton of Figure 1.1 is not deterministic because, by reading c from the state 1, we can go either to state 0 or to state 2. Another reason would be that it has two initial states, 0 and 1. However, for example, the automaton of Figure 1.2 is deterministic.

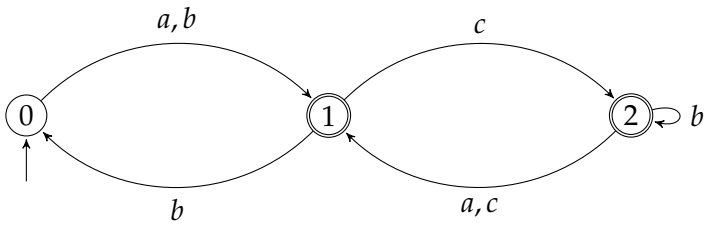


Figure 1.2: A deterministic and finite automaton.

A deterministic and finite automaton is complete if it is possible to read any letter of its alphabet from any state. In other words, for every letter a of its alphabet, and for every of its states q, there is an arrow labelled by a and starting at q. For example, the automaton of Figure 1.2 is not complete, because we cannot read the letter c from the state 0. To summarize, a finite automaton is complete and deterministic if there is exactly one possible state to go to by reading any letter from any state. For example, the automaton represented in Figure 1.3 is complete, deterministic, and finite.

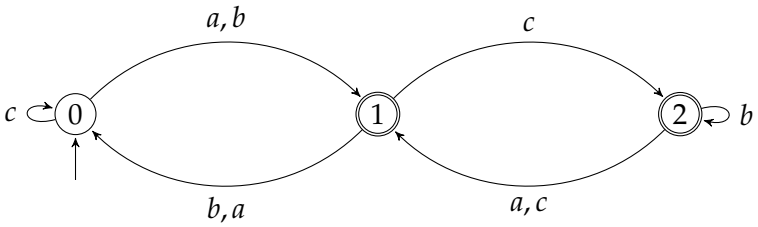


Figure 1.3: A complete, deterministic and finite automaton.

A language is recognized by a complete, deterministic, and finite automata (DFAs) if and only if it is regular. The reason why DFAs are so interesting is that checking if a word is recognized by one of them is easy. Indeed, we only need to read the word from its initial state, and to check whether the state in which we end up is final. Since there is only one way to read the word from the initial state, we do not need to worry whether there is another way to read it that would lead to a final state, as is the case with non-deterministic automata.

It is often interesting to work directly on regular languages to design algorithms. However, the space used to run algorithms is limited, and we are often concerned with the space we need to encode the objects we use. This raises the question of the space needed to code a regular language. Since every regular language is recognized by an infinity of DFAs, can we find the smallest DFA needed to recognize a regular language? And what does "smallest" mean exactly, for DFAs? In other words, what is the "size" of a DFA?

Intuitively, the size of a DFA should roughly depend on the size of its alphabet, and on how many states it has. However, by convention, we define the size of a DFA as the number of its states. Given a regular language L, the size of the smallest DFA (with respect to its size) that recognizes L is called the state complexity of L. Computing the state complexity of a regular language L is usually not obvious to do. In the rest of this thesis, we compute the state complexities of many regular languages, each time in a different way.

Nonetheless, the main questions we address in this thesis do not concern the state complexity of regular languages directly, but rather an extension of this notion to regular operations. A regular operation is an operation acting over regular languages and returning a regular language. The state complexity of a regular operation is the maximal state complexity of its outputs, given the state complexity of its inputs. In other words, the state complexity of a regular operation is a measure of how much more complex its output can be compared to its inputs.

The state complexity of operations has been extensively studied in the past years. To our knowledge, the first paper stating results about the state complexity of operations is [33]. In this paper, Maslov computed the state complexity of union, Kleene star, square root, catenation, among others, without rigorous proofs. Furthermore, the state complexities computed for the Kleene star and the square root were not accurate. Some time passed before other computer scientists decided to explore more deeply the state complexity of operations. Nonetheless, many papers computing the state complexity of different operations were eventually published. These include the Kleene star, reversal, powers,

proportional removals, catenation, binary boolean operations, among others [40, 38, 15, 28, 14, 23, 24, 25, 2]. A survey on the state complexity of operations has been published in 2017 [18]. Furthermore, the state complexities of some compositions of two or more well-known operations have been computed, like the star of union, the star of intersection, star-complement-star, multiple compositions of boolean operations and multiple compositions of catenation, among others [27, 19, 31, 11, 16, 29, 22, 8, 9, 10].

Even though the state complexity of many different operations are known, there are very little general results that can be applied to help us compute the state complexity of a new operation. This is mainly due to the fact that we do not understand well enough how to compute the state complexity of the composition of two operations, even when these operations are simple. Moreover, the monoid structure underlying the transitions of a DFA is well-known, but rarely used to formalize their results. Nonetheless, some similarities can still be observed between computations of state complexity for most operations. Two steps are usually needed when computing the state complexity of a regular operation. The first one is to compute an upper bound using some ingenious tricks. The second is to provide a family of languages, called a witness, whose image by the operation matches this upper bound. Brzozowski [2] pointed out that some particular witnesses could be used for several well-known operations. One of the explanations given by Brzozowski is that they are "complex" in a certain sense: their syntactic monoid is as large as possible. In this thesis, we take a step back and explain why this heuristic method works by adopting an algebraic point of view on witnesses and regular operations.

To that aim, in Chapter 4, we build an algebraic framework, so that the new notions we introduce may be more easily handled and deeply understood. To begin with, the operations that fall into the scope of our results, which we call 1-uniform, are the regular operations that commute with the inverse of every length-preserving morphism. This class contains many well-known regular operations, like the (set-theoretic) complement, the Kleene star, the Kleene positive closure, the cyclic shift, the mirror, all the boolean operations, the catenation, the shuffle, etc. Furthermore, this class of operations is stable by composition. Throughout our research, it appeared that this property of stability by composition played an important role, and gave an interesting algebraic structure to regular operations. Therefore, to formalize the results that come from this property, we decided to use operads, an algebraic tool that comes from the study of different types of algebras [32]. We introduce operads in Chapter 3. Operads are algebraic structures made to mimic operations of higher arity. Their definition captures the way operations of higher arity are composed with one another, in the same way that monoids captures the way by which some objects can be added or multiplied with one another. For example, language operations, regular operations, and 1-uniform operations alike form an operad.

In order to prove some results about the state complexity of 1-uniform operations, we define *modifiers*, a counterpart to 1-uniform operations in the space of operations over DFAs. Modifiers that are naturally associated with 1-uniform operations are called *coherent*. We use modifiers to work directly over DFAs, which makes state complexity computations easier. Furthermore, modifiers behave well with respect to composition. To be more precise, modifiers form an operad, and the association between coherent modifiers and 1-uniform operations, is described by a morphism of operads. The study of modifiers leads to the main result of Chapter 4, which states that every 1-uniform operation admits

a certain type of DFA, called *monster*, as witness.

As their name suggests, monsters are DFAs with a very large alphabet. In fact, their alphabet is composed of every possible transition function. Because of the size of their alphabets, monsters are not usually given as witnesses when computing state complexities. It is usually considered better to provide witnesses with small bounded alphabets (less than 5 or 6 letters). However, strictly speaking, if our goal is solely to compute the state complexity of a regular operation, we do not need to worry about the size of our witnesses' alphabets. Therefore, to reach that goal, monsters are good candidates for witnesses, since they give us as much leeway as possible for reachability and distinguishability proofs. Moreover, one can reduce the size of their alphabets later, once a first proof is done. Presenting certain state complexity computations this way may even be clearer, and easier to understand.

Using the framework built in Chapter 4, we devise a method to compute the state complexity of 1-uniform operations, and show that it works on simple examples in Chapter 5. Similar techniques have been used in numerous papers, the oldest of which would be a paper published in 1978 by Sakoda and Sipser [37]. The authors created languages composed of graphs, that "contain" in some way all the other languages recognized by the automata they study. This idea led to some state complexity results about conversions between different kinds of automata. Ravikumar [35] recognized the possible generality of this technique and applied it to several problems. Despite two of them being about the state complexity of operations (specifically about the Kleene star and the intersection), Ravikumar did not give a general class of regular operations for which this technique can be applied. Later on, many papers using this idea to prove results about the state complexity of operations were published [39, 26, 15, 3, 8]. However, none of them described a general framework that helps us knowing when we can apply such a technique, and the method itself was never formalized.

In addition, it is well worth noting that a very similar framework to the one we built, conveying similar ideas and using similar results, is described in the thesis of Sylvie Davies, albeit with a different point of view [13]. Sylvie Davies found these results concurrently and independently, and published them in [12]. This is not so surprising, as these ideas formalize a heuristic commonly used to compute state complexities. Coincidentally, this stresses the importance and fruitfulness of these ideas.

We put the method we devised to good use in Chapter 6, where we try to compute the state complexity of the star of a boolean operation. This problem is of course harder than computing the state complexity of either one of these operations, as there is no reason for state complexity of their composition to be equal to the composition of their state complexity. We manage to compute the exact state complexity of the star of symmetric difference [4], using some combinatorial results of [8]. This is an entirely new result. Combined with other results [16, 27], this gives us the state complexity of the star of every binary boolean operation. However, computing the state complexity of the star of every boolean operation is still out of our reach. We explain how our reasoning in the particular case of the symmetric difference may be generalized in Section 6.7.

As we see in Chapter 6, it can be difficult to compute the state complexity of the composition of two operations, even when these operations are well understood. As for the composition of three or more well-known regular operations, the problem becomes

highly complex, so much so that the most simple cases are already complicated [29, 22]. Therefore, in Chapter 7, we look for classes of operations that are non-trivial, but that have a simple structure with respect to composition. We examine how the algebraic structure of DFAs can be used to understand the algebraic structure of regular operations, in the hope of proving some state complexity results about large classes of operations. To that aim, we define two classes of modifiers, *friendly modifiers* and *product modifiers*, by simple algebraic properties pertaining to the algebraic structures of their input and output DFAs. The results of this chapter concerning friendly and product modifiers, are, to our knowledge, entirely new. These two classes are stable by composition, and thus form an operad. We show that friendly modifiers describe every composition of a boolean operation and some roots, and that the operation described by product modifiers are a slight generalization of boolean operations [6, 5] . Furthermore, we study in detail the underlying algebraic structure, and show how the structure induced by product modifiers fits into the structure induced by friendly modifiers. Finally, we compute the maximal state complexity of friendly operations, depending on their arity.

Chapter 2

Preliminaries

2.1 Notations and conventions

2.1.1 Standard notations and conventions

- The *cardinality* of a finite set E, denoted by $\#E$, is the number of elements of E.
- A set G is a *subset* of a set E if, for all $g \in G$, we have $g \in E$. In that case, we write $G \subseteq E$. The *set of all subsets* of E is denoted by 2^E.
- A *mapping* f from a set E to a set G associates every element of E with an element of G. We let $f(e)$ denote the element of G associated with an element e of E. We say that $f(e)$ is *the image* of e by f.
- The *set of all mappings* of a set E to a set F is denoted by F^E. If f is a mapping from E to E, we say that it is a mapping over E.
- The *identity over* E, denoted by Id_E, is the mapping such that, for all $e \in E$, we have $\mathrm{Id}_E(e) = e$. When E is clear from the context, to avoid cumbersome notations, we let Id_E denote Id.
- Let f be a mapping from a set E to a set F. For every subset G of E, we let $f(G)$ denote the set of all elements y of F such that there exists $x \in G$ with $y = f(x)$. We say that $f(G)$ is the *image* of G by f.
- A *partial function* from a set E to a set F is a mapping f from a subset $G \subseteq E$ to F. We say that G is the *domain* of f.
- Let E, F, and G be three sets, let f be a mapping from E to F, and g be a mapping from F to G. We let $f \circ g$ denote the mapping from E to G such that, for all $e \in E$, we have $(f \circ g)(e) = f(g(e))$. We call $\circ$ the *composition of functions*.
- The set of all non-negative integers is denoted by $\mathbb{N}$, and the set of all positive integers is denoted by $\mathbb{N} \setminus 0$.

- Let E be a set. A *sequence with values* in E is a mapping u from $\mathbb{N}$ to E. We usually let $(u_j)_{j\in\mathbb{N}}$ denote u, where $u_j = u(j)$ for all $j \in \mathbb{N}$.
- A sequence $(u_j)_{j\in\mathbb{N}}$ with values in a set E is *eventually periodic* if and only if there exist two non-negative integers p and N, such that for all $n \geq N$, we have $u_{n+p} = u_n$.
- For all non-negative integers k, a *k-tuple of elements* u of E, or a *finite sequence* with values in E, is a mapping from $\{1, \ldots, k\}$ to E, or equivalently an element of E^k. When the terminology of k-tuple is employed, we usually let $(u_1, \ldots, u_k)$ denote u, where $u_j = u(j)$ for all $j \in \{1, \ldots, k\}$. Furthermore, when the terminology of finite sequence is employed, similarly to the general case of sequences, we usually let $(u_j)_{j\in\{1,\ldots,k\}}$ denote u, where $u_j = u(j)$ for all $j \in \{1, \ldots, k\}$.
- For all non-negative integers k, we let E^k denote the set of all k-tuples of elements of E. The set E^1 is identified with E.
- A mapping f from a set E to a set F is *surjective* if and only if, for all $y \in F$, there exists $x \in E$ such that $y = f(x)$. A mapping f from a set E to a set F is *injective* if and only if, for any two distinct elements x and x' of E, we have $f(x) \neq f(x')$. A mapping is *bijective* if it is surjective and injective.
- Let f be a bijective mapping from E to F. We let f^{-1} denote the mapping from F to E such that, for all $y \in F$, $f^{-1}(y)$ is equal to the only element x of E that satisfies $f(x) = y$.
- Let E be a set, and G be a subset of E. The *complement* of G in E, denoted by $E \setminus G$, is the set of all elements e of E such that e is not in G.
- The *union* of two sets E and G, denoted by $E \cup G$, is the set of all elements that are either in E or in G.
- The *intersection* of two sets E and G, denoted by $E \cap G$, is the set of all elements that are in both E and G.
- The *symmetric difference* of two sets E and G, denoted by $E\Delta G$, is the set of all elements that are either in E or in G, but not both. Notice that we have $E\Delta G = (E \cup G)\setminus(E \cap G)$.
- A *monoid* is a set M equipped with an associative operation $\bullet$ such that there exists an element $e \in M$ that satisfies $e \bullet x = x \bullet e = x$, for all $x \in M$.
- A *group* is a monoid $(G, \bullet)$ such that, for all element $x \in G$, there exists an element y in G such that $x \bullet y = y \bullet x = e_G$. This element is called the *inverse* of x is denoted by x^{-1}.
- Let $(M, \bullet)$ be a monoid, and H be a subset of M. We say that M *is generated by* H if, for all $x \in M$, there exists a finite sequence $(h_j)_{j\in\{1,\ldots,m\}}$ of elements of H such that $x = h_1 \bullet h_2 \bullet \cdots \bullet h_m$.
- Let $(G, \bullet)$ be a group, and H be a subset of G. We let H^{-1} denote the set of all the elements x of G such that there exists $h \in H$ that satisfies $x = h^{-1}$. We say that G *is generated by* H if, for all $x \in G$, there exists a finite sequence $(h_j)_{j\in\{1,\ldots,m\}}$ of elements of $H \cup H^{-1}$ such that $x = h_1 \bullet h_2 \bullet \cdots \bullet h_m$.

2.1.2 Non-standard notations and conventions

- For every set E and every element g of some set, we let $[g \in E]$ denote the number equal to 1 if $g \in E$, and 0 otherwise.
- *A graded set* is a sequence of sets $(E_n)_{n\in\mathbb{N}}$ such that, for any two distinct non-negative integers i and j, we have $E_i \cap E_j = \emptyset$. We almost always identify a graded set $(E_n)_{n\in\mathbb{N}}$ with the set $E = \bigcup_{j\in\mathbb{N}} E_j$.
- Let E and F be two graded sets. A *graded mapping* f from E to F is a mapping from E to F such that, for any non-negative integer n, for any $e \in E_n$, we have $f(e) \in F_n$.
- Let $E_1, \ldots, E_k$ be k sets, and let $(f_1, \ldots, f_k) \in E_1^{E_1} \times \cdots \times E_k^{E_k}$. For any $(e_1, \ldots, e_k)$ in $E_1 \times \cdots \times E_k$, we let $(f_1, \ldots, f_k)(e_1, \ldots, e_k)$ denote the element $(f_1(e_1), \ldots, f_k(e_k))$ of $E_1 \times \cdots \times E_k$. In other words $(f_1, \ldots, f_k)$ denotes at the same time an element of $E_1^{E_1} \times \cdots \times E_k^{E_k}$ and an element of $(E_1 \times \cdots \times E_k)^{E_1 \times \cdots \times E_k}$. The reader should always be able to infer, from the context, which one we are referring to. Notice that, as a consequence, for any $(g_1, \ldots, g_k) \in E_1^{E_1} \times \cdots \times E_k^{E_k}$, we let $(f_1, \ldots, f_k) \circ (g_1, \ldots, g_k)$ denote the element $(f_1 \circ g_1, \ldots, f_k \circ g_k)$ of $E_1^{E_1} \times \cdots \times E_k^{E_k}$. Furthermore, if $(G_1, \ldots, G_k) \in 2^{E_1} \times \cdots \times 2^{E_k}$, we let $(f_1, \ldots, f_k)(G_1, \ldots, G_k)$ denote the element $(f_1(G_1), \ldots, f_k(G_k))$ of $2^{E_1} \times \cdots \times 2^{E_k}$. We say that $(f_1(G_1), \ldots, f_k(G_k))$ is the *image* of $(G_1, \ldots, G_k)$ by $(f_1, \ldots, f_k)$.
- Let X be a set, k be a non-negative integer. A *k-ary operation* over X is a pair (k, f) where f is a partial function from X^k to X.

  We put the arity k of an operation explicitly in the above definition, instead of relying only on the domain of f, because the domain of f could be empty. This trick lets us distinguish between two operations of different arity whose domains are empty. Nevertheless, a k-ary operation (k, f) is almost always denoted only by the function f, when the arity of the operation is clear from the context. We say that an operation is unary when it is 1-ary, and that it is binary when it is 2-ary.
- For all positive integers n, we let $[\![n]\!]$ denote the finite set $\{0, \ldots, n-1\}$.

2.2 Mappings over $[\![n]\!]$

Let n be a positive integer. For all finite sequences of $(i_j)_{j\in\{1,\ldots,m\}}$ of pairwise distinct elements of $[\![n]\!]$, we let $(i_1, \ldots, i_m)$ denote the mapping such that

$$\begin{cases} \text{for all } j \in \{1, \ldots, m\}, & (i_1, \ldots, i_m)(i_j) = \begin{cases} i_{j+1} & \text{if } 1 \le j \le m-1 \\ i_1 & \text{if } j = m \end{cases} \\ \text{for all } \ell \notin \{i_1, \ldots, i_m\}, & (i_1, \ldots, i_m)(\ell) = \ell \end{cases}$$

The mapping $(i_1, \ldots, i_m)$ is called an *m-cycle*. A 2-cycle is called a transposition. Furthermore, for all $i, j \in [\![n]\!]$, we let π_j^i denote the mapping such that, for all $\ell \in [\![n]\!]$, we have

$$\pi_j^i(\ell) = \begin{cases} j & \text{if } \ell = i \\ \ell & \text{otherwise.} \end{cases}$$

A *permutation* is a bijective mapping over $[\![n]\!]$, for some positive integer n. The set of all permutations over $[\![n]\!]$ is denoted by $\mathfrak{S}_n$. It is well-known that, for all positive integers n, $([\![n]\!]^{[\![n]\!]}, \circ)$ is a monoid, and that $(\mathfrak{S}_n, \circ)$ is a group. It is well-known that, for all integers n with $n \geq 2$, the monoid $(\mathfrak{S}_n, \circ)$ is generated by $\{(0,1), (0,1,\dots,n-1)\}$. For all positive integers n, we let Γ'_n denote the subset of $[\![n]\!]^{[\![n]\!]}$ equal to $\{\mathrm{Id}\}$ if $n = 1$, and to $\{(0,1), (0,1,\dots,n-1), \pi_0^{n-1}\}$ otherwise. It is well-known that for all positive integers n, the monoid $([\![n]\!]^{[\![n]\!]}, \circ)$ is generated by the set Γ'_n.

For all k-tuples of positive integers $(n_1,\dots,n_k)$, we let $\Gamma_{n_1,\dots,n_k}$ denote the set

$$[\![n_1]\!]^{[\![n_1]\!]} \times \cdots \times [\![n_k]\!]^{[\![n_k]\!]}.$$

Definition 1. *Let $(n_1,\dots,n_k)$ be a k-tuple of positive integers, and for all $j \in \{1,\dots,k\}$ with $n_j \geq 2$, let $t_j = (\underbrace{\mathrm{Id},\dots,\mathrm{Id}}_{j-1 \text{ elements}}, (0,1), \mathrm{Id},\dots,\mathrm{Id})$, $c_j = (\underbrace{\mathrm{Id},\dots,\mathrm{Id}}_{j-1 \text{ elements}}, (0,1,\dots,n_j-1), \mathrm{Id},\dots,\mathrm{Id})$, and $p_j = (\underbrace{\mathrm{Id},\dots,\mathrm{Id}}_{j-1 \text{ elements}}, \pi_0^{n_j-1}, \mathrm{Id},\dots,\mathrm{Id})$. We let $\Gamma'_{n_1,\dots,n_k}$ denote the set $\{(\mathrm{Id},\dots,\mathrm{Id})\}$ if $(n_1,\dots,n_k) = (1,\dots,1)$, and the set $\bigcup_{j\in\{1,\dots,k\}|n_j\geq 2} \{t_j, c_j, p_j\}$ otherwise.*

Proposition 1. *For all k-tuples of positive integers $(n_1,\dots,n_k)$, the monoid $(\Gamma_{n_1,\dots,n_k}, \circ)$ is generated by $\Gamma'_{n_1,\dots,n_k}$.*

Proof. Let $(n_1,\dots,n_k)$ be a k-tuple of integers.

For all $(\phi_1,\dots,\phi_k) \in \Gamma_{n_1,\dots,n_k}$, we have

$$(\phi_1,\dots,\phi_k) = (\phi_1,\mathrm{Id},\dots,\mathrm{Id}) \circ \cdots \circ (\underbrace{\mathrm{Id},\dots,\mathrm{Id}}_{j-1 \text{ elements}}, \phi_j, \mathrm{Id},\dots,\mathrm{Id}) \circ \cdots \circ (\mathrm{Id},\dots,\mathrm{Id},\phi_k).$$

However, for all $j \in \{1,\dots,k\}$, there exists an integer m_j and a finite sequence $(\phi_{j,\ell})_{\ell\in\{1,\dots,m_j\}}$ of elements of Γ'_{n_j} such that

$$(\underbrace{\mathrm{Id},\dots,\mathrm{Id}}_{j-1 \text{ elements}}, \phi_j, \mathrm{Id},\dots,\mathrm{Id}) = (\underbrace{\mathrm{Id},\dots,\mathrm{Id}}_{j-1 \text{ elements}}, \phi_{j,1}, \mathrm{Id},\dots,\mathrm{Id}) \circ \cdots \circ (\underbrace{\mathrm{Id},\dots,\mathrm{Id}}_{j-1 \text{ elements}}, \phi_{j,m_j}, \mathrm{Id},\dots,\mathrm{Id}).$$

However, by Definition 1, $\psi_{j,\ell} = (\underbrace{\mathrm{Id},\dots,\mathrm{Id}}_{j-1 \text{ elements}}, \phi_{j,l}, \mathrm{Id},\dots,\mathrm{Id})$ is an element of $\Gamma'_{n_1,\dots,n_k} \cup \{(\mathrm{Id},\dots,\mathrm{Id})\}$. Furthermore, we have

$$(\phi_1,\dots,\phi_k) = \psi_{1,1} \circ \cdots \circ \psi_{1,m_1} \circ \psi_{2,1} \circ \cdots \circ \psi_{2,m_2} \circ \cdots \circ \psi_{k,1} \circ \cdots \circ \psi_{k,m_k}.$$

Notice that the element $(\mathrm{Id},\dots,\mathrm{Id})$ of $\Gamma_{n_1,\dots,n_k}$ is an element of $\Gamma'_{n_1,\dots,n_k}$, if $(n_1,\dots,n_k) = (1,\dots,1)$, and is otherwise equal to

$$(\underbrace{\mathrm{Id},\dots,\mathrm{Id}}_{j-1 \text{ elements}}, (0,1), \mathrm{Id},\dots,\mathrm{Id}) \circ (\underbrace{\mathrm{Id},\dots,\mathrm{Id}}_{j-1 \text{ elements}}, (0,1), \mathrm{Id},\dots,\mathrm{Id}),$$

for all j such that $n_j \geq 2$. In any case, we can always write $(\mathrm{Id},\dots,\mathrm{Id})$ as a composition of elements of $\Gamma'_{n_1,\dots,n_k}$.

As a consequence, $(\Gamma_{n_1,\dots,n_k}, \circ)$ is generated by $\Gamma'_{n_1,\dots,n_k}$. □

2.3 Equivalence relations

A relation over a set E is a mapping from $E \times E$ to $\{0,1\}$. For any two elements q, q' of E, if $\mathbf{r}$ is a relation over E, we let $q\,\mathbf{r}\,q'$ denote the fact that $\mathbf{r}(q,q') = 1$.

Let $\mathbf{r}$ be a relation over a set E. The *reflexive closure* of $\mathbf{r}$ is the relation $\mathbf{v}$ such that, for all $q, q' \in E$, $q\,\mathbf{v}\,q'$ if and only if $q = q'$ or $q\,\mathbf{r}\,q'$. The *symmetric closure* of $\mathbf{r}$ is the relation $\mathbf{s}$ such that, for all $q, q' \in E$, $q\,\mathbf{s}\,q'$ if and only if $q\,\mathbf{r}\,q'$ or $q'\,\mathbf{r}\,q$. The *transitive closure* of $\mathbf{r}$ is the relation $\mathbf{t}$ such that, for all $q, q' \in E$, $q\,\mathbf{t}\,q'$ if and only if there exists a finite sequence $(q_j)_{j\in\{1,\dots,m\}}$ of elements of E that satisfies the following properties:

- $q_1 = q$,
- $q_m = q'$,
- for all $j \in \{1,\dots,m-1\}$, we have $q_j\,\mathbf{r}\,q_{j+1}$.

The *reflexive,symmetric, and transitive closure* of $\mathbf{r}$ is the transitive closure of the symmetric closure of the reflexive closure of $\mathbf{r}$.

An equivalence relation $\sim$ over a set E is a relation that is equal to its reflexive, symmetric and transitive closure. In other words, a relation $\sim$ over a set E is an equivalence relation if and only if

- for all $q \in E$, we have $q \sim q$,
- for all $q, q' \in E$, if $q \sim q'$, then $q' \sim q$
- for all $q_1, q_2, q_3 \in E$, if $q_1 \sim q_2$ and $q_2 \sim q_3$, then $q_1 \sim q_3$.

For all equivalence relations $\sim$ over a set E, for all $q \in E$, the *equivalence class* of q for $\sim$, denoted by $\widetilde{q}$, is the set of all $q' \in E$ such that $q \sim q'$. Furthermore, the set of all equivalence classes of E for $\sim$, denoted by $E/_\sim$, is the set of all $\tilde{q}$ with $q \in E$. Notice that, for all $q, q' \in E$, if $q \sim q'$, then $\widetilde{q} = \widetilde{q'}$.

We say that an equivalence relation $\sim$ over a graded set E is *graded for* E if, for all integers m and n, for all $q \in E_m$, and for all $q' \in E_n$ such that $q \sim q'$, we have $m = n$. Equivalently, $\sim$ is graded if, for all non-negative integers n, $(E/_\sim) \cap 2^{E_n} = E_n/_\sim$. Therefore, if $\sim$ is graded, the set $E/_\sim$ is naturally graded as follows: for all non-negative integers n, $(E/_\sim)_n$ is equal to the set $E_n/_\sim$.

2.4 Operations over languages and DFAs

2.4.1 Alphabets, words and languages

An *alphabet* is a finite set, whose elements are called *letters*. A *word* w over Σ is a finite sequence $(a_j)_{j\in\{1,\dots,\ell\}}$ with values in Σ. A word $w = (a_j)_{j\in\{1,\dots,\ell\}}$ is usually denoted by $a_1 \cdots a_\ell$. When $\ell = 0$, we call w the *empty* word, and it is denotes by ε. The *catenation* of two words $u = a_1 \cdots a_n$ and $v = b_1 \cdots b_m$, denoted by $u \cdot v$ or uv, is the word $a_1 \cdots a_n b_1 \cdots b_m$, that is the

finite sequence with values in Σ obtained by putting the finite sequence $(b_j)_{j\in\{1,\ldots,m\}}$ at the end of the finite sequence $(a_j)_{j\in\{1,\ldots,n\}}$. For all words w over an alphabet Σ, we define w^n inductively as $w \cdot w^{n-1}$ with $w^0 = \varepsilon$.

The set of all words over Σ is denoted by $\Sigma^\star$. A *language* over an alphabet Σ is a **pair** (L, Σ), where L is a subset of $\Sigma^\star$. To avoid cumbersome notations, we identify a language (L, Σ) with the first element L of the pair, when the alphabet Σ is clear from the context.

By convention, the *complement* of a language L over an alphabet Σ is the complement of L in $\Sigma^\star$.

The *catenation* of two languages L and L' over the same alphabet Σ, denoted by $L \cdot L'$, is the language (over Σ) of all the words ww' over Σ such that $w \in L$ and $w' \in L'$.

For every non-negative integer k and every language L over an alphabet Σ, the *k-th power* of L, denoted by L^k, is the language (over Σ) of all the words $w_1 \cdots w_k$ over Σ such that $w_j \in L$, for all $j \in \{1, \ldots, k\}$. In other words, for every non-negative integer k, $L^k = L \cdot L^{k-1}$ with $L^0 = \{\varepsilon\}$.

For any language L over an alphabet Σ, the *Kleene star* of L, denoted by $L^\star$, is the language (over Σ) of all the words $w_1 \cdots w_k$ over Σ, such that k is a non-negative integer, and $w_j \in L$ for all $j \in \{1, \ldots, k\}$. In other words, $L^\star = \bigcup_{k=0}^{+\infty} L^k$.

For every non-negative integer n, the *n-th root* of a language L over an alphabet Σ, denoted by $\sqrt[n]{L}$ is the language (over Σ) of all the words w such that $w^n \in L$. Notice that $\sqrt[0]{L} = \Sigma^\star$ if $\varepsilon \in L$ and $\emptyset$ otherwise, and $\sqrt[1]{L} = L$. By convention, we let $\sqrt{L}$ denote $\sqrt[2]{L}$.

2.4.2 Deterministic, finite and complete automata

A *complete and deterministic finite automaton* (DFA) is a 5-tuple $A = (\Sigma, Q, i, F, \delta)$ where Σ is an alphabet, Q is a finite non-empty set, i is an element of Q, F is a subset of Q, and δ is a mapping from $Q \times \Sigma$ to Q. The set Q is called the *set of states* of A, i is called its *initial state*, F is called its *set of final states*, and δ is called its *transition function*. The *size* of A, denoted by $\#A$, is the cardinality of Q. As an example, let us consider the DFA $A = (\{a, b\}, \{0, 1, 2\}, 0, \{0, 2\}, \delta)$, of size 3, where for all $j \in \{0, 1, 2\}$, we have $\delta(j, a) = (j + 1) \bmod 3$, and $\delta(j, b) = (2j) \bmod 3$. This DFA is represented in Figure 2.1.

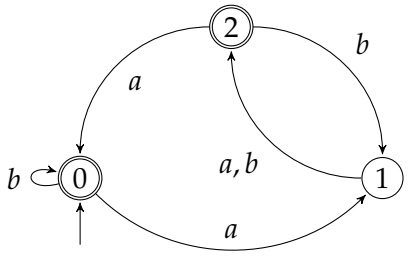


Figure 2.1: A DFA A.

As you can see in Figure 2.1, we represent the states of a DFA as circles with their name inside. We put on the initial state an arrow that does not come from any other state, and

we put around the final states another circle. Finally, we put an arrow labelled by a letter $a \in \Sigma$ between two states q and q', if and only if we have $\delta(q, a) = q'$.

For any DFA $A = (\Sigma, Q, i, F, \delta)$, any state $q \in Q$ and any letter $a \in \Sigma$, we let $\delta^a(q)$ denote $\delta(q, a)$. Notice that, for all letters $a \in \Sigma$, δ^a is mapping over Q. Furthermore, we extend the transition function δ to a mapping from $Q \times \Sigma^\star$ to Q in the following way: for all states $q \in Q$, and for all words $w = a_1 \cdots a_\ell$ over Σ, we let $\delta(q, w)$ denote the state $\delta^{a_\ell} \circ \cdots \circ \delta^{a_1}(q)$ (with $\delta(q, \varepsilon) = q$). We also let $\delta^w(q)$ denote $\delta(q, w)$.

We can associate a language with every DFA. Intuitively, we associate with a DFA every word such that, after beginning at the initial state and following the arrows by "reading" the letters one by one, we end up in a final state. More formally, the language *recognized* by a DFA $A = (\Sigma, Q, i, F, \delta)$, denoted by $\mathrm{L}(A)$, is the set of all words w such that $\delta^w(i) \in F$. A language is called *regular* if it is recognized by a DFA. Furthermore, the language $\mathrm{L}(A)$ is always over the alphabet of A. Two DFAs are *equivalent* if they recognize the same language.

For example, the language recognized by the DFA B of Figure 2.2 is the set of all words over the alphabet $\{a\}$ of odd length, *i.e.,* the set $\{a^{2k+1} \mid k \in \mathbb{N}\}$.

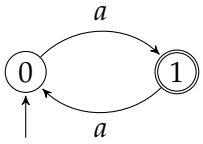


Figure 2.2: A DFA B.

We say that two DFAs are isomorphic if one is the same as the other up to a renaming of states. More formally, a DFA $A = (\Sigma, Q, i, F, \delta)$ is *isomorphic* to a DFA $A' = (\Sigma', Q', i', F', \delta')$ if $\Sigma = \Sigma'$, and if there exists a bijection ϕ from Q to Q' such that $\phi(i) = i'$, $\phi(F) = F'$, and, for all states $q \in Q$ and all letters $a \in \Sigma$, we have $\phi(\delta^a(q)) = \delta'^a(\phi(q))$.

Let $A = (\Sigma, Q, i, F, \delta)$ be a DFA, and let $\sim$ be an equivalence relation over Q. We say that $\sim$ is *compatible* with A if

- for all states $q, q' \in Q$ with $q \sim q'$, if $q \in F$, then $q' \in F$
- for all states $q, q' \in Q$ and all letters $a \in \Sigma$, if $q \sim q'$, then $\delta^a(q) \sim \delta^a(q')$.

Let $\sim$ be an equivalence relation compatible with A. We let $A/_\sim$ denote the DFA

$$(\Sigma, Q/_\sim, \tilde{i}, F/_\sim, \zeta),$$

where, for all $\widetilde{q} \in Q/_\sim$, we have $\zeta^a(\widetilde{q}) = \widetilde{\delta^a(q)}$. Since $\delta^a(q) \sim \delta^a(q')$ when $q \sim q'$, the DFA $A/_\sim$ is well-defined. We show by induction that, for all states $q \in Q$ and all words w over Σ, we have $\zeta^w(\widetilde{q}) = \widetilde{\delta^w(q)}$. Therefore, we have $\delta^w(i) \in F$ if and only if $\zeta^w(\tilde{i}) \in F/_\sim$. Hence, the DFA $A/_\sim$ recognizes the same language as A.

2.4.3 Accessible states, the Nerode equivalence and minimal DFAs

Let $A = (\Sigma, Q, i, F, \delta)$ be a DFA. A state $q \in Q$ is *accessible* in A if there exists a word w over Σ such that $q = \delta^w(i)$. The *accessible part* of A is the DFA obtained when restricting A to its accessible states, *i.e.,* the DFA $B = (\Sigma, Q', i, F', \delta')$, where Q' is the set of all accessible states of A, F' is the set of all final states of A that are accessible, and δ' is the restriction of δ to $Q' \times \Sigma$. The non-accessible states are not relevant for computing the language recognized by A. In other words, the accessible part of A recognizes the same language as A.

Let $A = (\Sigma, Q, i, F, \delta)$ be a DFA. We say that two states $q, q' \in Q$ are *distinguishable* in A if and only if there exists a word w over Σ such that the states $\delta^w(q)$ and $\delta^w(q')$ do not have the same finality, *i.e.,* such that either $\delta^w(q) \notin F$ and $\delta^w(q') \in F$, or $\delta^w(q) \in F$ and $\delta^w(q') \notin F$. In other words, two states $q, q'$are distinguishable if we can find a word w such that the two states obtained by "reading" w from q and q' do not have the same finality. Two states that are not distinguishable in A are said to be *indistinguishable* in A.

The *Nerode equivalence* induced by A is a relation $\sim$ over Q such that, for all states $q, q' \in Q$, we have $q \sim q'$ if and only if q and q' are indistinguishable in A. The Nerode equivalence $\sim$ induced by A is an equivalence relation, and is compatible with A. Therefore, if $\sim$ is the Nerode equivalence induced by A, the DFA $A/_\sim$ recognizes the same language as A.

Let L be a language over an alphabet Σ, and let A be a DFA that recognizes L. A *minimal DFA that recognizes* L, or a *minimal DFA equivalent to* A, is a DFA B such that B recognizes L, and such that the size of B is minimal out of all DFAs that recognize L, *i.e.,* such that $\mathrm{L}(B) = L$ and $\#B = \min\{\#C \mid \mathrm{L}(C) = L\}$. It is well-known [36] that any two minimal DFAs that recognize the same language L are isomorphic. Furthermore, given a DFA A, we can compute a minimal DFA equivalent to A.

Let A be a DFA, let B be the accessible part of A, let $\sim_A$ be the Nerode equivalence induced by A, and let $\sim_B$ be the Nerode equivalence induced by B. Then the DFA $B/_{\sim_B}$ is a minimal DFA equivalent to A, and so is the accessible part of $A/_{\sim_A}$ [36]. We make use of this result repeatedly in Chapters 5 and 7.

As an example, consider the DFA C of Figure 2.3. The only state that is not accessible in C is 5. Therefore, the DFA C' of Figure 2.4 is the accessible part of C. The states 1 and 2 are indistinguishable in C', as are the states 3 and 4. Furthermore, the states 1 and 3 are distinguishable in C' by reading the empty word. In addition, 0 is distinguishable from every other state. Therefore, the set of all equivalence classes of C', for the Nerode equivalence $\sim_{C'}$ induced by C', is equal to $\{\{0\}, \{1, 2\}, \{3, 4\}\}$. As a consequence, the DFA D of Figure 2.5 is equal to $C'/_{\sim_{C'}}$. Hence, D is a minimal DFA equivalent to C. Furthermore, the DFA D recognizes the language $\{a\}$. Therefore, D is a minimal DFA that recognizes $\{a\}$.

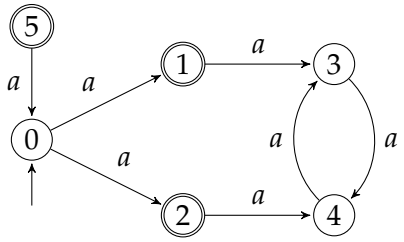


Figure 2.3: A DFA C

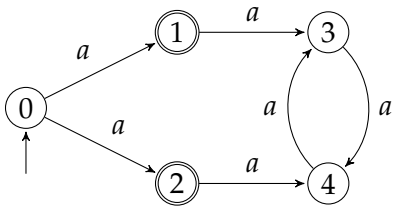


Figure 2.4: The DFA C': the accessible part of C

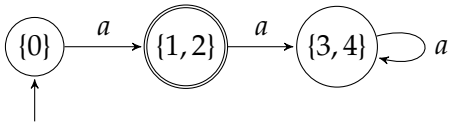


Figure 2.5: The DFA D: a minimal DFA equivalent to C.

2.4.4 Language operations and regular operations

What are languages operations and regular operations?

In computer science, and specifically in the research area of formal languages, a regular operation is usually an operation that acts on k regular languages and returns a single regular language, for some integer k. However, we have to clarify what we mean by that, because two main different point of views are used, with slightly different definitions that lead to different results. To this aim, we introduce some new notations. We let $\mathscr{L}$ denote the set of all languages, and by $\mathcal{L}$ the set of all regular languages. Furthermore, for any alphabet Σ, we let $\mathcal{P}(\Sigma^\star)$ denote the set of all languages over the alphabet Σ, and by $\mathbf{Rat}(\Sigma^\star)$ the set of all regular languages over the alphabet Σ. In addition, we let $\mathscr{L}_k^\Sigma$ denote the set $(\mathcal{P}(\Sigma^\star))^k$, and by $\mathcal{L}_k^\Sigma$ the set $(\mathbf{Rat}(\Sigma^\star))^k$. Finally, we let $\mathscr{L}_k$ denote the union of all the sets $(\mathcal{P}(\Sigma^\star))^k$, for all alphabets Σ, and by $\mathcal{L}_k$ the union of all the sets $(\mathbf{Rat}(\Sigma^\star))^k$, for all alphabets Σ.

From the *unrestricted* point of view, we see a k-ary language operation as an operation over $\mathscr{L}$ whose domain is $\mathscr{L}^k$, *i.e.*, a mapping from $\mathscr{L}^k$ to $\mathscr{L}$. Similarly, we see a regular operation as an operation over $\mathcal{L}$ whose domain is $\mathcal{L}^k$, *i.e.*, a mapping from $\mathcal{L}^k$ to $\mathcal{L}$. From the *restricted* point of view, we see a k-ary language operation as an operation that acts on

k languages over the **same** alphabet Σ, and returns a language over Σ. Similarly, we see a regular operation as an operation over $\mathcal{L}$ that acts on k regular languages over the **same** alphabet Σ, and returns a regular language over Σ. More formally, from the restricted point of view, a k-ary language operation $\otimes$ is an k-ary operation over $\mathscr{L}$ whose domain is $\mathscr{L}_k$ (*i.e.*, a mapping from $\mathscr{L}_k$ to $\mathscr{L}$), and that satisfies $\otimes((\mathcal{P}(\Sigma^\star))^k) \subseteq \mathcal{P}(\Sigma^\star)$, for any alphabet Σ. Similarly, a k-ary regular operation $\otimes$ is a k-ary operation over $\mathcal{L}$ whose domain is $\mathcal{L}_k$ (*i.e.*, a mapping from $\mathcal{L}_k$ to $\mathcal{L}$), and that satisfies $\otimes((\mathbf{Rat}(\Sigma^\star))^k) \subseteq \mathbf{Rat}(\Sigma^\star)$, for any alphabet Σ.

Even though the unrestricted point of view may seem simpler, it is probably easier, for technical reasons, to study the concepts we introduce from the restricted point of view. Therefore, it is the point of view that we adopt throughout this thesis. We stress here that **throughout the rest of this thesis, a k-ary language operation refers to a mapping $\otimes$ from $\mathscr{L}_k$ to $\mathscr{L}$ such that $\otimes((\mathcal{P}(\Sigma^\star))^k) \subseteq \mathcal{P}(\Sigma^\star)$, for all alphabets Σ and all non-negative integers k. Furthermore, a k-ary regular language operation refers to a mapping $\otimes$ from $\mathcal{L}_k$ to $\mathcal{L}$ such that $\otimes((\mathbf{Rat}(\Sigma^\star))^k) \subseteq \mathbf{Rat}(\Sigma^\star)$, for all alphabets Σ and all non-negative integers k.**

You may have noticed that we purposefully defined regular operations so that their domain was restricted to k-tuples of **regular** languages of $\mathscr{L}_k$. This decision is not without consequences throughout this thesis. In particular, two distinct language operations may have the same restriction to $\mathcal{L}_k$, and thus may be equal if considered as regular operations. Therefore, before using an operation over languages, it may be important to ascertain its domain. To avoid this inconvenience, we state now that **the domain of every k-ary operation over languages considered in Chapters 4, 5, 6, and 7 is $\mathcal{L}_k$**. Even though every operation we consider in these chapters is actually a regular operation, it is not obvious for some of them. Therefore, making the statement here that every operations considered are regular operations would require some of the proofs presented later on.

Boolean functions and boolean operations

An important class of language operations is the class of boolean operations. Boolean operations are language operations that generalize the complement, the union, and the intersection of languages. They are routinely used in computer science, and we use them extensively throughout this thesis. They provide good examples, as their properties are not trivial, but not overly complicated either when compared to other well-known operations. In order to define boolean operations over languages, we first need to define *boolean functions*.

Definition 2. *A k-ary boolean function $\mathbf{b}$ is a k-ary operation over $\{0,1\}$.*

In other words, a boolean function acts on k elements of $\{0,1\}$ and returns an element of $\{0,1\}$. From every boolean function, we define a corresponding operation over languages.

Definition 3. *Let $\mathbf{b}$ be a k-ary boolean function. We let $\otimes_{\mathbf{b}}$ denote the k-ary language operation such that, for all k-tuples of languages $(L_1, \ldots, L_k)$ over Σ, we have*

$$\otimes_{\mathbf{b}}(L_1, \ldots, L_k) = \{w \in \Sigma^* \mid \mathbf{b}([w \in L_1], \ldots, [w \in L_k]) = 1\}.$$

A regular operation is boolean *if it is equal to $\otimes_{\mathbf{b}}$, for some boolean function $\mathbf{b}$.*

Example 1. *The most usual boolean operations over languages can be defined as follows:*

- *Let* $\mathbf{c}$ *be the unary boolean function such that* $\mathbf{c}(0) = 1$ *and* $\mathbf{c}(1) = 0$*. The complement over languages is equal to* $\otimes_{\mathbf{c}}$*. Indeed, we have*

$$\{w \in \Sigma^* \mid \mathbf{c}([w \in L]) = 1\} = \{w \in \Sigma^* \mid [w \in L] = 0\} = \{w \in \Sigma^* \mid w \notin L\} = L^c.$$

- *For any two languages* L_1 *and* L_2 *over the same alphabet, we have* $\otimes_{\mathbf{union}}(L_1, L_2) = L_1 \cup L_2$, $\otimes_{\mathbf{inter}}(L_1, L_2) = L_1 \cap L_2$, $\otimes_{\mathbf{xor}}(L_1, L_2) = L_1 \Delta L_2$*, where* **union**, **inter** *and* **xor** *are the boolean functions respectively defined by Table 2.6, Table 2.7 and Table 2.8.*

union	0	1
0	0	1
1	1	1

Figure 2.6
Truth table
for **union**.

inter	0	1
0	0	0
1	0	1

Figure 2.7
Truth table
for **inter**.

xor	0	1
0	0	1
1	1	0

Figure 2.8
Truth table
for **xor**.

2.4.5 State complexity

There are as many types of state complexities as there are types of automata. However, in this thesis, we only study what is sometimes called deterministic state complexity, *i.e.*, the notion of state complexity that comes from DFAs.

The *state complexity* of a language L over an alphabet Σ, denoted by $\mathrm{sc}(L)$, is the size of a minimal DFA that recognizes L. For example, from Section 2.4.3, the state complexity of the language $\{a\}$ over the alphabet $\{a\}$ is 3 (the size of DFA D in Figure 2.5). We extend this notion to regular operations. The state complexity of a regular operation is a measure of how much it can increase the state complexity of its input languages. In other words, it is the maximal state complexity of the output, given that the state complexity of the inputs is less than or equal to a certain upper bound. More formally, the *state complexity* of a k-ary regular operation $\otimes$, denoted by $\mathrm{sc}_\otimes$, is a map from $(\mathbb{N} \setminus 0)^k$ to $\mathbb{N} \setminus 0$ such that, for all k-tuples of positive integers $(n_1, \ldots, n_k)$, we have

$$\mathrm{sc}_\otimes(n_1, \ldots, n_k) = \max\{\mathrm{sc}(\otimes(L_1, \ldots, L_k)) \mid (L_1, \ldots, L_k) \in \mathcal{L}_k \wedge \mathrm{sc}(L_j) \leq n_j, \forall j \in \{1, \ldots, k\}\}$$

In this thesis, we develop a method for computing the state complexity of some regular operations. In Chapters 5 and 7, we will see several examples of these kind of computations. However, there already are in the literature numerous examples (see for example [40, 15, 14, 23, 22, 27]). The usual way to compute the state complexity of a regular operation, is first, to find an upper bound, and then, to show that the upper bound is met by providing a witness. A *witness* for a k-ary regular operation $\otimes$ is a k-tuple of DFAs $(A_1, \ldots, A_k)$ (whose alphabets are all equal), such that

$$\mathrm{sc}_\otimes(n_1, \ldots, n_k) = \mathrm{sc}(\otimes(\mathrm{L}(A_1), \ldots, \mathrm{L}(A_k))),$$

where n_j is the size of A_j, for all $j \in \{1, \ldots, k\}$. In other words, the usual way to compute the state complexity of an operation $\otimes$ is to prove that there is an upper bound b such that $\mathrm{sc}_\otimes(n_1, \ldots, n_k) \leq b$, and then to find a k-tuple of DFAs $(A_1, \ldots, A_k)$ such that each A_j is of size n_j, and such that $\mathrm{sc}(\otimes(\mathrm{L}(A_1), \ldots, \mathrm{L}(A_k))) = b$.

2.4.6 Morphisms

Let Σ and Γ be two alphabets. A morphism is a function ϕ from Σ^* to Γ^*, such that, for all $w, v \in \Sigma^*$, $\phi(wv) = \phi(w)\phi(v)$. Notice that ϕ is completely defined by its value on letters, since $\phi(\varepsilon) = \varepsilon$. We say that a morphism ϕ from Σ^* to Γ^* is 1-*uniform* if, for all $a \in \Sigma$, $\phi(a) \in \Gamma$.

Proposition 2. *Let L be a regular language over alphabet Σ recognized by the DFA $A = (\Sigma, Q, i, F, \delta)$ and let ϕ be a 1-uniform morphism from Γ^* to Σ^*. Then, $\phi^{-1}(L)$ is the regular language recognized by the DFA $B = (\Gamma, Q, i, F, \delta')$ where, for all $a \in \Gamma$ and $q \in Q$, $\delta'(q, a) = \delta(q, \phi(a))$.*

Proof. A word w is recognized by B if and only if $\delta'^w(i) \in F$. However, an easy induction shows that $\delta'^w(q) = \delta(q, \phi(w))$. Therefore, a word w is recognized by B if and only if $\delta^{\phi(w)}(i) \in F$. However, for any word v, $\delta^v(i) \in F$ if and only if v is recognized by A. Therefore, w is recognized by B if and only if $\phi(w)$ is recognized by A. Hence, $\mathrm{L}(B) = \phi^{-1}(\mathrm{L}(A))$. □

The next proposition is a direct consequence of Proposition 2.

Proposition 3. *Let L be a regular language and ϕ be a 1-uniform morphism. We have* $\mathrm{sc}(\phi^{-1}(L)) \leq \mathrm{sc}(L)$.

Chapter 3

Operads

3.1 What is an operad?

An operad is a structure designed to mimic the composition of k-ary functions. To our knowledge, the word *operad* first appeared in a book of May about iterated loop spaces [34]. This notion was also connected to the works of G.M. Kelly on coherence in categories, and the definitions used in [34] arose from discussions between the two mathematicians. In the 1990's, the interest in operads was renewed by the works of Ginzburg and Kapranov, who connected a duality phenomena in rational homotopy theory to the Koszul duality for operads [20]. Since then, operads were widely used to encode different types of algebras, like Poisson algebras, Lie and pre-Lie algebras, or Jordan algebras, to name a few. Using operads to study these kinds of algebras allows us to simplify results and generalize them. Nowadays, operads appear in many different areas of research, like algebraic topology, differential geometry, combinatorial algebra, or computer science. One can find more information on operads in [32]. The results of this chapter are well-known results from the theory of operads that can be found in e.g. [21]. Nonetheless, a proof is given here for everyone of them, in order to familiarize the reader with the notion of operads.

An operad is composed of a graded set $\mathcal{O}$ and some binary operations $\odot_j$. The idea is that the binary operations $\odot_j$ behave like a composition, and that an element in the set $\mathcal{O}_n$ behaves like an n-ary operation. To explain intuitively the rules defining an operad, we represent their elements as trees. An element of $\mathcal{O}_n$ is represented by a tree whose root has n children (Figure 3.1).

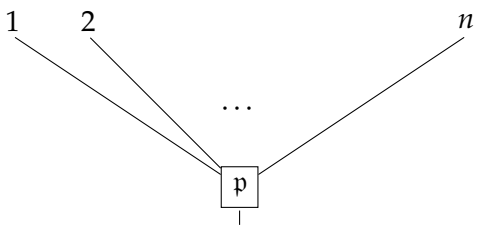


Figure 3.1: A representation of an element $\mathfrak{p}$ of $\mathcal{O}_n$.

Furthermore, we represent the composition of two elements $\mathfrak{p} \circ_j \mathfrak{q}$ by replacing the j-th child of the root of $\mathfrak{p}$ with $\mathfrak{q}$ (Figure 3.2).

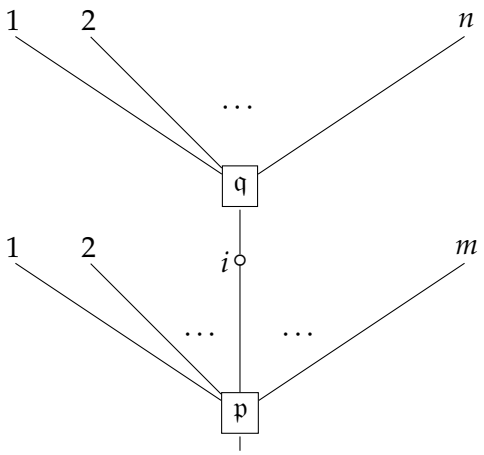


Figure 3.2: A representation of $\mathfrak{p} \circ_i \mathfrak{q}$ with $\mathfrak{p} \in \mathcal{O}_m$ and $\mathfrak{q} \in \mathcal{O}_n$.

For $\circ_j$ to "behave like" a composition, we set three rules that an operad must satisfy. The first one states that there must exists an element $\mathbf{1}$ of $\mathcal{O}_1$ that behaves like the identity. It is represented in Figure 3.3 below.

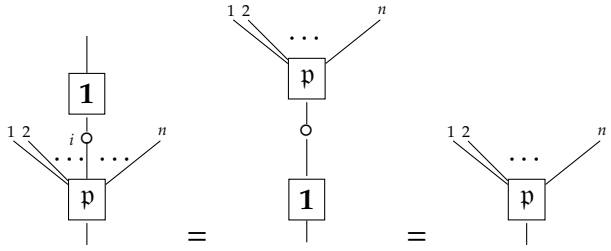


Figure 3.3: The "identity" rule.

The second rule is an associativity rule. It gives two different ways of composing three elements of $\mathcal{O}$, that correspond intuitively to arranging these elements in the shape of a "line" (right part of Figure 3.4). That rule is represented in Figure 3.4 below.

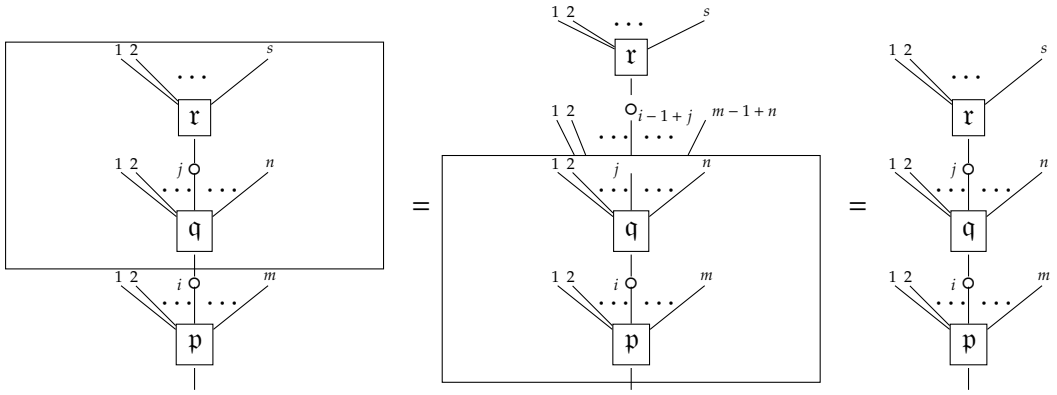


Figure 3.4: The "aligned" associativity rule.

Notice that, in the middle part of Figure 3.4, the element corresponding to the rectangle is the element $\mathfrak{p} \circ_i \mathfrak{q}$. As it is an element of $\mathcal{O}_{m+n-1}$, it is represented as a tree with $m+n-1$ children, and then composed with $\mathfrak{r}$.

We can also compose three elements of $\mathcal{O}$ in the shape of a "triangle" (like in the right part of Figure 3.5). Therefore, another associativity rule is needed (Figure 3.5).

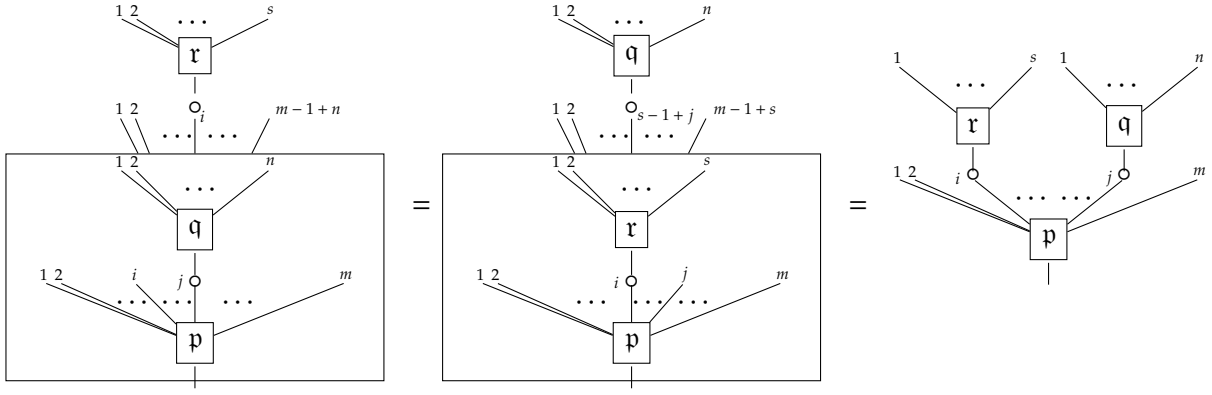


Figure 3.5: The "triangular" associativity rule.

The "identity" rule, the "aligned" associativity rule, and the "triangular" associativity rule, are the three rules that an operad must satisfy. We formalize this idea in the following definition.

Definition 4. *An* operad *is a graded set $\mathcal{O}$ equipped with a sequence $(\odot_j)_{j\in\mathbb{N}\setminus 0}$ of binary operations over $\mathcal{O} = \bigcup_{n\in\mathbb{N}} \mathcal{O}_n$. For all $j \in \mathbb{N}\setminus 0$, $\odot_j$ is defined for every pair of functions in $\mathcal{O}_m \times \mathcal{O}_n$, with $m \geq j$ and $n \in \mathbb{N}$, and returns an element of $\mathcal{O}_{m+n-1}$. Furthermore, the sequence of binary operations $(\odot_j)_{j\in\mathbb{N}\setminus 0}$, must satisfy the following conditions:*

- *the "identity" rule (Figure 3.3): there exists an element $\mathrm{Id} \in \mathcal{O}_1$ such that, for all positive integers n and $i \leq n$, and all elements $\mathfrak{p}$ of $\mathcal{O}_n$, we have $\mathrm{Id} \odot_1 \mathfrak{p} = \mathfrak{p} \odot_i \mathrm{Id} = \mathfrak{p}$,*

- *the "aligned" associativity rule (Figure 3.4): we have* $\mathfrak{p} \odot_i (\mathfrak{q} \odot_j \mathfrak{r}) = (\mathfrak{p} \odot_i \mathfrak{q}) \odot_{i+j-1} \mathfrak{r}$*, for all positive integers* $m, n, s, i \leq m, j \leq n$*, and all* $\mathfrak{p}, \mathfrak{q}, \mathfrak{r}$*, elements of* $\mathcal{O}_m, \mathcal{O}_n, \mathcal{O}_s$ *respectively,*
- *the "triangular" associativity rule (Figure 3.5): we have* $(\mathfrak{p} \odot_j \mathfrak{q}) \odot_i \mathfrak{r} = (\mathfrak{p} \odot_i \mathfrak{r}) \odot_{j+s-1} \mathfrak{q}$*, for all positive integers* m, n, s, i*, and* j *such that* $i < j \leq m$*, and all* $\mathfrak{p}, \mathfrak{q}, \mathfrak{r}$*, elements of* $\mathcal{O}_m, \mathcal{O}_n, \mathcal{O}_s$ *respectively.*

Notice that, for every operad, the element satisfying the "identity" rule is unique. It is called the *identity* of $(\mathcal{O}, \odot)$.

We end this section by giving an example of an operad.

Example 2. *For all integers* n*, we let* $\mathcal{B}_n$ *denote the set of all subsets of* $\{0,1\}^n$*. Furthermore, we let* $\mathcal{B}$ *denote the set* $\bigcup_{j=0}^{+\infty} \mathcal{B}_k$*, naturally graded with the sequence* $(\mathcal{B}_k)_{k \in \mathbb{N}}$*. For all non-negative integers* m, n *and* j *with* $1 \leq j \leq m$*, for all* $E \in \mathcal{B}_m$ *and* $E' \in B_n$*, we let* $E \circledcirc_j E'$ *denote the set of all* $(e_1, \ldots, e_{m+n-1}) \in \mathcal{B}_{m+n-1}$ *such that*

- *either* $(e_1, \ldots, e_{j-1}, 1, e_{j+n}, \ldots, e_{m+n-1}) \in E$ *and* $(e_j, \ldots, e_{j+n-1}) \in E'$
- *or* $(e_1, \ldots, e_{j-1}, 0, e_{j+n}, \ldots, e_{m+n-1}) \in E$ *and* $(e_j, \ldots, e_{j+n-1}) \notin E'$*.*

In order to illustrate the above definition, we show here that the graded set $\mathcal{B}$ *equipped with* $\circledcirc$ *is an operad, by checking that the three "rules" of Definition 4 hold. However, even though the example seems simple, the direct proof given here is rather heavy. A faster proof is given in Example 6, using the tools we develop in the next section.*

- *The element* $\{1\}$ *of* $\mathcal{B}_1$ *satisfies the "identity" rule of Definition 4. In other words, for all positive integers* n *and* $i \leq n$*, and for all elements* E *of* $\mathcal{B}_n$*, we have* $\{1\} \circledcirc_1 E = E \circledcirc_i \{1\} = E$*.*
- *For all positive integers* $m, n, s, i \leq m, j \leq n$*, and all* E_1, E_2, E_3*, elements of* $\mathcal{B}_m, \mathcal{B}_n, \mathcal{B}_s$ *respectively, we have* $(e_1, \ldots, e_{m+n+s-2}) \in E_1 \circledcirc_i (E_2 \circledcirc_j E_3)$ *if and only if*
  - *either* $(e_1, \ldots, e_{i-1}, 1, e_{i+n+s-1}, \ldots, e_{m+n+s-2}) \in E_1$ *and* $(e_i, \ldots, e_{i+n+s-2}) \in E_2 \circledcirc_j E_3$
  - *or* $(e_1, \ldots, e_{i-1}, 0, e_{i+n+s-1}, \ldots, e_{m+n+s-2}) \in E_1$ *and* $(e_i, \ldots, e_{i+n+s-2}) \notin E_2 \circledcirc_j E_3$*.*

  However, for all non-negative integers x*,* y *and* z *with* $1 \leq z \leq x$*, for all* $E \in \mathcal{B}_x$*, for all* $E' \in B_y$*, and for all* $(e_1, \ldots, e_{x+y-1}) \in \mathcal{B}_{x+y-1}$*, we have either* $(e_j, \ldots, e_{z+y-1}) \in E'$ *or* $(e_j, \ldots, e_{z+y-1}) \notin E'$*. If* $(e_j, \ldots, e_{z+y-1}) \in E'$*, then* $(e_1, \ldots, e_{x+y-1}) \notin E \circledcirc_j E'$ *if and only if*

$$(e_1, \ldots, e_{z-1}, 1, e_{z+y}, \ldots, e_{x+y-1}) \notin E,$$

  and if $(e_j, \ldots, e_{z+y-1}) \notin E'$*, then* $(e_1, \ldots, e_{x+y-1}) \notin E \circledcirc_j E'$ *if and only if*

$$(e_1, \ldots, e_{z-1}, 0, e_{z+y}, \ldots, e_{x+y-1}) \notin E.$$

  Therefore, we have $(e_1, \ldots, e_{x+y-1}) \notin E \circledcirc_j E'$ *if and only if*

  - *either* $(e_1, \ldots, e_{z-1}, 0, e_{z+y}, \ldots, e_{x+y-1}) \notin E$ *and* $(e_j, \ldots, e_{z+y-1}) \notin E'$
  - *or* $(e_1, \ldots, e_{z-1}, 1, e_{z+y}, \ldots, e_{x+y-1}) \notin E$ *and* $(e_j, \ldots, e_{z+y-1}) \in E'$*.*

Therefore, $(e_1, \dots, e_{m+n+s-2}) \in E_1 \circledcirc_i (E_2 \circledcirc_j E_3)$ *if and only if*

- *either* $(e_1, \dots, e_{i-1}, 1, e_{i+n+s-1}, \dots, e_{m+n+s-2}) \in E_1$ *and* $(e_i, \dots, e_{i+j-2}, 1, e_{i+j+n-1}, \dots, e_{i+n+s-2}) \in E_2$ *and* $(e_{i+j-1}, \dots, e_{i+j+s-2}) \in E_3$
- *or* $(e_1, \dots, e_{i-1}, 1, e_{i+n+s-1}, \dots, e_{m+n+s-2}) \in E_1$ *and* $(e_i, \dots, e_{i+j-2}, 0, e_{i+j+s-1}, \dots, e_{i+n+s-2}) \in E_2$ *and* $(e_{i+j-1}, \dots, e_{i+j+s-2}) \notin E_3$
- *or* $(e_1, \dots, e_{i-1}, 0, e_{i+n+s-1}, \dots, e_{m+n+s-2}) \in E_1$ *and* $(e_i, \dots, e_{i+j-2}, 0, e_{i+j+s-1}, \dots, e_{i+n+s-2}) \notin E_2$ *and* $(e_{i+j-1}, \dots, e_{i+j+s-2}) \notin E_3$
- *or* $(e_1, \dots, e_{i-1}, 0, e_{i+n+s-1}, \dots, e_{m+n+s-2}) \in E_1$ *and* $(e_i, \dots, e_{i+j-2}, 1, e_{i+j+s-1}, \dots, e_{i+n+s-2}) \notin E_2$ *and* $(e_{i+j-1}, \dots, e_{i+j+s-2}) \in E_3$

Furthermore, $(e_1, \dots, e_{m+n+s-2}) \in (E_1 \circledcirc_i E_2) \circledcirc_{i+j-1} E_3$ *if and only if*

- *either* $(e_1, \dots, e_{i+j-2}, 1, e_{i+j+s-1}, \dots, e_{m+n+s-2}) \in E_1 \circledcirc_i E_2$ *and* $(e_{i+j-1}, \dots, e_{i+j+s-2}) \in E_3$
- *or* $(e_1, \dots, e_{i+j-2}, 0, e_{i+j+s-1}, \dots, e_{m+n+s-2}) \in E_1 \circledcirc_i E_2$ *and* $(e_{i+j-1}, \dots, e_{i+j+s-2}) \notin E_3$.

Thus, since $i+n \geq i+j$*, by removing the* $\circledcirc$ *in the above characterization of* $(E_1 \circledcirc_i E_2) \circledcirc_{i+j-1} E_3$*, similarly to what we did in the case of* $E_1 \circledcirc_i (E_2 \circledcirc_j E_3)$*, we get that* $E_1 \circledcirc_i (E_2 \circledcirc_j E_3) = (E_1 \circledcirc_i E_2) \circledcirc_{i+j-1} E_3$.

- *For all positive integers* m*,* n*,* s*,* i*, and* j *such that* $i < j \leq m$*, and all* E_1*,* E_2*,* E_3*, elements of* $\mathcal{B}_m$*,* $\mathcal{B}_n$*,* $\mathcal{B}_s$ *respectively, we have* $(e_1, \dots, e_{m+n+s-2}) \in (E_1 \circledcirc_j E_2) \circledcirc_i E_3$ *if and only if*

  - *either* $(e_1, \dots, e_{i-1}, 1, e_{i+s}, \dots, e_{m+n+s-2}) \in E_1 \circledcirc_j E_2$ *and* $(e_i, \dots, e_{i+s-1}) \in E_3$
  - *or* $(e_1, \dots, e_{i-1}, 0, e_{i+s}, \dots, e_{m+n+s-2}) \in E_1 \circledcirc_j E_2$ *and* $(e_i, \dots, e_{i+s-1}) \notin E_3$.

  Therefore, $(e_1, \dots, e_{m+n+s-2}) \in (E_1 \circledcirc_j E_2) \circledcirc_i E_3$ *if and only if*

  - *either* $(e_1, \dots, e_{i-1}, 1, e_{i+s}, \dots, e_{j+s-2}, 1, e_{j+s+n-1}, \dots, e_{m+n+s-2}) \in E_1$ *and* $(e_{j+s-1}, \dots, e_{j+s+n-2}) \in E_2$ *and* $(e_i, \dots, e_{i+s-1}) \in E_3$
  - *or* $(e_1, \dots, e_{i-1}, 1, e_{i+s}, \dots, e_{j+s-2}, 0, e_{j+s+n-1}, \dots, e_{m+n+s-2}) \in E_1$ *and* $(e_{j+s-1}, \dots, e_{j+s+n-2}) \notin E_2$ *and* $(e_i, \dots, e_{i+s-1}) \in E_3$
  - *or* $(e_1, \dots, e_{i-1}, 0, e_{i+s}, \dots, e_{j+s-2}, 1, e_{j+s+n-1}, \dots, e_{m+n+s-2}) \in E_1$ *and* $(e_{j+s-1}, \dots, e_{j+s+n-2}) \in E_2$ *and* $(e_i, \dots, e_{i+s-1}) \notin E_3$
  - *or* $(e_1, \dots, e_{i-1}, 0, e_{i+s}, \dots, e_{j+s-2}, 0, e_{j+s+n-1}, \dots, e_{m+n+s-2}) \in E_1$ *and* $(e_{j+s-1}, \dots, e_{j+s+n-2}) \notin E_2$ *and* $(e_i, \dots, e_{i+s-1}) \notin E_3$.

  Furthermore, $(e_1, \dots, e_{m+n+s-2}) \in (E_1 \circledcirc_i E_3) \circledcirc_{j+s-1} E_2$ *if and only if*

  - *either* $(e_1, \dots, e_{j+s-2}, 1, e_{j+n+s-1}, \dots, e_{m+n+s-1}) \in E_1 \circledcirc_i E_3$ *and* $(e_{j+s-1}, \dots, e_{j+n+s-2}) \in E_2$
  - *or* $(e_1, \dots, e_{j+s-2}, 0, e_{j+n+s-1}, \dots, e_{m+n+s-1}) \in E_1 \circledcirc_i E_3$ *and* $(e_{j+s-1}, \dots, e_{j+n+s-2}) \notin E_2$.

  As a consequence, by removing the $\circledcirc$ *in the above characterization of* $(E_1 \circledcirc_i E_3) \circledcirc_{j+s-1} E_2$*, similarly to what we did in the case of* $(E_1 \circledcirc_j E_2) \circledcirc_i E_3$*, we get that* $(E_1 \circledcirc_i E_3) \circledcirc_{j+s-1} E_2 = (E_1 \circledcirc_j E_2) \circledcirc_i E_3$.

We have thus proved that the graded set $\mathcal{B}$ *equipped with* $\circledcirc$ *is an operad.*

3.2 Operations over a set

Operations are extensively used throughout mathematics and computer science, the simplest and most well-known example being the addition on integers, which is a binary operation. In order to give to operations over a set a structure of operad, we need to define a composition. This composition is a generalization of the composition of partial functions, that we recall below.

Definition 5. *Let X be a set, Y and Z be two subsets of X, f be a mapping from Y to X and g be a mapping from Z to X. Let D be the set of all elements z of Z such that $g(z)$ is in Y. We let $f \circ g$ denote the function from D to X such that $(f \circ g)(x) = f(g(x))$.*

Notice that, in the above definition, the domain of $f \circ g$ is not X nor is it the domain of g, but rather the elements x in the domain of g such that $g(x)$ is in the domain of f. We generalize this idea to k-ary operations, and define the domain of $\mathfrak{p} \circ_j \mathfrak{q}$ similarly, based the domains of $\mathfrak{p}$ and $\mathfrak{q}$, but also on the operation $\mathfrak{q}$. We illustrate this in Figure 3.6, where $\mathfrak{q}$ is a unary operation over X of domain Z, $\mathfrak{p}$ is a binary operation over X of domain Y, and $\mathbf{D}$ is the domain of $\mathfrak{p} \circ_1 \mathfrak{q}$. As you can see, $\mathbf{D}$ is the set of all pairs (x, y) in X^2 such that $x \in Z$ and $(\mathfrak{q}(x), y) \in Y$.

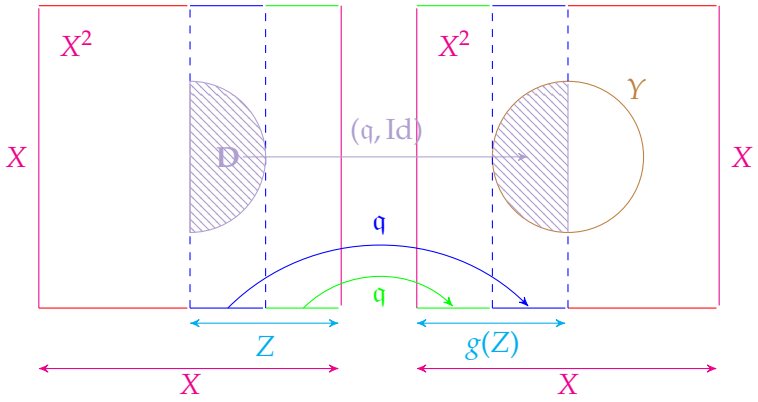


Figure 3.6: The domain $\mathbf{D}$ of $\mathfrak{p} \circ_1 \mathfrak{q}$.

As for the output of the composition $\mathfrak{p} \circ_j \mathfrak{q}$ in the general case where $\mathfrak{p}$ is m-ary and $\mathfrak{q}$ is n-ary, we only need to follow Figure 3.2.

Definition 6. *Let X be a set, let m, n and j be three integers with $1 \leq j \leq m$, let $\mathfrak{p}$ be a m-ary operation over X of domain Y, and let $\mathfrak{q}$ be a n-ary operation over X of domain Z. Let D be the set of all $m + n - 1$-tuples $(e_1, \ldots, e_{m+n-1})$of elements of X such that*

$$\begin{cases} (e_j, \ldots, e_{n+j-1}) \in Z \\ (e_1, \ldots, e_{j-1}, \mathfrak{q}(e_j, \ldots, e_{n+j-1}), e_{n+j}, \ldots, e_{n+m-1}) \in Y \end{cases}$$

We let $\mathfrak{p} \circ_j \mathfrak{q}$ denote the $m + n - 1$-ary operation $\mathfrak{r}$ from D into X such that, for any element $e = (e_1, \ldots, e_{m+n-1})$ of D,

$$\mathfrak{r} = \mathfrak{p}(e_1, \ldots, e_{j-1}, \mathfrak{q}(e_j, \ldots, e_{n+j-1}), e_{n+j}, \ldots, e_{m+n-1}).$$

We call the sequence of binary operations $(\circ_j)_{j\in\mathbb{N}\setminus\{0\}}$ *the* composition of operations.

Example 3. *Let* $\mathfrak{o}_+$ *be the binary operation over* $\mathbb{Z}$ *of domain* $\mathbb{Z}^2$ *such that, for any two integers* (i,j), $\mathfrak{o}_+(i,j) = i+j$. *Furthermore, let* $\mathfrak{o}_/$ *be the binary operation over* $\mathbb{Z}$ *of domain* $\mathbb{Z}\times(\mathbb{Z}\setminus\{0\})$ *such that, for any two integers* (i,j) *with* $j\neq 0$, $\mathfrak{o}_/(i,j) = \frac{i}{j}$. *From Definition 6, the domain* D *of* $\mathfrak{o}_/ \circ_2 \mathfrak{o}_+$ *is the set of all elements* (x,y,z) *of* $\mathbb{Z}^3$ *such that* $\mathfrak{o}_+(y,z)\neq 0$ *(and equivalently such that* $y+z\neq 0$*). Furthermore, for any element* (x,y,z) *of* D, $(\mathfrak{o}_/ \circ_2 \mathfrak{o}_+)(x,y,z) = \mathfrak{o}_/(x,\mathfrak{o}_+(y,z)) = \frac{x}{y+z}$. *We can see that the domain* D *that comes from Definition 6 indeed corresponds to the elements* (x,y,z) *for which* $\frac{x}{y+z}$ *is well-defined.*

When $\mathfrak{p}$ is a unary operation over X, and $\mathfrak{q}$ is any operation over X, for simplicity, we let $\mathfrak{p}\circ\mathfrak{q}$ denote the operation $\mathfrak{p}\circ_1\mathfrak{q}$. We now prove that operations over a set indeed form an operad.

Proposition 4. *Let* X *be any set, and let* $\boldsymbol{O}$ *be a graded set such that, for any integer* $n\in\mathbb{N}$, $\boldsymbol{O}_n$ *is the set of all* n*-ary operations over* X. *The graded set* $\boldsymbol{O}$ *equipped with the composition of operations is an operad. We let* $(\mathfrak{Map}_X, \circ)$ *denote this operad.*

Proof. Notice that, by Definition 6, $\circ_j$ is indeed defined for every element of $\boldsymbol{O}_m\times\boldsymbol{O}_n$ with $j\leq m$, and returns an element of O_{m+n-1}. We now prove successively that $(\mathfrak{Map}_X,\circ)$ satisfies the three conditions of Definition 4.

- The identity over X satisfies the first condition of Definition 4.
- By Definition 6, we have

$$\begin{gathered}\mathfrak{p}\odot_i(\mathfrak{q}\odot_j\mathfrak{r})(e_1,\dots,e_{m+n+s-1}) =\\ ((\mathfrak{p}\odot_i\mathfrak{q})\odot_{i+j-1}\mathfrak{r})(e_1,\dots,e_{m+n+s-1}) =\\ \mathfrak{p}(e_1,\dots,e_{i-1},\mathfrak{q}(e_i,\dots,e_{i+j-2},\mathfrak{r}(e_{i+j-1},\dots,e_{i+j+s-2}),e_{i+j+s-1},\dots,e_{i+n+s-2}),\\ e_{i+n+s-1},\dots,e_{m+n+s-2})\end{gathered}$$

  Furthermore, the domain of $\mathfrak{p}\odot_i(\mathfrak{q}\odot_j\mathfrak{r})$ and of $(\mathfrak{p}\odot_i\mathfrak{q})\odot_{i+j-1}\mathfrak{r}$ is the set of all $(e_1,\dots,e_{m+n+s-1})$ with

$$\begin{cases}(e_{i+j-1},\dots,e_{i+j+s-2})\in D_{\mathfrak{r}}\\ (e_i,\dots,e_{i+j-2},\mathfrak{r}(e_{i+j-1},\dots,e_{i+j+s-2}),e_{i+j+s-1},\dots,e_{i+n+s-2})\in D_{\mathfrak{q}}\\ (e_1,\dots,e_{i-1},\mathfrak{q}(e_i,\dots,e_{i+j-2},\mathfrak{r}(e_{i+j-1},\dots,e_{i+j+s-2}),e_{i+j+s-1},\dots,e_{i+n+s-2}),\\ e_{i+n+s-1},\dots,e_{m+n+s-2})\in D_{\mathfrak{p}}\end{cases}$$

  where $D_{\mathfrak{r}}$, $D_{\mathfrak{q}}$, and $D_{\mathfrak{p}}$ are the domains of $\mathfrak{r}$, $\mathfrak{q}$ and $\mathfrak{p}$ respectively.

- By Definition 6, we have

$$\begin{gathered}((\mathfrak{p}\odot_j\mathfrak{q})\odot_i\mathfrak{r})(e_1,\dots,e_{m+n+s-1}) =\\ ((\mathfrak{p}\odot_i\mathfrak{r})\odot_{j+s-1}\mathfrak{q})(e_1,\dots,e_{m+n+s-1}) =\\ \mathfrak{p}(e_1,\dots,e_{i-1},\mathfrak{r}(e_i,\dots,e_{i+s-1}),e_{i+s},\dots,e_{j+s-2},\mathfrak{q}(e_{j+s-1},\dots,e_{n+s-2}),\\ e_{n+s-1},\dots,e_{m+n+s-2})\end{gathered}$$

Furthermore, the domain of $(\mathfrak{p} \odot_j \mathfrak{q}) \odot_i \mathfrak{r}$ and of $(\mathfrak{p} \odot_i \mathfrak{r}) \odot_{j+s-1} \mathfrak{q}$ is the set of all $(e_1, \ldots, e_{m+n+s-1})$ with

$$\begin{cases} (e_i, \ldots, e_{i+s-1}) \in D_{\mathfrak{r}} \\ (e_{j+s-1}, \ldots, e_{n+s-2}) \in D_{\mathfrak{q}} \\ (e_1, \ldots, e_{i-1}, \mathfrak{r}(e_i, \ldots, e_{i+s-1}), e_{i+s}, \ldots, e_{j+s-2}, \mathfrak{q}(e_{j+s-1}, \ldots, e_{n+s-2}), \\ e_{n+s-1}, \ldots, e_{m+n+s-2}) \in D_{\mathfrak{p}} \end{cases}$$

where $D_{\mathfrak{r}}$, $D_{\mathfrak{q}}$, and $D_{\mathfrak{p}}$ are the domains of $\mathfrak{r}$, $\mathfrak{q}$ and $\mathfrak{p}$ respectively.

□

As a consequence of the above proposition, the set of operations over languages, the set of operations operations over DFAs, and the set boolean functions are operads, when equipped with $\circ$.

We now define a generalization of the commutativity of two functions to two operations, when one of them is unary.

Definition 7. *Let X be a set, let n be an integer, and let $\mathfrak{p}$ and $\mathfrak{r}$ be two operations over X, respectively n-ary and unary. We say that $\mathfrak{p}$* commutes *with $\mathfrak{r}$ if, for any n elements $e_1, \ldots, e_n$ of X, we have $\mathfrak{p}(\mathfrak{r}(e_1), \ldots, \mathfrak{r}(e_n)) = \mathfrak{r}(\mathfrak{p}(e_1, \ldots, e_n))$.*

We illustrate this definition with Figure 3.7 below.

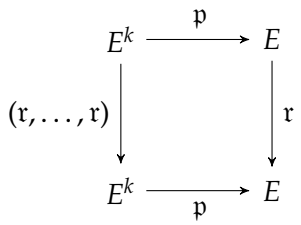


Figure 3.7: An illustration of Definition 7.

We next prove that, if two operations commute with a unary operation, then their composition also commutes with this unary operation.

Lemma 1. *Let E be any set, let n, m, and j be three positive integers such that $j \leq m$, and let $\mathfrak{p}$, $\mathfrak{q}$, and $\mathfrak{r}$ be three operations over E, respectively n-ary, m-ary, and unary. If $\mathfrak{p}$ and $\mathfrak{q}$ commute with $\mathfrak{r}$, then $\mathfrak{p} \circ_j \mathfrak{q}$ commutes with $\mathfrak{r}$.*

Proof. For any $n + m - 1$ elements $e_1, \ldots, e_{n+m-1}$ of E, we have

$$\begin{aligned} &\mathfrak{p} \circ_j \mathfrak{q}(\mathfrak{r}(e_1), \ldots, \mathfrak{r}(e_{m+n-1})) \\ = \mathfrak{p}(\mathfrak{r}(e_1), \ldots, \mathfrak{r}(e_{j-1}), \mathfrak{q}(\mathfrak{r}(e_j), \ldots, \mathfrak{r}(e_{j+m-1})), \mathfrak{r}(e_{j+m}), \ldots, \mathfrak{r}(e_{n+m-1})) \\ = \mathfrak{p}(\mathfrak{r}(e_1), \ldots, \mathfrak{r}(e_{j-1}), \mathfrak{r}(\mathfrak{q}(e_j, \ldots, e_{j+m-1})), \mathfrak{r}(e_{j+m}), \ldots, \mathfrak{r}(e_{n+m-1})) \\ = \mathfrak{r}(\mathfrak{p}(e_1, \ldots, e_{j-1}, \mathfrak{q}(e_j, \ldots, e_{j+m-1}), e_{j+m}, \ldots, e_{n+m-1})) \end{aligned}$$

□

3.3 Morphisms, quotient operads, and suboperads

Similarly to subgroups or subrings, we define the notion of suboperad.

Definition 8. *An operad $(\mathcal{O}', \odot')$ is a suboperad of another operad $(\mathcal{O}, \odot)$ if and only if*

- *$\mathcal{O}'_n$ is a subset of $\mathcal{O}_n$, for all $n \in \mathbb{N}$,*
- *the identity of $\mathcal{O}'$ is the identity of $\mathcal{O}$.*

The next proposition is straightforward from Definition 8, and is very similar to the cases of monoids, groups or rings. It states that, to prove that a subset of an operad is a suboperad, we only need to examine the stability by composition of this subset, and to check that the identity of the operad is in the subset. Therefore, we first need to define what "stability by composition" means in the case of operads, even though it is extremely intuitive.

Definition 9. *Let $(\mathcal{O}, \odot)$ be an operad. We say that a subset E of $\mathcal{O}$ is* stable by $\odot$ *if, for any two integers n and $j \leq n$, for any $\mathfrak{p} \in (\mathcal{O}_n \cap E)$ and any $\mathfrak{q} \in E$, we have $\mathfrak{p} \odot_j \mathfrak{q} \in E$.*

Proposition 5. *Let $(\mathcal{O}, \odot_j)$ be an operad, and let E be a subset of $\mathcal{O}$ stable by $\odot_j$ such that the identity of $\mathcal{O}$ is in E. Let $\mathcal{E}$ be the graded set such that $\mathcal{E}_n = \mathcal{O}_n \cap E$, for any $n \in \mathbb{N}$. We have $(\mathcal{E}, \odot_j)$ is a suboperad of $(\mathcal{O}, \odot_j)$.*

As a consequence of the above proposition, the set of language operations (graded by their arity), and the set of regular operations (graded by their arity), are operads when equipped with the composition of operations $\circ$. These are important examples that we use throughout the rest of this thesis. We also develop below the example of boolean operations.

Example 4. *For any positive integers m and j with $j \leq m$, for any m-ary boolean function $\mathbf{b}$, and any boolean function $\mathbf{b}'$, we have, from Definition 3, $\otimes_{\mathbf{b} \circ_j \mathbf{b}'} = \otimes_{\mathbf{b}} \circ_j \otimes_{\mathbf{b}'}$. Furthermore, the identity over languages is equal to $\otimes_{\mathbf{b}}$, where $\mathbf{b}$ is the identity over $\{0, 1\}$. Therefore, the set of boolean operations $\mathfrak{O}_{\mathbf{b}}$ equipped with $\circ$ is an suboperad of the set of operations over languages equipped with $\circ$. Thus, $(\mathfrak{O}_{\mathbf{b}}, \circ)$ is an operad.*

We define now the notion of morphism between operads, in a similar way to the notions of morphisms between monoids, groups, or rings. As expected, similar properties arise.

Definition 10. *Let $(\mathcal{O}, \odot)$ and $(\mathcal{O}', \odot')$ be two operads, and ϕ be a graded mapping from $\mathcal{O}$ to $\mathcal{O}'$. We say that ϕ is a* morphism of operads *if and only if*

- *the image by ϕ of the identity of $\mathcal{O}$ is the identity of $\mathcal{O}'$,*
- *for any three integers m, n and j with $1 \leq j \leq m$, for any element $\mathfrak{p}$ of $\mathcal{O}_m$, and for any element $\mathfrak{q}$ of $\mathcal{O}_n$, we have $\phi(\mathfrak{p}) \odot'_j \phi(\mathfrak{q}) = \phi(\mathfrak{p} \odot_j \mathfrak{q})$.*

Furthermore, we say that ϕ is an isomorphism of operads *if it is bijective.*

We mentioned in Example 4 that, for any positive integers m and j with $j \leq m$, for any m-ary boolean function $\mathbf{b}$, and any boolean function $\mathbf{b}'$, we have $\otimes_{\mathbf{b}\circ_j\mathbf{b}'} = \otimes_{\mathbf{b}} \circ_j \otimes_{\mathbf{b}'}$. Furthermore, $\otimes_{\mathrm{Id}_{\{0,1\}}}$ is the identity over languages. Therefore, the mapping from the set of boolean functions to $\mathfrak{O}_{\mathbf{b}}$, that maps a boolean function function $\mathbf{b}$ to the boolean operation $\otimes_{\mathbf{b}}$, is a morphism of operads. Another example is given below.

Example 5. *Let ϕ be the mapping from the set of boolean functions to $\mathcal{B}$ (defined in Example 2) such that, for any k-ary boolean function $\mathbf{b}$, we have*

$$\phi(\mathbf{b}) = \{(e_1, \ldots, e_k) \in \{0,1\}^k \mid \mathbf{b}(e_1, \ldots, e_k) = 1\}.$$

For any non-negative integers k, k', j with $1 \leq j \leq k$, and any boolean functions $\mathbf{b}$ and $\mathbf{b}'$, respectively k-ary and k'-ary, we have

$$\phi(\mathbf{b} \circ_j \mathbf{b}') =$$
$$\{(e_1, \ldots, e_{k+k'-1}) \in \{0,1\}^{k+k'-1} \mid \mathbf{b}(e_1, \ldots, e_{j-1}, \mathbf{b}'(e_j, \ldots, e_{j+k'-1}), e_{j+k'}, \ldots, e_{k+k'+1}) = 1\}.$$

Hence, $\phi(\mathbf{b} \circ_j \mathbf{b}')$ is the set of all $(e_1, \ldots, e_{k+k'-1}) \in \{0,1\}^{k+k'-1}$ such that,

- *either $\mathbf{b}(e_1, \ldots, e_{j-1}, 1, e_{j+n}, \ldots, e_{m+n-1}) = 1$ and $\mathbf{b}'(e_j, \ldots, e_{j+n-1}) = 1$,*
- *or $\mathbf{b}(e_1, \ldots, e_{j-1}, 0, e_{j+n}, \ldots, e_{m+n-1}) = 1$ and $\mathbf{b}'(e_j, \ldots, e_{j+n-1}) = 0$*

It follows from the Definition of $\mathcal{B}$ and $\circledcirc$ that $\phi(\mathbf{b} \circ_j \mathbf{b}') = \phi(\mathbf{b}) \circledcirc_j \phi(\mathbf{b}')$. Furthermore, $\phi(\mathrm{Id}_{\{0,1\}}) = \{1\}$. As a consequence, the mapping ϕ is an isomorphism of operads.

As expected, several properties of group or monoid morphisms have a counterpart for morphisms of operads. For example, the image of an operad by a morphism of operads is an operad.

Proposition 6. *Let $(O, \odot)$ and $(O', \odot')$ be two operads, and let ϕ be a morphism of operads from $(O, \odot)$ to $(O', \odot)$. We let $\phi(O)$ denote the graded set such that $(\phi(O))_n = \phi(O_n) = \phi(O) \cap O'_n$, for any integer n. We have $(\phi(O), \odot')$ is an operad, and ϕ is a morphism of operads from $(O, \odot)$ to $(\phi(O), \odot')$.*

Proof. Let m, j and n be three non-negative integers with $1 \leq j \leq m$, and let $\mathfrak{o}'_1 \in (\phi(O))_m$ and any $\mathfrak{o}'_2 \in (\phi(O))_n$. There exists $\mathfrak{o}_1, \mathfrak{o}_2 \in O$ such that $\phi(\mathfrak{o}_1) = \mathfrak{o}'_1$ and $\phi(\mathfrak{o}_2) = \mathfrak{o}'_2$. Furthermore, by Definition 10, as $\mathfrak{o}'_1 \in O'_m$ and $\mathfrak{o}'_2 \in O'_n$, we have $\mathfrak{o}_1 \in O_m$ and $\mathfrak{o}_2 \in O_n$. Therefore, we have $\mathfrak{o}'_1 \odot'_j \mathfrak{o}'_2 = \phi(\mathfrak{o}_1) \odot'_j \phi(\mathfrak{o}_2) = \phi(\mathfrak{o}_1 \odot_j \mathfrak{o}_2)$. Hence, $\mathfrak{o}_1 \odot_j \mathfrak{o}_2 \in O_{m+n-1}$, and we have $\mathfrak{o}'_1 \odot'_j \mathfrak{o}'_2 \in \phi(O_{m+n-1})$. Furthermore, the image by ϕ of the identity of O is the identity of O'. As a consequence, from Proposition 5, $(\phi(O), \odot')$ is an operad. The fact that ϕ is a morphism of operads from $(O, \odot)$ to $(\phi(O), \odot')$ is straightforward from Definition 10. □

Furthermore, as expected, the composition of two morphisms of operads is a morphism of operads.

Proposition 7. *Let $(O, \odot)$, $(O', \odot')$ and $(O'', \odot'')$ be three operads, let ϕ be a morphism of operads from O to O', and let ψ be a morphism of operads from $(O', \odot)$ to $(O'', \odot'')$. The application $\psi \circ \phi$ is a morphism of operads from $(O, \odot)$ to $(O'', \odot'')$.*

Proof. The fact that the image by $\psi \circ \phi$ of the identity of $\mathcal{O}$ is the identity of $\mathcal{O}''$ is straightforward from Definition 10.

For any integer m, for any $\mathfrak{o} \in \mathcal{O}_m$, we have $\phi(\mathfrak{o}) \in \mathcal{O}'_m$, and therefore $(\psi \circ \phi)(\mathfrak{o}) \in \mathcal{O}''_m$. Hence, $\psi \circ \phi$ is a graded mapping.

Furthermore, for any integers m, j with $1 \leq j \leq m$, for any $\mathfrak{o}_1 \in \mathcal{O}_m$ and any $\mathfrak{o}_2 \in \mathcal{O}$, by Definition 10, we have $\phi(\mathfrak{o}_1 \odot_j \mathfrak{o}_2) = \phi(\mathfrak{o}_1) \odot'_j \phi(\mathfrak{o}_2)$. Therefore, again by Definition 10, we have $\psi(\phi(\mathfrak{o}_1) \odot'_j \phi(\mathfrak{o}_2)) = \psi(\phi(\mathfrak{o}_1)) \odot''_j \psi(\phi(\mathfrak{o}_2))$, and thus $(\psi \circ \phi)(\mathfrak{o}_1 \odot_j \mathfrak{o}_2) = (\psi \circ \phi)(\mathfrak{o}_1) \odot''_j (\psi \circ \phi)(\mathfrak{o}_2)$. As consequence, $\psi \circ \phi$ is a morphism of operads. □

Another expected property is that the inverse of an isomorphism of operads is also an isomorphism of operads.

Proposition 8. *Let ϕ be an isomorphism of operads. Then its inverse ϕ^{-1} is an isomorphism of operads.*

Proof. Let $(\mathcal{O}, \odot)$ and $(\mathcal{O}', \odot')$ be two operads, and ϕ be an isomorphism of operads from $\mathcal{O}$ into $\mathcal{O}'$. The fact that ϕ^{-1} is a graded mapping comes directly from the fact that ϕ is a graded mapping. We check that ϕ^{-1} satisfies the two conditions of Definition 10.

- The image of the identity of $(\mathcal{O}, \odot)$ by ϕ is the identity of $(\mathcal{O}', \odot')$, and thus the image by ϕ^{-1} of the identity of $(\mathcal{O}', \odot')$ is the identity of $(\mathcal{O}, \odot)$.
- For any two integers m and n, for any element $\mathfrak{p}$ of $\mathcal{O}_m$ and any element $\mathfrak{q}$ of $\mathcal{O}_n$, for any $j \in \{1, \dots, m\}$, we have

  $$\phi^{-1}(\mathfrak{p}) \circ_j \phi^{-1}(\mathfrak{q}) = \phi^{-1}(\phi(\phi^{-1}(\mathfrak{p}) \circ_j \phi^{-1}(\mathfrak{q})))$$

  Therefore, by Definition 10,

  $$\phi^{-1}(\mathfrak{p}) \circ_j \phi^{-1}(\mathfrak{q}) = \phi^{-1}(\phi(\phi^{-1}(\mathfrak{p})) \circ_j \phi(\phi^{-1}(\mathfrak{q}))) = \phi^{-1}(\mathfrak{p} \circ_j \mathfrak{q})$$

□

The next proposition shows that a bijection from an operad into any set induces a structure of operad on its image.

Proposition 9. *Let $(\mathcal{O}, \odot)$ be an operad, let $\mathcal{O}'$ be a graded set, and let ϕ be a graded bijection from $\mathcal{O}$ to $\mathcal{O}'$. Let $(\odot')_{j \in \mathbb{N} \setminus 0}$ be the sequence of binary operations defined by $\phi(\mathfrak{p}) \odot'_j \phi(\mathfrak{q}) = \phi(\mathfrak{p} \odot_j \mathfrak{q})$, for any integers m, j with $1 \leq j \leq m$, any $\mathfrak{p} \in \mathcal{O}_m$, and any $\mathfrak{q} \in \mathcal{O}$. The graded set $\mathcal{O}'$ equipped with $\odot'$ is an operad.*

Proof. We check that $(\mathcal{O}', \odot')$ satisfies the three conditions of Definition 4.

- Let Id be the identity of $(\mathcal{O}, \odot)$. For any two positive integers n and $i \leq n$, and any element $\mathfrak{p}'$ of $\mathcal{O}'_n$, there exists $\mathfrak{p} \in \mathcal{O}_n$ such that $\phi(\mathfrak{p}) = \mathfrak{p}'$, and we have

  $$\phi(\mathrm{Id}) \odot'_1 \mathfrak{p}' = \phi(\mathrm{Id}) \odot'_1 \phi(\mathfrak{p}) = \phi(\mathrm{Id} \odot'_1 \mathfrak{p}) = \phi(\mathfrak{p} \odot_i \mathrm{Id}) = \mathfrak{p}' \odot'_i \phi(\mathrm{Id}).$$

  Furthermore, we similarly have

  $$\phi(\mathrm{Id}) \odot'_1 \mathfrak{p}' = \phi(\mathrm{Id}) \odot'_1 \phi(\mathfrak{p}) = \phi(\mathrm{Id} \odot'_1 \mathfrak{p}) = \phi(\mathfrak{p}) = \mathfrak{p}'.$$

  Therefore, $\phi(\mathrm{Id})$ satisfies the "identity" rule for $\mathcal{O}'$.

- For any positive integers m, n, s, $i \leq m$, $j \leq n$, and any $\mathfrak{p}'$, $\mathfrak{q}'$, $\mathfrak{r}'$, elements of $\mathcal{O}'_m$, $\mathcal{O}'_n$, $\mathcal{O}'_s$ respectively, there exists $\mathfrak{p} \in \mathcal{O}_m$, $\mathfrak{q} \in \mathcal{O}_n$, and $\mathfrak{r} \in \mathcal{O}_s$ such that $\phi(\mathfrak{p}) = \mathfrak{p}'$, $\phi(\mathfrak{q}) = \mathfrak{q}'$, $\phi(\mathfrak{r}) = \mathfrak{r}'$, and we have

$$\mathfrak{p}' \odot_i (\mathfrak{q}' \odot_j \mathfrak{r}') = \phi(\mathfrak{p}) \odot_i (\phi(\mathfrak{q}) \odot_j \phi(\mathfrak{r})) = \phi(\mathfrak{p} \odot_i (\mathfrak{q} \odot_j \mathfrak{r})) = \phi((\mathfrak{p} \odot_i \mathfrak{q}) \odot_{i+j-1} \mathfrak{r}) = (\mathfrak{p}' \odot_i \mathfrak{q}') \odot_{i+j-1} \mathfrak{r}'.$$

- For any positive integers m, n, s, i, and j such that $i < j \leq m$, and any $\mathfrak{p}'$, $\mathfrak{q}'$, $\mathfrak{r}'$, elements of $\mathcal{O}'_m$, $\mathcal{O}'_n$, $\mathcal{O}'_s$ respectively, there exists $\mathfrak{p} \in \mathcal{O}_m$, $\mathfrak{q} \in \mathcal{O}_n$, and $\mathfrak{r} \in \mathcal{O}_s$ such that $\phi(\mathfrak{p}) = \mathfrak{p}'$, $\phi(\mathfrak{q}) = \mathfrak{q}'$, $\phi(\mathfrak{r}) = \mathfrak{r}'$, and we have

$$(\mathfrak{p}' \odot_j \mathfrak{q}') \odot_i \mathfrak{r}' = (\phi(\mathfrak{p}) \odot_j \phi(\mathfrak{q})) \odot_i \phi(\mathfrak{r}) = \phi((\mathfrak{p} \odot_j \mathfrak{q}) \odot_i \mathfrak{r}) = (\mathfrak{p}' \odot_i \mathfrak{r}') \odot_{j+s-1} \mathfrak{q}'.$$

□

Example 6. *Let ϕ be the bijection defined in Example 5,* i.e., *the mapping from the set of boolean functions to $\mathcal{B}$ such that, for any k-ary boolean function* **b**, *we have*

$$\phi(\mathbf{b}) = \{(e_1, \ldots, e_k) \in \{0,1\}^k \mid \mathbf{b}(e_1, \ldots, e_k) = 1\}.$$

Without knowing that ϕ is an isomorphism of operads, or even that $(\mathcal{B}, \circledcirc)$ is an operad, we could follow the reasoning of Example 5 to prove that for any positive integers m and j with $1 \leq j \leq m$, for any m-ary boolean function **b**, *and any boolean function* **b**′, *we have $\phi(\mathbf{b}) \circledcirc_j \phi(\mathbf{b}') = \phi(\mathbf{b} \circ_j \mathbf{b})$. With this statement, Proposition 9 immediately gives us a new proof that $(\mathcal{B}, \circledcirc)$ is an operad.*

We now introduce quotient operads. To this aim, we characterize equivalence relations that preserve the operadic structure, so that the quotient of an operad remains an operad.

Definition 11. *Let $(\mathcal{O}, \odot)$ be an operad, and let $\sim$ be a graded equivalence relation on $\mathcal{O}$. We say that* $\sim$ respects $\odot$ *if, for any positive integers m and j with $1 \leq j \leq m$, for any $\mathfrak{p}, \mathfrak{p}' \in \mathcal{O}_m$ such that $\mathfrak{p} \sim \mathfrak{p}'$, and any $\mathfrak{q}, \mathfrak{q}' \in \mathcal{O}$ such that $\mathfrak{q} \sim \mathfrak{q}'$, we have $\mathfrak{p} \odot_j \mathfrak{q} \sim \mathfrak{p}' \odot_j \mathfrak{q}'$. Furthermore, we let $\odot/_\sim$ denote the sequence of binary operations over $\mathcal{O}/_\sim$ such that, for any integers j and m with $1 \leq j \leq m$, for any $\mathfrak{p} \in \mathcal{O}_m$ and $\mathfrak{q} \in \mathcal{O}$, we have $\widetilde{\mathfrak{p}} \, (\odot/_\sim)_j \, \widetilde{\mathfrak{q}} = \widetilde{\mathfrak{p} \odot_j \mathfrak{q}}$.*

Proposition 10. *Let $(\mathcal{O}, \odot)$ be an operad, and let $\sim$ be a graded equivalence relation on $\mathcal{O}$ that respects $\odot$. The set $\mathcal{O}/_\sim$ equipped with $\odot/_\sim$ is an operad. It is called the* quotient *of $(\mathcal{O}, \odot)$ by $\sim$.*

Proof. We check that $(\mathcal{O}/_\sim, \odot/_\sim)$ satisfies the three conditions of Definition 4.

- Let Id be the identity of $(\mathcal{O}, \odot)$. For any positive two integers n and $j \leq n$, and any element $\mathfrak{p}$ of $\mathcal{O}_n$, we have

$$\widetilde{\mathfrak{p}} (\odot/_\sim)_j \widetilde{\mathrm{Id}} = \widetilde{\mathfrak{p} \odot_j \mathrm{Id}} = \widetilde{\mathfrak{p}},$$

  and similarly:

$$\widetilde{\mathrm{Id}} (\odot/_\sim)_1 \widetilde{\mathfrak{p}} = \widetilde{\mathrm{Id} \odot_1 \mathfrak{p}} = \widetilde{\mathfrak{p}}.$$

  Therefore, $\widetilde{\mathrm{Id}}$ satisfies the "identity" rule for $(\mathcal{O}/_\sim, \odot/_\sim)$.

- For any positive integers $m, n, s, i \leq m, j \leq n$, and any $\mathfrak{p}, \mathfrak{q}, \mathfrak{r}$, elements of O_m, O_n, O_s respectively, we have

$$\widetilde{\mathfrak{p}}(\odot/_\sim)_i(\widetilde{\mathfrak{q}}(\odot/_\sim)_j\widetilde{\mathfrak{r}}) = \widetilde{\mathfrak{p}}(\odot/_\sim)_i\widetilde{(\mathfrak{q}\odot_j\mathfrak{r})} = \widetilde{\mathfrak{p}\odot_i(\mathfrak{q}\odot_j\mathfrak{r})} = \widetilde{(\mathfrak{p}\odot_i\mathfrak{q})\odot_{i+j-1}\mathfrak{r}}$$
$$= \widetilde{(\mathfrak{p}\odot_i\mathfrak{q})}(\odot/_\sim)_{i+j-1}\widetilde{\mathfrak{r}} = (\widetilde{\mathfrak{p}}(\odot/_\sim)_i\widetilde{\mathfrak{q}})(\odot/_\sim)_{i+j-1}\widetilde{\mathfrak{r}}.$$

- For any positive integers m, n, s, i, and j such that $i < j \leq m$, and any $\mathfrak{p}, \mathfrak{q}, \mathfrak{r}$, elements of O_m, O_n, O_s respectively, we have

$$(\widetilde{\mathfrak{p}}(\odot/_\sim)_j\widetilde{\mathfrak{q}})(\odot/_\sim)_i\widetilde{\mathfrak{r}} = \widetilde{(\mathfrak{p}\odot_j\mathfrak{q})}(\odot/_\sim)_i\widetilde{\mathfrak{r}} = \widetilde{(\mathfrak{p}\odot_j\mathfrak{q})\odot_i\mathfrak{r}} = \widetilde{(\mathfrak{p}\odot_i\mathfrak{r})\odot_{j+s-1}\mathfrak{q}}$$
$$= \widetilde{(\mathfrak{p}\odot_i\mathfrak{r})}(\odot/_\sim)_{j+s-1}\widetilde{\mathfrak{q}} = (\widetilde{\mathfrak{p}}(\odot/_\sim)_i\widetilde{\mathfrak{r}})(\odot/_\sim)_{j+s-1}\widetilde{\mathfrak{q}}.$$

□

Finally, we show how a morphisms of operad ϕ from $(O, \odot)$ to $(O', \odot')$, induces an equivalence relation $\sim^\phi$ on O, by which $(O, \odot)$ can be divided. Furthermore, we show that there is a injective morphism $\widehat{\phi}$ that makes the diagram of Figure 3.8 commutative.

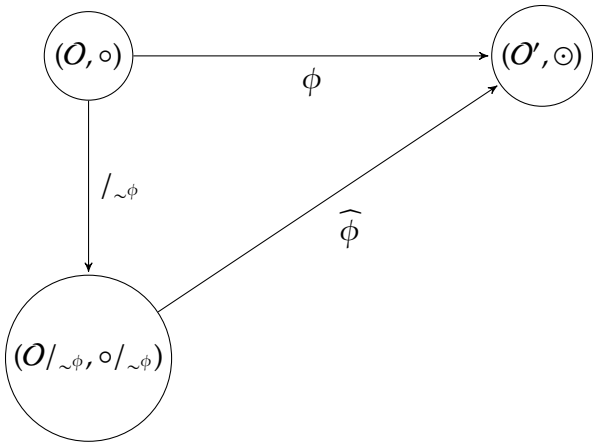


Figure 3.8: Commutative diagram for ϕ, $/_{\sim^\phi}$, and $\widehat{\phi}$.

Definition 12. *Let $(O, \odot)$ and $(O', \odot')$ be two operads, and let ϕ be a morphism of operads from $(O, \odot)$ to $(O', \odot')$. We let $\sim^\phi$ denote the equivalence relation over O such that, for any $\mathfrak{o}, \mathfrak{o}' \in O$, $\mathfrak{o} \sim^\phi \mathfrak{o}'$ if and only if $\phi(\mathfrak{o}) = \phi(\mathfrak{o}')$. Furthermore, we let $\widehat{\phi}$ denote the function from $O/_{\sim^\phi}$ to O' such that, for any $\mathfrak{o} \in O$, we have $\widehat{\phi}(\widetilde{\mathfrak{o}}^\phi) = \phi(\mathfrak{o})$.*

Notice that, in the above definition, $\widehat{\phi}$ is indeed well-defined, since for any $\mathfrak{o}, \mathfrak{o}' \in O$ such that $\widetilde{\mathfrak{o}} = \widetilde{\mathfrak{o}'}$, we have $\phi(\mathfrak{o}) = \phi(\mathfrak{o}')$, and therefore $\widehat{\phi}(\widetilde{\mathfrak{o}}^\phi) = \widehat{\phi}(\widetilde{\mathfrak{o}'}^\phi)$.

Proposition 11. *Let $(O, \odot)$ and $(O', \odot')$ be two operads, and let ϕ be a morphism of operads from $(O, \odot)$ to $(O', \odot')$. The set $O/_{\sim_\phi}$ equipped with $\odot/_{\sim_\phi}$ is an operad, and $\widehat{\phi}$ is an injective morphism of operads from $(O/_{\sim_\phi}, \odot/_{\sim_\phi})$ to $(O', \odot')$.*

Proof. For any two integers m and n, any $\mathfrak{o} \in O_m$ and any $\mathfrak{o}' \in O_n$, if $\mathfrak{o} \sim^\phi \mathfrak{o}'$, then $\phi(\mathfrak{o}) = \phi(\mathfrak{o}')$, and therefore $m = n$. The equivalence relation $\sim^\phi$ is therefore graded for O.

Furthermore, for any integers m and j with $1 \leq j \leq m$, for any $\mathfrak{o}_1, \mathfrak{o}'_1 \in O_m$ and for any $\mathfrak{o}_2, \mathfrak{o}'_2 \in \mathfrak{O}$ such that $\mathfrak{o}_1 \sim^\phi \mathfrak{o}'_1$ and $\mathfrak{o}_2 \sim^\phi \mathfrak{o}'_2$, we have

$$\phi(\mathfrak{o}_1 \odot_j \mathfrak{o}_2) = \phi(\mathfrak{o}_1) \odot'_j \phi(\mathfrak{o}_2) = \phi(\mathfrak{o}'_1) \odot'_j \phi(\mathfrak{o}'_2) = \phi(\mathfrak{o}'_1 \odot_j \mathfrak{o}'_2),$$

and therefore $(\mathfrak{o}_1 \odot_j \mathfrak{o}_2) \sim^\phi (o'_1 \odot_j \mathfrak{o}'_2)$. As a consequence, by Proposition 10, $(O/_{\sim_\phi}, \odot/_{\sim_\phi})$ is an operad. Furthermore, for any $\mathfrak{o}_1, \mathfrak{o}_2 \in O/_{\sim_\phi}$, we have

$$\widehat{\phi}(\widetilde{\mathfrak{o}_1}^\phi (\odot/_{\sim_\phi})_j \widetilde{\mathfrak{o}_2}^\phi) = \widehat{\phi}(\widetilde{\mathfrak{o}_1 \odot_j \mathfrak{o}_2}^\phi) = \phi(\mathfrak{o}_1 \odot_j \mathfrak{o}_2) = \phi(\mathfrak{o}_1) \odot'_j \phi(\mathfrak{o}_2) = \widehat{\phi}(\widetilde{\mathfrak{o}_1}^\phi) \odot'_j \widehat{\phi}(\widetilde{\mathfrak{o}_2}^\phi).$$

Therefore, $\widehat{\phi}$ is a morphism of operads, and the fact that it $\widehat{\phi}$ is injective comes directly from its definition. □

We now have all the tools required to handle operads for the remainder of this thesis.

Chapter 4

Modifiers and 1-uniform operations

In this chapter, we build a framework to compute the state complexity of a certain kind of regular operations, that we call 1-*uniform*. For this purpose, we define *modifiers*, a counterpart to 1-uniform operations in the space of operations over DFAs. By studying modifiers themselves and the link between modifiers and 1-uniform operations, we prove that every 1-uniform operation admits a certain kind of DFA as witness, called *monster*. Monsters are k-tuples of DFAs built to have alphabets as large as possible, while not having two letters with the same transition functions. Furthermore, throughout this chapter, we pay a close attention to the algebraic structure behind the notions we introduce, and we summarize this structure in Figure 4.19. In fact, this is the main difference between our approach and the work of Sylvie Davies in [13]. Indeed, even though Theorem 1 and Theorem 2 (which are the main results of this chapter) are also proved in [13], we introduce them Theorems with the point of view of operads. This allows us to cleanly capture how the composition of operations is carried from modifiers to 1-uniform operations. Furthermore, we introduce some new notations in Section 4.2.3 that are different from the ones used in [13], and show that their use is correct (Proposition 17). We use these notations extensively throughout the rest of the thesis, and we believe in particular that they make Chapter 7 much easier to read.

4.1 About 1-uniform operations

In this section, we define 1-uniform operations and give some examples of well-known 1-uniform operations. We also give an example of regular operation that is not 1-uniform, and prove that 1-uniform operations are stable by composition.

Definition 13. *A k-ary operation $\otimes$ is* 1*-uniform if it is regular, and if it commutes with every inverse* 1*-uniform morphism,* i.e.*, for any k-tuple of regular languages $(L_1, \ldots, L_k)$, for any* 1*-uniform morphism ϕ, $\otimes(\phi^{-1}(L_1), \ldots, \phi^{-1}(L_k)) = \phi^{-1}(\otimes(L_1, \ldots, L_k))$.*

In the rest of this thesis, the operations for which we give state complexity results are all 1-uniform. Many well-known operations are 1-uniform. Let us illustrate this concept by proving that the Kleene star is 1-uniform.

Proposition 12. *The Kleene star is* 1*-uniform.*

Proof. Let Σ and Γ be two alphabets. Let L be a regular language over Σ, and let ϕ be a 1-uniform morphism from Γ^* to Σ^*. We first prove $\phi^{-1}(L)^\star \subseteq \phi^{-1}(L^\star)$. Indeed, if v is a word in $\phi^{-1}(L)^\star$, then there exists an integer n and n words $u_1, \dots, u_n$ such that $v = u_1 \cdots u_n$. Therefore, there exists n words of L $t_1, \dots, t_n$ such that $\phi(u_i) = t_i$ for all $i \in \{1, \dots, n\}$. We thus have $\phi(v) = w$ with $w = t_1 \cdots t_n$ and $v \in \phi^{-1}(L^\star)$.

Conversely, let v be a word of $\phi^{-1}(L^\star)$. There exists an integer n and $t_1, \dots, t_n$ n words of L such that $\phi(v) = w$, with $w = t_1 \cdots t_n$. As ϕ is 1-uniform, $\phi(v) = \phi(v_1) \cdots \phi(v_{|v|})$, and each $\phi(v_j)$ are letters of Σ. Therefore, v and w have the same length, and $\phi(v_j) = w_j$, for all $j \in \{1, \dots, |v|\}$. As a consequence, for all $i \in \{1, \dots, n\}$, if $u_i = v_{|t_1|+|t_2|+\cdots+|t_{i-1}|+1} \cdots v_{|t_1|+|t_2|+\cdots+|t_i|}$, we have $\phi(u_i) = t_i$ and $v = u_1 \cdots u_n$. We thus have $v \in \phi^{-1}(L)^\star$, and $\phi^{-1}(L^\star) \subseteq \phi^{-1}(L)^\star$. □

Another example of a 1-uniform operation is given by the union of two languages (*i.e.*, the operation $\otimes$ defined by $\otimes(L_1, L_2) = L_1 \cup L_2$), which is a binary operation. This follows from the following remark:

Remark 1. *For any function ϕ from a set E to a set F, $\phi^{-1}(X \cup Y) = \phi^{-1}(X) \cup \phi^{-1}(Y)$, for all $X, Y \subseteq F$.*

Indeed, to prove that the union of languages is a 1-uniform operation, we only have to apply the above remark to the particular case of $E = \Gamma^*$ and $F = \Sigma^*$ where Σ and Γ are any two alphabets, and ϕ is any 1-uniform morphism from Γ^* to Σ^*.

We see many other examples of 1-uniform operations throughout this chapter. We refrain from mentioning them all in this section, as we develop in the following sections tools that make proving the 1-uniformity of regular operations very easy. However, even though many well-known regular operations are 1-uniform, some of them are not. The right quotient is an example of a regular operation that is not 1-uniform.

Example 7. *The right quotient of two languages is defined by $L_1 \cdot L_2^{-1} = \{u \mid uv \in L_1 \text{ for some } v \in L_2\}$. We show that the operation $\otimes(L_1, L_2) = L_1 \cdot L_2^{-1}$ is regular but not 1-uniform.*

Let $\Gamma = \Sigma = \{a, b\}$, and let ϕ be the 1-uniform morphism from Γ^* to Σ^* such that $\phi(a) = \phi(b) = a$. Furthermore, let $L_1 = \{ab\}$ and $L_2 = \{b\}$. We have $\phi^{-1}(L_1) = \phi^{-1}(L_2) = \emptyset$, and therefore, $\phi^{-1}(L_1) \cdot (\phi^{-1}(L_2))^{-1} = \emptyset$. However, $L_1 \cdot L_2^{-1} = \{a\}$, and therefore $\phi^{-1}(L_1 \cdot L_2^{-1}) = \{a, b\}$. We thus have $\phi^{-1}(L_1) \cdot \phi^{-1}(L_2)^{-1} \neq \phi^{-1}(L_1 \cdot L_2^{-1})$. As a consequence, the right quotient operation $\otimes$ is not 1-uniform.

Let $\mathfrak{O}_\mathbf{u}$ be the set of all 1-uniform operations. From Lemma 1 and Definition 13, $\mathfrak{O}_\mathbf{u}$ is stable by the composition of operations $\circ$. Therefore, by Proposition 5, $(\mathfrak{O}_\mathbf{u}, \circ)$ is a suboperad of the operad of regular operations.

Proposition 13. *The set of 1-uniform operations equipped with the composition of operations $\circ$ is an operad.*

The above proposition is quite important, as it means that the main result of this chapter (Theorem 2) can be applied to a large number of interesting operations, some already studied (for example the star of union [27], or star-complement-star [22, 29]), and some that we have yet to study (for example the star of symmetric difference studied later in Chapter 5).

4.2 Modifiers

4.2.1 Definition

The definition of operational state complexity involves directly the state complexity of languages. The definition of the state complexity of languages, in turn, directly involves the notion of minimal DFA. The easiest way to compute the minimal DFA associated with a language relies on first giving a DFA that recognizes this language, and then on minimizing this DFA. Therefore, the most convenient way to compute the state complexity of regular operations involves doing computations directly on DFAs. Hence, in order to prove state complexity results for 1-uniform operations, it is convenient to link these operations over languages with operations acting directly over DFAs. To this aim, we define a counterpart of 1-uniform operations, constituted of operations over DFAs called *modifiers*. To give more details, we first define the notions of *state configuration* and *transition configuration*.

Definition 14. *A* state configuration *is a* 3*-tuple* (Q, i, F) *such that* Q *is a finite set,* $i \in Q$ *and* $F \subseteq Q$*. The state configuration of a DFA* $A = (\Sigma, Q, i, F, \delta)$ *is the triplet* (Q, i, F)*. A* transition configuration *is a* 4*-tuple* (Q, i, F, δ) *such that* (Q, i, F) *is a state configuration and* $\delta \in Q^Q$*. If* $A = (\Sigma, Q, i, F, \delta)$ *is a DFA, the transition configuration of a letter* $a \in \Sigma$ *in a* A *is the* 4*-tuple* (Q, i, F, δ^a)*.*

In the graphical representation of a DFA, the state configuration corresponds to keeping the states and removing the arrows, with the initial state and final states marked. A transition configuration corresponds to state configuration equipped with a transition function over its states.

We let $\mathcal{A}_k^\Sigma$ denote the set of all k-tuples of DFAs $(A_1, \ldots, A_k)$ such that, for any $j \in \{1, \ldots, k\}$, the alphabet of A_j is Σ. We also let $\mathcal{A}_k$ denote the union over all alphabets Σ of $\mathcal{A}_k^\Sigma$, *i.e.,* the set of all k-tuples of DFAs having the same alphabet.

A k-modifier is a k-ary operation over DFAs, that associates a k-tuple in A_k^Σ with a DFA whose alphabet is also Σ. However, intuitively, modifiers do not directly associate multiple DFAs with a DFA, but instead use intermediate steps. Indeed, they first associate tuples of state configurations with a state configuration, and then associate tuples of transition configurations with a transition function. Finally, they naturally construct a DFA from multiple DFAs using the previous steps. In other words, a modifier $\mathfrak{m}$ is an operation over DFAs that must satisfy the following conditions:

- the state configuration of the output of $\mathfrak{m}$ only depends on the state configuration of its inputs,
- the transition function of a letter a in the output of $\mathfrak{m}$ only depends on the transition configurations of the letter a in the inputs of $\mathfrak{m}$.

Furthermore, in order for the conditions above to make sense, we set that all the inputs of a modifier must have the same alphabet as its output. More formally,

Definition 15. *A* k-modifier *is a* k*-ary operation* $\mathfrak{m}$ *of domain* $\mathcal{A}_k$ *that satisfies the following conditions:*

- *for any* $(A_1, \ldots, A_k) \in \mathcal{A}_k^\Sigma$, *the alphabet of* $\mathfrak{m}(A_1, \ldots, A_k)$ *is* Σ,
- *for any* $(A_1, \ldots, A_k) \in \mathcal{A}_k^\Sigma$ *and any* $(B_1, \ldots, B_k) \in \mathcal{A}_k^\Gamma$, *such that* A_j *has the same state configuration as* B_j *for any* $j \in \{1, \ldots, k\}$, *we have*
  - *the state configurations of* $\mathfrak{m}(A_1, \ldots, A_k)$ *and of* $\mathfrak{m}(B_1, \ldots, B_k)$ *are the same,*
  - *if there exist two letters* $a \in \Sigma$ *and* $b \in \Gamma$, *such that, for any* $j \in \{1, \ldots, k\}$, *the transition function of* a *in* A_j *is equal to the transition function of* b *in* B_j, *then the transition function of* a *in* $\mathfrak{m}(A_1, \ldots, A_k)$ *is equal to the transition function of* b *in* $\mathfrak{m}(B_1, \ldots, B_k)$.

In order to familiarize ourselves with this notion, we give many examples in the next section.

4.2.2 Examples

At first sight, the definition of modifiers seem very broad, and may even seem to include all operations over DFAs. However, this is not the case. First, it is straightforward that a unary operation over DFAs that maps two DFAs with the same state configuration to two DFAs with different state configurations is not a modifier. Furthermore, the third condition of Definition 15 cannot be removed. Indeed, a unary operation $\mathfrak{o}$ over DFAs that maps the DFA A of Figure 4.1 to the DFA B, and the DFA C to the DFA D, is not a modifier, even though it satisfies so far the first and second conditions of Definition 15. That is because the transition configurations of the letter a in DFA A and DFA C are the same, while the transition configurations of the letter a in DFA B and DFA D are not the same.

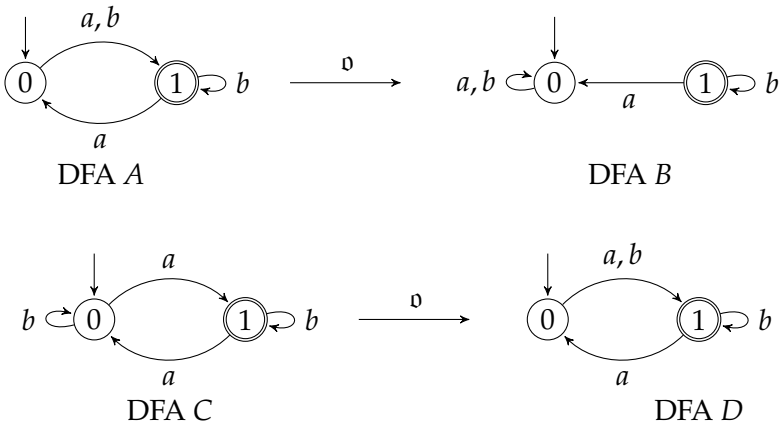


Figure 4.1: An operation over DFAs that is not a modifier.

Several well-known constructions over DFAs can be seen as modifiers. The simplest is probably the (set-theoretic) complement.

Example 8. *For any DFA* $A = (\Sigma, Q, i, F, \delta)$, *we define*

$$\mathfrak{Comp}(A) = (\Sigma, Q, i, Q \setminus F, \delta).$$

This modifier follows the classical construction for the complement of a DFA [36]. We use this construction to compute the complement $\mathfrak{Comp}(A)$ *(Figure 4.3) of the DFA A (Figure 4.2).*

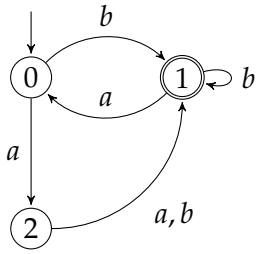


Figure 4.2: A DFA A.

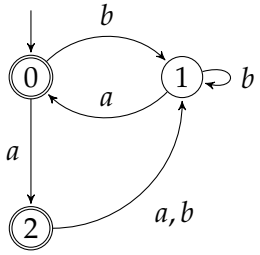


Figure 4.3: $\mathfrak{Comp}(A)$.

Example 9. *For any DFA* $A = (\Sigma, Q, i, F, \delta)$*, we define*

$$\mathfrak{Sqrt}_k(A) = (\Sigma, Q^Q, \mathrm{Id}_Q, \{\phi \in Q^Q | \phi^k(i) \in F\}, \delta'),$$

where for any $a \in \Sigma$, $\delta'^a(\phi) = \delta^a \circ \phi$*. Furthermore, we let* $\mathfrak{Sqrt}$ *denote the modifier* $\mathfrak{Sqrt}_2$*. The DFA of Figure 4.5, where* $[ij]$ *represents the function* ϕ *such that* $\phi(0) = i$ *and* $\phi(1) = j$*, is the image of the DFA B (Figure 4.4) by the modifier* $\mathfrak{Sqrt}$*.*

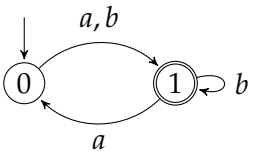


Figure 4.4: A DFA B.

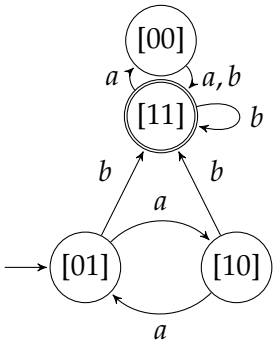


Figure 4.5: The DFA $\mathfrak{Sqrt}(B)$.

The above modifier $\mathfrak{Sqrt}$ *follows the classical construction on DFAs for the square root operation on regular languages. This remains true in the general case of the k-th root.*

Proposition 14. *For any DFA A, we have* $\sqrt[k]{\mathrm{L}(A)} = \mathrm{L}(\mathfrak{Sqrt}_k(A))$.

Proof. Let $A = (\Sigma, Q, i, F, \delta)$ be a DFA, let $\mathfrak{Sqrt}_k(A) = (\Sigma, Q', i', F', \delta')$, and let w be any word of Σ^*. The word w^k is accepted by A if and only if $\delta^{w^k}(i) \in F$. However, we have

$$\delta^{w^k} = \underbrace{\delta^w \circ \ldots \circ \delta^w}_{k \text{ times}} = (\delta^w)^k.$$

Therefore, $w^k \in \mathrm{L}(A)$ if and only if $\delta^w \in F'$. But $\delta'^w(i') = \delta^w \circ \mathrm{Id}_Q = \delta^w$. We thus have $w^k \in \mathrm{L}(A)$ if and only if $\delta'^w(i') \in F'$. Hence, we have $\sqrt[k]{\mathrm{L}(A)} = \mathrm{L}(\mathfrak{Sqrt}_k(A))$. □

Example 10. *We generalize the case of the complement to boolean operations. We define a modifier from a boolean function by using the classical "product automaton" construction, and we show that these modifiers correspond to boolean operations.*

Definition 16. *Let* $\mathbf{b}$ *be a k-ary boolean function. We let* $\mathfrak{m}_{\mathbf{b}}$ *denote the modifier such that, for any k-tuple of DFAs* $(A_1, \ldots, A_k)$ *with* $A_j = (\Sigma, Q_j, i_j, F_j, \delta_j)$,

$$\mathfrak{m}_{\mathbf{b}}(A_1, \ldots, A_k) = (\Sigma, Q_1 \times Q_2 \times \cdots \times Q_k, (i_1, i_2, \ldots, i_k), F', (\delta_1, \delta_2, \ldots, \delta_k)),$$

where $(q_1, \ldots, q_k) \in F'$ *if and only if* $\mathbf{b}([q_1 \in F_1], \ldots, [q_k \in F_k]) = 1$.

Figures 4.8, 4.9, and 4.9, display the image of the pair of DFAs (A_1, A_2) *defined in Figures 4.6 and 4.7 by the modifiers* $\mathfrak{m}_{\mathbf{union}}$, $\mathfrak{m}_{\mathbf{inter}}$, *and* $\mathfrak{m}_{\mathbf{xor}}$.

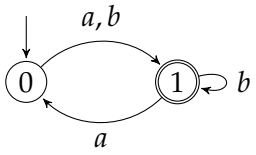


Figure 4.6: DFA A_1.

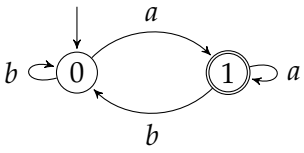


Figure 4.7: DFA A_2.

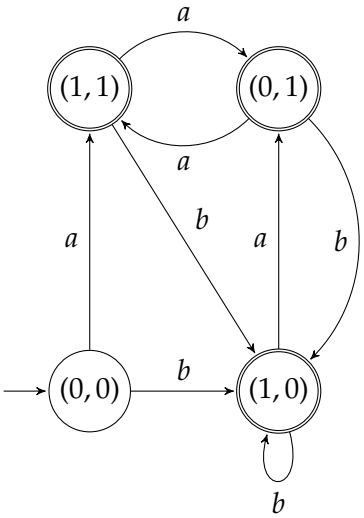


Figure 4.8
The DFA $\mathfrak{m}_{\mathbf{union}}(A_1, A_2)$.

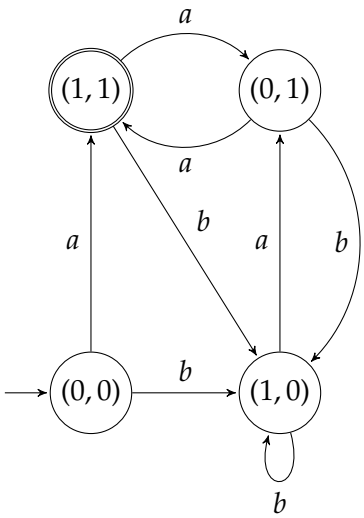


Figure 4.9
The DFA $\mathfrak{m}_{\mathbf{inter}}(A_1, A_2)$.

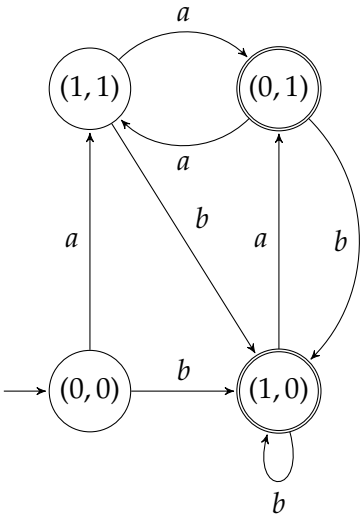


Figure 4.10
The DFA $\mathfrak{m}_{\mathbf{xor}}(A_1, A_2)$.

The modifier $\mathfrak{m}_{\mathbf{b}}$ *corresponds to the operation* $\otimes_{\mathbf{b}}$. *In other words,*

Proposition 15. *For any boolean function* $\mathbf{b}$, *and for any k-tuple of DFAs* $(A_1, \ldots, A_k)$, *we have*

$$\mathrm{L}(\mathfrak{m}_{\mathbf{b}}(A_1, \ldots, A_k)) = \otimes_{\mathbf{b}}(\mathrm{L}(A_1), \ldots, \mathrm{L}(A_k)).$$

Proof. Let $\mathbf{b}$ be a boolean function, and let $A_1, \dots, A_k$ be k DFAs with $A_j = (\Sigma, Q_j, i_j, F_j, \delta_j)$ for all $j \in \{1, \dots, k\}$, let $w = a_1 \cdots a_n$ be a word in Σ^*, and let $\mathfrak{m}_{\mathbf{b}}(A_1, \dots, A_k) = (\Sigma, Q, i, F, \delta)$. From Definition 3, the word w is in $\otimes_{\mathbf{b}}(\mathrm{L}(A_1), \dots, \mathrm{L}(A_k))$ if and only if $\mathbf{b}([\delta_1^w(i_1) \in F_1], \dots, [\delta_k^w(i_k) \in F_k]) = 1$. However, using Definition 16, an easy induction shows that we have $\delta^w(i_1, \dots, i_k) = (\delta_1^w(i_1), \dots, \delta_k^w(i_k))$. Therefore, from Definition 16, w is in $\otimes_{\mathbf{b}}(\mathrm{L}(A_1), \dots, \mathrm{L}(A_k))$ if and only if $\delta^w(i_1, \dots, i_k) \in F$. Consequently, as $(i_1, \dots, i_k) = i$, w is in $\otimes_{\mathbf{b}}(\mathrm{L}(A_1), \dots, \mathrm{L}(A_k))$ if and only if w is in $\mathrm{L}(\mathfrak{m}_{\mathbf{b}}(A_1, \dots, A_k))$. □

4.2.3 Alternative notations

We let $\mathfrak{M}$ denote the set of all modifiers. Before giving more examples, we clarify and formalize the algebraic structure behind modifiers. First notice that modifiers naturally form an operad.

Proposition 16. *The set of modifiers equipped with the composition of operations is an operad.*

Proof. We show that $\mathfrak{M}$ is stable by the composition of operations $\circ$. Let k, k' and j be three positive integers with $j \leq k$, and let $\mathfrak{m}$ and $\mathfrak{m}'$ be two modifiers, respectively k-ary and k'-ary. By Definition 6, as the domains of $\mathfrak{m}$ and $\mathfrak{m}'$ are respectively $\mathcal{A}_k$ and $\mathcal{A}_{k'}$, the domain of the $k + k' - 1$-ary operation $\mathfrak{m} \circ_j \mathfrak{m}'$ is $\mathcal{A}_{k+k'-1}$. Furthermore, recall that, by Definition 6, for any $(A_1, \dots, A_{k+k'-1}) \in \mathcal{A}_{k+k'-1}$, we have

$$\mathfrak{m} \circ_j \mathfrak{m}'(A_1, \dots, A_{k+k'-1}) = \mathfrak{m}(A_1, \dots, A_{j-1}, \mathfrak{m}'(A_j, \dots, A_{k'+j-1}), A_{k'+j}, \dots, A_{k+k'-1}). \tag{4.1}$$

This equality helps us check that $\mathfrak{m} \circ_j \mathfrak{m}'$ satisfies the three points of Definition 15.

- For any alphabet Σ, and any $(A_1, \dots, A_{k+k'-1}) \in \mathcal{A}^{\Sigma}_{k+k'-1}$, we have $(A_j, \dots, A_{k'+j-1}) \in \mathcal{A}^{\Sigma}_{k'}$, and therefore the alphabet of $\mathfrak{m}'(A_1, \dots, A_{k+k'-1})$ is Σ. As a consequence,

  $$(A_1, \dots, A_{j-1}, \mathfrak{m}'(A_j, \dots, A_{k'+j-1}), A_{k'+j}, \dots, A_{k+k'-1}) \in A^{\Sigma}_k,$$

  and by (4.1), the alphabet of $\mathfrak{m} \circ_j \mathfrak{m}'(A_1, \dots, A_{k+k'-1})$ is Σ.

- For any two elements $(A_1, \dots, A_{k+k'-1})$ of $\mathcal{A}^{\Sigma}_{k+k'-1}$ and $(B_1, \dots, B_{k+k'-1})$ of $\mathcal{A}^{\Gamma}_{k+k'-1}$, such that, for any $j \in \{1, \dots, k + k' - 1\}$, A_j and B_j have the same state configuration, we have

  - the state configurations of $\mathfrak{m}'(A_j, \dots, A_{k+j-1})$ and of $\mathfrak{m}'(B_j, \dots, B_{k+j-1})$ are the same, and therefore, by (4.1), the state configurations of $\mathfrak{m} \circ_j \mathfrak{m}'(A_1, \dots, A_{k+k'-1})$ and $\mathfrak{m} \circ_j \mathfrak{m}'(B_1, \dots, B_{k+k'-1})$ are the same.
  - If there exist two letters $a \in \Sigma$ and $b \in \Gamma$ such that, for any $j \in \{1, \dots, k+k'-1\}$, the transition function of a in A_j is the same as the transition function of b in B_j, then the transition function of a in $\mathfrak{m}'(A_1, \dots, A_k)$ is equal to the transition function of b in $\mathfrak{m}'(B_1, \dots, B_k)$, and therefore, by (4.1), the transition function of a in $\mathfrak{m} \circ_j \mathfrak{m}'(A_1, \dots, A_{k+k'-1})$ is equal to the transition function of b in $\mathfrak{m} \circ_j \mathfrak{m}'(B_1, \dots, B_{k+k'-1})$.

Therefore, $\mathfrak{M}$ is stable by $\circ$. Furthermore, it is easy to check that the identity over DFAs is a modifier. In addition, by Proposition 4, the operations over DFAs equipped with the composition of operations is an operad. As a consequence, by Proposition 5, $(\mathfrak{M}, \circ)$ is an operad. □

Even though modifiers behave well with respect to the composition of operations, they are not easy to handle in an algebraic context. One of the reasons is that they naturally contain more information than necessary.

We let $\mathcal{TC}$ denote the set of all transition configurations. We seek to extract the essential information contained in a modifier. To this aim, we associate with every modifier an operation over $\mathcal{TC}$, by examining the way it maps transition configurations to transition configurations.

Definition 17. *Let $\mathfrak{m}$ be a k-ary modifier. We let $\overline{\mathfrak{m}}$ denote the k-ary operation over $\mathcal{TC}$ that satisfies the following property: for any k-tuple of transition configurations*

$$((Q_1, i_1, F_1, \phi_1), \dots, (Q_k, i_k, F_k, \phi_k)),$$

if, for any $j \in \{1, \dots, k\}$, we let δ_j denote the mapping such that $\delta_j(q_j, a) = \phi_j(q_j)$ for any $q_j \in Q_j$, and by $(\{a\}, Q, i, F, \delta)$ the DFA

$$\mathfrak{m}((\{a\}, Q_1, i_1, F_1, \delta_1), \dots, (\{a\}, Q_k, i_k, F_k, \delta_k)),$$

then we have

$$\overline{\mathfrak{m}}((Q_1, i_1, F_1, \phi_1), \dots, (Q_k, i_k, F_k, \phi_k)) = (Q, i, F, \delta^a).$$

It is interesting to take another look at Definition 15 to understand the above definition. Indeed, in the above definition, if we let Σ denote any non-empty alphabet, b denote any letter of Σ, $(A_1, \dots, A_k)$ denote any k-tuple of DFAs such that $A_j = (\Sigma, Q_j, i_j, F_j, \delta_j)$ with $\delta_j^b = \phi_j$, and $(\Sigma, Q', i', F', \delta')$ denote the DFA

$$\mathfrak{m}((\Sigma, Q_1, i_1, F_1, \delta_1), \dots, (\Sigma, Q_k, i_k, F_k, \delta_k)),$$

then we have

$$\overline{\mathfrak{m}}((Q_1, i_1, F_1, \phi_1), \dots, (Q_k, i_k, F_k, \phi_k)) = (Q', i', F', \delta'^b) = (Q, i, F, \delta^a).$$

In other words, for every transition configurations $t_1, \dots, t_k$, we can get $\overline{\mathfrak{m}}(t_1, \dots, t_k)$ by first applying $\mathfrak{m}$ to any k-tuple of DFAs $(A_1, \dots, A_k)$ such that the transition configuration of a certain letter b in A_j is equal t_j, and then by looking at the transition configuration of b in the resulting DFA.

Example 11. *Using the notations of Figure 4.11, if $\mathfrak{m}$ is a unary modifier such that $\mathfrak{m}(A) = B$, then $\overline{\mathfrak{m}}(t) = s$.*

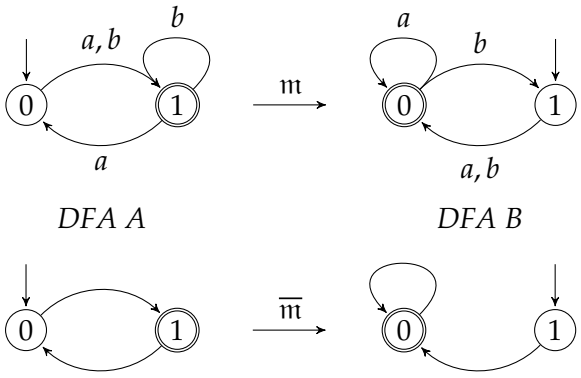


Figure 4.11: An illustration of Definition 17.

We now formalize the idea that $\mathfrak{m}$ and $\overline{\mathfrak{m}}$ contain the same information. To this aim, we prove that the application $\mathfrak{m} \mapsto \overline{\mathfrak{m}}$ is injective, and that it behaves well with respect to the composition of operations. Recall that $\mathfrak{Map}_{\mathcal{TC}}$ is the set of all operations over $\mathcal{TC}$.

Lemma 2. *The application $\mathfrak{m} \mapsto \overline{\mathfrak{m}}$ is an injective morphism from the operad $(\mathfrak{M}, \circ)$ to $(\mathfrak{Map}_{\mathcal{TC}}, \circ)$.*

Proof. We first prove that the application $\mathfrak{m} \mapsto \overline{\mathfrak{m}}$ is injective. Let $\mathfrak{m}$ and $\mathfrak{m}'$ be two different modifiers. If $\mathfrak{m}$ and $\mathfrak{m}'$ do not have the same arity, then $\overline{\mathfrak{m}}$ and $\overline{\mathfrak{m}'}$ do not have the same arity either, and are, therefore, different. If $\mathfrak{m}$ and $\mathfrak{m}'$ have the same arity k, there exists a k-tuple of DFAs $(A_1, \dots, A_k)$, where all A_j have the same alphabet Σ, such that $\mathfrak{m}(A_1, \dots, A_k) \neq \mathfrak{m}'(A_1, \dots, A_k)$. As $\mathfrak{m}(A_1, \dots, A_k)$ and $\mathfrak{m}'(A_1, \dots, A_k)$ have the same alphabet Σ, there exists a letter $a \in \Sigma$ such that the transition configuration of a in $\mathfrak{m}(A_1, \dots, A_k)$ is different from the transition configuration of a in $\mathfrak{m}'(A_1, \dots, A_k)$. Therefore, if, for any $j \in \{1, \dots, k\}$, t_j^a is the transition configuration of a in A_j, we have $\overline{\mathfrak{m}}(t_1^a, \dots, t_k^a) \neq \overline{\mathfrak{m}'}(t_1^a, \dots, t_k^a)$. As a consequence, the application $\mathfrak{m} \mapsto \overline{\mathfrak{m}}$ is injective.

We now prove that $\mathfrak{m} \mapsto \overline{\mathfrak{m}}$ is a morphism of operads from $(\mathfrak{M}, \circ)$ to $(\mathfrak{Map}_{\mathcal{TC}}, \circ)$. Let $\mathfrak{m}$ and $\mathfrak{m}'$ be two modifiers, respectively k-ary and k'-ary. We show that $\overline{\mathfrak{m} \circ_j \mathfrak{m}'} = \overline{\mathfrak{m}} \circ_j \overline{\mathfrak{m}'}$.

First notice that, by Definition 17, the domains of $\overline{\mathfrak{m}}$ and $\overline{\mathfrak{m}'}$ are respectively $\mathcal{TC}^k$ (*i.e.*, $\underbrace{\mathcal{TC} \times \cdots \times \mathcal{TC}}_{\text{k } \textit{times}}$) and $\mathcal{TC}^{k'}$. Therefore, by Definition 6, the domain of $\overline{\mathfrak{m}} \circ \overline{\mathfrak{m}'}$ is $\mathcal{TC}^{k+k'-1}$. Furthermore, by Proposition 16, $\mathfrak{m} \circ_j \mathfrak{m}'$ is a $k+k'-1$ modifier, by Definition 17, the domain of $\overline{\mathfrak{m} \circ_j \mathfrak{m}'}$ is also $\mathcal{TC}^{k+k'-1}$. Therefore, the domains of $\overline{\mathfrak{m}} \circ_j \overline{\mathfrak{m}'}$ and $\overline{\mathfrak{m} \circ_j \mathfrak{m}'}$ are the same.

Let $(t_1, \dots, t_{k+k'-1})$ be any $(k + k' - 1)$-tuple of transition configurations, with $t_j = (Q_j, i_j, F_j, \phi_j)$ for any $j \in \{1, \dots, k + k' - 1\}$, and let $(A_1, \dots, A_{k+k'-1})$ be a $k + k' - 1$-tuple of DFAs such that, for any $j \in \{1, \dots, k + k' - 1\}$, $A_j = (\{a\}, Q_j, i_j, F_j, \delta_j)$, with $\delta_j^a = \phi_j$. By Definition 17, the transition configuration of a in $\mathfrak{m}'(A_j, \dots, A_{k'+j-1})$ is $\overline{\mathfrak{m}'}(t_j, \dots, t_{k'+j-1})$. Thus, similarly, by Definition 17, the transition configuration of a in

$$\mathfrak{m}(A_1, \dots, A_{j-1}, \mathfrak{m}'(A_j, \dots, A_{k'+j-1}), A_{k'+j}, \dots, A_{k+k'-1})$$

is $\overline{\mathfrak{m}}(t_1, \ldots, t_{j-1}, \overline{\mathfrak{m}'}(t_j, \ldots, t_{k'+j-1}), t_{k'+j}, \ldots, t_{k+k'-1})$. Hence, we have $\overline{\mathfrak{m}} \circ_j \overline{\mathfrak{m}'}(A_1, \ldots, A_{k+k'-1}) = \overline{\mathfrak{m} \circ_j \mathfrak{m}'}(A_1, \ldots, A_{k+k'-1})$. In addition, $\mathfrak{m} \mapsto \overline{\mathfrak{m}}$ maps the identity of $\mathfrak{M}$ to the identity of $\mathfrak{Map}_{\mathcal{TC}}$, and it is a graded mapping. Hence, by Definition 10, $\mathfrak{m} \mapsto \overline{\mathfrak{m}}$ is a morphism of operads. □

However, $\mathfrak{m} \mapsto \overline{\mathfrak{m}}$ is not an isomorphism of operads from $(\mathfrak{M}, \circ)$ to $(\mathfrak{Map}_{\mathcal{TC}}, \circ)$. In order to make it an isomorphism, we exhibit its image and its inverse on this image. We let $\mathfrak{FT}$ denote the set of all operations over $\mathcal{TC}$ whose first three coordinates depend only on the first three coordinate of their input. In other words,

Definition 18. *We let $\mathfrak{FT}$ denote the set of all operations $\mathfrak{c}$ over $\mathcal{TC}$ that satisfy the following property: if $(t_1, \ldots, t_k)$ and $(t'_1, \ldots, t'_k)$ are two k-tuples of transitions configurations (where k is the arity of $\mathfrak{c}$) such that, for any $j \in \{1, \ldots, k\}$, the state configurations of t_j and t'_j are equal, then the state configuration of $\mathfrak{c}(t_1, \ldots, t_k)$ is equal to the state configuration of $\mathfrak{c}(t'_1, \ldots, t'_k)$.*

Example 12. *As we can see in Figure 4.12, any unary operation $\mathfrak{o}$ over $\mathcal{TC}$, such that $\mathfrak{o}(t) = s$ and $\mathfrak{o}(t') = s'$, is not in $\mathfrak{FT}$, since the state configurations of t and t' are equal, but the state configurations of s and s' are not.*

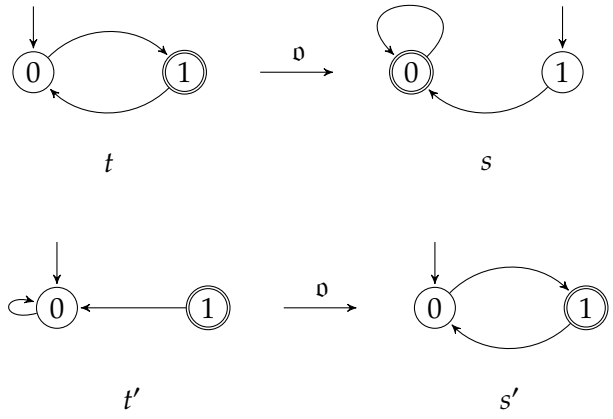


Figure 4.12: An operation in $\mathfrak{Map}_{\mathcal{TC}}$ but not in $\mathfrak{FT}$.

An operation over $\mathcal{TC}$ associated with a modifier is always in $\mathfrak{FT}$.

Lemma 3. *For any modifier $\mathfrak{m}$, we have $\overline{\mathfrak{m}} \in \mathfrak{FT}$.*

Proof. Let $\mathfrak{m}$ be a k-ary modifier, let $(t_1, \ldots, t_k)$ and $(s_1, \ldots, s_k)$ be two k-tuples of transitions configurations such that the state configurations of t_j and s_j are the same for any $j \in \{1, \ldots, k\}$. Furthermore, let $(A_1, \ldots, A_k)$ and $(B_1, \ldots, B_k)$ be two k-tuples of DFAs such that, for any $j \in \{1, \ldots, k\}$, the alphabet of A_j and B_j is $\{a\}$, and the transition configurations of a in A_j and B_j are respectively t_j and s_j. Notice that, as the state configurations of t_j and s_j are the same for any $j \in \{1, \ldots, k\}$, the state configurations of A_j and B_j are also the same. Therefore, by Definition 15, the state configurations of $\mathfrak{m}(A_1, \ldots, A_k)$ and $\mathfrak{m}(B_1, \ldots, B_k)$ are equal. As a consequence, by Definition 17, the state configuration of $\overline{\mathfrak{m}}(t_1, \ldots, t_k)$ is equal to the state configuration of $\overline{\mathfrak{m}}(s_1, \ldots, s_k)$. In conclusion, by Definition 18, $\overline{\mathfrak{m}}$ is in $\mathfrak{FT}$. □

Conversely, we can naturally associate an operation over DFAs with an operation in $\mathfrak{FT}$. We show that such an operation over DFAs is always a modifier.

Definition 19. *Let k be an integer, and let $\mathfrak{tc}$ be an element of $\mathfrak{FT}$. For any k-tuple of DFAs $(A_1, \dots, A_k)$ with $A_j = (\Sigma, Q_j, i_j, F_j, \delta_j)$. We let $[\mathfrak{tc}](A_1, \dots, A_k)$ denote the DFA A such that*

- *the alphabet of A is Σ,*
- *the state configuration of A is the state configuration of*

$$\mathfrak{tc}((Q_1, i_1, F_1, \mathrm{Id}_{Q_1}), \dots, (Q_k, i_k, F_k, \mathrm{Id}_{Q_k})),$$

- *for any letter a of Σ, the transition configuration of a in A is equal to*

$$\mathfrak{tc}((Q_1, i_1, F_1, \delta_1^a), \dots, (Q_k, i_k, F_k, \delta_k^a)).$$

Notice that the third condition of the definition above does not imply the second condition in the case where Σ is the empty alphabet.

Example 13. *Using the notations of Figure 4.13, if $\mathfrak{tc}$ is a unary operation in $\mathfrak{FT}$ such that $\mathfrak{tc}(t_1) = s_1$ and $\mathfrak{tc}(t_2) = s_2$, then $[\mathfrak{tc}](A) = B$.*

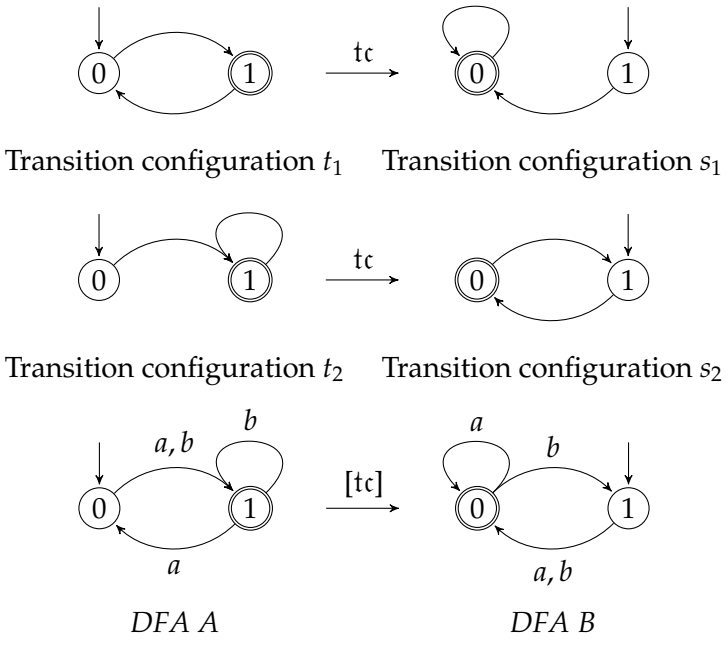


Figure 4.13: An illustration of Definition 19.

Lemma 4. *For any operation $\mathfrak{tc}$ of $\mathfrak{FT}$, we have $[\mathfrak{tc}] \in \mathfrak{M}$, and $\overline{[\mathfrak{tc}]} = \mathfrak{tc}$.*

Proof. Let $\mathfrak{tc}$ be a k-ary operation of $\mathfrak{FT}$. We first show that $[\mathfrak{tc}] \in \mathfrak{M}$ by checking that it satisfies the last two points of Definition 22, since the first one stems directly from Definition 19. Let $(A_1, \dots, A_k)$ and $(B_1, \dots, B_k)$ be two elements of $\mathcal{A}_k$, such that $A_j = (\Sigma, Q_j, i_j, F_j, \delta_j)$ and $B_j = (\Gamma, Q_j, i_j, F_j, \zeta_j)$ for any $j \in \{1, \dots, k\}$.

- By Definition 19, the state configurations of $[\mathfrak{tc}](A_1, \dots, A_k)$ and of $[\mathfrak{tc}](B_1, \dots, B_k)$ are equal to the state configuration of $\mathfrak{tc}((Q_1, i_1, F_1, \mathrm{Id}_{Q_1}), \dots, (Q_k, i_k, F_k, \mathrm{Id}_{Q_k}))$.
- Suppose that a and b are two letters of Σ and Γ respectively, such that, for any $j \in \{1, \dots, k\}$, $\delta_j^a = \zeta_j^b = \phi_j$. By Definition 19, the transition configurations of a and b in $\mathfrak{m}(A_1, \dots, A_k)$ are equal to $\mathfrak{tc}((Q_1, i_1, F_1, \phi_1), \dots, (Q_k, i_k, F_k, \phi_k))$.

Therefore, we have $[\mathfrak{tc}] \in \mathfrak{M}$. We now show that $\overline{[\mathfrak{tc}]} = \mathfrak{tc}$. Let $(t_1, \dots, t_k)$ be any k-tuple of transition configurations, and let $(A_1, \dots, A_k) \in \mathcal{A}_k^{\{a\}}$ such that the transition configuration of a in A_j is t_j. By Definition 19, the transition configuration of a in $[\mathfrak{tc}](A_1, \dots, A_k)$ is equal to $\mathfrak{tc}(A_1, \dots, A_k)$. However, by Definition 17, $\overline{[\mathfrak{tc}]}$ is the transition configuration of a in $[\mathfrak{tc}](A_1, \dots, A_k)$. Thus, we have $\overline{[\mathfrak{tc}]}(t_1, \dots, t_k) = \mathfrak{tc}(t_1, \dots, t_k)$. □

As a consequence of Lemma 3 and Lemma 4, the image of $\mathfrak{m} \mapsto \overline{\mathfrak{m}}$ is $\mathfrak{FT}$, and therefore, by Lemma 2, $\mathfrak{m} \mapsto \overline{\mathfrak{m}}$ is an isomorphism of operads between $\mathfrak{M}$ and $\mathfrak{FT}$. As a consequence, by Proposition 8, its inverse $\mathfrak{tc} \mapsto [\mathfrak{tc}]$ is also an isomorphism. To summarize,

Proposition 17. *The application* $\mathfrak{tc} \mapsto [\mathfrak{tc}]$ *is a isomorphism of operads from* $(\mathfrak{FT}, \circ)$ *to* $(\mathfrak{M}, \circ)$*, and its inverse is* $\mathfrak{m} \mapsto \overline{\mathfrak{m}}$*.*

We can now define any modifier by defining the application of $\mathfrak{FT}$ it is associated with. In addition to providing alternative definitions of $\mathfrak{Comp}$, $\mathfrak{Sqrt}_k$ and $\mathfrak{m}_{\mathfrak{b}}$ (where $\mathbf{b}$ is a boolean function) with this new formalism, we provide a definition of modifiers $\mathfrak{Conc}$ and $\mathfrak{Star}$ that respectively follow the classical constructions of the catenation and the Kleene Star [36].

For simplicity, in the following examples, an application $\mathfrak{tc}$ of $\mathfrak{FT}$ is denoted directly by the 4-tuple $(\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d})$ such that

$$\mathfrak{tc}(t_1, \dots, t_k) = (\mathfrak{Q}(t_1, \dots, t_k), \mathfrak{i}(t_1, \dots, t_k), \mathfrak{f}(t_1, \dots, t_k), \mathfrak{d}(t_1, \dots, t_k)).$$

Furthermore, we do not write the dependency of $\mathfrak{Q}$, $\mathfrak{i}$, and $\mathfrak{f}$ over the fourth coordinates of their input k-tuples, *i.e.,* we do not write $\mathfrak{Q}((Q_1, i_1, F_1, \delta_1), \dots, (Q_k, i_k, F_k, \delta_k))$, but we write $\mathfrak{Q}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k))$. We do not write either the dependency of $\mathfrak{d}$ on $Q_1, \dots, Q_k$, as this information is already contained in $(\delta_1, \dots, \delta_k)$, *i.e.,* we do not write $\mathfrak{d}((Q_1, i_1, F_1, \delta_1), \dots, (Q_k, i_k, F_k, \delta_k))$, but we write $\mathfrak{d}((i_1, F_1, \delta_1), \dots, (i_k, F_k, \delta_k))$.

Example 14. *The modifier* $\mathfrak{Comp}$ *of Example 8 corresponding to the (set-theoretic) complement is equal to* $[\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$*, where, for any transition configuration* (Q, i, F, δ) *,*

$$\mathfrak{Q}(Q, i, F) = Q, \mathfrak{i}(Q, i, F) = i, \mathfrak{f}(Q, i, F) = Q \setminus F, \mathfrak{d}(i, F, \delta) = \delta.$$

Example 15. *We define the modifier* $\mathfrak{Star}$ *by* $\mathfrak{Star} = [\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$*, where, for any state configuration* (Q, i, F) *and any* $\delta \in Q^Q$*,* $\mathfrak{Q}(Q, i, F) = 2^Q$*,* $\mathfrak{i}(Q, i, F) = \emptyset$*,* $\mathfrak{f}(Q, i, F) = \{E \subseteq Q \mid E \cap F \neq \emptyset\} \cup \{\emptyset\}$*, and, for all* $E \subseteq Q$*,*

$$\mathfrak{d}(i, F, \delta)(E) = \begin{cases} \{\delta(i)\} & \text{if } E = \emptyset \text{ and } \delta(i) \notin F \\ \{\delta(i), i\} & \text{if } E = \emptyset \text{ and } \delta(i) \in F \\ \delta(E) & \text{if } E \neq \emptyset \text{ and } \delta(E) \cap F = \emptyset \\ \delta(E) \cup \{i\} & \text{if } E \neq \emptyset \text{ and } \delta(E) \cap F \neq \emptyset \end{cases}$$

The modifier $\mathfrak{Star}$ *follows the classical construction for DFAs of the Kleene star operation [36], i.e., for any DFA* A, $\mathrm{L}(\mathfrak{Star}(A)) = (\mathrm{L}(A))^\star$. *Figure 4.15 represents the image of the DFA* A *of Figure 4.14 by the modifier* $\mathfrak{Star}$.

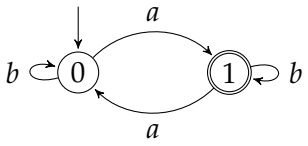


Figure 4.14: A DFA A.

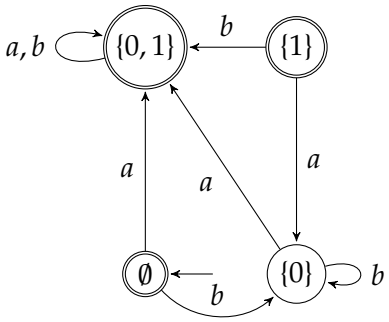


Figure 4.15: The DFA $\mathfrak{Star}(A)$.

Example 16. *The modifier* $\mathfrak{Sqrt}_k$ *defined in Example 9 is equal to* $[\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}_k, \mathfrak{d}]$, *where, for any state configuration* (Q, i, F) *and any* $\phi \in Q^Q$, $\mathfrak{Q}(Q, i, F) = Q^Q$, $\mathfrak{i}(Q, i, F) = Id_Q$, $\mathfrak{f}_k(Q, i, F) = \{\phi \in Q^Q | \phi^k(i) \in F\}$, *and, for any* $\psi \in Q^Q$, $\mathfrak{d}(i, F, \phi)(\psi) = \phi \circ \psi$.

Example 17. *For any k-ary boolean function* $\mathbf{b}$, $\mathfrak{m}_\mathbf{b}$ *is equal to* $[\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$, *where, for any k-tuple* $((Q_1, i_1, F_1, \delta_1), \ldots, (Q_k, i_k, F_k, \delta_k))$ *of elements of* $\mathcal{TC}$,

- $\mathfrak{Q}((Q_1, i_1, F_1), \ldots, (Q_k, i_k, F_k)) = Q_1 \times \cdots \times Q_k$,
- $\mathfrak{i}((Q_1, i_1, F_1), \ldots, (Q_k, i_k, F_k)) = (i_1, \ldots, i_k)$,
- $$\mathfrak{f}((Q_1, i_1, F_1), \ldots, (Q_k, i_k, F_k)) = \{(q_1, \ldots, q_k) \in Q_1 \times \cdots \times Q_k \mid \mathbf{b}([q_1 \in F_1], \ldots, [q_k \in F_k]) = 1\},$$
- *and, for any* $(q_1, \ldots, q_k) \in Q_1 \times \cdots \times Q_k$,

$$\mathfrak{d}((i_1, F_1, \delta_1), \ldots, (i_k, F_k, \delta_k))(q_1, \ldots, q_k) = (\delta_1(q_1), \ldots, \delta_k(q_k)).$$

Example 18. *We define the modifier* $\mathfrak{Conc} = [\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$, *where, for any two elements of* $\mathcal{TC}$ $(Q_1, i_1, F_1, \delta_1)$ *and* $(Q_2, i_2, F_2, \delta_2)$,

- $\mathfrak{Q}((Q_1, i_1, F_1), (Q_2, i_2, F_2)) = Q_1 \times 2^{Q_2}$
- $\mathfrak{i}((Q_1, i_1, F_1), (Q_2, i_2, F_2)) = \begin{cases} (i_1, \emptyset) & \text{if } i_1 \notin F_1 \\ (i_1, \{i_2\}) & \text{if } i_1 \in F_1 \end{cases}$
- $\mathfrak{f}((Q_1, i_1, F_1), (Q_2, i_2, F_2)) = \{(q_1, E) \in Q_1 \times 2^{Q_2} \mid E \cap F_2 \neq \emptyset\}$
- *and, for any* $(q_1, E) \in Q_1 \times 2^{Q_2}$,

$$\mathfrak{d}((i_1, F_1, \delta_1), (i_2, F_2, \delta_2))(q_1, E) = \begin{cases} (\delta_1(q_1), \delta_2(E)) & \delta_1(q_1) \notin F_1 \\ (\delta_1(q_1), \delta_2(E) \cup \{i_2\}) & \text{otherwise.} \end{cases}$$

This modifier follows the classical construction on DFAs corresponding to the catenation operation on languages [36]. In other words, for any two DFAs A_1 and A_2, $\mathrm{L}(\mathfrak{Conc}(A_1, A_2)) = \mathrm{L}(A_1) \cdot \mathrm{L}(A_2)$. Figure 4.18 shows the DFA $\mathfrak{Conc}(A_1, A_2)$, where A_1 and A_2 are the DFAs of Figures 4.16 and 4.17.

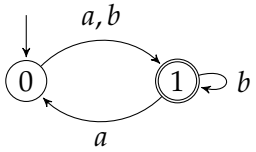


Figure 4.16: DFA A_1.

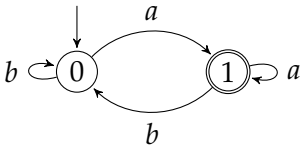


Figure 4.17: DFA A_2.

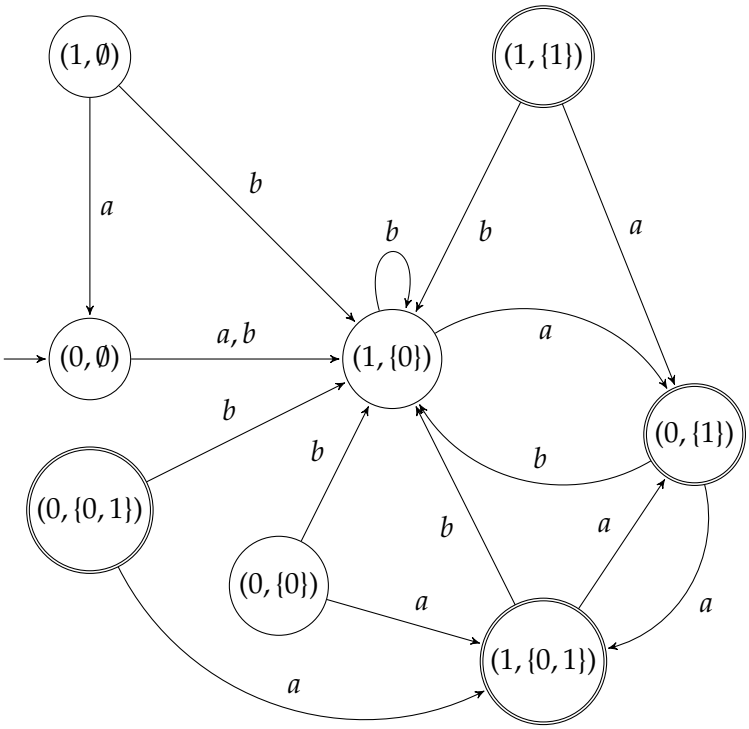


Figure 4.18: The DFA $\mathfrak{Conc}(A_1, A_2)$.

4.2.4 From modifiers to regular operations

We now explain how we associate a regular operation with a modifier. This association follows a very natural intuition. We say that a modifier is coherent if it can naturally be associated with a regular operation.

Definition 20. *A k-modifier $\mathfrak{m}$ is* coherent *if, for every pair of k-tuples of DFAs $(A_1, \ldots, A_k)$ and $(B_1, \ldots, B_k)$ such that $\mathrm{L}(A_j) = \mathrm{L}(B_j)$ for all $j \in \{1, \ldots, k\}$, we have $\mathrm{L}(\mathfrak{m}(A_1, \ldots, A_k)) = \mathrm{L}(\mathfrak{m}(B_1, \ldots, B_k))$.*

We let $\mathfrak{M}_c$ denote the set of coherent modifiers.

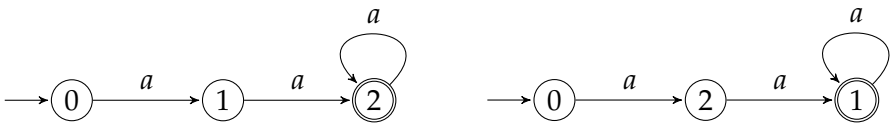
Definition 21. *For any coherent modifier* $\mathfrak{m}$*, the operation* $\otimes$ *such that, for all k-tuples of DFAs* $(A_1, \ldots, A_k)$*,* $\otimes(\mathrm{L}(A_1), \ldots, \mathrm{L}(A_k)) = \mathrm{L}(\mathfrak{m}(A_1, \ldots, A_k))$*, is well-defined. We say that* $\mathfrak{m}$ describes *the operation* $\otimes$*, and we let* **desc** *denote the mapping from* $\mathfrak{M}_c$ *into the set of operations over languages such that* $\mathbf{desc}(\mathfrak{m}) = \otimes$*.*

Every modifier given in an example above is coherent. Furthermore, by construction, $\mathbf{desc}(\mathfrak{m}_\mathfrak{b})$ is equal to $\otimes_\mathfrak{b}$ for any boolean function **b**, $\mathbf{desc}(\mathfrak{Conc})$ is the catenation of two languages, $\mathbf{desc}(\mathfrak{Star})$ is the Kleene star, and $\mathbf{desc}(\mathfrak{Sqrt})$ is the square root.

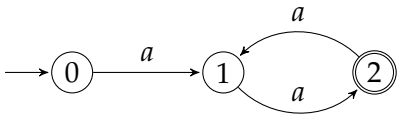
Remark 2. *Some modifiers are not coherent. For instance, consider the* 1*-modifier* $\mathfrak{Fto1} = [\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$ *such that, for any transition configuration* (Q, i, F, δ)*, we have*

$$\mathfrak{Q}(Q) = Q, \mathfrak{i}(Q, i, F) = i, \mathfrak{f}(Q, i, F) = F, \text{ and } \mathfrak{d}(i, F, \delta)(q) = \begin{cases} \delta(q) & \text{if } q \notin F \\ 1 & \text{if } 1 \in Q \text{ and } q \in F \\ \delta(q) & \text{otherwise.} \end{cases}$$

In other words, this modifier changes the transition functions of its input DFA such that the image of a final state by a transition function of the output DFA is 1 *if possible. If* A_1 *and* A'_1 *are two deterministic automata recognizing the same language then we have in general* $L(\mathfrak{Fto1}(A_1)) \neq L(\mathfrak{Fto1}(A'_1))$ *because the recognized language depends on the labels of the states of* A_1 *and* A'_1*. For instance, the two following automata recognize the same language* a^2a^**.*



But applying $\mathfrak{Fto1}$ *on the first one gives*



which recognizes $(aa)^+$ *while* $\mathfrak{Fto1}$ *lets the second automaton unchanged.*

Proposition 18. *The set of all coherent modifiers equipped with the composition of operations* $\circ$ *is an operad. Furthermore, the mapping* **desc** *is a morphism of operads.*

Proof. Let $\mathfrak{m}_1$ and $\mathfrak{m}_2$ be two coherent modifiers, respectively k_1-ary and k_2-ary, and let $\otimes = \mathbf{desc}(\mathfrak{m}_1)$ and $\oplus = \mathbf{desc}(\mathfrak{m}_2)$. Let $L_1, \ldots, L_{k_1+k_2-1}$ be regular languages recognized respectively by DFAs $A_1, \ldots, A_{k_1+k_2-1}$. We have

$$\begin{aligned} &\mathfrak{m}_1 \circ_j \mathfrak{m}_2(A_1, \ldots, A_{k_1+k_2-1}) \\ = \mathfrak{m}_1(A_1, \ldots, A_{j-1}, &\mathfrak{m}_2(A_j, \ldots, A_{j+k_2-1}), A_{j+k_2}, \ldots, A_{k_1+k_2-1}). \end{aligned}$$

As a consequence, if $\mathfrak{m}_1$ and $\mathfrak{m}_2$ are coherent, then $\mathfrak{m}_1 \circ_j \mathfrak{m}_2$ is coherent as well. Furthermore:

$$\begin{aligned} &\otimes \circ_j \oplus(L_1, \ldots, L_{k_1+k_2-1}) \\ = \ &\otimes(L_1, \ldots, L_{j-1}, \oplus(L_j, \ldots, L_{j+k_2-1}), L_{j+k_2}, \ldots, L_{k_1+k_2-1}) \\ = \ &\mathrm{L}(\mathfrak{m}_1(A_1, \ldots A_{j-1}, \mathfrak{m}_2(A_j, \ldots, A_{j+k_2-1}), A_{j+k_2}, \ldots, A_{k_1+k_2-1})) \\ = \ &\mathrm{L}(\mathfrak{m}_1 \circ_j \mathfrak{m}_2(A_1, \ldots, A_{k_1+k_2-1})). \end{aligned}$$

Thus, $\mathbf{desc}(\mathfrak{m}_1 \circ_j \mathfrak{m}_2) = \mathbf{desc}(\mathfrak{m}_1) \circ_j \mathbf{desc}(\mathfrak{m}_2)$. Therefore, from Definition 20, $\mathfrak{m}_1 \circ_j \mathfrak{m}_2$ is coherent. Furthermore, the identity over $\mathfrak{M}$, denoted by Id, is coherent, and that $\mathbf{desc}(\mathrm{Id})$ is the identity over the set of regular languages $\mathcal{L}$. Thus, by Proposition 5 and Proposition 16, $(\mathfrak{M}_c, \circ)$ is an operad. In addition, by Definition 10, **desc** is a morphism of operads. □

4.3 The link with operational state complexity

In this section, we first define some specific DFAs with large alphabets called *monsters*. Then, we use this tool to show that the regular operations described by coherent modifiers are exactly all 1-uniform operations. Finally, we show that a 1-uniform operation always has a witness that is also a monster.

4.3.1 Monsters

We first define the DFAs we use to find witnesses for 1-uniform operations, called *monsters*. As we also want to deal with k-ary operations, and not only unary ones, monsters will not necessarily be DFAs, but, in all generality, k-tuples of DFAs. The idea is to define k-tuples of DFAs over the same alphabet, so that this alphabet is as large as possible, while not having two letters with the same transition functions. In other words, every possible k-tuple of transition functions of a monster should correspond to a single letter of its alphabet. This will give us as much leeway as possible to prove reachability and distinguishability results when minimizing the output DFA of 1-uniform operations. To use simple and intuitive notations, the alphabet of a k-monster are k-tuples of functions. Furthermore, the k-tuple constituted of the transition functions of a letter in the DFAs of the k-monster is the letter itself. More formally, we have the next definition.

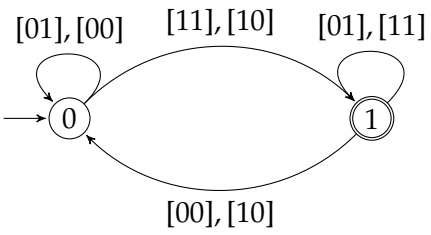
Definition 22. *Let $(n_1, \dots, n_k)$ be a k-tuple of positive integers, and let $(F_1, \dots, F_k)$ be a k-tuple of sets such that $F_j \subseteq [\![n_j]\!]$, for any $j \in \{1, \dots, k\}$. We let $\mathrm{Mon}^{F_1,\dots,F_k}_{n_1,\dots,n_k}$ denote the k-tuple of DFAs $(\mathbb{M}_1, \dots, \mathbb{M}_k)$ such that, for any $j \in \{1, \dots, k\}$, we have $\mathbb{M}_j = (\Gamma_{n_1,\dots,n_k}, [\![n_j]\!], 0, F_j, \delta_j)$, where*

- $\Gamma_{n_1,\dots,n_k} = [\![n_1]\!]^{[\![n_1]\!]} \times \cdots \times [\![n_k]\!]^{[\![n_k]\!]}$,
- *for any $(a_1, \dots, a_k) \in \Sigma$ and for any $q_j \in [\![n_j]\!]$, we have $\delta_j(q_j, (a_1, \dots, a_k)) = a_j(q_j)$.*

We say that a k-tuple of DFAs is a k-monster if and only if it is equal to $\mathrm{Mon}^{F_1,\dots,F_k}_{n_1,\dots,n_k}$, for some k-tuple of positive integers $(n_1, \dots, n_k)$, and some k-tuple of sets $(F_1, \dots, F_k)$ with $F_j \subseteq [\![n_j]\!]$ for any $j \in \{1, \dots, k\}$.

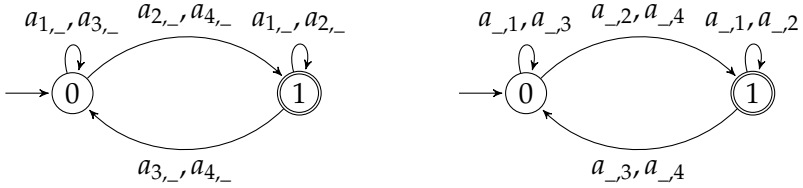
Remark 3. *When F_j is different from $\emptyset$ and Q_j, $\mathbb{M}_j$ is minimal.*

Example 19. *The 1-monster $\mathrm{Mon}^{\{1\}}_2$ is*



where, for all $i, j \in \{0,1\}$, the label $[ij]$ denotes the mapping that maps 0 to i and 1 to j, which is also a letter in the DFA above.

Example 20. *The 2-monster $\mathrm{Mon}_{(2,2)}^{(\{1\},\{1\})}$ is given by the following pair of automata on an alphabet with $2^2 \times 2^2 = 16$ symbols where $a_{i,\_}$ (respectively $a_{\_,j}$) denotes the set of transitions $a_{i,x}$ (respectively $a_{x,j}$) for $x \in \{1,\dots,4\}$:*



Each symbol codes a pair of functions, denoted by the word of their image.

$$\begin{array}{llll} a_{1,1} = [01,01] & a_{1,2} = [01,11] & a_{1,3} = [01,00] & a_{1,4} = [01,10] \\ a_{2,1} = [11,01] & a_{2,2} = [11,11] & a_{2,3} = [11,00] & a_{2,4} = [11,10] \\ a_{3,1} = [00,01] & a_{3,2} = [00,11] & a_{3,3} = [00,00] & a_{3,4} = [00,10] \\ a_{4,1} = [10,01] & a_{4,2} = [10,11] & a_{4,3} = [10,00] & a_{4,4} = [10,10]. \end{array}$$

For instance, $a_{1,2} = [01,11]$ means that the symbol $a_{1,2}$ labels a transition from 0 to 0 and a transition from 1 to 1 in the first automaton and a transition from 0 to 1 and a transition from 1 to 1 in the second automaton.

Notice that monsters only differ from one another only by the size and final states of their DFAs. Therefore, when using them as witnesses, we will only need to discuss their final states.

Monsters convey the key idea of our general approach in the sense that every k-tuple of languages is in some way contained in a k-monster. This universality-like property is formalized in the following lemma:

Lemma 5. *Let $(L_1,\dots,L_k)$ be any k-tuple of regular languages over the same alphabet, and let $(A_1,\dots,A_k)$ be any k-tuple of DFAs over the same alphabet, such that A_j satisfies the following properties for any $j \in \{1,\dots,k\}$:*

- *A_j recognizes the language L_j,*
- *the states of A_j is $[\![n_j]\!]$ for some integer n_j,*
- *the initial state of A_j is 0.*

We let $(\Sigma, [\![n_j]\!], 0, F_j, \delta_j)$ denote A_j, and we let $(\mathbb{M}_1,\dots,\mathbb{M}_k)$ denote $\mathrm{Mon}_{n_1,\dots,n_k}^{F_1,\dots,F_k}$. Furthermore, we let ϕ denote the 1-uniform morphism from Σ to $\Gamma_{n_1,\dots,n_k}$ such that, for all $a \in \Sigma$, we have $\phi(a) = (\delta_1^a, \delta_2^a, \dots, \delta_k^a)$. For any $j \in \{1,\dots,k\}$, the language L_j is the preimage of $\mathbb{M}_j$ by the 1-uniform morphism ϕ, i.e., *we have*

$$(L_1,\dots,L_k) = (\phi^{-1}(\mathrm{L}(\mathbb{M}_1)),\dots,\phi^{-1}(\mathrm{L}(\mathbb{M}_k))). \tag{4.2}$$

Proof. Let j be an integer of $\{1,\dots,k\}$. By Definition 22, the transition function ξ_j of $\mathbb{M}_j$ satisfies $\xi_j^{(\delta_1^a,\dots,\delta_k^a)} = \delta_j^a$. Therefore, by Proposition 2, a word is in $\phi^{-1}(\mathrm{L}(\mathbb{M}_j))$ if and only if it is recognized by the DFA $B_j = (\Sigma, [\![n_j]\!], 0, F_j, \zeta_j)$, with, for any $l \in [\![n_j]\!]$ and any $a \in \Sigma$, we have

$$\zeta_j^a = \xi_j^{\phi(a)} = \xi_j^{(\delta_1^a,\dots,\delta_k^a)} = \delta_j^a.$$

To conclude, $A_j = B_j$ and $L_j = \phi^{-1}(\mathrm{L}(\mathbb{M}_j))$, for all $j \in \{1,\dots,k\}$. □

4.3.2 Modifiers and 1-uniform operations

The next proposition shows that a coherent modifier always describes a 1-uniform operation, and that every 1-uniform operation is described by a coherent modifier. It is the beginning of our algebraic analysis. This algebraic landscape is furthered in Chapter 7. We let O_u denote the set of 1-uniform operations.

Theorem 1. $\mathbf{desc}(\mathfrak{M}_c) = O_u$.

Proof. Let $\otimes$ be a k-ary 1-uniform operation. We define a k-modifier $\mathfrak{m}$ as follows.

Let $(A_1,\dots,A_k)$ be a k-tuple of DFAs with $A_j = (\Sigma, Q_j^A, i_j^A, F_j^A, \delta_{A,j})$, for any $j \in \{1,\dots,k\}$. We can rename the states of each DFA in this k-tuple so that each A_j becomes $D_j = (\Sigma, [\![n_j]\!], 0, F_j, \delta_j)$. If $\mathrm{Mon}_{n_1,\dots,n_k}^{F_1,\dots,F_k} = (\mathbb{M}_1,\dots,\mathbb{M}_k)$, we let $B = (\Gamma_{n_1,\dots,n_k}, Q', i', F', \delta')$ denote the minimal DFA of $\otimes(\mathrm{L}(\mathbb{M}_1),\dots,\mathrm{L}(\mathbb{M}_k))$. We set $\mathfrak{m}(A_1,\dots,A_k) = (\Sigma, Q', i', F', \tilde{\delta}')$, with $\tilde{\delta}'(q,a) = \delta'(q,(\delta_1^a,\dots,\delta_k^a))$.

Notice that $\mathfrak{m}$ is indeed a coherent modifier. First, (Q', i', F') depends only on every (Q_j^A, i_j^A, F_j^A) for $j \in \{1,\dots,k\}$. Second, $\tilde{\delta}'$ depends only on $(\delta_1^a,\dots,\delta_k^a)$ and on δ', which in turn depend only on (Q_j^A, i_j^A, F_j^A) for $j \in \{1,\dots,k\}$, and on $(\delta_{A,1}^a,\dots,\delta_{A,k}^a)$.

Furthermore, by Proposition 2, $\mathrm{L}(\mathfrak{m}(A_1,\dots,A_k)) = \phi^{-1}(\mathrm{L}(B))$, where ϕ is the 1-uniform morphism such that $\phi(a) = (\delta_1^a,\dots,\delta_k^a)$ for all $a \in \Sigma$. Therefore, we have

$$\mathrm{L}(\mathfrak{m}(A_1,\dots,A_k)) = \phi^{-1}(\otimes(\mathrm{L}(\mathbb{M}_1),\dots,\mathrm{L}(\mathbb{M}_k))).$$

And, since $\otimes$ is 1-uniform, we obtain from Lemma 5 and Definition 13

$$\mathrm{L}(\mathfrak{m}(A_1,\dots,A_k)) = \otimes(\phi^{-1}(\mathrm{L}(\mathbb{M}_1)),\dots,\phi^{-1}(\mathrm{L}(\mathbb{M}_k))) = \otimes(L_1,\dots,L_k).$$

We thus have $\mathbf{desc}(\mathfrak{m}) = \otimes$.

Conversely, let $\mathfrak{m}$ be a coherent k-modifier, and let $\otimes = \mathbf{desc}(\mathfrak{m})$. We must prove that $\otimes$ is 1-uniform. Let Γ and Σ be two alphabets. Consider a 1-uniform morphism ϕ from Γ^* to Σ^* and a k-tuple of languages $(L_1,\dots,L_k)$ over Σ. Let $(A_1,\dots,A_k)$ be a k-tuple of DFAs such that $A_j = (\Sigma, Q_j, i_j, F_j, \delta_j)$ and A_j recognizes L_j for any $j \in \{1,\dots,k\}$, and let $(B_1,\dots,B_k)$ be the k-tuple of DFAs such that, for any $j \in \{1,\dots,k\}$, $B_j = (\Gamma, Q_j, i_j, F_j, \tilde{\delta}_j)$, with $\tilde{\delta}_j^a = \delta_j^{\phi(a)}$ for any letter $a \in \Gamma$. We have $\mathrm{L}(B_j) = \phi^{-1}(\mathrm{L}(A_j))$ for $j \in \{1,\dots,k\}$.

Let $\mathfrak{m}(A_1,\dots,A_k) = (\Sigma, Q, i, F, \delta)$ and $\mathfrak{m}(B_1,\dots,B_k) = (\Gamma, Q', i', F', \delta')$. Since the state configuration of each A_j is the same as the state configuration of each B_j, we have $(Q, i, F) = (Q', i', F')$. Furthermore, because the transition configuration of any letter $a \in \Gamma$ in B is equal

to the transition configuration of $\phi(a)$ in A, we have $\delta'^{a} = \delta^{\phi(a)}$. Hence, $\mathrm{L}(\mathfrak{m}(B_1, \ldots, B_k)) = \phi^{-1}(\mathrm{L}(\mathfrak{m}(A_1, \ldots, A_k)))$, which implies that $\otimes(\mathrm{L}(B_1), \ldots, \mathrm{L}(B_k)) = \phi^{-1}(\otimes(\mathrm{L}(A_1), \ldots, \mathrm{L}(A_k)))$. Therefore, $\otimes(\phi^{-1}(\mathrm{L}(A_1)), \ldots, \phi^{-1}(\mathrm{L}(A_k))) = \phi^{-1}(\otimes(\mathrm{L}(A_1), \ldots, \mathrm{L}(A_k)))$, as expected. □

To end this section, we give in Figure 4.19 a summary diagram of some propositions above. Every set represented in this figure is an operad equipped with the composition of operations $\circ$. Furthermore, every represented mapping is a morphism of operads. Mappings represented with a two-headed arrow (*i.e.*, $\twoheadrightarrow$) are surjections, and mappings represented with a double-ended arrow (*i.e.*, $\leftrightarrow$) are bijections.

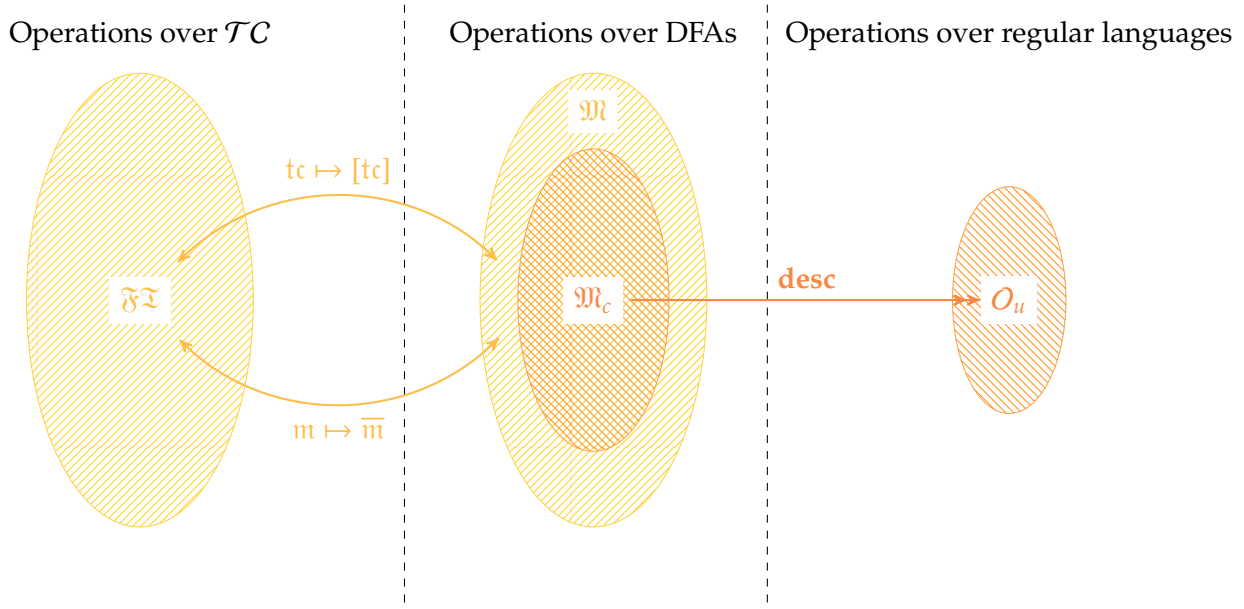


Figure 4.19: The algebraic structure linking operations over transitions configurations, operations over DFAs and regular operations.

4.3.3 Computing the state complexity of 1-uniform operations

The following theorem is the main result of this section. It is used later on to design a method for computing the state complexity of 1-uniform operations.

Theorem 2. *Every k-ary 1-uniform operation admits a family of monster k-languages as witness.*

Proof. Suppose now that $\otimes$ is a k-ary 1-uniform operation. Let $(L_1, \ldots, L_k)$ be any k-tuple of regular languages over Σ, and let $(A_1, \ldots, A_k)$ be the k-tuple of DFAs such that each $A_j = (\Sigma, [\![n_j]\!], 0, F_j, \delta_j)$ is a minimal DFA that recognizes L_j. Let ϕ the 1-uniform morphism such that, for all $a \in \Sigma$, $\phi(a) = (\delta_1^a, \ldots, \delta_k^a)$, and let $\mathrm{Mon}_{n_1,\ldots,n_k}^{F_1,\ldots,F_k} = (\mathbb{M}_1, \ldots, \mathbb{M}_k)$. We have $\otimes(L_1, \ldots, L_k) = \otimes(\phi^{-1}(\mathrm{L}(\mathbb{M}_1)), \ldots, \phi^{-1}(\mathrm{L}(\mathbb{M}_k)))$ by Lemma 5, and so $\otimes(L_1, \ldots, L_k) = \phi^{-1}(\otimes(\mathrm{L}(\mathbb{M}_1), \ldots, \mathrm{L}(\mathbb{M}_k)))$ by Definition 13. It follows that

$$\mathrm{sc}(\otimes(L_1, \ldots, L_k)) = \mathrm{sc}(\phi^{-1}(\otimes(\mathrm{L}(\mathbb{M}_1), \ldots, \mathrm{L}(\mathbb{M}_k)))) \leq \mathrm{sc}(\otimes(\mathrm{L}(\mathbb{M}_1), \ldots, \mathrm{L}(\mathbb{M}_k)))$$

by Proposition 3. In addition, each $\mathrm{L}(\mathbb{M}_j)$ has the same state complexity as L_j. □

We show, in the next section, how the above theorem may be used to compute the state complexity of a 1-uniform operation, by giving some simple examples.

Chapter 5

Examples

In the last section, we showed that 1-uniform operations are described by coherent modifiers (Theorem 1). By combining this result with Theorem 2, we design a method for computing the state complexity of a 1-uniform operation $\otimes$.

1. Construct a modifier $\mathfrak{m}$ that describes $\otimes$, and consider the DFAs obtained by applying it to monsters;

2. Find an upper bound, using Theorem 2;

3. Give a monster that allows us to reach this upper bound, by choosing its final states appropriately.

One should not view this method as a hard and fast rule, but rather as a starting point of research. Steps 2 and 3 can be intertwined for legibility purposes, like in Chapter 6. The approach of many papers computing the state complexity of some 1-uniform operation can be tied to this method [26, 15, 3, 8].

The state complexity of Kleene star and catenation were among the first to be computed [33, 40]. The state complexity of union and intersection have been computed in [1], and the state complexity of symmetric difference has been computed in [2], for example. These three operations give us the state complexity of all binary boolean operations. As for the state complexity of boolean operations of higher arity, a large part of the answer is provided by [16]. In that paper, the authors compute the state complexity of every boolean operation that "depends on each of its operands" [16]. We show that the method we designed can be used to easily find all these results again. Furthermore, we use it in Section 5.2 to compute the state complexity of boolean operations in the general case. In [13], Sylvie Davies gives these examples and proves them in a very similar manner, using the approach she developed (which is also very similar to ours). Nonetheless, we still found it important to give them here using our formalism.

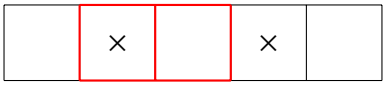
5.1 Star

5.1.1 Applying the star modifier to monsters

Let $n \geq 2$ be an integer, and G be a subset of $[\![n]\!]$, and let $A = (\Sigma, Q, i, F, \delta) = \mathfrak{Star}(\mathrm{Mon}_n^G)$. By the definition of $\mathfrak{Star}$ (Example 15), we have $\Sigma = \Gamma_n = [\![n]\!]^{[\![n]\!]}$, $Q = 2^{[\![n]\!]}$, $i = \emptyset$, $F = \{E \in Q \mid E \cap G \neq \emptyset\} \cup \{\emptyset\}$ and, for any $E \in Q$ and any $\phi \in \Sigma$,

$$\delta^\phi(E) = \begin{cases} \{\phi(0)\} & \text{if } E = \emptyset \text{ and } \phi(0) \notin G \\ \{\phi(0), 0\} & \text{if } E = \emptyset \text{ and } \phi(0) \in G \\ \phi(E) & \text{if } E \neq \emptyset \text{ and } \phi(E) \cap G = \emptyset \\ \phi(E) \cup \{0\} & \text{if } E \neq \emptyset \text{ and } \phi(E) \cap G \neq \emptyset \end{cases}$$

We can represent a subset E of $[\![n]\!]$ as a "line" of squares, that may be empty or filled with a cross. An empty square at position i represents that $i \notin E$, and a square filled with a cross at position i represents that $i \in E$. All squares representing a position that is in G are in red. For example, if $n = 5$, and $G = \{1, 2\}$, the subset $\{1, 3\}$ of $[\![5]\!]$ is represented with the following line:



With this representation, we can alternatively display Figures 4.14 and 4.15 as Figures 5.1 and 5.2. In Figure 5.1, we identify the states 0 and 1 with the subsets $\{0\}$ and $\{1\}$ of $\{0, 1\}$.

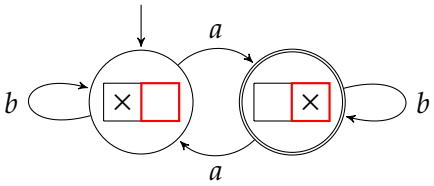


Figure 5.1: A DFA A.

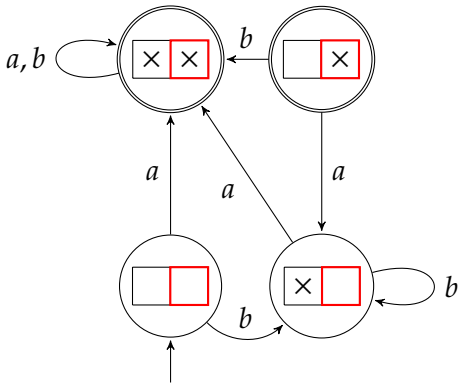


Figure 5.2: The DFA $\mathfrak{Star}(A)$.

This representation gives us an interpretation of the transition function of A. Indeed, in A, to go with the letter ϕ from the representation of a subset E, to the representation of a subset E' (in the case where $E \neq \emptyset$), one can change the positions of the crosses of E by applying the function ϕ to them, and then put a cross at the beginning of the line if and only if there is a cross in a red square. For example, if $n = 4$, $G = \{1, 2\}$, $\phi(1) = 2$, $\phi(2) = 3$, we have $\delta^\phi(\{1, 2\}) = \{0, 2, 3\}$, which is represented by Figure 5.3.

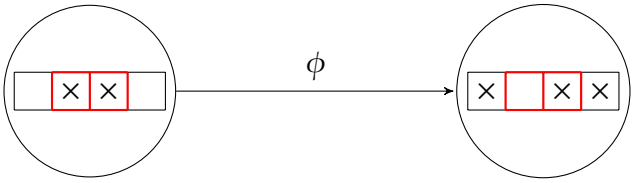


Figure 5.3: A transition in $\mathfrak{Star}(\mathrm{Mon}_4^{\{1,2\}})$.

5.1.2 An upper bound

We first establish an upper bound on the state complexity of the Kleene star. Our reasoning is based on the following remark:

Remark 4. *If reading a letter ϕ from any state of A leads to a state with a cross in a red square, then this state also has cross in the leftmost square. More formally, if E is any element of Q and ϕ any letter of Γ_n, if $G \cap \phi(E) \neq \emptyset$, then $0 \in \delta^\phi(E)$. Therefore, an easy induction shows that any state E of A such that $E \cap G \neq \emptyset$ and $0 \notin E$ is not accessible in A. For example, the state at the left of Figure 5.3 is not accessible in the DFA $\mathfrak{Star}(\mathrm{Mon}_4^{\{2,3\}})$.*

Lemma 6. *For any integer $n \geq 2$, the state complexity $\mathrm{sc}_{\mathbf{Star}}$ of the Kleene star satisfies $\mathrm{sc}_{\mathbf{Star}}(n) \leq 2^{n-1} + 2^{n-2}$.*

Proof. We distinguish several cases. First suppose that $G = \emptyset$. Then A has no final states, and the size of the minimal DFA associated with A is 1. We next show that, in every other case, the number of accessible states of A is less than or equal to $2^{n-1} + 2^{n-2}$.

Second, suppose that $G = \{0\}$. Let E be a state of A that is a singleton $\{j\}$. If $\phi(j) = 0$, $\phi(E) \cap G \neq \emptyset$, and $\delta^\phi(E) = \{0\} \cup \{0\} = \{0\}$. If $\phi(j) = l \neq 0$, then $\phi(E) \cap G = \emptyset$ and $\delta^\phi(E) = \{\phi(j)\}$. In both cases, $\delta^\phi(E)$ is a singleton. This can be seen by relying on our representation. Indeed, in this case, adding a cross to a line with only one cross *via* a transition is impossible. From this, an easy induction shows that every accessible state of A is a singleton. Therefore, the number of accessible states of A is at most $n+1 \leq 2^{n-1}+2^{n-2}$.

Finally, suppose that $G \notin \{\emptyset, \{0\}\}$. From Remark 4, we deduce an upper bound on the number of accessible states of A. We distinguish two cases:

- Suppose $0 \notin G$. The number of states E of A such that $E \cap G \neq \emptyset$ and $0 \notin E$ is $2^{n-1} - 2^{n-1-\#G}$. Therefore, since $\#G \geq 1$, the number of accessible states of A is at most $2^n - 2^{n-1} - 2^{n-1-\#G} = 2^{n-1} + 2^{n-1-\#G} \leq 2^{n-1} + 2^{n-2}$.

- Suppose $0 \in G$. The number of states E of A such that $E \cap G \neq \emptyset$ and $0 \notin E$ is $2^{n-1} - 2^{n-\#G}$. Therefore, the number of accessible states of A is at most $2^n - 2^{n-1} - 2^{n-\#G} = 2^{n-1} + 2^{n-\#G}$. However, as $G \notin \{\emptyset, \{0\}\}$, we have $\#G \geq 2$. As a consequence, the number of accessible states of A is at most $2^{n-1} + 2^{n-2}$.

Therefore, we have proven, in every case, that the size of the minimal DFA associated with A is less or equal than $2^{n-1} + 2^{n-2}$. Therefore, by Theorem 2, $\mathrm{sc}_{\mathbf{Star}}(n) \leq 2^{n-1} + 2^{n-2}$. □

5.1.3 A lower bound

We now prove that the language recognized by $\mathrm{Mon}_n^{\{n-1\}}$ is a witness of the Kleene star, and the above upper bound is met.

Lemma 7. *Le n be an integer greater than or equal to* 2*. If* $G = \{n-1\}$*, then the size of the minimal DFA associated with* A *is* $2^{n-1} + 2^{n-2}$*.*

Proof. Let $G = \{n-1\}$, so that $A = \mathfrak{Star}(\mathrm{Mon}_n^{\{n-1\}})$. Let S be the set of all states E of A such that, if $n-1 \in E$, then $0 \in E$. We show that every state E of S is accessible in A, by induction on the number of elements of E. We follow the intuition provided by Figure 5.4.

The empty set is initial in A. Every singleton of elements of $[\![n]\!]$ is in S, except for the singleton $\{n-1\}$. However, if $j \in [\![n-1]\!]$, $\{j\}$ is accessible from the empty set by reading any letter ϕ such that $\phi(0) = j$. Therefore, any element E of S such that $\#E \leq 1$ is accessible in A.

Now let $j \in \{1, \ldots, n-1\}$, and suppose that any element E of S such that $\#E \leq j$ is accessible in A. Let E' be any element of S such that $\#E' = j+1$. If $E' = \{0, n-1\}$, then it is accessible from $\{0\}$ by reading the letter $(0, n-1)$. Otherwise, let l and l' be two distinct elements of E' such that $l \neq n-1$, and let $E = (0, l) \circ (l', n-1)(E')$. We have $0 \in E$, $n-1 \in E$, and $\#E = \#E'$. Furthermore, E is accessible from the set $E'' = E \setminus \{n-1\}$ by reading the letter $(0, n-1)$. Therefore, E' is accessible from E'', by reading the letter $(0, l)$ followed by the letter $(l', n-1)$. Furthermore, $\#E'' = j$, and $n-1 \notin E''$, which implies that $E'' \in S$. As a consequence, E' is accessible in A. Thus, we have shown that every element of S is accessible in A.

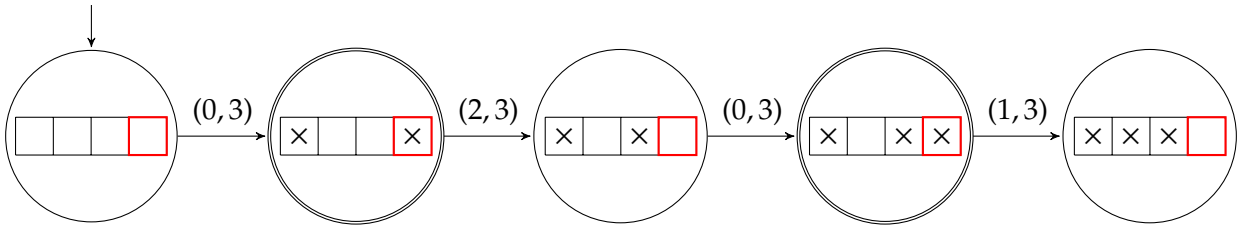


Figure 5.4: A run in $\mathfrak{Star}(\mathrm{Mon}_4^{\{3\}})$ from the initial state $\emptyset$ to the state $\{0, 1, 2\}$.

As a consequence, by Remark 4, the accessible states of A are exactly the states in S. Now we show that the states of S are pairwise distinguishable in A. We follow the intuition of Figure 5.5.

Let E and E' be any two different non-empty elements of S. There exists an integer j such that, either $j \in E$ and $j \notin E'$, or $j \notin E$ and $j \in E'$. As both cases are symmetrical, we suppose that j is an integer such that $j \in E$ and $j \notin E'$. Let ϕ be the letter of Γ_n such that $\phi(j) = n-1$, and such that, for any $l \in [\![n]\!]$ that is not equal to j, $\phi(l) = 0$. Reading the letter ϕ from the state E leads to the state $\{0, n-1\}$. Furthermore, reading the letter ϕ from the state E' leads to the state $\{0\}$. However, both $\{n-1\}$ and $\{0, n-1\}$ are final in A, while $\{0\}$ is not final in A. Therefore, E and E' are distinguishable in A. Furthermore, reading the empty word ε from the state $\emptyset$ in A leads to $\emptyset$, which is final, but reading the empty

word from the state $\{0\}$ in A leads to $\{0\}$, which is not final. As a consequence, $\emptyset$ and $\{0\}$ are distinguishable in A. Hence, any two distinct states of A are distinguishable.

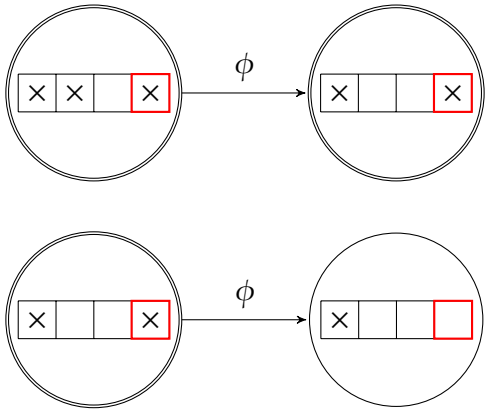


Figure 5.5: How to distinguish between two states of $\mathfrak{Star}(\mathrm{Mon}_4^{\{3\}})$, with the letter ϕ such that $\phi(1) = 3$, and $\phi(0) = \phi(2) = \phi(3) = 0$.

Thus, the size of the minimal DFA associated with A is the size of S, which is $2^{n-1} + 2^{n-2}$. □

Therefore, by Lemma 6 and Lemma 7, $\mathrm{Mon}_n^{\{n-1\}}$ is a witness for the Kleene star, and the upper bound of Lemma 6 is reached.

Proposition 19. *For any integer $n \geq 2$, the state complexity $\mathrm{sc}_{\mathbf{Star}}$ of the Kleene star satisfies $\mathrm{sc}_{\mathbf{Star}}(n) = 2^{n-1} + 2^{n-2}$.*

5.2 Boolean Operations

5.2.1 Applying the modifiers describing boolean operations to monsters

Let $\mathbf{b}$ be a k-ary boolean function, let $(n_1, \ldots, n_k)$ be a k-tuple of positive integers, let $(F_1, \ldots, F_k)$ be a k-tuple of finite sets such that $F_j \subseteq [\![n_j]\!]$, let $\mathrm{Mon}_{n_1,\ldots,n_k}^{F_1,\ldots,F_k} = (\mathbb{M}_1, \ldots, \mathbb{M}_k)$ and let $A = (\Gamma_{n_1,\ldots,n_k}, Q, i, F, \delta) = \mathfrak{m}_{\mathbf{b}}(\mathbb{M}_1, \ldots, \mathbb{M}_k)$. Notice that every DFA that is equal to $\mathfrak{m}_{\mathbf{b}}(\mathrm{Mon}_{n_1,\ldots,n_k}^{F'_1,\ldots,F'_k})$, for some $(F'_1, \ldots, F'_k)$ with $F'_j \subseteq [\![n_j]\!]$, has the same set of states, the same initial state, and the same transition function. It is only its set of final states that depends on $F'_1, \ldots, F'_k$.

Remark 5. *By Definition 22, if $\phi = (\phi_1, \ldots, \phi_k)$ is a letter of $\Gamma_{n_1,\ldots,n_k}$, for any $j \in \{1, \ldots, k\}$, the transition function of ϕ in $\mathbb{M}_j$ is ϕ_j. Therefore, by Definition 16, for any element $(q_1, \ldots, q_k)$ of Q, $\delta^{\phi}(q_1, \ldots, q_k) = (\phi_1(q_1), \ldots, \phi_k(q_k))$. In other words, the transition function of $(\phi_1, \ldots, \phi_k)$ in A is itself, $(\phi_1, \ldots, \phi_k)$. This implies that, if $w = a_1 \cdots a_n$ is a word over $\Gamma_{n_1,\ldots,n_k}$ with $a_l = (\zeta_{l,1}, \ldots, \zeta_{l,k})$, we have $\delta^w = \delta^b$, where*

$$b = (\zeta_{k,1} \circ \cdots \circ \zeta_{1,1}, \ldots, \zeta_{k,k} \circ \cdots \circ \zeta_{1,k}).$$

As a consequence, if there is a run from a state q to a state q' in A, then there exists a letter b of $\Gamma_{n_1,\dots,n_k}$ such that $\delta^b(q) = q'$. Therefore, all accessible states in A are accessible with one letter only, and if, in addition, $F_j \notin \{\emptyset, [\![n_j]\!]\}$ for any $j \in \{1,\dots,k\}$, any two distinguishable states of A are distinguishable with one letter only.

A consequence of the above remark is that any state $(q_1,\dots,q_k)$ of A is accessible from its initial state $(0,\dots,0)$ by reading a letter $(\phi_1,\dots,\phi_k)$ such that $\phi_j(0) = q_j$, for all $j \in \{1,\dots,k\}$. Therefore, to minimize the DFA A, we only need to compute the induced Nerode equivalence.

5.2.2 An upper bound

Definition 23. *For any positive integer k and $\ell \in \{1,\dots,k\}$, we say that a k-ary boolean function $\mathbf{b}$ depends on its ℓ-th coordinate if there exist two k-tuples $(u_1,\dots,u_k)$ and $(v_1,\dots,v_k)$ of elements of $\{0,1\}$, such that $u_j = v_j$ for any $j \neq \ell$, and $\mathbf{b}(u_1,\dots,u_k) \neq \mathbf{b}(v_1,\dots,v_k)$.*

Lemma 8. *For any $\ell \in \{1,\dots,k\}$, if $\mathbf{b}$ does not depend on its ℓ-th coordinate, then any two states $s_1 = (q_1,\dots,q_k)$ and $s_2 = (q'_1,\dots,q'_k)$ of Q with $q_j = q'_j$ for all $j \neq \ell$ are not distinguishable in A.*

Proof. For any letter $\phi = (\phi_1,\dots,\phi_k)$ of $\Gamma_{n_1,\dots,n_k}$ and any two states $s_1 = (q_1,\dots,q_k)$ and $s_2 = (q'_1,\dots,q'_k)$ of Q with $q_j = q'_j$ for all $j \neq l$, we have, for any $j \neq l$, $\phi_j(q_j) = \phi_j(q'_j)$ and $[\phi_j(q_j) \in F_j] = [\phi_j(q'_j) \in F_j]$. Therefore, as $\mathbf{b}$ does not depend on its l-th coordinate, we have

$$\mathbf{b}([\phi_1(q_1) \in F_1],\dots,[\phi_k(q_k) \in F_k]) = \mathbf{b}([\phi_1(q'_1) \in F_1],\dots,[\phi_k(q'_k) \in F_k]),$$

and

$$\mathbf{b}([q_1 \in F_1],\dots,[q_k \in F_k]) = \mathbf{b}([q'_1 \in F_1],\dots,[q'_k \in F_k]).$$

Thus, by Definition 16, $s_1 \in F$ if and only if $s_2 \in F$, and $\delta^\phi(s_1)$ is in F if and only if $\delta^\phi(s_2)$ is in F. Hence, s_1 and s_2 cannot be distinguished in A by reading the empty word. As a consequence, by Remark 5, s_1 and s_2 are not distinguishable in A. □

Let $E_{\mathbf{b}}$ be the set of all integers j in $\{1,\dots,k\}$ such that $\mathbf{b}$ depends on its j-th coordinate. As a consequence of the above lemma, any state $(q_1,\dots,q_k)$ of A is equivalent (in the sense of the Nerode equivalence) to the state $(q'_1,\dots,q'_k)$, where

$$\begin{cases} q'_j = q_j, & \text{for all } j \in E_{\mathbf{b}} \\ q'_j = 0 & \text{otherwise} \end{cases}$$

However, there are exactly $\prod_{j \in E_{\mathbf{b}}} n_j$ elements $(q_1,\dots,q_k)$ of Q, such that $q_j = 0$ for all $j \notin E_{\mathbf{b}}$. It follows that this number is an upper bound for the number of states of a minimal DFA equivalent to A, and therefore an upper bound for the state complexity of $\otimes_{\mathbf{b}}$:

Corollary 1. *Let $E_{\mathbf{b}}$ be the set of all integers j in $\{1,\dots,k\}$ such that $\mathbf{b}$ depends on its j-th coordinate. We have $\mathrm{sc}_{\otimes_{\mathbf{b}}}(n_1,\dots,n_k) \leq \prod_{j \in E_{\mathbf{b}}} n_j$.*

5.2.3 A lower bound

We now prove that this upper bound is tight. Furthermore, we prove that $\mathrm{Mon}^{F_1,\dots,F_k}_{n_1,\dots,n_k}$ is a witness for $\otimes_{\mathbf{b}}$, if $F_j \notin \{\emptyset, [\![n_j]\!]\}$ for all $j \in \{1,\dots,k\}$.

Lemma 9. *If, for all $j \in \{1,\dots,k\}$, $F_j \notin \{[\![n_j]\!], \emptyset\}$, then the size of the minimal DFA equivalent to A is $\prod_{j \in E_{\mathbf{b}}} n_j$.*

Proof. We prove that any two distinct states of the set of all elements $(q_1,\dots,q_k)$ of Q such that $q_j = 1$ for all $j \notin E_{\mathbf{b}}$, are distinguishable. Let $(q_1,\dots,q_k)$ and $(q'_1,\dots,q'_k)$ be two states of Q such that there exists $j \in E_{\mathbf{b}}$ with $q_j \neq q'_j$. As $\mathbf{b}$ depends on j, there exists two k-tuples of elements of $\{0,1\}$, $(u_1,\dots,u_k)$ and $(u'_1,\dots,u'_k)$ such that $\mathbf{b}(u_1,\dots,u_k) = 1$, $\mathbf{b}(u'_1,\dots,u'_k) = 0$ and such that $u_l = v_l$ for all $l \neq j$. Let $\phi = (\phi_1,\dots,\phi_k)$ be a letter $\Gamma_{n_1,\dots,n_k}$ such that

- For all $l \neq j$, $\phi_l(q_l)$ and $\phi_l(q'_l)$ are equal to an element of Q_l that is in F_l if and only if $u_l = u'_l = 1$,
- $\phi_j(q_j)$ is equal to an element of Q_j that is in F_j if and only if $u_j = 1$,
- $\phi_j(q'_j)$ is equal to an element of Q_j that is in F_j if and only if $u'_j = 1$.

We have $([\phi_1(q_1) \in F_1],\dots,[\phi_k(q_k) \in F_k]) = (u_1,\dots,u_k)$ and $([\phi_1(q'_1) \in F_1],\dots,[\phi_k(q'_k) \in F_k]) = (u'_1,\dots,u'_k)$. Therefore, $\mathbf{b}([\phi_1(q_1) \in F_1],\dots,[\phi_k(q_k) \in F_k]) = 1$ and $\mathbf{b}([\phi_1(q'_1) \in F_1],\dots,[\phi_k(q'_k) \in F_k]) = 0$, which implies, by Definition 16, that $(\phi_1(q_1),\dots,\phi_k(q_k)) \in F$ and $(\phi_1(q'_1),\dots,\phi_k(q'_k)) \notin F$. As a consequence, $(q_1,\dots,q_k)$ and $(q'_1,\dots,q'_k)$ are distinguishable in A. □

The state complexity computed in the above lemma coincides with the upper bound of Corollary 1. As a consequence, $\mathrm{Mon}^{F_1,\dots,F_k}_{n_1,\dots,n_k}$ is a witness for every boolean operation, if $F_j \notin \{[\![n_j]\!], \emptyset\}$ for all $j \in \{1,\dots,k\}$. We have thus computed the exact state complexity of every boolean operation.

Proposition 20. *We have* $\mathrm{sc}_{\otimes_{\mathbf{b}}}(n_1,\dots,n_k) = \prod_{j \in E_{\mathbf{b}}} n_j$.

5.3 Catenation

5.3.1 Applying the catenation modifier to monsters

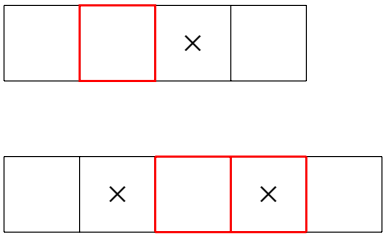
Let (n_1,n_2) be a pair of integers greater than or equal to 2, let (F_1,F_2) be a pair of finite sets such that $F_1 \subseteq [\![n_1]\!]$ and $F_2 \subseteq [\![n_2]\!]$, let $\mathrm{Mon}^{F_1,F_2}_{n_1,n_2} = (\mathbb{M}_1,\mathbb{M}_2)$ and let $A = (\Sigma,Q,i,F,\delta) = \mathfrak{Conc}(\mathbb{M}_1,\mathbb{M}_2)$. From Example 18 and Definition 22, we have

- $\Sigma = \Gamma_{n_1,n_2} = [\![n_1]\!]^{[\![n_1]\!]} \times [\![n_2]\!]^{[\![n_2]\!]}$,
- $Q = [\![n_1]\!] \times 2^{[\![n_2]\!]}$,
- $i = \begin{cases} (0,\emptyset) & \text{if } 0 \notin F_1 \\ (0,\{0\}) & \text{if } 0 \in F_1 \end{cases}$,

- $F = \{(q_1, E) \in [\![n_1]\!] \times 2^{[\![n_2]\!]} \mid E \cap F_2 \neq \emptyset\}$,
- and, for any $(q_1, E) \in [\![n_1]\!] \times 2^{[\![n_2]\!]}$ and any $(\phi_1, \phi_2) \in \Gamma_{n_1,n_2}$,

$$\delta^{(\phi_1,\phi_2)}(q_1, E) = \begin{cases} (\phi_1(q_1), \phi_2(E)) & \text{if } \phi_1(q_1) \notin F_1 \\ (\phi_1(q_1), \phi_2(E) \cup \{0\}) & \text{otherwise.} \end{cases}$$

To represent states of A, we use a representation very similar to the case of the Kleene star (Figures 5.1, 5.2 and 5.3). However, this time, a state (q, E) of A is represented by two "lines" of squares. The first one represents q, and thus has always exactly one cross. The second one represents E, and thus has as many crosses as the size of E. For example, if $n_1 = 4$, $n_2 = 5$, $F_1 = 1$ and $F_2 = \{2, 3\}$, the state $(2, \{1, 3\})$ of A is represented with the following pair of lines:



This representation gives us an interpretation of the transition function of A. Indeed, in A, to go from a state (q, E) to a state (q', E') by reading a letter (ϕ_1, ϕ_2) (in the case where $E \neq \emptyset$), one can change the positions of the crosses on the two lines representing (q, E) by applying ϕ_1 to the cross of the first one, and ϕ_2 to the crosses of the second one, and then by putting a cross at the beginning of the second line if and only if the cross of the first line is in a red square. For example, if $n_1 = 3$, $n_2 = 4$, $F_1 = \{2\}$,$F_2 = \{1\}$, $\phi_1(1) = 2$, $\phi_2(1) = 2$, $\phi_2(2) = 3$, we have $\delta^{\phi}((1, \{1, 2\})) = (2, \{0, 2, 3\})$, which is represented by Figure 5.6.

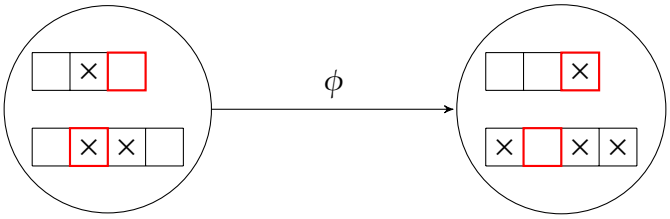


Figure 5.6: A transition in $\mathfrak{Conc}(\mathrm{Mon}_{3,4}^{\{2\},\{1\}})$.

5.3.2 An upper bound

We first establish an upper bound on the state complexity of Catenation. Our reasoning is based on the following remark:

Remark 6. *If reading a letter from any state in A leads to a state whose cross on the first line is in a red square, then this state's second line has a cross in the leftmost square. More formally, if (q, E) is an element of Q, if (ϕ_1, ϕ_2) is any letter of Γ_{n_1,n_2}, and if $\phi_1(q) \in F_1$, then, denoting by (q', E') the*

state $\delta^{(\phi_1,\phi_2)}(q,E)$*, we have* $0 \in E'$*. Therefore, an easy induction shows that any state* (q,E) *of* A *such that* $q \in F_1$ *and* $0 \notin E$ *is not accessible in* A*.*

Lemma 10. *The state complexity* $\mathrm{sc}_{\mathbf{Conc}}$ *of Catenation satisfies* $\mathrm{sc}_{\mathbf{Conc}}(n_1,n_2) \leq (n_1-1)2^{n_2}+2^{n_2-1}$*.*

Proof. If $F_1 = \emptyset$, any state (q,E) of A such that $E \neq \emptyset$ is not accessible. Therefore, the number of accessible states of A is less or equal to n_1. Suppose that $F_1 \neq \emptyset$. The number of all states (q,E) of A such that, if $q \in F_1$ then $0 \in E$ is $(n_1 - \#F_1)\times 2^{n_2} + \#F_1 \times 2^{n_2-1}$. Therefore, by Remark 6, the number of accessible states of A is less or equal to $(n_1 - \#F_1) \times 2^{n_2} + \#F_1 \times 2^{n_2-1} \leq (n_1-1)\times 2^{n_2} + 2^{n_2-1}$. In all cases, the number of accessible states of A is less or equal to $(n_1-1)\times 2^{n_2}+2^{n_2-1}$. Therefore, by Theorem 2, we have $\mathrm{sc}_{\mathbf{Conc}}(n_1,n_2) \leq (n_1-1)2^{n_2}+2^{n_2-1}$. □

5.3.3 A lower bound

We now prove that $\mathrm{Mon}_{n_1,n_2}^{\{n_1-1\},\{n_2-1\}}$ is a witness for Catenation, and the above upper bound is met.

Lemma 11. *If* $F_1 = \{n_1-1\}$ *and* $F_2 = \{n_2-1\}$*, then the size of the minimal DFA associated with* A *is* $(n_1-1)2^{n_2}+2^{n_2-1}$*.*

Proof. Recall that n_1 and n_2 are both greater than or equal to 2. Let $F_1 = \{n_1-1\}$ and $F_2 = \{n_2-1\}$, so that $A = \mathfrak{Conc}(\mathrm{Mon}_{n_1,n_2}^{\{n_1-1\},\{n_2-1\}})$. Let S be the set of all states (j,E) of A such that, if $j = n_1-1$, then $0 \in E$. We show that every state (j,E) of S is accessible in A, by induction on the number of elements of E. We follow the intuition provided by Figure 5.7.

The state $(0,\emptyset)$ is initial in A. A state $(j,\emptyset)$ is in S if and only if $j \in [\![n_1-1]\!]$. However, if $j \in [\![n_1-1]\!]$, $(j,\emptyset)$ is accessible from $(0,\emptyset)$ by reading the letter $((0,j),\mathrm{Id})$. Therefore, any element $(j,\emptyset)$ of S is accessible in A.

The state $(n_1-1,\{0\})$ is accessible in A from $(0,\emptyset)$ by reading the letter $((0,n_1-1),\mathrm{Id})$. Furthermore, any state $(j,\{m\})$ of S, where $j \in [\![n_1-1]\!]$ and $m \in [\![n_2]\!]$ is reached from the state $(n_1-1,\{0\})$ by reading the letter $((n_1-1,j),(0,m))$. Thus, any state (j,E) of S with $\#E \leq 1$ is accessible in A.

Now let $l \in \{1,\ldots,n_2-1\}$, and suppose that every element (j,E) of S such that $\#E \leq l$ is accessible in A. Let (j',E') be any element of S such that $\#E' = l+1$. Let r be any element of E', let r' be any non-zero element of $(0,r)(E')$, and let $E = (0,r)(E')\setminus\{r'\}$. The state (j',E') is reached in A from the state (n_1-1,E) by reading the letter $(\mathrm{Id}_{[\![n_1-1]\!]},(0,r'))$ and then the letter $((n_1-1,j'),(0,r))$. Furthermore, $0 \in E$, which implies that (n_1-1,E) is in S, and $\#E = l$. As a consequence, (j',E') is accessible in A. Thus, we have shown by induction that every element of S is accessible in A.

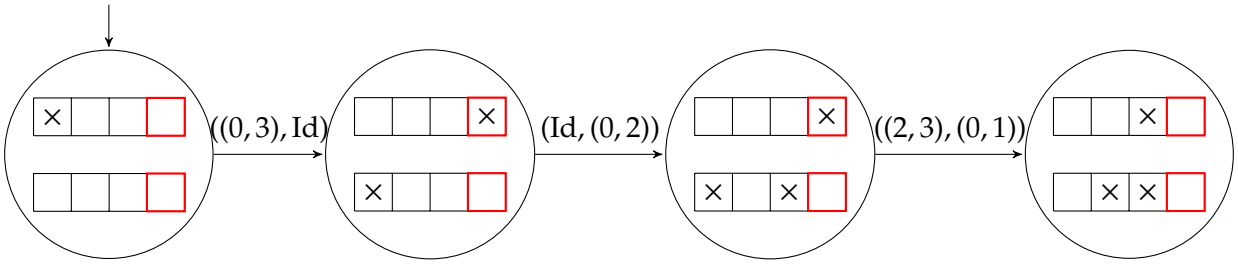


Figure 5.7: A run in $\mathfrak{Star}(\mathrm{Mon}_{4,4}^{\{3\},\{3\}})$ from the initial state $(0, \emptyset)$ to the state $(2, \{1, 2\})$; where $\mathrm{Id} = \mathrm{Id}_{[\![4]\!]}$.

As a consequence, by Remark 6, the accessible states of A are exactly the states in S. Now we show that the states of S are pairwise distinguishable in A.

Let (j, E) and (j', E') be any two different elements of S. We distinguish two cases, and we follow the intuition given by Figures 5.8 and 5.9.

- First suppose that $j \neq j'$. Let ϕ_1 be any function of $[\![n_1]\!]^{[\![n_1]\!]}$ such that $\phi_1(j) = 0$ and $\phi_1(j') = n_1 - 1$, and let ϕ_2 be the function of $[\![n_2]\!]^{[\![n_2]\!]}$ such that $\phi_2(l) = n_2 - 1$, for any $l \in [\![n_2]\!]$. Reading the letter (ϕ_1, ϕ_2) from the state (j, E) leads to the state $(0, \{n_2 - 1\})$. Furthermore, reading the letter (ϕ_1, ϕ_2) from the state (j', E') leads to the state $(n_1 - 1, \{0, n_2 - 1\})$. Therefore, if $\phi_3 = \mathrm{Id}_{[\![n_1]\!]}$ and $\phi_4 = (0, n_2 - 1)$, we have $\delta^{(\phi_1,\phi_2)(\phi_3,\phi_4)}(j, E) = (0, \{0\})$ and $\delta^{(\phi_1,\phi_2)(\phi_3,\phi_4)}(j', E') = (n_1 - 1, \{0, n_2 - 1\})$. However, $(n_1 - 1, \{0, n_2 - 1\})$ is final in A, while $(0, \{0\})$ is not, since $n_2 \geq 2$. As a consequence, (j, E) and (j', E') are distinguishable in A.

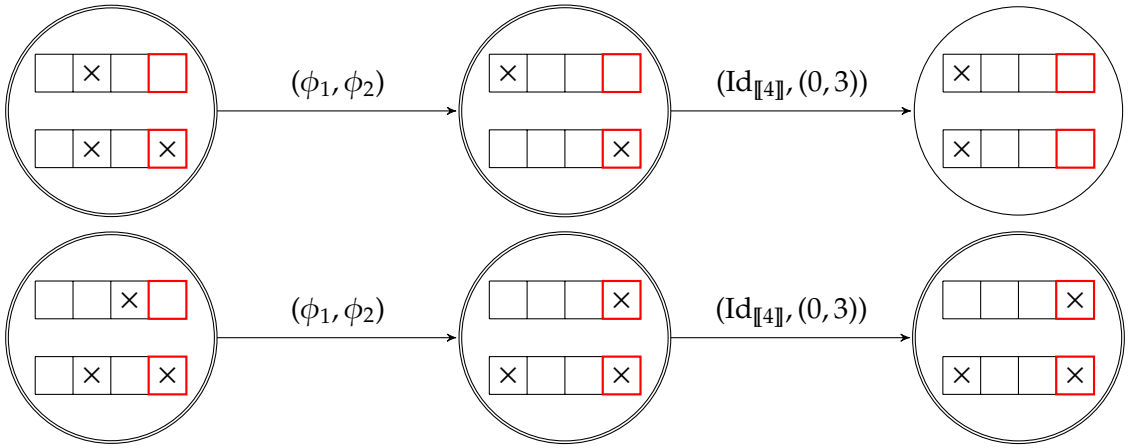


Figure 5.8: How to distinguish between two states $(1, \{1, 3\})$ and $(2, \{1, 3\})$ of $\mathfrak{Conc}(\mathrm{Mon}_{4,4}^{\{3\},\{3\}})$ when $j \neq j'$, with $\phi_1(2) = 3$, $\phi_1(1) = 0$, and $\phi_2(1) = \phi_2(3) = 3$.

- Suppose now that $E \neq E'$. There exists an integer j such that, either $j \in E$ and $j \notin E'$, or $j \notin E$ and $j \in E'$. As both cases are symmetrical, we suppose that j is an integer such that $j \in E$ and $j \notin E'$. Let ϕ_1 be the function of $[\![n_1]\!]^{[\![n_1]\!]}$ such that $\phi_1(l) = 0$, for

any $l \in [\![n_1]\!]$. Let ϕ_2 be the function of $[\![n_2]\!]^{[\![n_2]\!]}$ such that $\phi_2(j) = n_2 - 1$, and such that, for any $l \in [\![n_2]\!]$ that is not equal to j, $\phi_2(l) = 0$. Reading the letter (ϕ_1, ϕ_2) from the state (j, E) leads to the state $(0, \{n_2 - 1\})$ or to the state $(0, \{0, n_2 - 1\})$. Furthermore, reading the letter (ϕ_1, ϕ_2) from the state (j', E') leads to the state $(0, \{0\})$. However, $(0, \{n_2 - 1\})$ and $(0, \{0, n_2 - 1\})$ are final in A, while $(0, \{0\})$ is not final in A. Therefore, (j, E) and (j', E') are distinguishable in A.

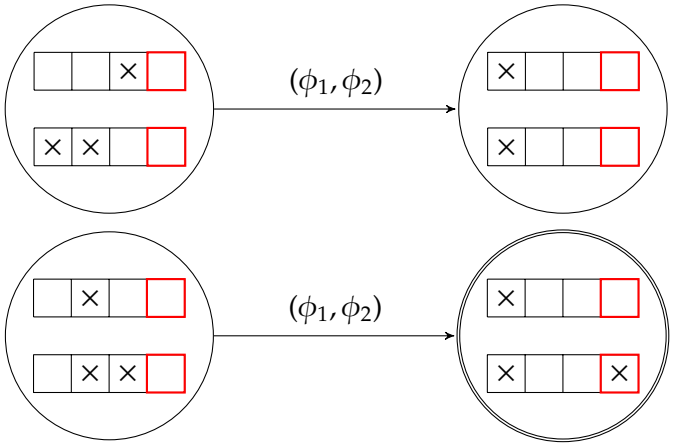


Figure 5.9: How to distinguish between two states (j, E) and (j', E') of $\mathfrak{Conc}(\mathrm{Mon}_{4,4}^{\{3\},\{3\}})$ when $E \neq E'$, with $\phi_1(1) = \phi_1(2) = 0$, $\phi_2(1) = \phi_2(0) = 0$, and $\phi_2(2) = 3$.

Thus, we have shown that the states of S are pairwise distinguishable in A. Therefore, the size of the minimal DFA associated with A is the size of S, which is $(n_1 - 1)2^{n_2} + 2^{n_2-1}$. □

Therefore, by Lemma 10 and Lemma 11, $\mathrm{Mon}_{n_1,n_2}^{\{n_1-1\},\{n_2-1\}}$ is a witness for Catenation, and the upper bound of Lemma 10 is reached.

Proposition 21. *For any pair of positive integers (n_1, n_2) with $n_2 \geq 2$, the state complexity $\mathrm{sc}_{\mathbf{Conc}}$ of Catenation satisfies $\mathrm{sc}_{\mathbf{Conc}}(n_1, n_2) = (n_1 - 1)2^{n_2} + 2^{n_2-1}$.*

Chapter 6

On the star of boolean operations

We have computed in Section 5.2 the state complexity of boolean operations. A natural question arises: what happens when we take the Kleene star of a boolean operation? Although these combined operations do not seem much more complicated, their state complexities are actually much harder to compute. In [16], the authors compute the state complexity of the star of union and make some advances towards computing the state complexity of the star of intersection. This state complexity was later computed in [27]. However, in both cases, the methods used seem to be very specific to both of these operations. Furthermore, in [17], the authors compute the state complexity of the star of multiple unions, generalizing the method used in [16].

The results discussed above are not sufficient to compute the state complexity of the star of every binary boolean operation. We completely answer this problem by computing the state complexity of the star of symmetric difference, in a way that is inspired by [8]. This computation requires some work, because it involves more complicated combinatorial objects than in the case of the star of union or the star of intersection. Furthermore, we give a witness for the star of symmetric difference with an alphabet of size 16. Finally, we give some ideas to generalize our reasoning to the star of boolean operations in Section 6.7.

6.1 The star of the symmetric difference: a first analysis

We let $\circledast$ denote the composition of **Star** and $\otimes_{\mathbf{xor}}$. In other words, $\circledast$ is the binary regular operation such that, for any pair of languages (L_1, L_2) over the same alphabet, we have $\circledast(L_1, L_2) = (L_1 \Delta L_2)^\star$. By Example 15, Example 10, and Proposition 13, $\circledast$ is a 1-uniform operation. Furthermore, by Proposition 18, $\mathbf{desc}(\mathfrak{Star} \circ_1 \mathfrak{m}_{\mathbf{xor}}) = \circledast$, *i.e.*, the modifier $\mathfrak{Star} \circ_1 \mathfrak{m}_{\mathbf{xor}}$ describes $\circledast$. Therefore, to prove state complexity results about $\circledast$, we first give a formula that describes the application of $\mathfrak{St}\mathfrak{X} = \mathfrak{Star} \circ_1 \mathfrak{m}_{\mathbf{xor}}$ to a pair of DFAs. This formula is obtained directly from Example 15, Example 17, and Definition 6.

Let (A_1, A_2) be a pair of DFAs with $A_1 = (\Sigma, Q_1, i_1, F_1, \delta_1)$ and $A_2 = (\Sigma, Q_2, i_2, F_2, \delta_2)$. We have

$$\mathfrak{St}\mathfrak{X}(A_1, A_2) = (\Sigma, 2^{Q_1 \times Q_2}, \emptyset, \{E \subseteq Q_1 \times Q_2 \mid E \cap F \neq \emptyset\} \cup \{\emptyset\}, \delta),$$

where $F = (F_1 \times Q_2)\Delta(Q_1 \times F_2)$ and where, for any letter a of Σ,

$$\delta^a(\emptyset) = \begin{cases} \{(\delta_1^a(i_1), \delta_2^a(i_2))\} \text{ if } (\delta_1^a(i_1), \delta_2^a(i_2)) \notin F \\ \{(\delta_1^a(i_1), \delta_2^a(i_2)), (i_1, i_2)\} \text{ otherwise} \end{cases}$$

$$\text{and, for all } E \neq \emptyset,\ \delta^a(E) = \begin{cases} (\delta_1^a, \delta_2^a)(E) \text{ if } (\delta_1^a, \delta_2^a)(E) \cap F = \emptyset \\ (\delta_1^a, \delta_2^a)(E) \cup \{(i_1, i_2)\} \text{ otherwise.} \end{cases}$$

Notice first that if $n_1 = 1$ or $n_2 = 1$, we can compute $\mathrm{sc}_{\circledast}(n_1, n_2)$ with the results of Section 5.1. Suppose $n_1 = 1$ and $n_2 \geq 2$, for example, and let L_1 be a language over an alphabet Σ of state complexity equal to 1. We have either $L_1 = \emptyset$ or $L_2 = \Sigma^*$. Thus, for any language L_2 over Σ, if $L_1 = \emptyset$, then we have $\circledast(L_1, L_2) = L_2^\star$. Furthermore, if $L_1 = \Sigma^\star$, then we have $\circledast(L_1, L_2) = (L_2^c)^\star$. However, for any positive integer n_2 and any language L_2 of state complexity inferior or equal to n_2, we know, from Example 8, that the state complexity of L_2^c is at most n_2. Hence, from Proposition 19, we have $\mathrm{sc}((L_2^c)^\star) \leq 2^{n_2-1} + 2^{n_2-2}$ and $\mathrm{sc}(L_2^\star) \leq 2^{n_2-1} + 2^{n_2-2}$. To summarize, we have $\mathrm{sc}(\circledast(L_1, L_2)) \leq 2^{n_2-1} + 2^{n_2-2}$ for any positive integer n_2 and any language L_2 of state complexity inferior or equal to n_2. However, from Lemma 7, we also have $\mathrm{sc}(\circledast(\emptyset, \mathrm{L}(\mathrm{Mon}_{n_2}^{\{n_2-1\}}))) = 2^{n_2-1} + 2^{n_2-2}$, for any integer $n_2 \geq 2$. As a consequence, we have $\mathrm{sc}_{\circledast}(n_1, n_2) = 2^{n_2-1} + 2^{n_2-2}$, if $n_1 = 1$ and $n_2 \geq 2$. Similarly, if $n_2 = 1$ and $n_1 \geq 2$, we have $\mathrm{sc}_{\circledast}(n_1, n_2) = 2^{n_1-1} + 2^{n_1-2}$. Therefore, for the remainder of this section, we only concern ourselves with computing $\mathrm{sc}_{\circledast}(n_1, n_2)$ when n_1 and n_2 are superior or equal to 2.

Since $\circledast$ is a binary 1-uniform operation admits a family of 2-monsters as witness. Let (n_1, n_2) be two integers greater than or equal to 2, and let $(\mathbb{M}_1, \mathbb{M}_2) = \mathrm{Mon}_{n_1,n_2}^{\{n_1-1\},\{0\}}$. We are going to show that $(\mathrm{L}(\mathbb{M}_1)), \mathrm{L}(\mathbb{M}_2))$ is indeed a witness for $\circledast$. This allows us to compute its state complexity. To be more precise, here is the outline of our proof. For any $F_1 \subseteq [\![n_1]\!]$ and any $F_2 \subseteq [\![n_2]\!]$, we let M_{F_1,F_2} denote the DFA $\mathfrak{Stx}(\mathrm{Mon}_{n_1,n_2}^{F_1,F_2})$. We are going to minimize the DFA $\mathrm{M}_{\{n_1-1\},\{0\}}$ by first computing its Nerode equivalence, and then by computing its accessible part. We will therefore have computed the minimal DFA equivalent to $\mathrm{M}_{\{n_1-1\},\{0\}}$, and computing its size allows us to compute the state complexity of $\mathrm{L}(\mathrm{M}_{\{n_1-1\},\{0\}})$. We then show that the state complexity of $\mathrm{L}(\mathrm{M}_{\{n_1-1\},\{0\}})$ is the greatest out of all the state complexities of $\mathrm{L}(\mathrm{M}_{F_1,F_2})$, with $(F_1, F_2) \subseteq [\![n_1]\!] \times [\![n_2]\!]$. Theorem 2 allows us to conclude that the state complexity of $\mathrm{L}(\mathrm{M}_{\{n_1-1\},\{0\}})$ is indeed $\mathrm{sc}_{\circledast}(n_1, n_2)$.

6.2 Computing the Nerode equivalence of $\mathrm{M}_{\{n_1-1\},\{0\}}$

In order to give a visual representation of the next proofs, we associate elements of $2^{[\![n_1]\!]\times[\![n_2]\!]}$ with boolean matrices of size $n_1 \times n_2$. Such a matrix is called a tableau of size $n_1 \times n_2$ when crosses are put in place of 1s, and 0s are erased. The same symbol denotes the element of $2^{[\![n_1]\!]\times[\![n_2]\!]}$, the associated boolean matrix, and the associated tableau. If T is an element of $2^{[\![n_1]\!]\times[\![n_2]\!]}$, we let $T_{x,y}$ denote the value of the boolean matrix T at row x and column y. Therefore, the three following assertions mean the same thing: a cross is at the coordinates (x, y) in T, $T_{x,y} = 1$, $(x, y) \in T$.

We say that a cross at coordinates (x, y) of a tableau, is in the final zone of $\mathrm{M}_{\mathrm{F}_1,\mathrm{F}_2}$, if $(x, y) \in (F_1 \times [\![n_2]\!])\Delta([\![n_1]\!] \times F_2)$. We notice that a tableau T of size $n_1 \times n_2$ is final in $\mathrm{M}_{\mathrm{F}_1,\mathrm{F}_2}$ if

and only if T has a cross in the final zone of $\mathrm{M}_{\mathrm{F}_1,\mathrm{F}_2}$, or T is empty. We fix for the remainder of this chapter two integers n_1 and n_2 greater than or equal to 2.

Definition 24. *A tableau T of size $n_1 \times n_2$ is right-triangle free if $\forall x, x' \in [\![n_1]\!]$ such that $x \neq x'$ and $\forall y, y' \in [\![n_2]\!]$, such that $y \neq y'$, we have $\#(\{(x,y),(x,y'),(x',y),(x',y')\} \cap T) \neq 3$.*

Example 21. *Figure 6.1 is an example of a right triangle.*

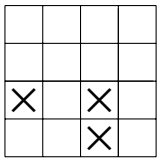


Figure 6.1: A tableau with a right triangle.

Definition 25. *We let $\rightarrow$ denote the relation over tableaux of size $n_1 \times n_2$ that satisfies the following property: for any two tableaux T and T' of size $n_1 \times n_2$, we have $T \rightarrow T'$ if and only if there exist two pairs of integers (i,j) and (i',j') in $[\![n_1]\!] \times [\![n_2]\!]$, such that $T' = T \cup \{(i',j')\}$ and $\{(i,j),(i',j),(i,j')\} \subseteq T$. We let $\overset{*}{\leftrightarrow}$ denote the reflexive, symmetric and transitive closure of $\rightarrow$.*

For any tableau T, we define $\mathrm{Sat}(T)$ as the smallest tableau (relatively to inclusion) with no right triangle containing T. The existence of $\mathrm{Sat}(T)$ comes from the fact that $T \subseteq 2^{[\![n_1]\!]\times[\![n_2]\!]}$, which is a right-triangle free tableau, and the uniqueness of $\mathrm{Sat}(T)$ comes from the fact that the intersection of two right-triangle free tableaux is right-triangle free. Two tableaux T and T' are therefore equivalent if $\mathrm{Sat}(T) = \mathrm{Sat}(T')$.

Lemma 12. *A tableau T of size $n_1 \times n_2$ is right-triangle free if and only if, for all $i, i' \in [\![n_1]\!]$, the lines i and i' are either the same* (i.e., *for all $j \in [\![n_2]\!]$, we have $T_{i,j} = T_{i',j}$), or disjoint* (i.e., *for all $j \in [\![n_2]\!]$, we have $T_{i,j} = 0 \vee T_{i',j} = 0$).*

Proof. If T has a right triangle, that is there exists $i, i' \in [\![n_1]\!]$ and $j, j' \in [\![n_2]\!]$ such that $\{(i,j),(i',j),(i,j')\} \subseteq T$ but $(i',j') \notin T'$, then the lines i and i' are neither the same nor disjoint.

Conversely, if there are two lines i and i' that are neither the same nor disjoint, then $i \neq i'$. Since the two lines are not disjoint, there exists $j \in [\![n_2]\!]$ such that $T_{i,j} = T_{i',j} = 1$. Furthermore, since they are not the same either, there exists $j' \in [\![n_2]\!]$ such that $T_{i,j'} \neq T_{i',j'}$. As a consequence, T has a right triangle and is not right-triangle free. □

Lemma 13. *Let $F_1 \subseteq [\![n_1]\!]$ with $F_1 \notin \{\emptyset, [\![n_1]\!]\}$, let $F_2 \subseteq [\![n_2]\!]$ with $F_2 \notin \{\emptyset, [\![n_2]\!]\}$, and let T and T' be any two non-empty states of $\mathrm{M}_{\mathrm{F}_1,\mathrm{F}_2}$ such that $T \rightarrow T'$. Then T is final if and only if T' is final.*

Proof. As $T \subseteq T'$, if T is final, T' is also final. Now let us suppose T' is final and let i, j, i', j' be the integers of Definition 25. If another cross of T' than (i',j') is in the final zone, then T is also final. Let us now prove that (i',j') is in the final zone implies T is final. Thus, let us suppose, for example that $i' \in F_1$, and $j' \notin F_2$. Then, either $j \notin F_2$ and (i',j) is in the final zone of T, either $j \in F_2$ and $i \notin F_1$ and (i,j) is in the final zone of T, or either $j \in F_2$ and $i \in F_1$ and (i,j') is in the final zone of T. In every case, T is final. □

Let us recall that the alphabet of $\mathrm{M}_{\mathrm{F}_1,\mathrm{F}_2}$ is $\Gamma_{n_1,n_2} = [\![n_1]\!]^{[\![n_1]\!]} \times [\![n_2]\!]^{[\![n_2]\!]}$.

Lemma 14. *Let $F_1 \subseteq [\![n_1]\!]$ with $F_1 \notin \{\emptyset, [\![n_1]\!]\}$, let $F_2 \subseteq [\![n_2]\!]$ with $F_2 \notin \{\emptyset, [\![n_2]\!]\}$, and let T and T' be two non-empty states of M_{F_1,F_2} such that $T \to T'$. Then, for any $a \in \Gamma_{n_1,n_2}$, we have $\delta^a(T) \to \delta^a(T')$ or $\delta^a(T) = \delta^a(T')$, where δ is the transition function of M_{F_1,F_2}.*

Proof. Let i, j, i', j' be the integers of Definition 25. Let $a = (f, g) \in \Gamma_{n_1,n_2}$. We have, either $(f,g)(T) \to (f,g)(T')$ or $(f,g)(T) = (f,g)(T')$. If $(f,g)(T) = (f,g)(T')$, then obviously have $\delta^a(T) = \delta^a(T')$. Therefore, we suppose now that $(f,g)(T) \to (f,g)(T')$.

Suppose $(f,g)(T)$ is not final. Then $(f,g)(T')$ is not final and $\delta^a(T) = (f,g)(T) \to (f,g)(T') = \delta^a(T')$. Otherwise, if $(f,g)(T)$ is final, then $(f,g)(T')$ is also final and $\delta^a(T') = (f,g)(T') \cup \{(0,0)\} = (f,g)(T) \cup \{(f(i'), g(j')), (0,0)\} = \delta^a(T) \cup \{(f(i'), g(j'))\}$. Hence, the lemma follows from the fact that

$$\{(f(i), g(j)), (f(i), g(j')), (f(i'), g(j))\} \subseteq \delta^a(T).$$

□

Proposition 22. *Let $F_1 \subseteq [\![n_1]\!]$ with $F_1 \notin \{\emptyset, [\![n_1]\!]\}$, let $F_2 \subseteq [\![n_2]\!]$ with $F_2 \notin \{\emptyset, [\![n_2]\!]\}$, and let T, T' be two non-empty states of M_{F_1,F_2}. If $T \overset{*}{\leftrightarrow} T'$, then T and T' are indistinguishable in M_{F_1,F_2}.*

Proof. Let δ be the transition function of M_{F_1,F_2}. From Lemma 14, we show by a straightforward induction that, for any word w, if $T \to T'$, then $\delta^w(T) \to \delta^w(T')$ or $\delta^w(T) = \delta^w(T')$. From Lemma 13, if $T \to T'$, then T and T' are indistinguishable. However, $\overset{*}{\leftrightarrow}$ is the reflexive, symmetric and transitive closure of $\to$. Hence, if $T \overset{*}{\leftrightarrow} T'$, then T and T' are not distinguishable in M_{F_1,F_2}. □

We let $\sim$ denote the equivalence relation over tableaux of size $n_1 \times n_2$ such that we have $T \sim T'$ if and only if, either $T, T' \in \{\emptyset, \{(0,0)\}\}$ or $T \overset{*}{\leftrightarrow} T'$. Notice that, since $\{(0,0)\}$ is final in $\mathrm{M}_{\{n_1-1\},\{0\}}$, $\emptyset$ and $\{(0,0)\}$ are not distinguishable in $\mathrm{M}_{\{n_1-1\},\{0\}}$. Therefore, $\sim$ is compatible with $\mathrm{M}_{\{n_1-1\},\{0\}}$ (see Section 2.4.2 for a definition of *compatible*), and it makes sense to consider $\mathrm{M}_{\{n_1-1\},\{0\}}/_\sim$.

Recall here that we let (i, j) denote the transposition exchanging i and j, by $(i_1, \dots, i_m)$ the permutation such that $(i_1, \dots, i_m)(i_j) = i_{(j+1) \bmod m}$, and by π_i^j the application that sends j to i, and does not change any other elements (for more details, see Section 2.2).

Lemma 15. *Any two distinct non-empty states of $\mathrm{M}_{\{n_1-1\},\{0\}}/_\sim$ are distinguishable in $\mathrm{M}_{\{n_1-1\},\{0\}}/_\sim$.*

Proof. Let us first recall that the final zone of $\mathrm{M}_{\{n_1-1\},\{0\}}$ is the set of all $(i, j) \in [\![n_1]\!] \times [\![n_2]\!]$ such that either $i = n_1 - 1$ or $j = 0$, but not both. We use this fact statement in the rest of the proof. Let δ be the transition function of $\mathrm{M}_{\{n_1-1\},\{0\}}$. Let T and T' be two right-triangle free non-empty tableaux of size $n_1 \times n_2$, such that $T \neq T'$. Let (i, j) be such that $T_{i,j} \neq T'_{i,j}$. We prove that T and T' are distinguishable in $\mathrm{M}_{\{n_1-1\},\{0\}}$. We suppose that $T_{i,j} = 1$ (the case $T'_{i,j} = 1$ is symmetrical), and we consider two cases.

Suppose that for all $i' \in [\![n_1]\!]$, $T'_{i',j} = 0$. Then, since T' is not empty, $(\pi_0^{n_1-1}, (0, j))(T')$ does not have any cross on line $n_1 - 1$ nor on column 0. However, $(\pi_0^{n_1-1}, (0, j))(i, j) = (\pi_0^{n_1-1}(i), 0)$. Therefore, $(\pi_0^{n_1-1}, (0, j))(T)$ does have a cross on column 0, but not on line $n_1 - 1$ (the cross that comes from the coordinates (i, j) in T). As a consequence, $\delta^{(\pi_0^{n_1-1}, (0,j))}(T)$ is final in

$\mathrm{M}_{\{n_1-1\},\{0\}}$, and $\delta^{(\pi_0^{n_1-1},(0,j))}(T')$ is not final in $\mathrm{M}_{\{n_1-1\},\{0\}}$ since T' is not empty, which implies that T and T' are distinguishable in $\mathrm{M}_{\{n_1-1\},\{0\}}$.

We now suppose that there exists $x \in [\![n_1]\!]$ such that $T'_{x,j} = 1$. Let $\{i_1, \dots, i_\ell\} = \{\alpha \mid T'_{\alpha,j} = 1\}$ and let $\{j_1, \dots, j_p\} = \{\beta \mid T'_{i_1,\beta} = 1\}$. The two next properties are illustrated by Figure 6.2. We designate them by Property 1 and Property 2 in the rest of the proof.

1. By Lemma 12, lines $i_1, \dots, i_\ell$ are the same, as they all have a cross on the column j. Columns $\{j_1, \dots, j_p\}$ are also the same, as they all have a cross on line i_1. It follows that, if
$$(i', j') \in \Big(\{i_1, \dots, i_\ell\} \times \Big(\{0, \dots, n_2 - 1\} \setminus \{j_1, \dots, j_p\}\Big)\Big) \cup \Big((\{0, \dots, n_1 - 1\} \setminus \{i_1, \dots, i_\ell\}) \times \{j_1, \dots, j_p\}\Big),$$
then we have $T'_{i',j'} = 0$.
2. We have $j \in \{j_1, \dots, j_p\}$ and $i \notin \{i_1, \dots, i_\ell\}$.

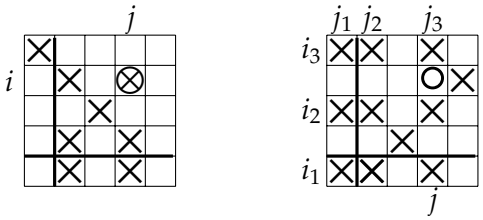


Figure 6.2: An example of two tableaux T and T'.

We let (f, g) denote the pair of mappings such that, for any $(i', j') \in [\![n_1]\!] \times [\![n_2]\!]$, we have
$$f(i') = \begin{cases} n_1 - 1 & \text{if } i' \in \{i_1, \dots, i_\ell\} \\ 0 & \text{otherwise} \end{cases} \quad \text{and } g(j') = \begin{cases} 0 & \text{if } j' \in \{j_1, \dots, j_p\} \\ n_2 - 1 & \text{otherwise.} \end{cases}$$

If $(f(i'), g(j'))$ is in the final zone of $\mathrm{M}_{\{n_1-1\},\{0\}}$, then
$$(i', j') \in \Big(\{i_1, \dots, i_\ell\} \times \Big(\{0, \dots, n_2 - 1\} \setminus \{j_1, \dots, j_p\}\Big)\Big) \cup \Big((\{0, \dots, n_1 - 1\} \setminus \{i_1, \dots, i_\ell\}) \times \{j_1, \dots, j_p\}\Big).$$

Hence, by Property 1, we have $T'_{i',j'} = 0$. As a consequence, $(f, g)(T')$ has at most two crosses, one at $(n_1 - 1, 0)$ and one at $(0, n_2 - 1)$. Neither $(n_1 - 1, 0)$ nor $(0, n_2 - 1)$ is in the final zone of $\mathrm{M}_{\{n_1-1\},\{0\}}$. As a consequence, $(f, g)(T') = \delta^{(f,g)}(T')$ is not final in $\mathrm{M}_{\{n_1-1\},\{0\}}$. However, by Property 2, since $T_{i,j} = 1$, we have $((f, g)(T))_{0,0} = 1$. Therefore, $(f, g)(T) = \delta^{(f,g)}(T)$ is final in $\mathrm{M}_{\{n_1-1\},\{0\}}$. Thus, T and T' are distinguishable in $\mathrm{M}_{\{n_1-1\},\{0\}}$.

In both cases, T and T' are distinguishable in $\mathrm{M}_{\{n_1-1\},\{0\}}$. Therefore, $\overset{*}{\leftrightarrow} T$ and $\overset{*}{\leftrightarrow} T'$ are distinguishable in $\mathrm{M}_{\{n_1-1\},\{0\}}/_{\overset{*}{\leftrightarrow}}$. To conclude, since the representative of any state of $\mathrm{M}_{\{n_1-1\},\{0\}}/_{\overset{*}{\leftrightarrow}}$ is a right-triangle free tableau, any two distinct states of $\mathrm{M}_{\{n_1-1\},\{0\}}/_{\overset{*}{\leftrightarrow}}$ are distinguishable. □

In particular, from the above lemma, we know that $\{(0,0)\}$ is distinguishable from any other state of $\mathrm{M}_{\{n_1-1\},\{0\}}$ (except for $\emptyset$). Hence, the equivalence class of $\emptyset$ for the Nerode equivalence induced by $\mathrm{M}_{\{n_1-1\},\{0\}}$ is $\{\emptyset, \{(0,0)\}\}$. As a consequence, the following corollary is straightforward from Lemma 15 and of Lemma 22.

Corollary 2. *The Nerode equivalence induced by* $\mathrm{M}_{\{n_1-1\},\{0\}}$ *is equal to* $\sim$.

6.3 Computing the accessible states of $\mathrm{M}_{\{n_1-1\},\{0\}}$

Lemma 16. *The set of accessible states of* $\mathrm{M}_{\{n_1-1\},\{0\}}$ *is the set of all tableaux* T *of size* $n_1 \times n_2$ *such that, if* T *has a cross in the final zone of* $\mathrm{M}_{\{n_1-1\},\{0\}}$*, then* T *has cross at* $(0,0)$.

Proof. Recall first that the final zone of $\mathrm{M}_{\{n_1-1\},\{0\}}$ is the set of all $(i,j) \in [\![n_1]\!] \times [\![n_2]\!]$ such that either $i = n_1 - 1$ or $j = 0$, but not both. We use this repeatedly in the rest of the proof. We let E denote the set of all tableau T of size $n_1 \times n_2$ such that, if T has a cross in the final zone of $\mathrm{M}_{\{n_1-1\},\{0\}}$, then $T_{0,0} = 1$. It follows from the definition of the transition function of $\mathfrak{Stx}$ that every accessible tableau T of $\mathrm{M}_{\{n_1-1\},\{0\}}$ is in E. We now prove that every tableau in E is accessible.

Let δ be the transition function of $\mathrm{M}_{\{n_1-1\},\{0\}}$. For any tableau T of size $n_1 \times n_2$, let $\#_{\mathrm{nf}}T$ be the number of crosses of T which are not in the final zone of $\mathrm{M}_{\{n_1-1\},\{0\}}$. Let $<$ be the strict partial order on tableaux such that $T < T'$ if and only if, either $\#T < \#T'$, or $\#T = \#T'$ and $\#_{\mathrm{nf}}T < \#_{\mathrm{nf}}T'$). We prove that every tableau in E is accessible, by induction on non-empty tableaux for the strict partial order $<$ (the empty tableau is the initial state of $\mathrm{M}_{\{n_1-1\},\{0\}}$, and so it is accessible). The only minimal element for the strict partial order $<$ is the empty tableau. Therefore, for any non-empty tableau $T' \in E$, we define a tableau T such that $T < T'$, and T' is accessible from T. We distinguish the following cases :

- Suppose that T' has no cross in the final zone of $\mathrm{M}_{\{n_1-1\},\{0\}}$. In particular, we have $T'_{0,0} = 0$. Let (i,j) be any cross of T', let $(f,g) = ((0,i),(0,j))$, and let $T = (f,g)(T')$. We have $T_{0,0} = 1$ and $(f,g)(T) = (f,g)((f,g)(T')) = T'$. Therefore, since $T \in E$, $\#T = \#T'$, and $\#_{\mathrm{nf}}T < \#_{\mathrm{nf}}T'$, and since T' is not final, we have $T' = \delta^{(f,g)}(T)$ and $T < T'$.

- Suppose now that T' has at least one cross in the final zone of $\mathrm{M}_{\{n_1-1\},\{0\}}$.

  - Suppose there exists (i,j) in the final zone of $\mathrm{M}_{\{n_1-1\},\{0\}}$, such that $(i,j) \neq (0,0)$ and $T'_{i,j} = 1$. Notice that this implies that $T'_{0,0} = 1$. Let $(f,g) = ((0,i),(0,j))$, let T'' be the cross matrix obtained from T' by deleting the cross at $(0,0)$, and let $T = (f,g)(T'')$. Since $((f,g)(T''))_{0,0} = T''_{i,j} = 1$, we have $T_{0,0} = 1$, and therefore $T \in E$. Furthermore, $(f,g)(T) = (f,g)((f,g)(T'')) = T''$. In addition, T'' has a cross at coordinates (i,j), which is in the final zone. Therefore, $\delta^{(f,g)}(T) = T'' \cup \{(0,0)\} = T'$. To summarize, since $\#T < \#T'$, we have $\delta^{(f,g)}(T) = T'$, $T \in E$, and $T < T'$.
  - Suppose now that $(0,0)$ is the only cross of T' in the final zone of $\mathrm{M}_{\{n_1-1\},\{0\}}$.
    - If $T' = \{(0,0)\}$, then it is accessible from the initial state $T = \emptyset$ by reading $(\mathrm{Id}, \mathrm{Id})$. We obviously have $\#T < \#T'$, and thus $T < T'$.

* If $T' = \{(0,0),(n_1-1,0)\}$, then T' is reached from the state $T = \{(0,0)\}$ by reading, for example, the letter, $((0,n_1-1),\mathrm{Id})$, then the letter $(\mathrm{Id},(0,n_2-1))$, and finally the letter $(\mathrm{Id},\pi_0^{n_2-1})$. Indeed, $\delta^{((0,n_1-1),\mathrm{Id})}(\{(0,0)\}) = \{(n_1-1,0)\}$, $\delta^{(\mathrm{Id},(0,n_2-1))}(\{(n_1-1,0)\}) = \{(0,0),(n_1-1,n_2-1)\}$, and $\delta^{(\mathrm{Id},\pi_0^{n_2-1})}(\{(0,0),(n_1-1,n_2-1)\}) = \{(0,0),(n_1-1,0)\}$. We obviously have $\#T < \#T'$, and thus $T < T'$.
* If $T'_{n_1-1,0} = 1$, $T' \neq \{(0,0)\}$, and $T' \neq \{(0,0),(n_1-1,0)\}$, then T' has a cross at (i,j), where $i \in [\![n_1-1]\!]$ and $j \in \{1,\dots,n_2-1\}$. Let $(f,g) = ((i,n_1-1),\mathrm{Id})$, and let $T = (f,g)(T')$. Since $T_{n_1-1,j} = 1$, T has a cross in the final zone distinct from $(0,0)$, and therefore $\#_{\mathrm{nf}}T < \#_{\mathrm{nf}}T'$. Furthermore, $(f,g)(T) = T'$, which implies, since $(0,0) \in (f,g)(T) = T'$, that we have $\delta^{(f,g)}(T) = T'$. In addition, $(0,0)$ is equal to either $(f,g)(n_1-1,0)$ if $i = 0$, or $(f,g)(0,0)$ if $i \neq 0$. In both cases, we have $T_{0,0} = 1$, which implies that $T \in E$. To summarize, since $\#T = \#T'$, we have $T \in E$, $T' = \delta^{(f,g)}(T)$ and $T < T'$. We illustrate this case with Figure 6.3.

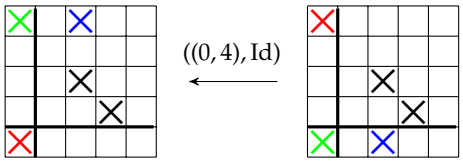


Figure 6.3: How to go from T' to T.

* Suppose now that $T'_{n_1-1,0} = 0$ and $T' \neq \{(0,0)\}$ (this naturally implies that $T' \neq \{(0,0),(n_1-1,0)\}$). Then there exists (i,j), with $i \in [\![n_1-1]\!]$ and $j \in \{1,\dots,n_2-1\}$, such that $T'_{i,j} = 1$. Let $(f,g) = (\pi_i^{n_1-1},\mathrm{Id})$, and let T be the tableau of size $n_1 \times n_2$ such that, for any $(i',j') \in [\![n_1]\!] \times [\![n_2]\!]$, we have

$$T_{i',j'} = \begin{cases} T'_{i,j'} & \text{if } i' = n_1-1 \\ 0 & \text{if } i' = i \\ T'_{i',j'} & \text{otherwise} \end{cases}$$

For any $i \in [\![n_1]\!] \setminus \{0\}$, and any $j \in [\![n_2]\!]$, we have $(f,g)(T)_{i,j} = T_{i,j} = T'_{i,j}$. We also have $(f,g)(T)_{n_1-1,j} = 0 = T'_{n_1-1,j}$, and $(f,g)(T)_{0,j} = T_{n_1-1,j} = T'_{0,j}$, for any $j \in [\![n_2]\!]$. Hence, since $T'_{0,0} = 1$, we have $T_{0,0} = 1$, $(f,g)(T) = T'$, $\#T = \#T'$, and $\#_{\mathrm{nf}}T < \#_{\mathrm{nf}}T'$. Therefore, since $(0,0) \in \delta^{(f,g)}(T)$, we have $T \in E$, $T' = \delta^{(f,g)}(T)$ and $T < T'$. We illustrate this case with Figure 6.4.

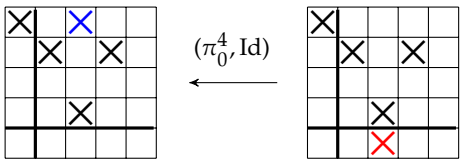


Figure 6.4: How to go from T' to T.

□

For all $(F_1, F_2) \subseteq [\![n_1]\!] \times [\![n_2]\!]$, we let $\widehat{\mathrm{M}}_{F_1,F_2}$ denote the restriction of the DFA M_{F_1,F_2} to all the states T that satisfy: if T has a cross in the final zone of M_{F_1,F_2}, then T has cross at $(0,0)$. The following corollary is straightforward from Corollary 2 and Lemma 16.

Corollary 3. *$\widehat{\mathrm{M}}_{\{n_1-1\},\{0\}}/_\sim$ is a minimal DFA equivalent to $\mathrm{M}_{\{n_1-1\},\{0\}}$.*

Before moving on, we state the following remark that stems from the formula given for $\mathfrak{Stx}$.

Remark 7. *The accessible part of M_{F_1,F_2} is included in $\widehat{\mathrm{M}}_{F_1,F_2}$.*

This remark is useful later on to get an upper bound on the state complexity of M_{F_1,F_2}.

6.4 Computing the state complexity of the language recognized by $\mathrm{M}_{\{n_1-1\},\{0\}}$

Let $\mathcal{R}_{n_1,n_2}$ be the set of non-empty right-triangle free tableaux of size $n_1 \times n_2$. For any $(F_1, F_2) \subseteq [\![n_1]\!] \times [\![n_2]\!]$, we let $\mathcal{T}_{F_1,F_2}$ denote the set of all $T \in \mathcal{R}_{n_1,n_2}$ such that, if T has a cross in the final zone of M_{F_1,F_2}, then $T_{0,0} = 1$. The state $\{(0,0)\}$ is final in $M_{\{n_1-1\},\{0\}}$, and therefore is not distinguishable from $\{\emptyset\}$ in $M_{\{n_1-1\},\{0\}}$. As a consequence, the size of $\widehat{\mathrm{M}}_{\{n_1-1\},\{0\}}/_\sim$ is the cardinality of $\mathcal{T}_{\{n_1-1\},\{0\}}$.

However, for any non-empty tableaux T of size $n_1 \times n_2$, T has no cross in the final zone of $M_{\{n_1-1\},\{0\}}$ if and only if T does not have any cross on line $n_1 - 1$ or on column 0, except maybe for a cross at $(n_1 - 1, 0)$. Therefore the set of all tableaux T of size $n_1 \times n_2$ such that T has no cross in the final zone of $M_{\{n-1\},\{0\}}$, is equal to $E \cup E'$, where

$$E = \{T \in \mathcal{R}_{n_1,n_2} \mid \forall i \in [\![n_1]\!], \forall j \in [\![n_2]\!], T_{i,0} = 0 \wedge T_{n_1-1,j} = 0\} \text{ and}$$
$$E' = \{T \in \mathcal{R}_{n_1,n_2} \mid T_{n_1-1,0} = 1 \text{ and } \forall i \in [\![n_1 - 1]\!], \forall j \in \{1, \dots, n_2 - 1\}, T_{i,0} = 0 \wedge T_{n_1-1,j} = 0\}.$$

However, $\#E = \alpha_{n_1-1,n_2-1} - 1$, and $\#E' = \alpha_{n_1-1,n_2-1}$, where $\alpha_{x,y}$ is the number of right-triangle free tableaux of size $x \times y$, for any positive integers x and y. Furthermore, since $(0,0)$ is in the final zone of $M_{\{n-1\},\{0\}}$, $\mathcal{T}_{\{n_1-1\},\{0\}}$ is equal to the disjoint union of the two following sets: the set of all right-triangle tableaux that do not have any cross in the final zone of $M_{\{n-1\},\{0\}}$, and the set of all right-triangle free tableaux that have a cross at $(0,0)$. Therefore, we have $\#\mathcal{T}_{\{n_1-1\},\{0\}} = 2\alpha_{n_1-1,n_2-1} + \alpha'_{n_1,n_2} - 1$, where $\alpha'_{x,y}$ the number of right-triangle free tableaux of size $x \times y$ having a cross at $(0,0)$, for any positive integers x and y. Therefore, Corollary 3 gives us the state complexity of $\mathrm{L}(\mathrm{M}_{\{n_1-1\},\{0\}})$.

Lemma 17. *The state complexity of $\mathrm{L}(\mathrm{M}_{\{n_1-1\},\{0\}})$ is $2\alpha_{n_1-1,n_2-1} + \alpha'_{n_1,n_2} - 1$.*

Closed formulas for $\alpha(x, y)$ and $\alpha'(x, y)$ are given in Corollary 20 and Proposition 22 of [8].

In the next section, we prove that $(\{n_1 - 1\}, \{0\})$ is a pair of final states (F_1, F_2) that maximizes the size of any minimal DFA equivalent to any DFA M_{F_1,F_2}, with $(F_1, F_2) \subseteq [\![n_1]\!] \times [\![n_2]\!]$.

6.5 Discussing the monsters' final states

Definition 26. *Let T be a right-triangle free tableau of size $n_1 \times n_2$, let ℓ_1, ℓ_2 be two elements of $[\![n_1]\!]$, and let c_1, c_2 be two elements of $[\![n_2]\!]$. We let* $\mathrm{mergel}(T, \ell_1, \ell_2)$ *and* $\mathrm{mergec}(T, c_1, c_2)$ *denote the two tableaux such that, for any $(i, j) \in [\![n_1]\!] \times [\![n_2]\!]$, we have*

$$\mathrm{mergel}(T, \ell_1, \ell_2)_{i,j} = \begin{cases} \max(T_{\ell_1,j}, T_{\ell_2,j}) & \text{if } \begin{cases} i = \ell_1, \\ i = \ell_2, \\ \exists c \mid T_{i,c} = T_{\ell_1,c} = 1, \\ \text{or } \exists c \mid T_{i,c} = T_{\ell_2,c} = 1, \end{cases} \\ T_{i,j} & \text{otherwise,} \end{cases}$$

and

$$\mathrm{mergec}(T, c_1, c_2)_{i,j} = \begin{cases} \max(T_{i,c_1}, T_{i,c_2}) & \text{if } \begin{cases} j = c_1, \\ j = c_2, \\ \exists \ell \mid T_{\ell,j} = T_{\ell,c_1} = 1, \\ \text{or } \exists l \mid T_{\ell,j} = T_{\ell,c_2} = 1, \end{cases} \\ T_{i,j} & \text{otherwise.} \end{cases}$$

The idea behind this definition is that $\mathrm{mergel}(T, \ell_1, \ell_2)$ (resp. $\mathrm{mergec}(T, c_1, c_2)$) merges the lines ℓ_1 and ℓ_2 (respectively the columns c_1 and c_2) of T. In other words, in light of Lemma 12, the function mergel applied to (T, ℓ_1, ℓ_2) replaces all lines that are the same as line ℓ_1 or line ℓ_2, by a line containing all the crosses of ℓ_1 and ℓ_2. Similarly, the function mergec applied to (T, c_1, c_2) replaces all columns that are the same as column c_1 or column c_2, by a column containing all the crosses of c_1 and c_2. Notice that by Lemma 12, $\mathrm{mergel}(T, \ell_1, \ell_2)$ and $\mathrm{mergec}(T, c_1, c_2)$ are right-triangle free tableaux. If S is a set of pairs of integers then we let $\min(S)$ denote the minimal element using the lexicographic order. The next lemma seems weirdly obvious, but we have not found any trivial proof.

Lemma 18. *Let $F_1 \subseteq [\![n_1]\!]$ and $F_2 \subseteq [\![n_2]\!]$ such that $F_1, F_2 \neq \emptyset$, $F_1 \neq [\![n_1]\!]$, and $F_2 \neq [\![n_2]\!]$, and let $F = (F_1 \times [\![n_2]\!]) \Delta ([\![n_1]\!] \times F_2)$ (F is the final zone of M_{F_1,F_2}). We have*

- *if $(0,0) \in F$, then $\mathcal{T}_{F_1,F_2} \leq \mathcal{T}_{\{n_1-1\},\{0\}}$,*
- *and otherwise if $(0,0) \notin F$, then $\mathcal{T}_{F_1,F_2} \leq \mathcal{T}_{\{n_1-1\},\{0\}} - 1$.*

Proof. Let $F_1 \subseteq [\![n_1]\!]$, $F_2 \subseteq [\![n_2]\!]$, such that $F_1, F_2 \neq \emptyset$, $F_1 \neq [\![n_1]\!]$ and $F_2 \neq [\![n_2]\!]$. We let F denote the final zone of M_{F_1,F_2}, *i.e.*, the set $(F_1 \times [\![n_2]\!]) \Delta ([\![n_1]\!] \times F_2)$. We have

$$\begin{aligned} \#\mathcal{T}_{F_1,F_2} &= \#\{T \in \mathcal{R}_{n_1,n_2} \mid (\forall (i,j) \in F, T_{i,j} = 0) \vee T_{0,0} = 1\} \\ &= \#\{T \in \mathcal{R}_{n_1,n_2} \mid (\forall (i,j) \in F, T_{i,j} = 0 \wedge T_{0,0} = 0) \vee T_{0,0} = 1\} \\ &= \#\{T \in \mathcal{R}_{n_1,n_2} \mid T_{0,0} = 1\} + \#\{T \in \mathcal{R}_{n_1,n_2} \mid T_{0,0} = 0 \wedge \forall (i,j) \in F, T_{i,j} = 0\}. \end{aligned} \tag{6.1}$$

In particular, we have

$$\begin{aligned} \#\mathcal{T}_{\{n_1-1\},\{0\}} &= \#\{T \in \mathcal{R}_{n_1,n_2} \mid T_{0,0} = 1\} + \\ \#\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \neq (n_1-1, 0), (i = n_1 - 1 \vee j = 0) &\implies T_{i,j} = 0\}. \end{aligned} \tag{6.2}$$

However, if $(0,0) \notin F$, then

$$\begin{aligned} &\#\{T \in \mathcal{R}_{n_1,n_2} \mid T_{0,0} = 0 \wedge \forall (i,j) \in F, T_{i,j} = 0\} \\ &= \#\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \in F, T_{i,j} = 0\} - \#\{T \in \mathcal{R}_{n_1,n_2} \mid T_{0,0} = 1 \wedge \forall (i,j) \in F, T_{i,j} = 0\} \\ &\leq \#\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \in F, T_{i,j} = 0\} - 1. \end{aligned} \tag{6.3}$$

Therefore, to prove the lemma, it suffices to prove that

$$\begin{aligned} &\#\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \in F, T_{i,j} = 0\} \leq \\ &\#\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \neq (n_1 - 1, 0), (i = n_1 - 1 \vee j = 0) \implies T_{i,j} = 0\}. \end{aligned} \tag{6.4}$$

We notice first that, in the case where $n_1 = 2$, since $F_1, F_2 \neq \emptyset$, $F_1 \neq [\![n_1]\!]$, and $F_2 \neq [\![n_2]\!]$, we have

$$\#\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \in F, T_{i,j} = 0\} = 2^{n_2} - 1.$$

Therefore, in this case, $\#\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \in F, T_{i,j} = 0\}$ does not depend on F_1 and F_2, and (6.4) holds. Inequation (6.4) similarly holds if $n_2 = 2$.

We now suppose that n_1 and n_2 are greater or equal to 3. Notice that we have

$$\#\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \in F, T_{i,j} = 0\} = \#\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \in H_1, T_{i,j} = 0\}, \tag{6.5}$$

where $H_1 = (([\![n_1]\!] \setminus F_1) \times [\![n_2]\!]) \Delta ([\![n_1]\!] \times ([\![n_2]\!] \setminus F_2))$. Indeed, the set $\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \in H_1, T_{i,j} = 0\}$ corresponds to a rotation of 180 degrees of every tableau T in the set $\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \in F, T_{i,j} = 0\}$. Furthermore, we also have

$$\#\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \in F, T_{i,j} = 0\} = \#\{T \in \mathcal{R}_{n_2,n_1} \mid \forall (j,i) \in H_2, T_{j,i} = 0\}, \tag{6.6}$$

where $H_2 = (([\![n_2]\!] \setminus F_2) \times [\![n_1]\!]) \Delta ([\![n_2]\!] \times ([\![n_1]\!] \setminus F_1))$. Indeed, the set $\{T \in \mathcal{R}_{n_2,n_1} \mid \forall (j,i) \in H_2, T_{j,i} = 0\}$ corresponds roughly to an axial symmetry of every tableau T in the set $\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \in F, T_{i,j} = 0\}$, with respect to a straight line going through the coordinates $(0,0)$ and $(\min(n_1,n_2), \min(n_1,n_2))$ of the tableau T. From the two above remarks, we claim that, to prove the lemma, it suffices to prove (6.4) for all F_1 and F_2 such that

1. either $\#F_1 \leq n_1 - 2$ and $\#F_2 \leq n_2 - 2$,
2. or $\#F_1 = 1$ and $\#F_2 = n_2 - 1$

Indeed, suppose that $\#F_1 = n_1 - 1$ or $\#F_2 = n_2 - 1$. If $\#F_1 = n_1 - 1$, we use (6.5) and "exchange" F_1 and F_2 for $[\![n_1]\!] \setminus F_1$ and $[\![n_2]\!] \setminus F_2$ respectively, and we fall into the scope of the first case above if $\#F_2 \neq 1$, and of the second case if $\#F_2 = 1$. If $\#F_2 = n_2 - 1$, we use (6.5) and "exchange" F_1 and F_2 for $[\![n_2]\!] \setminus F_2$ and $[\![n_1]\!] \setminus F_1$ respectively, and we fall into the scope of the first case above if $\#F_1 \neq 1$, and of the second case if $\#F_1 = 1$.

In addition, for any $F_1' \subseteq [\![n_1]\!]$ such that $\#F_1' = \#F_1$, and any $F_2' \subseteq [\![n_2]\!]$ such that $\#F_2' = \#F_2$, we have

$$\#\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \in F, T_{i,j} = 0\} = \#\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \in F', T_{i,j} = 0\},$$

where $F' = (F_1' \times [\![n_2]\!]) \Delta ([\![n_1]\!] \times F_2')$. To summarize, to prove the lemma, since $F_1, F_2 \neq \emptyset$, $F_1 \neq [\![n_1]\!]$, and $F_2 \neq [\![n_2]\!]$, it suffices to prove (6.4) when

- $F_1 = \{\ell_1, \dots, n_1 - 1\}$ for some ℓ_1 with $1 \leq \ell_1 \leq n_1 - 1$.
- $F_2 = \{0, \dots, \ell_2\}$, for some ℓ_2 with $0 \leq \ell_2 \leq n_2 - 2$.
- either $\#F_1 \leq n_1 - 2$ and $\#F_2 \leq n_2 - 2$, or $\#F_1 = 1$ and $\#F_2 = n_2 - 1$.

Hence, in the rest of the proof, we suppose that F_1 and F_2 satisfy the three conditions above.

In the following, we define a mapping ϕ from the set $\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \in F, T_{i,j} = 0\}$ into the set

$$\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \neq (n_1 - 1, 0), (i = n_1 - 1 \vee j = 0) \implies T_{i,j} = 0\}.$$

Furthermore, we also define a partial function ψ from the set

$$\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \neq (n_1 - 1, 0), (i = n_1 - 1 \vee j = 0) \implies T_{i,j} = 0\}$$

to the set $\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \in F, T_{i,j} = 0\}$, such that $\psi \circ \phi$ is the identity over $\{T \in \mathcal{R}_{n_1,n_2} \mid \forall (i,j) \in F, T_{i,j} = 0\}$. We have thus proven that ϕ is an injection. The inequation (6.4) follows.

We consider the following cases:

- If $\#F_1 = \#F_2 = 1$, then $F_1 = \{n_1 - 1\}$ and $F_2 = \{0\}$, and we set ϕ as the identity, which is obviously an injection.
- If $\#F_1 = 1$ and $\#F_2 = n_2 - 1$, then $F_1 = \{n_1 - 1\}$ and $F_2 = [\![n_2 - 1]\!]$, and we set

$$\phi(T)_{i,j} = \begin{cases} 0 & \text{if } (i,j) \in [\![n_1 - 1]\!] \times \{0\}, \\ 0 & \text{if } (i,j) \in \{n_1 - 1\} \times ([\![n_2]\!] \setminus \{0\}), \\ \text{mergel}(T, n_1 - 1, 0)_{i,j} & \text{otherwise.} \end{cases}$$

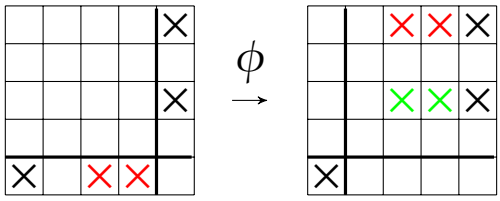


Figure 6.5: An example for $(n_1, n_2) = (5, 5)$.

In this case, we define ψ by

$$\psi(T)_{i,j} = \begin{cases} 0 & \text{if } (i,j) \in [\![n_1 - 1]\!] \times [\![n_2 - 1]\!], \\ 0 & \text{if } (i,j) = (n_1 - 1, n_2 - 1), \\ T_{i,j} & \text{if } (i,j) \in [\![n_1 - 1]\!] \times \{n_2 - 1\}, \\ T_{0,j} & \text{if } (i,j) \in \{n_1 - 1\} \times ([\![n_2 - 1]\!] \setminus \{0\}), \\ T_{i,j} & \text{if } (i,j) = (n_1 - 1, 0). \end{cases}$$

We have

$$(\psi \circ \phi)(T)_{i,j} = \begin{cases} 0 & \text{if } (i,j) \in [\![n_1 - 1]\!] \times [\![n_2 - 1]\!], \\ 0 & \text{if } (i,j) = (n_1 - 1, n_2 - 1), \\ \phi(T)_{i,j} & \text{if } (i,j) \in [\![n_1 - 1]\!] \times \{n_2 - 1\}, \\ \phi(T)_{0,j} & \text{if } (i,j) \in \{n_1 - 1\} \times ([\![n_2 - 1]\!] \setminus \{0\}), \\ \phi(T)_{i,j} & \text{if } (i,j) = (n_1 - 1, 0). \end{cases}$$

We check that $\psi \circ \phi$ is the identity by considering the following cases.

- If $(i,j) \in [\![n_1-1]\!]\times[\![n_2-1]\!]$, or if $(i,j) = (n_1-1, n_2-1)$, we have $\psi(\phi(T))_{i,j} = 0 = T_{i,j}$.
- If $(i,j) \in [\![n_1 - 1]\!] \times \{n_2 - 1\}$, for all $c < n_2 - 1$, we have $T_{i,c} = 0$. Recall also that $T_{n_1-1,n_2-1} = 0$. Therefore, since $i \in [\![n_1 - 1]\!]$, we have

  $$\text{mergel}(T, n_1 - 1, 0)_{i,j} = \begin{cases} \max(T_{n_1-1,j}, T_{0,j}) & \text{if } \begin{cases} i = 0, \\ T_{i,n_2-1} = T_{0,n_2-1} = 1, \end{cases} \\ T_{i,j} & \text{otherwise.} \end{cases}$$

  Furthermore, since $j = n_2 - 1$, we have

  $$\psi(\phi(T))_{i,j} = \text{mergel}(T, n_1 - 1, 0)_{i,j} = \begin{cases} T_{0,j} & \text{if } \begin{cases} i = 0, \\ T_{i,j} = T_{0,j} = 1, \end{cases} \\ T_{i,j} & \text{otherwise.} \end{cases}$$

  Hence, we have $\psi(\phi(T))_{i,j} = T_{i,j}$.
- If $(i,j) \in \{n_1 - 1\} \times ([\![n_2 - 1]\!] \setminus \{0\})$, we have $T_{0,j} = 0$. Therefore, $\psi(\phi(T))_{i,j} = \text{mergel}(T, n_1 - 1, 0)_{0,j} = \max(T_{n_1-1,j}, T_{0,j}) = T_{n_1-1,j} = T_{i,j}$.
- If $(i,j) = (n_1 - 1, 0)$, we have $\psi(\phi(T))_{i,j} = \phi(T)_{i,j} = T_{i,j}$.

- We can now suppose that $\#F_1 \leq n_1 - 1$ and $\#F_2 \leq n_2 - 1$. We distinguish several cases for the definitions of both ϕ and ψ. We first define the function ϕ, and later the function ψ. The function ψ is only defined on the image of ϕ. For each case, $\phi(T)$ falls into the case of the same number in the definition of ψ. From there, in each case, we can prove that $\psi \circ \phi$ is the identity function, in a similar way than what we did for the case of $\#F_1 = 1$ and $\#F_2 = n_2 - 1$, by considering several cases for i and j based on the definition of ϕ and ψ.

  1. If $T_{i,j} = 0$ for all $(i,j) \neq (n_1 - 1, 0)$ such that $i = n_1 - 1 \vee j = 0$, then $\phi(T) = T$. In all of the following cases, we suppose that there exists $(i,j) \neq (n_1 - 1, 0)$ such that $i = n_1 - 1 \vee j = 0$ with $T_{i,j} = 1$.
  2. If $T_{n_1-1,0} = 0$, and
     (a) If there exists $(\ell, c) \in ([\![n_1]\!] \setminus F_1) \times ([\![n_2]\!] \setminus F_2)$ such that $T_{\ell,c} = 0$, then

     $$\phi(T)_{i,j} = \begin{cases} 0 \text{ if } (i,j) \in [\![n_1 - 1]\!] \times \{0\}, \\ 0 \text{ if } (i,j) \in \{n_1 - 1\} \times ([\![n_2]\!] \setminus \{0\}), \\ \text{mergel}(\text{mergec}(T, 0, j_1), n_1 - 1, i_1)_{i,j} & \text{otherwise,} \end{cases}$$

with $(i_1, j_1) = \min\{(\ell, c) \in (\llbracket n_1 \rrbracket \setminus F_1) \times (\llbracket n_2 \rrbracket \setminus F_2) \mid T_{\ell,c} = 0\}$

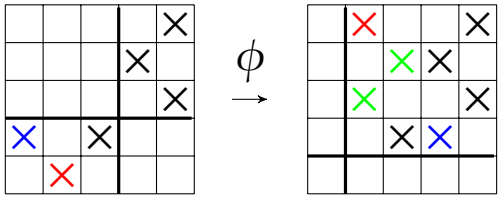


Figure 6.6: A first 5×5 example with $F_1 = \{3, 4\}$ and $F_2 = \{0, 1, 2\}$.

(b) Otherwise

$$\phi(T)_{i,j} = \begin{cases} 0 & \text{if } (i,j) \in (\llbracket n_1 \rrbracket \setminus F_1) \times (\llbracket n_2 \rrbracket \setminus F_2), \\ T_{i,0} & \text{if } (i,j) \in (F_1 \setminus \{n_1 - 1\}) \times \{n_2 - 1\}, \\ T_{n_1-1,j} & \text{if } (i,j) \in \{\ell_1 - 1\} \times (F_2 \setminus \{0\}), \\ 0 & \text{if } (i,j) \in (\llbracket n_1 \rrbracket \setminus \{n_1 - 1\}) \times \{0\}, \\ 0 & \text{if } (i,j) \in \{n_1 - 1\} \times (\llbracket n_2 \rrbracket \setminus \{0\}), \\ T_{i,j} & \text{otherwise,} \end{cases}$$

where ℓ_1 is the minimal element of F_1.

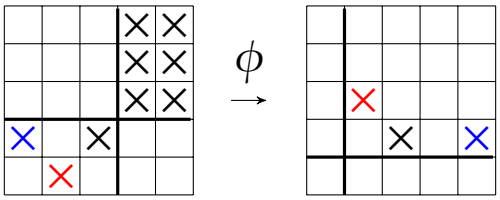


Figure 6.7: A second 5×5 example with $F_1 = \{3, 4\}$ and $F_2 = \{0, 1, 2\}$.

3. If $T_{n_1-1,0} = 1$, and

(a) if there exists $(\ell, c) \in (\llbracket n_1 \rrbracket \setminus F_1) \times (\llbracket n_2 \rrbracket \setminus F_2)$ such that $T_{\ell,c} = 1$, then

$$\phi(T)_{i,j} = \begin{cases} 0 \text{ if } (i,j) \in \llbracket n_1 - 1 \rrbracket \times \{0\}, \\ 0 \text{ if } (i,j) \in \{n_1 - 1\} \times (\llbracket n_2 \rrbracket \setminus \{0\}), \\ \text{mergel}(\text{mergec}(T, 0, j_1), n_1 - 1, i_1)_{i,j} \quad \text{otherwise,} \end{cases}$$

with $(i_1, j_1) = \min\{(\ell, c) \in (\llbracket n_1 \rrbracket \setminus F_1) \times (\llbracket n_2 \rrbracket \setminus F_2) \mid T_{\ell,c} = 1\}$.

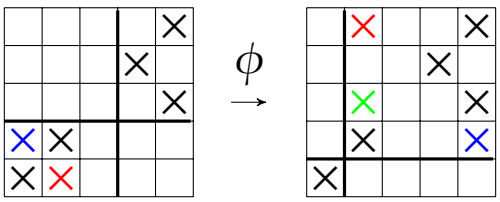


Figure 6.8: A third 5×5 example with $F_1 = \{3, 4\}$ and $F_2 = \{0, 1, 2\}$.

(b) Otherwise,

$$\phi(T)_{i,j} = \begin{cases} 0 & \text{if } (i,j) \in [\![n_1 - 1]\!] \times \{0\}, \\ 0 & \text{if } (i,j) \in \{n_1 - 1\} \times ([\![n_2]\!] \setminus \{0\}), \\ 1 & \text{if } (i,j) = (\ell_1 - 1, n_2 - 1), \\ 1 & \text{if } (i,j) = (0, \ell_2 + 1), \\ T_{i,0} & \text{if } j = n_2 - 1 \wedge i \in F_1 \setminus \{n_1 - 1\}, \\ T_{n_1-1,j} & \text{if } i = \ell_1 - 1 \wedge j \in F_2 \setminus \{0\}, \\ T_{i,j} & \text{otherwise,} \end{cases}$$

where ℓ_1 is the minimal element of F_1 and ℓ_2 is the maximal element of F_2.

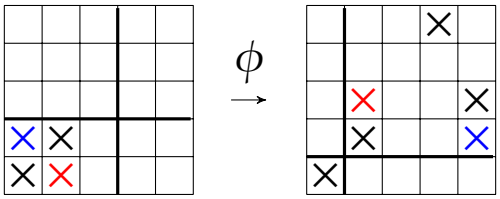


Figure 6.9: An example for $(n_1, n_2) = (5, 5)$, $F_1 = \{3, 4\}$ and $F_2 = \{0, 1, 2\}$.

The application ψ is defined as follows :

1. If $T_{i,j} = 0$ for all $(i,j) \in (F_1 \times [\![n_2]\!]) \Delta ([\![n_1]\!] \times F_2)$, then $\psi(T) = T$. We suppose in all the following cases that there exists $(i,j) \in (F_1 \times [\![n_2]\!]) \Delta ([\![n_1]\!] \times F_2)$ with $T_{i,j} = 1$.

2. If $T_{n_1-1,0} = 0$, then there exists $(i,j) \in ([\![n_1]\!] \setminus F_1) \times ([\![n_2]\!] \setminus F_2)$ with $T_{i,j} = 0$. Let $(i_1, j_1) = \min\{(i,j) \in ([\![n_1]\!] \setminus F_1) \times ([\![n_2]\!] \setminus F_2) \mid T_{i,j} = 0\}$.

   (a) If there exists $(i,j) \in \{i_1\} \times (F_2 \setminus \{0\}) \cup (F_1 \setminus \{n_1 - 1\}) \times \{j_1\}$ such that $T_{i,j} = 1$, then

$$\psi(T)_{i,j} = \begin{cases} 0 & \text{if } (i,j) \in F_1 \times [\![n_2]\!] \setminus F_2, \\ 0 & \text{if } (i,j) \in [\![n_1]\!] \setminus F_1 \times F_2, \\ T_{i_1,j} & \text{if } i = n_1 - 1 \wedge j \in F_2 \setminus \{0\}, \\ T_{i,j_1} & \text{if } j = 0 \wedge i \in F_1 \setminus \{n_1 - 1\}, \\ T_{i,j} & \text{otherwise.} \end{cases}$$

   (b) Otherwise

$$\psi(T)_{i,j} = \begin{cases} 0 & \text{if } (i,j) \in F_1 \times [\![n_2]\!] \setminus F_2, \\ 0 & \text{if } (i,j) \in [\![n_1]\!] \setminus F_1 \times F_2, \\ 1 & \text{if } (i,j) \in ([\![n_1]\!] \setminus F_1) \times ([\![n_2]\!] \setminus F_2) \\ T_{\ell_1-1,j} & \text{if } i = n_1 - 1 \wedge j \in F_2 \setminus \{0\}, \\ T_{i,n_2-1} & \text{if } j = 0 \wedge i \in F_1 \setminus \{n_1 - 1\}, \\ T_{i,j} & \text{otherwise,} \end{cases}$$

where ℓ_1 is the minimal element of F_1.

3. If $T_{n_1-1,0} = 1$, then there exists $(i,j) \in (\llbracket n_1 \rrbracket \setminus F_1) \times (\llbracket n_2 \rrbracket \setminus F_2)$ with $T_{i,j} = 1$. Let $(i_1, j_1) = \min\{(i,j) \in (\llbracket n_1 \rrbracket \setminus F_1) \times (\llbracket n_2 \rrbracket \setminus F_2) \mid T_{i,j} = 1\}$.

   (a) If there exists $(i,j) \in \{i_1\} \times (F_2 \setminus \{0\}) \cup (F_1 \setminus \{n_1 - 1\}) \times \{j_1\}$ such that $T_{i,j} = 1$, then

$$\psi(T)_{i,j} = \begin{cases} 0 & \text{if } (i,j) \in F_1 \times \llbracket n_2 \rrbracket \setminus F_2, \\ 0 & \text{if } (i,j) \in \llbracket n_1 \rrbracket \setminus F_1 \times F_2, \\ T_{i_1,j} & \text{if } i = n_1 - 1 \wedge j \in F_2 \setminus \{0\}, \\ T_{i,j_1} & \text{if } j = 0 \wedge i \in F_1 \setminus \{n_1 - 1\}, \\ T_{i,j} & \text{otherwise.} \end{cases}$$

   (b) Otherwise,

$$\psi(T)_{i,j} = \begin{cases} 0 & \text{if } (i,j) \in F_1 \times \llbracket n_2 \rrbracket \setminus F_2, \\ 0 & \text{if } (i,j) \in \llbracket n_1 \rrbracket \setminus F_1 \times F_2, \\ 0 & \text{if } (i,j) \in (\llbracket n_1 \rrbracket \setminus F_1) \times (\llbracket n_2 \rrbracket \setminus F_2) \\ T_{\ell_1-1,j} & \text{if } i = n_1 - 1 \wedge j \in F_2 \setminus \{0\}, \\ T_{i,n_2-1} & \text{if } j = 0 \wedge i \in F_1, \\ T_{i,j} & \text{otherwise,} \end{cases}$$

   where ℓ_1 is the minimal element of F_1.

□

We know, from Proposition 22, that for any $F_1 \subseteq \llbracket n_1 \rrbracket$ with $F_1 \notin \{\emptyset, \llbracket n_1 \rrbracket\}$, and any $F_2 \subseteq \llbracket n_2 \rrbracket$ with $F_2 \notin \{\emptyset, \llbracket n_2 \rrbracket\}$, the equivalence relation $\overset{*}{\leftrightarrow}$ is compatible with M_{F_1,F_2}. As a consequence, by Remark 7, if $(0,0)$ is not in the final zone of M_{F_1,F_2}, we have

$$\mathrm{sc}(\mathrm{L}(\mathrm{M}_{F_1,F_2})) \leq \#\widehat{\mathrm{M}}_{F_1,F_2}/_{\overset{*}{\leftrightarrow}} \leq \mathcal{T}_{F_1,F_2} + 1 \leq \mathcal{T}_{\{n_1-1\},\{0\}}.$$

Furthermore, if $(0,0)$ is in the final zone of M_{F_1,F_2}, then $\emptyset$ is not distinguishable from $\{(0,0)\}$ in M_{F_1,F_2}, and thus $\sim$ is compatible with M_{F_1,F_2}. Hence, in this case, we have

$$\mathrm{sc}(\mathrm{L}(\mathrm{M}_{F_1,F_2})) \leq \#\widehat{\mathrm{M}}_{F_1,F_2}/_{\sim} \leq \mathcal{T}_{F_1,F_2} \leq \mathcal{T}_{\{n_1-1\},\{0\}}.$$

However, by Corollary 3, we have

$$\mathcal{T}_{\{n_1-1\},\{0\}} = \#\widehat{\mathrm{M}}_{\{n_1-1\},\{0\}}/_{\sim} = \mathrm{sc}(\mathrm{L}(\mathrm{M}_{\{n_1-1\},\{0\}}))$$

Thus, in both case, we have $\mathrm{sc}(\mathrm{L}(\mathrm{M}_{F_1,F_2})) \leq \mathrm{sc}(\mathrm{L}(\mathrm{M}_{\{n_1-1\},\{0\}}))$.

Notice also that if $F_1 = \emptyset$ or if $F_1 = \llbracket n_1 \rrbracket$, then $\mathrm{sc}(\mathrm{L}(M_{F_1,F_2})) \leq \mathrm{sc}_{\circledast}(1, n_2) \leq 2^{n_2-1} + 2^{n_2-2}$. Furthermore, if $F_2 = \emptyset$ or if $F_2 = \llbracket n_2 \rrbracket$, then $\mathrm{sc}(\mathrm{L}(M_{F_1,F_2})) \leq \mathrm{sc}_{\circledast}(n_1, 1) \leq 2^{n_1-1} + 2^{n_1-2}$. Hence, it is easy to check that, in both cases, we also have $\mathrm{sc}(\mathrm{L}(\mathrm{M}_{F_1,F_2})) \leq \mathrm{sc}(\mathrm{L}(\mathrm{M}_{\{n_1-1\},\{0\}}))$.

To summarize, for any $F_1 \subseteq \llbracket n_1 \rrbracket$ and any $F_2 \subseteq \llbracket n_2 \rrbracket$, $\mathrm{sc}(\mathrm{L}(\mathrm{M}_{F_1,F_2})) \leq \mathrm{sc}(\mathrm{L}(\mathrm{M}_{\{n_1-1\},\{0\}}))$. Hence, Theorem 2 gives us the state complexity of $\circledast$.

Theorem 3. *For any integer n_1 and n_2 greater than or equal to 2, we have $\mathrm{sc}_{\circledast}(n_1, n_2) = 2\alpha_{n_1-1,n_2-1} + \alpha'_{n_1,n_2} - 1$. Furthermore, $\mathrm{L}(\mathrm{Mon}_{n_1,n_2}^{\{n_1-1\},\{0\}})$ is a witness for $\circledast$,* i.e.,*for any integers n_1 and n_2 greater or equal to 2, we have $\mathrm{sc}_{\circledast}(n_1, n_2) = \mathrm{sc}(\circledast(\mathrm{L}(\mathbb{M}_1), \mathrm{L}(\mathbb{M}_2)))$, where $(\mathbb{M}_1, \mathbb{M}_2) = \mathrm{Mon}_{n_1,n_2}^{\{n_1-1\},\{0\}}$.*

6.6 Witnesses with a finitely bounded alphabet size

We now prove that $\circledast$ admits a witness with a finitely bounded alphabet. Let n_1, n_2 be two positive integers greater than or equal to 2, and let $(\mathbb{M}_1, \mathbb{M}_2) = \mathrm{Mon}_{n_1,n_2}^{\{n_1-1\},\{0\}}$. Let B_1 and B_2 be the DFAs obtained by restricting the letters of respectively $\mathbb{M}_1$ and $\mathbb{M}_2$ to the alphabet

$$\Sigma' = \Big\{((0,\dots,n_1-2),\mathrm{Id}),((1,\dots,n_1-2),\mathrm{Id}),(\mathrm{Id},(1,\dots,n_2-2)),((1,\dots,n_1-1),\mathrm{Id}),$$
$$(\mathrm{Id},(1,\dots,n_2-1)),((0,n_1-1),\mathrm{Id}),(\mathrm{Id},(0,n_2-1)),((0,1),(0,1)),((0,1),\mathrm{Id}),$$
$$(\mathrm{Id},(0,1)),(\pi_0^1,\mathrm{Id}),(\mathrm{Id},\pi_0^1),(\pi_{n_1-1}^{n_1-2},\mathrm{Id}),(\mathrm{Id},\pi_{n_2-1}^{n_2-2}),(\pi_0^{n_1-1},\mathrm{Id}),(\mathrm{Id},\pi_0^{n_2-1})\Big\}.$$

We let F denote the final zone of $\mathrm{M}_{\{n_1-1\},\{0\}}$, by B the DFA $\mathfrak{Stx}(B_1,B_2)$, by δ the transition function of B, and by $\widehat{B}$ the DFA obtained by restricting B to the accessible states of $\mathrm{M}_{\{n_1-1\},\{0\}}$, *i.e.*, the states T such that, if T has a cross in F, then T has cross at $(0,0)$. We can obtain B by restricting the letters of $\mathrm{M}_{\{n_1-1\},\{0\}}$ to the alphabet Σ'. Therefore, since the equivalence relation $\sim$ is compatible with $\mathrm{M}_{\{n_1-1\},\{0\}}$, it is also compatible with B (see Section 2.4.2 for a definition of *compatible*). Furthermore, the DFA $A = \widehat{B}/_\sim$ is obtained by restricting the letters of $\widehat{\mathrm{M}}_{\{n_1-1\},\{0\}}/_\sim$ to the alphabet Σ'. We show that A is a minimal DFA equivalent to B.

Recall first that every letter of Σ' can be seen as a mapping over the set of all tableaux of size $n_1 \times n_2$. Every word w of Σ' can also be seen as such a mapping: if $w = a_1 \cdots a_m$, for any tableau T of size $n_1 \times n_2$, we let $w(T)$ denote the tableau $a_m \circ \cdots \circ a_1(T)$. Therefore, every word w of Σ' also designates a mapping over the set of all tableaux of size $n_1 \times n_2$. When it exists, we let w^{-1} denote the inverse of the mapping w. For any word $w = a_1 \cdots a_m$, and any $i, j \in \{1,\dots,m\}$ with $i \le j$, we let $w[i,j]$ denote the subword $a_i \cdots a_j$. By convention, if $j = 0$ or $j < i$, $w[i,j] = \varepsilon$. The two following lemmas are proven by a straightforward induction on the length of w.

Lemma 19. *Let w be a word over Σ', and T be a non-empty state of B. Suppose that, for any integer $1 \le k \le |w|$, we have either $(w[1,k](T))_{0,0} = 1$, or $w[1,k](T)$ has no cross in F. Then we have $\delta^w(T) = w(T)$.*

Lemma 20. *For any word w over Σ', for any non-empty tableau T of size $n_1 \times n_2$, for any $(i,j) \in [\![n_1]\!] \times [\![n_2]\!]$, if $(w(T))_{i,j} = 1$, then we have $(\delta^w(T))_{i,j} = 1$, where δ is the transition function of B.*

We now show that the accessible part of B is equal to $\widehat{B}$.

Lemma 21. *All the states of $\widehat{B}$ are accessible.*

Proof. This proof follows the same steps as the proof of Lemma 16. However, we now have to make sure that all the letters we use are in Σ'. We let E denote the set of all tableaux T of size $n_1 \times n_2$ such that, if T has a cross in the final zone of $\mathrm{M}_{\{n_1-1\},\{0\}}$, then $T_{0,0} = 1$. The fact that every accessible tableau T of B is in E comes directly from the formula given for $\mathfrak{Stx}$ in Section 6.1. We now prove that every tableau of E is accessible in B.

Let δ be the transition function of B. For any tableau T of size $n_1 \times n_2$, let $\#_{\mathrm{nf}}T$ be the number of crosses of T which are not in the final zone of $\mathrm{M}_{\{n_1-1\},\{0\}}$. Let $<$ be the strict partial order on tableaux such that $T < T'$ if and only if, either $\#T < \#T'$, or $\#T = \#T'$ and

$\#_{\mathrm{nf}}T < \#_{\mathrm{nf}}T'$.
We prove that every tableau in E is accessible, by induction on non-empty tableaux for the partial order $<$ (the empty tableau is the initial state of $\mathrm{M}_{\{n_1-1\},\{0\}}$, and so it is accessible). The only minimal element for the strict partial order $<$ is the empty tableau. Therefore, for any non-empty tableau $T' \in E$, we define a tableau T such that $T < T'$, and T' is accessible from T. We distinguish the same cases as in the proof of Lemma 16. In order not to repeat ourselves too much, we focus here only on the cases of the proof of Lemma 16 where the letters used are not in Σ'.

- If T' has no cross in the final zone, according the previous remark, we have only to examine the case where $T'_{n_1-1,0} = 0$. Otherwise, we used the letters $((0, n_1 - 1), \mathrm{Id})$, $(\mathrm{Id}, (0, n_2 - 1))$, and $(\mathrm{Id}, \pi_0^{n_2-1})$, which are in Σ'. Let (i, j) be the index of a cross of T'. Let $w = (\mathrm{Id}, (0, 1))((0, \dots, n_1 - 2), \mathrm{Id})^i(\mathrm{Id}, (1, \dots, n_2 - 1))^{j-1}$ and let $T = w^{-1}(T')$. We have $(i, 1) \in (\mathrm{Id}, (1, \dots, n_2 - 1))^{j-1}(T')$, $(1, 1) \in ((0, \dots, n_1 - 2), \mathrm{Id})^i(\mathrm{Id}, (1, \dots, n_2 - 1))^{j-1}(T')$, and therefore $\{0, 0\} \in T$. Furthermore, for all $k \in \{1, \dots, |w|\}$, for all $(i, j) \neq (n_1 - 1, 0)$ such that $i = n_1 - 1$ or $j = 0$, we have $(w[1, k](T))_{i,j} = (w[k+1, |w|]^{-1}(T'))_{i,j} = 0$. Thus, by Lemma 19, we have $\delta^w(T) = T'$. We also have $T < T'$, since $\#T = \#T'$ and $\#_{\mathrm{nf}}T < \#_{\mathrm{nf}}T'$.

- Suppose now that T' has at least one cross in the final zone of $\mathrm{M}_{\{n_1-1\},\{0\}}$. Recall then that $T'_{0,0} = 1$. We consider the following cases.

  - Suppose that T' has a cross in the final zone other than $(0, 0)$. Let (i, j) be such a cross. We consider two cases.

    * If $j = 0$, then $1 \leq i \leq n_1 - 2$. We consider the word $w_1 = ((1, \dots, n_1 - 2), \mathrm{Id})^{i-1}$. Let T'' be the tableau obtained from $w_1^{-1}(T')$ by removing the cross at $(0, 0)$ (*i.e.*, $T'' = w_1^{-1}(T') \setminus \{(0, 0)\}$), let $w_2 = ((0, 1), \mathrm{Id})$, and let $T = w_2(T'')$. We have $(0, 0), (1, 0) \in w_1^{-1}(T)$. As a consequence, $T = ((0, 1), \mathrm{Id})(T'') \setminus \{(0, 1)\}$, and $(0, 0) \in T$. We thus have $T'' = ((0, 1), \mathrm{Id})(T) \cup \{(0, 0)\}$. Furthermore, $(0, 1) \in ((0, 1), \mathrm{Id})(T)$, which implies that $((0, 1), \mathrm{Id})(T)$ is final. Therefore, $T'' = \delta^{((0,1),\mathrm{Id})}(T')$. In addition, every symbol of w_1 fixes $(0, 0)$. Hence, $(0, 0) \in w_1[1, k](T'')$, for any $k \in \{1, \dots, i - 1\}$. Since $T' = w_1(T'')$, by Lemma 19, we have $T' = \delta^{w_1}(T'') = \delta^{w_2 w_1}(T)$. Furthermore, $T < T'$ as $\#T < \#T'$.
    * Otherwise, let $w_1 = ((1, \dots, n_1 - 1), \mathrm{Id})^{i-1}(\mathrm{Id}, (1, \dots, n_2 - 1))^{j-1}$, and let T'' be the tableau obtained from $w_1^{-1}(T')$ by removing the cross at $(0, 0)$, *i.e.*, $T'' = w_1^{-1}(T') \setminus \{(0, 0)\}$. Let $w_2 = ((0, 1), (0, 1))$, and let $T = w_2(T'')$.
      Notice first that $T_{0,0} = 1$, which implies that $T \in E$. Furthermore, since $(w_2 w_1(T))_{0,0} = T'_{0,0} = 1$, by Lemma 20, we have $(\delta^{w_2 w_1}(T))_{0,0} = 1$. Let k be the minimal integer in $\{0, \dots, |w_1|\}$ such that $(\delta^{w_1[1,k]}(T''))_{0,0} = 1$. Notice that, for any integers $\ell \in \{0, \dots, k - 1\}$, the tableau $\delta^{w_1[1,\ell]}(T'')$ has no cross in the final zone. Therefore, by Lemma 19, we have $w_2(w_1[1, l](T)) = \delta^{w_2 \cdot w_1[1,l]}(T)$, where $w_2 \cdot w_1[1, l]$ is the catenation of w_2 and $w_1[1, l]$. Furthermore $\delta^{w_2 \cdot w_1[1,k]}(T) = w_1[1, k](\delta^{w_2}(T)) \cup \{(0, 0)\}$. Hence, since letters of $w_1[k + 1, |w_1|]$ do not change line 0 or column 0, for any $\ell \in \{k, \dots, |w_1|\}$, we have $(\delta^{w_1[k+1,\ell]}(\delta^{w_2 \cdot w_1[1,k]}(T)))_{0,0} = 1$. Hence, from Lemma 19, we have $\delta^{w_2 w_1}(T) = w_1[k+1, |w_1|](w_1[1, k](\delta^{w_2}(T)) \cup \{(0, 0)\}) = w_1(\delta^{w_2}(T)) \cup \{(0, 0)\} = w_1(T'') \cup \{(0, 0)\} = T'$. Moreover, as $\#T < \#T'$, we have $T < T'$.

– Suppose that the only cross of T' in the final zone of $\mathrm{M}_{\{n_1-1\},\{0\}}$ is at $(0,0)$. According to the remark made just before the beginning of the induction, in addition to the case where $T' = \{(0,0)\}$, we only have to consider the case where $T' \neq \{(0,0),(n_1-1,0)\}$, and for all $j \in \{1,\dots,n_2-1\}$, $T'_{0,j} = 0$.

  * If $T' = \{(0,0)\}$, then it is accessible from $\emptyset$ by reading the word $((0,n_1-1),\mathrm{Id})((0,n_1-1),\mathrm{Id})$.
  * Suppose that $T' \neq \{(0,0),(n_1-1,0)\}$, and for all $j \in \{1,\dots,n_2-1\}$, $T'_{0,j} = 0$. It follows that there exists $(i,j) \neq (n_1-1,0)$ such that $i \neq 0$ and $T'_{i,j} = 1$. Let $w = ((1,\dots,n_1-1),\mathrm{Id})^i$ and let $T = w^{-1}(T')$. By Lemma 19, since $(w[1,k](T))_{0,0} = 1$ for any $k \in \{0,\dots,|w|\}$, we have $T \in E$ and $\delta^w(T) = T'$. Furthermore, as $\#T = \#T'$ and $\#_{\mathrm{nf}}T < \#_{\mathrm{nf}}T'$, we have $T < T'$.

□

We now show that any two distinct states of A are distinguishable in A, by proving that any two distinct right-triangle free states of $\widehat{B}$ are distinguishable in $\widehat{B}$. This result is a refinement of Lemma 15, and unsurprisingly, its proof is also a refinement of the proof of Lemma 15. The idea of the proof below is to replace the letters used in Lemma 15 with words over Σ'. In order to make the proof easier, we first state the following lemma, which is proven by a straightforward induction on the length of w.

Lemma 22. *Let w be a word over Σ', and T be a non-empty tableau of size $n_1 \times n_2$. Let $(i,j) \in [\![n_1]\!] \times [\![n_2]\!]$ be a pair of integers such that $T_{i,j} = 1$, and let $(i',j') = w(i,j)$. We have $\delta^w(T)_{i',j'} = 1$.*

Lemma 23. *Any two distinct states of A are distinguishable in A.*

Proof. It is sufficient to prove that any two distinct non-empty right-triangle free tableaux of $\widehat{B}$ are distinguishable in $\widehat{B}$. Recall that a tableau T is final in $\widehat{B}$ if and only if $T \cap F \neq \emptyset$. We use this statement repeatedly in the rest of the proof. Let T and T' be two distinct non-empty right-triangle free tableaux of size $n_1 \times n_2$. We show that T and T' are distinguishable in $\widehat{B}$. Let (i,j) be such that $T_{i,j} \neq T'_{i,j}$. We suppose that $T_{i,j} = 1$ (the case $T'_{i,j} = 1$ is symmetrical). We distinguish the same cases as in the proof of Lemma 15.

Suppose now that for all $i' \in [\![n_1]\!]$, we have $T'_{i',j} = 0$.

- If $j = 0$, then T' is not final, as T' does not have any cross on column 0 or on line $n_1 - 1$. The same thing can be said of $(\pi_0^{n_1-1},\mathrm{Id})(T')$. As a consequence, $\delta^{(\pi_0^{n_1-1},\mathrm{Id})}(T')$ is not final in $\widehat{B}$. Furthermore, $(\pi_0^{n_1-1},\mathrm{Id})(i,j)$ is equal to $(i,0)$ if $i < n_1 - 1$, and to $(0,0)$ if $i = n_1 - 1$. Therefore, by Lemma 22, $\delta^{(\pi_0^{n_1-1},\mathrm{Id})}(T)$ is final in $\widehat{B}$. As a consequence, T and T' are distinguishable in $\widehat{B}$.

- If $j \neq 0$, then let $w = (\pi_0^{n_1-1},\mathrm{Id})(\mathrm{Id},(1,\dots,n_2-1))^{n_2-j}(\mathrm{Id},(0,1))$. Let $T'' = (\pi_0^{n_1-1},\mathrm{Id})(T')$. If $T'_{0,0} = 1$ or $T'_{n_1-1,0} = 1$, then $T''_{0,0} = 1$. Furthermore, if $T'_{0,0} = 0$ and $T'_{n_1-1,0} = 0$, then T' does not have any cross on line $n_1 - 1$ or on column 0. Therefore, T'' does not have any cross on line $n_1 - 1$ or on column 0 either. In both cases, we thus have $T'' = \delta^{(\pi_0^{n_1-1},\mathrm{Id})}(T')$.

Notice that, since T'' does not have any cross on line $n_1 - 1$, for any $k \in \mathbb{N}$, $(\mathrm{Id}, (1, \dots, n_2 - 1))^k(T'')$ does not have any cross on line $n_1 - 1$ either. Furthermore, $(\mathrm{Id}, (1, \dots, n_2 - 1))$ does not change column 0. Therefore, either T'' has a cross at $(0, 0)$ and $(\mathrm{Id}, (1, \dots, n_2 - 1))^k(T'')$ is final in $\widehat{B}$ for any $k \in \mathbb{N}$, or T'' does not have any cross either on column 0 or on line $n_1 - 1$, and $(\mathrm{Id}, (1, \dots, n_2 - 1))^k(T'')$ is not final in $\widehat{B}$ for any $k \in \mathbb{N}$. In both cases, by Lemma 19, we have $(\mathrm{Id}, (1, \dots, n_2 - 1))^{n_2 - j}(T'') = \delta^{(\mathrm{Id}, (1, \dots, n_2 - 1))^{n_2 - j}}(T'')$. Furthermore, T'' does not have any cross on column j, and recall that it does not have any cross on line $n_1 - 1$ either. As a consequence, $\delta^{(\mathrm{Id}, (1, \dots, n_2 - 1))^{n_2 - j}}(T'')$ does not have any cross on column 1 or on line $n_1 - 1$. Therefore, $(\mathrm{Id}, (0, 1))((\delta^{(\mathrm{Id}, (1, \dots, n_2 - 1))^{n_2 - j}}(T''))$ does not have any cross either on column 0 or on line $n_1 - 1$. Hence, $(\mathrm{Id}, (0, 1))((\delta^{(\mathrm{Id}, (1, \dots, n_2 - 1))^{n_2 - j}}(T'')) = \delta^{(\mathrm{Id}, (0,1))(\mathrm{Id}, (1, \dots, n_2 - 1))^{n_2 - j}}(T'')$. To summarize, $\delta^w(T') = w(T')$, and $\delta^w(T')$ is not final in $\widehat{B}$. However, $T_{i,j} = 1$, $w(i, j) = (0, 0)$ if $i = n_1 - 1$, and $w(i, j) = (i, 0)$ if $1 \leq i \leq n_1 - 1$. Therefore, by Lemma 22, we have $(\delta^w(T))_{0,0} = 1$, and $\delta^w(T)$ is final in $\widehat{B}$. The tableaux T and T' are thus distinguishable in $\widehat{B}$.

We now suppose that there exists $x \in [\![n_1]\!]$ such that $T'_{x,j} = 1$. Let $\{i_1, \dots, i_\ell\} = \{\alpha \mid T'_{\alpha,j} = 1\}$ and let $\{j_1, \dots, j_p\} = \{\beta \mid T'_{i_1,\beta} = 1\}$. We state the same two properties that we noticed in the proof of Lemma 15. We designate them by Property 1 and Property 2 in the rest of the proof.

1. By Lemma 12, lines $i_1, \dots, i_\ell$ are the same, as they all have a cross on the column j. Columns $j_1, \dots, j_p$ are also the same, as they all have a cross on line i_1. It follows that, if
$$(i', j') \in \Big(\{i_1, \dots, i_\ell\} \times \Big(\{0, \dots, n_2 - 1\} \setminus \{j_1, \dots, j_p\}\Big)\Big) \cup \Big((\{0, \dots, n_1 - 1\} \setminus \{i_1, \dots, i_\ell\}) \times \{j_1, \dots, j_p\}\Big),$$
then we have $T'_{i',j'} = 0$.
2. We have $j \in \{j_1, \dots, j_p\}$ and $i \notin \{i_1, \dots, i_\ell\}$.

The same reasoning as in Lemma 15 can be made. We let (f, g) denote the pair of mappings such that, for any $(i', j') \in [\![n_1]\!] \times [\![n_2]\!]$, we have
$$f(i') = \begin{cases} n_1 - 1 & \text{if } i' \in \{i_1, \dots, i_\ell\} \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad g(j') = \begin{cases} 0 & \text{if } j' \in \{j_1, \dots, j_p\} \\ n_2 - 1 & \text{otherwise.} \end{cases}$$
If $(f(i'), g(j'))$ is in F, then
$$(i', j') \in \Big(\{i_1, \dots, i_\ell\} \times \Big(\{0, \dots, n_2 - 1\} \setminus \{j_1, \dots, j_p\}\Big)\Big) \cup \Big((\{0, \dots, n_1 - 1\} \setminus \{i_1, \dots, i_\ell\}) \times \{j_1, \dots, j_p\}\Big).$$
Hence, by Property 1, we have $T'_{i',j'} = 0$. As a consequence, $(f, g)(T')$ has at most two crosses, one at $(n_1 - 1, 0)$ and one at $(0, n_2 - 1)$. Neither $(n_1 - 1, 0)$ nor $(0, n_2 - 1)$ is in the final

zone of $\mathrm{M}_{\{n_1-1\},\{0\}}$. As a consequence, $(f,g)(T') = \delta^{(f,g)}(T')$ is not final in $\mathrm{M}_{\{n_1-1\},\{0\}}$. However, by Property 2, since $T_{i,j} = 1$, we have $((f,g)(T))_{0,0} = 1$. Therefore, $(f,g)(T) = \delta^{(f,g)}(T)$ is final in $\mathrm{M}_{\{n_1-1\},\{0\}}$. We now simulate the letter (f,g) with letters in Σ'. More precisely, we show that there exists a word w over Σ' such that $\delta^w(T') = (f,g)(T')$, and such that $a_m \circ \cdots \circ a_1 = (f,g)$, where $a_1 \cdots a_m = w$. If w is such a word, from Lemma 22, we have $(\delta^w(T))_{0,0} = 1$, which implies that $\delta^w(T)$ is final in $\widehat{B}$. Proving the existence of such a word w is therefore sufficient to conclude our proof.

We consider two main cases. In the first case, the word used to simulate (f,g) is $w_1w_2w_3w_4w_5$. In the second case, we read a word w_0 from the two tableaux T and T', so that the two resulting tableaux fall into the scope of the first case.

Let $\{k_1,\dots,k_s\} = \{i_1,\dots,i_l\} \setminus \{0, n_1-1\}$, $\{k'_1,\dots,k'_{n_1-2-s}\} = [\![n_1]\!] \setminus \{0, n_1-1, k_1,\dots,k_s\}$, $\{c_1,\dots,c_d\} = \{j_1,\dots,j_p\} \setminus \{0, n_2-1\}$, and $\{c'_1,\dots,c'_{n_2-2-d}\} = [\![n_2]\!] \setminus \{0, n_2-1, c_1,\dots,c_d\}$. Recall that $[x \in \{i_1,\dots,i_l\}]$ is equal to 1 if $x \in \{i_1,\dots,i_l\}$, and 0 otherwise.

1. $(0,0) \in T'$. In this case, we define a five words w_1, w_2, w_3, w_4, w_5, and we let w denote the word $w_1w_2w_3w_4w_5$.

   (a) $0 \in \{i_1,\dots,i_l\}$ and $0 \in \{j_1,\dots,j_p\}$

$$w_1 = \prod_{v=1}^{s} \Big[((1,\dots,n_1-2),\mathrm{Id})^{n_1-k_v-1}(\pi_0^1,\mathrm{Id})((1,\dots,n_1-2),\mathrm{Id})^{k_v-1}\Big] \\ (\pi_0^{n_1-1},\mathrm{Id})^{[(n_1-1)\in\{i_1,\dots,i_l\}]}.$$

   The word w_1 does cyclic permutations on the lines so that the lines $\{k_1,\dots,k_s\}$ which are the same as line 0 are permuted one after the other on line 1 and then merged with line 0. The last line is merged with line 0 if it is in $\{i_1,\dots,i_l\}$.

$$w_2 = \prod_{v=1}^{d} \Big[(\mathrm{Id},(1,\dots,n_2-2))^{n_2-c_v-1}(\mathrm{Id},\pi_0^1)(\mathrm{Id},(1,\dots,n_2-2))^{c_v-1}\Big] \\ (\mathrm{Id},\pi_0^{n_2-1})^{[(n_2-1)\in\{j_1,\dots,j_p\}]}.$$

   The word w_2 does the same with columns $\{c_1,\dots,c_d\}$, and with the last column.

$$w_3 = \prod_{v=1}^{n_1-2-s} ((1,\dots,n_1-2),\mathrm{Id})^{n_1-k'_v-2}(\pi_{n_1-1}^{n_1-2},\mathrm{Id})((1,\dots,n_1-2),\mathrm{Id})^{k'_v}.$$

   The word w_3 merges the lines $\{k'_1,\dots,k'_{n_1-2-s}\}$ with line n_1-1, but they are not necessarily the same.

$$w_4 = \prod_{v=1}^{n_2-2-d} (\mathrm{Id},(1,\dots,n_2-2))^{n_2-c'_v-2}(\mathrm{Id},\pi_{n_2-1}^{n_2-2})(\mathrm{Id},(1,\dots,n_2-2))^{c'_v}.$$

   The word w_4 merges the columns $\{c'_1,\dots,c'_{n_2-2-d}\}$ with column n_2-1.

$$w_5 = ((0,n_1-1),\mathrm{Id}).$$

(b) $0 \notin \{i_1, \ldots, i_l\}$ and $0 \notin \{j_1, \ldots, j_p\}$.

$$w_1 = \prod_{v=1}^{n_1-2-s} \left[((1, \ldots, n_1 - 2), \mathrm{Id})^{n_1-k'_v-1}(\pi_0^1, \mathrm{Id})(1, \ldots, n_1 - 2), \mathrm{Id})^{k'_v-1}\right] (\pi_0^{n_1-1}, \mathrm{Id})^{[(n_1-1)\notin\{i_1,\ldots,i_l\}]}.$$

$$w_2 = \prod_{v=1}^{n_2-2-d} \left[(\mathrm{Id}, (1, \ldots, n_2 - 2))^{n_2-c'_v-1}(\mathrm{Id}, \pi_0^1)(\mathrm{Id}, (1, \ldots, n_2 - 2))^{c'_v-1}\right] (\mathrm{Id}, \pi_0^{n_2-1})^{[(n_2-1)\notin\{j_1,\ldots,j_p\}]}.$$

$$w_3 = \prod_{v=1}^{s}((1, \ldots, n_1 - 2), \mathrm{Id})^{n_1-k_v-2}(\pi_{n_1-1}^{n_1-2}, \mathrm{Id})(1, \ldots, n_1 - 2), \mathrm{Id})^{k_v}.$$

$$w_4 = \prod_{v=1}^{d}(\mathrm{Id}, (1, \ldots, n_2 - 2))^{n_2-c_v-2}(\mathrm{Id}, \pi_{n_2-1}^{n_2-2})(\mathrm{Id}, (1, \ldots, n_2 - 2))^{c_v}.$$

$$w_5 = (\mathrm{Id}, (0, n_2 - 1)).$$

2. $(0, 0) \notin T'$. In every case except for case 2(a), we find a word w_0 such that the two tableaux $\delta^{w_0}(T)$ and $\delta^{w_0}(T')$ fall into the scope of case 1.

   (a) $\{i_1, \ldots, i_l\} = \{n_1 - 1\}$.
   This implies that $j = 0, 0 \leq i \leq n_1-2$, and $\{j_1, \ldots, j_p\} = \{0\}$. Let $w_0 = ((0, n_1-1), \mathrm{Id})$, $T_1 = \delta^{w_0}(T)$, and $T'_1 = \delta^{w_0}(T')$. Since $(0, 0) \in w_0(T')$, we have $T'_1 = w_0(T')$. Hence, by Lemma 22, T_1 has a cross at the coordinates $w_0(i, j)$, while T'_1 does not. Therefore, the tableaux T_1 and T'_1 fall into the scope of case 1(a).

   (b) $\{i_1, \ldots, i_l\} \neq \{n_1 - 1\}$.
   This implies that there exists $x \in \{i_1, \ldots, i_l\}$ such that $x \in \{0, \ldots, n_1 - 2\}$. Furthermore, since $(0, 0) \notin T'$, we have $1 \leq j_1$.

   - If $x = 0$, let $w_0 = (\mathrm{Id}, (1, \ldots, n_2 - 1))^{n_1-j_1-1}(\mathrm{Id}, (1, 0))$. The tableau T' has no crosses in F since $(0, 0) \notin T'$. Furthermore, the letter $(\mathrm{Id}, (1, \ldots, n_2 - 1))$ does not change the column 0 or line $n_1 - 1$ if they are empty. In addition, $w_0(i_1, j_1) = (0, 0)$, which implies, by Lemma 22, that $w_0(T)_{0,0} = 1$. Thus, by Lemma 19, we have $\delta^{w_0}(T') = w_0(T')$. Let $T_1 = \delta^{w_0}(T)$ and $T'_1 = \delta^{w_0}(T') = w_0(T')$. Furthermore, by Lemma 22, T_1 has a cross at the coordinates $w_0(i, j) \neq (0, 0)$, while $T'_1 = w_0(T')$ does not, since $(i, j) \notin T'$. Therefore, the tableaux T_1 and T'_1 fall into the scope of case 1.
   - If $x \neq 0$, then $n_1 \geq 3$. Let

     $$w_0 = ((1, \ldots, n_1 - 2), \mathrm{Id})^{n_1-x-1}((1, 0), \mathrm{Id})(\mathrm{Id}, (1, \ldots, n_2 - 1))^{n_1-j_1-1}(\mathrm{Id}, (1, 0)).$$

     The tableau T' has no crosses in F since $(0, 0) \notin T'$. Furthermore, the letters $((1, \ldots, n_1 - 2), \mathrm{Id})$, $((1, 0), \mathrm{Id}$, and $(\mathrm{Id}, (1, \ldots, n_2 - 1))$ do not change the column 0 or line $n_1 - 1$ if they are empty. In addition, $w_0(i_1, j_1) = (0, 0)$, which implies, by Lemma 22, that $w_0(T)_{0,0} = 1$. Thus, by Lemma 19, we have

$\delta^{w_0}(T') = w_0(T')$. Let $T_1 = \delta^{w_0}(T)$ and $T'_1 = \delta^{w_0}(T') = w_0(T')$. Furthermore, by Lemma 22, T_1 has a cross at the coordinates $w_0(i,j) \neq (0,0)$, while $T'_1 = w_0(T')$ does not, since $(i,j) \notin T'$. Therefore, the tableaux T_1 and T'_1 fall into the scope of case 1.

From a direct computation, we get $a_m \circ \cdots \circ a_1 = (f,g)$, where $a_1 \cdots a_m = w_1w_2w_3w_4w_5$. Furthermore, in case 1, for all $k \in \{0,\dots,|w|-1\}$, we have $w[1,k](T')_{0,0} = 1$. Furthermore, $w[1,|w|-1](T') \subseteq \{(0,0),(n_1-1,n_2-1)\}$. As a consequence, $w(T') \cap F = \emptyset$. Therefore, from Lemma 19, in case 1, we have $\delta^w(T') = (f,g)(T')$. Hence, $\delta^w(T')$ is not final in $\widehat{B}$. Furthermore, since $T'_{i,j} = 0$, we have $i \notin \{i_1,\dots,i_l\}$, and $j \in \{j_1,\dots,j_p\}$, which implies that $w(i,j) = (0,0)$. Therefore, from Lemma 22, we have $(\delta^w(T))_{0,0} = 1$, which implies that $\delta^w(T)$ is final in $\widehat{B}$. Therefore, in case 1, T and T' are distinguishable in $\widehat{B}$. Furthermore, in case 2, we have shown that the two tableaux $\delta^{w_0}(T)$ and $\delta^{w_0}(T')$ fall into the scope of case 1. Therefore, in these cases, $\delta^{w_0}(T)$ and $\delta^{w_0}(T')$ are distinguishable in $\widehat{B}$. As a consequence, in all cases, T and T' are distinguishable in $\widehat{B}$. To conclude, any two distinct states of A are distinguishable. □

Lemma 21 and Lemma 23 imply that A is minimal. However, A has the same size as $\widehat{\mathrm{M}}_{\{n_1-1\},\{0\}}/_{\overset{*}{\leftrightarrow}}$. Hence, by Theorem 3 A is witness for $\circledast$.

Theorem 4. *The operation $\circledast$ has a witness with sixteen letters. More precisely, for any integers n_1 and n_2 greater than or equal to 2, we have $\mathrm{sc}_{\circledast}(n_1,n_2) = \mathrm{sc}(\circledast(\mathrm{L}(B_1),\mathrm{L}(B_2)))$.*

6.7 Towards the general case

Our guess is that the method we use to compute the state complexity of the star of the symmetric difference can be generalized to compute the state complexity of the star of every boolean operations. However, the aim of this section is not to describe precisely how to generalize it, but just to give a few thoughts.

First notice that, since the Kleene star and boolean operations are 1-uniform (see Example 15 and Example 10), and since 1-uniform operations are closed under composition by Proposition 13, for any boolean function $\mathbf{b}$, the regular operation $\circledast_{\mathbf{b}} = (\otimes_{\mathbf{b}}(L_1,\dots,L_k))^\star$ is 1-uniform. Therefore, all these operations fall into the scope of our framework. A formula for the modifier $\mathfrak{Star} \circ \mathfrak{m}_{\mathbf{b}}$ can be obtained directly from Example 15, Example 17, and Definition 6, and is remarkably similar to the case of the symmetric difference (Section 6.1).

Let $\mathbf{b}$ be a k-ary boolean function, and let $(A_1,\dots,A_k)$ be a k-tuple of DFAs with $A_j = (\Sigma, Q_j, i_j, F_j, \delta_j)$. Recall that, if a is a letter of Σ and E a subset of $Q_1 \times \cdots \times Q_k$, we let $(\delta_1^a,\dots,\delta_k^a)(E)$ denote the set $\{(\delta_1^a(q_1),\dots,\delta_k^a(q_k)) \mid (q_1,\dots,q_k) \in E\}$. We have

$$\mathfrak{Star} \circ \mathfrak{m}_{\mathbf{b}}(A_1,\dots,A_k) = (\Sigma, 2^{Q_1\times\cdots\times Q_k}, \emptyset, \{E \subseteq Q_1 \times \cdots \times Q_k \mid E \cap F \neq \emptyset\} \cup \{\emptyset\}, \delta)$$

where F is the set of all elements $(q_1,\dots,q_k)$ of $Q_1 \times \cdots \times Q_k$ that satisfy $\mathbf{b}([q_1 \in F_1],\dots,[q_k \in$

$F_k]) = 1$, and where, for any letter a of Σ, we have

$$\delta^a(\emptyset) = \begin{cases} \{(\delta_1^a(i_1), \dots, \delta_k^a(i_k))\}, & \text{if } (\delta_1^a(i_1), \dots, \delta_k^a(i_k)) \notin F; \\ \{(\delta_1^a(i_1), \dots, \delta_k^a(i_k)), (i_1, \dots, i_k)\}, & \text{otherwise ;} \end{cases}$$

and for any $E \in 2^{Q_1 \times \cdots \times Q_k} \setminus \{\emptyset\}$, we have

$$\delta^a(E) = \begin{cases} (\delta_1^a, \dots, \delta_k^a)(E), & \text{if } (\delta_1^a, \dots, \delta_k^a)(E) \cap F = \emptyset; \\ (\delta_1^a, \dots, \delta_k^a)(E) \cup \{(i_1, \dots, i_k)\}, & \text{otherwise.} \end{cases}$$

Notice that, if $\mathbf{b}$ is a binary boolean operation, the states of $\mathfrak{Star}(\mathfrak{m}_{\mathbf{b}}(\mathrm{Mon}_{n_1,\dots,n_k}^{F_1,\dots,F_k}))$ are still labelled by tableaux. What essentially changes between two binary boolean operations in the above formula is what we called the "final zone" in previous sections, *i.e.*, the set F of all elements $(q_1, \dots, q_k)$ of $Q_1 \times \cdots \times Q_k$ such that $\mathbf{b}([q_1 \in F_1], \dots, [q_k \in F_k]) = 1$ (in the case of the symmetric difference, we had $F = (F_1 \times Q_2)\Delta(Q_1 \times F_2)$). In that sense, the general case where $\mathbf{b}$ is any k-ary boolean operation is similar to the binary case, albeit with a generalization of tableaux to higher dimensions: a tableaux of size $n_1 \times \cdots \times n_k$ is a subset of $[\![n_1]\!] \times \cdots \times [\![n_k]\!]$.

In the remainder of this section, $\mathbf{b}$ is a boolean function, $(n_1, \dots, n_k)$ is a k-tuple of integers greater than or equal to 2, and $(F_1, \dots, F_k)$ is a k-tuple of sets such that $F_j \subseteq [\![n_j]\!]$ and $F_j \notin \{\emptyset, [\![n_j]\!]\}$, for any $j \in \{1, \dots, k\}$. Our idea is to link the Nerode equivalence of $A = \mathfrak{Star}(\mathfrak{m}_{\mathbf{b}}(\mathrm{Mon}_{n_1,\dots,n_k}^{F_1,\dots,F_k}))$ to the Nerode equivalence of the DFA $B = \mathfrak{Star}(\mathfrak{m}_{\mathbf{b}}(\mathrm{Mon}_{2,\dots,2}^{G_1,\dots,G_k}))$, where, for any $j \in \{1, \dots, k\}$, we have

$$G_j = \begin{cases} \{0\} & \text{if} \quad 0 \in F_j \text{ and } F_j \neq [\![n_j]\!] \\ \{1\} & \text{if} \quad 0 \notin F_j \text{ and } F_j \notin \{\emptyset, [\![n_j]\!]\}. \end{cases}$$

We defined the DFA B so that it would "mimic" the DFA A, but with a much smaller size. We speculate that the Nerode equivalence of B has enough information, so that we may build upon it the Nerode equivalence of A. Using this idea, we begin to generalize the reasoning presented in the previous sections of this chapter. This leads to Claim 1, a generalization of Proposition 22.

We do not want the ideas introduced in this section to be sunk in an ocean of cumbersome proofs and notations. Therefore, we state Claim 1 without proof. The proof of Claim 1 is not conceptually difficult and highly resembles the proof of Proposition 22; it will be published in further work. It is worth noting that in all of the known cases (star of intersection, star of symmetric difference, star of multiple unions), the upper bound that can be naturally derived from Claim 1 is the actual state complexity of the operation considered. Even though there is no reason to expect this to hold true in the general case, Claim 1 could be used to compute non-trivial upper-bounds for the state complexity of $\circledast_{\mathbf{b}}$, for some boolean functions $\mathbf{b}$.

The first step to generalize Proposition 22 is to generalize the equivalence relation $\overset{*}{\leftrightarrow}$ defined in Definition 25. We do not generalize $\leftrightarrow$ directly, but we define another equivalence relation $\bowtie$ such that its reflexive, symmetric and transitive closure $\overset{*}{\bowtie}$ coincides with $\overset{*}{\leftrightarrow}$. To that aim, we introduce a way to link tableaux of size $n_1 \times \cdots \times n_k$ to tableaux of size $2 \times \cdots \times 2$.

Definition 27. *A* cube *is a subset C of $[\![n_1]\!] \times \cdots \times [\![n_k]\!]$ such that there exists two k-tuples $(a_1, \ldots, a_k)$ and $(b_1, \ldots, b_k)$ in $[\![n_1]\!] \times \cdots \times [\![n_k]\!]$, with $a_j < b_j$ for any $j \in \{1, \ldots, k\}$, that satisfy $C = \{a_1, b_1\} \times \cdots \times \{a_k, b_k\}$.*

For any cube $C = \{a_1, b_1\} \times \cdots \times \{a_k, b_k\}$, with $a_j < b_j$ for any $j \in \{1, \ldots, k\}$, we let by ξ^C denote the k-tuple $(\xi_1^C, \ldots, \xi_k^C)$, where ξ_j^C is the mapping from $\{a_j, b_j\}$ to $\{0, 1\}$ such that $\xi_j^C(b_j) = 1$ and $\xi_j^C(a_j) = 0$.

Definition 28. *For any cube $C = \{a_1, b_1\} \times \cdots \times \{a_k, b_k\}$, with $a_j < b_j$ for any $j \in \{1, \ldots, k\}$, and any tableau T of size $n_1 \times \cdots \times n_k$, we let T^C denote the tableau of size $2 \times \cdots \times 2$ such that $T^C = \xi^C(T \cap C)$.*

Example 22. *We represent T^C in Figure 6.10, where T is the tableau of size 5×5 represented in Figure 6.10, and C is the cube represented by the red squares of Figure 6.10.*

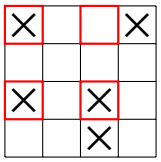


Figure 6.10: A tableau T of size 5×5.

Figure 6.11: The tableau T^C, where $C = \{(0,0), (0,2), (2,0), (2,2)\}$;

Definition 29. *We let $\bowtie$ denote the relation over the set of non-empty tableaux of size $n_1 \times \cdots \times n_k$ such that, for any two non-empty tableaux T and T' of size $n_1 \times \cdots \times n_k$, we have $T \bowtie T'$ if and only if there exists a cube C that satisfies*

- *the tableaux T^C and T'^C are indistinguishable in B,*
- *for all $\ell \notin C$, we have $\ell \in T$ if and only if $\ell \in T'$.*

Notice that, in the case where $\mathbf{b}$ is the symmetric difference, we have $T \bowtie T'$ if and only if there exists a cube C such that T^C and T'^C are either equal, or both of size at least 3. Therefore, if $T \to T'$, then $T \bowtie T'$. Furthermore, if $\#T^C = \#T'^C = 4$, then $T^C = T'^C$. Therefore, if $T \bowtie T'$, then we have

$$\begin{cases} T' = T & \text{if } \#T^C = \#T'^C = 4 \\ T' \to T & \text{if } \#T'^C = 3 \text{ and } \#T^C = 4 \\ T \to T' & \text{if } \#T'^C = 4 \text{ and } \#T^C = 3 \\ T \to (T \cup C) \text{ and } T' \to (T' \cup C) & \text{if } \#T^C = 3 \text{ and } \#T'^C = 3 \end{cases}$$

However, we have $T \cup C = T' \cup C$. Thus, if $T \bowtie T'$, then $T \overset{*}{\leftrightarrow} T'$. Hence, $\overset{*}{\bowtie}$ is indeed equal to $\overset{*}{\leftrightarrow}$ when $\mathbf{b}$ is the symmetric difference.

It is also worth noting here that, when $\mathbf{b}$ is the union of k languages, $A/_{\overset{*}{\bowtie}}$ provides the construction used in [17] to describe $\circledast_{\mathbf{b}}$, and later on to compute its state complexity. Furthermore, when $\mathbf{b}$ is the intersection of two languages, we have $T \overset{*}{\bowtie} T'$ if and only if

$T = T'$. As a consequence, $A/_{\overset{*}{\bowtie}}$ also provides the construction used to compute the state complexity of $\circledast_{\mathbf{b}}$ [27]. Therefore, studying $A/_{\overset{*}{\bowtie}}$ may be a good starting point to study the state complexity of $\circledast_{\mathbf{b}}$, for any boolean operation $\mathbf{b}$.

Now we state without proof a generalization of Proposition 22.

Claim 1. *For any two non-empty tableaux T and T' of size $n_1 \times \cdots \times n_k$, if $T \overset{*}{\bowtie} T'$, then T and T' are indistinguishable in A.*

The above claim is a generalization of Proposition 22. Claim 1 could be used to compute non-trivial upper bounds for the state complexities of the star of other boolean operations. For example, we believe it is possible to obtain Theorem 3.1 of [17] (the upper bound part of the state complexity) by using this proposition.

We do not yet have a complete proof that would allow us to compute the state complexity of the star of every boolean operation. We suspect that Lemma 16 cannot be easily generalized as is. Furthermore, if we want to carry on our reasoning, we have to understand better why Lemma 18 holds true, because our proof seems too technical to be generalized. Finally, if we want to push our reasoning to the end, we have to find a way to generalize Corollary 20 and Proposition 22 of [8], which seems to be a difficult problem of combinatorics. We do believe, nonetheless, that it would not be very difficult to compute the state complexity of the star of union [16] and the state complexity of the star of intersection [27], by beginning from Proposition 1, and by following the steps taken from Section 6.3 onwards. However, the proofs brought about by these reasonings would be of little interest, as they would be very similar to those already existing.

We end this section with a table of the state complexity of the star of every binary boolean operation, Table 6.1, which is made possible by combining the values of the state complexity of the star of union [16], the state complexity of the star of intersection [27], and the state complexity of the star of symmetric difference.

<table>
<tr><td rowspan="2">The binary boolean operation b</td><td colspan="3">The state complexity of $\circledast_{\mathbf{b}}$, i.e., $\mathrm{sc}_{\circledast_{\mathbf{b}}}(n_1, n_2)$</td></tr>
<tr><td>$n_1 = 1, n_2 \geq 2$</td><td>$n_1 \geq 2, n_2 = 1$</td><td>$n_1 \geq 2, n_2 \geq 2$</td></tr>
<tr><td>$\emptyset$</td><td>2</td><td>2</td><td>2</td></tr>
<tr><td>$L_1^\star$</td><td>2</td><td>$2^{n_1-1} + 2^{n_1-2}$</td><td>$2^{n_1-1} + 2^{n_1-2}$</td></tr>
<tr><td>$L_2^\star$</td><td>$2^{n_2-1} + 2^{n_2-2}$</td><td>2</td><td>$2^{n_2-1} + 2^{n_2-2}$</td></tr>
<tr><td>$(L_1 \cap L_2)^\star$</td><td colspan="3" rowspan="4">$2^{n_1 n_2-1} + 2^{n_1 n_2-2}$</td></tr>
<tr><td>$(L_1 \cap L_2^c)^\star$</td></tr>
<tr><td>$(L_1^c \cap L_2)^\star$</td></tr>
<tr><td>$(L_1^c \cap L_2^c)^\star$</td></tr>
<tr><td>$(L_1 \cup L_2)^\star$</td><td rowspan="8">$2^{n_2-1} + 2^{n_2-2}$</td><td rowspan="8">$2^{n_1-1} + 2^{n_1-2}$</td><td rowspan="4">$2^{n_1+n_2-1} - 2^{n_1-1} - 2^{n_2-1} + 1$</td></tr>
<tr><td>$(L_1^c \cup L_2)^\star$</td></tr>
<tr><td>$(L_1 \cup L_2^c)^\star$</td></tr>
<tr><td>$(L_1^c \cup L_2^c)^\star$</td></tr>
<tr><td>$(L_1 \Delta L_2)^\star$</td><td rowspan="4">$2\alpha_{n_1-1,n_2-1} + \alpha'_{n_1,n_2} - 1$</td></tr>
<tr><td>$(L_1^c \Delta L_2)^\star$</td></tr>
<tr><td>$(L_1 \Delta L_2^c)^\star$</td></tr>
<tr><td>$(L_1^c \Delta L_2^c)^\star$</td></tr>
<tr><td>$\Sigma^\star$</td><td>1</td><td>1</td><td>1</td></tr>
</table>

Table 6.1: The state complexities of the star of the 16 binary boolean operations.

Chapter 7

Friendly and product modifiers

In this chapter, we examine two suboperads of modifiers defined by simple algebraic properties: product modifiers and friendly modifiers. We first characterize the operations described by friendly modifiers, and we explore their algebraic structure. We then show that we can fit into this structure the algebraic structure of product modifiers. Finally, we show some state complexity results for friendly and product modifiers.

7.1 Friendly modifiers

In this section, we study the structure of the operad of coherent friendly modifiers $(\mathfrak{M}_c^f, \circ)$. We prove that we can change every friendly modifier into a "standard" form, without changing the operation it describes. Furthermore, we define *friendly operations* as the composition of a generalized version of boolean operations and some roots. We prove that the set of friendly operations $\mathfrak{O}^f$ is exactly the set of all the operations described by friendly modifiers. Furthermore, we study the algebraic structure behind coherent friendly modifiers, with respect to the composition of operations, in detail. We define a quotient operad of $(\mathfrak{M}_c^f, \circ)$ that is isomorphic to $(\mathfrak{O}^f, \circ)$. We also prove both of these operads are isomorphic to an operad equal to the set of all k-tuples of eventually periodic sequences with values in $\{0, 1\}$, equipped with some binary operation $\odot$.

7.1.1 Friendly modifiers: an operad

Definition 30. *We say that a k-modifier* $\mathfrak{m} = [\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$ *is* friendly *if, for any two k-tuples of transition configurations* $(Q_1, i_1, F_1, \phi_1), \ldots, (Q_k, i_k, F_k, \phi_k)$ *and* $(Q_1, i_1, F_1, \psi_1), \ldots, (Q_k, i_k, F_k, \psi_k)$*, we have*

$$\mathfrak{d}((i_1, F_1, \phi_1 \circ \psi_1), \ldots, (i_k, F_k, \phi_k \circ \psi_k)) =$$
$$\mathfrak{d}((i_1, F_1, \phi_1), \ldots, (i_k, F_k, \phi_k)) \circ \mathfrak{d}((i_1, F_1, \psi_1), \ldots, (i_k, F_k, \psi_k))$$

The idea of the above definition is that $\mathfrak{d}$ is a morphism of monoids with respect to its third coordinate. For instance, it is easy to see that the modifier $\mathfrak{Sqrt}$ is friendly, and that any modifier $\mathfrak{m}_\mathbf{b}$, where $\mathbf{b}$ is a boolean function, is friendly.

Lemma 24. *The set of friendly modifiers is stable by the composition of operations* $\circ$.

Proof. Let $\mathfrak{m}_1 = [\mathfrak{Q}_1, \mathfrak{i}_1, \mathfrak{f}_1, \mathfrak{d}_1]$ and $\mathfrak{m}_2 = [\mathfrak{Q}_2, \mathfrak{i}_2, \mathfrak{f}_2, \mathfrak{d}_2]$ be two friendly modifiers, respectively k-ary and k'-ary, let $j \leq k$ be a positive integer, and let $\mathfrak{m}_1 \circ_j \mathfrak{m}_2 = [\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$. For any $k + k' - 1$-tuple of transition configurations $(t_1, \dots, t_{k+k'-1})$ with $t_j = (Q_j, i_j, F_j, \delta_j)$ for any $j \in \{1, \dots, k + k' - 1\}$, we have

$$\begin{aligned}\mathfrak{d}(t_1, \dots, t_{k+k'-1}) = \mathfrak{d}_1(t_1, \dots, t_{j-1}, (\mathfrak{Q}_2(t_j, \dots, t_{k'+j-1}), \mathfrak{i}_2(t_j, \dots, t_{k'+j-1}),\\ \mathfrak{f}_2(t_j, \dots, t_{k'+j-1}), \mathfrak{d}_2(t_j, \dots, t_{k'+j-1})), t_{k'+j}, \dots, t_{k+k'-1}).\end{aligned} \tag{7.1}$$

However, for any two $k + k' - 1$-tuples of transition configurations $(t_1, \dots, t_{k+k'-1})$ and $(t'_1, \dots, t'_{k+k'-1})$, with $t_j = (Q_j, i_j, F_j, \phi_j)$ and $t'_j = (Q_j, i_j, F_j, \phi'_j)$ for any $j \in \{1, \dots, k + k' - 1\}$, we have, since $\mathfrak{m}_2$ is friendly,

$$\begin{aligned}\mathfrak{d}_2(t''_j, \dots, t''_{k'+j-1}) =\\ \mathfrak{d}_2(t_j, \dots, t_{k'+j-1}) \circ \mathfrak{d}_2(t'_j, \dots, t'_{k'+j-1}),\end{aligned} \tag{7.2}$$

where $t''_\ell = (Q_\ell, i_\ell, F_\ell, \phi_\ell \circ \phi'_\ell)$. Therefore, we get that $\mathfrak{m}_1 \circ_j \mathfrak{m}_2$ is friendly from (7.1), replacing δ_l by $\phi_l \circ \psi_l$, for any $l \in \{1, \dots, k + k' - 1\}$, and using (7.2) and the fact that $\mathfrak{m}_1$ is friendly. □

We let $\mathfrak{M}^{\mathfrak{f}}$ denote the set of friendly modifiers (graded by their arity). Since the identity over DFAs is friendly, from Proposition 5 and Lemma 24, $(\mathfrak{M}^{\mathfrak{f}}, \circ)$ is a suboperad of the operad of modifiers $(\mathfrak{M}, \circ)$.

Proposition 23. *The set of friendly modifiers equipped with the composition of operations* $\circ$ *is a suboperad of* $(\mathfrak{M}, \circ)$.

7.1.2 Standard friendly modifiers

Our first goal is to get a grasp on the operations friendly modifiers describe. To reach this goal, we prove that we can always change a friendly modifier into a form that is easier work with, a *standard friendly modifier*, without changing the operation it describes. Furthermore, notice that Theorem 5, proven later in Section 7.1.4, shows that every operation described by a friendly modifier is described by a unique standard friendly modifier. In other words, a standard friendly modifier is a canonical form for every coherent friendly modifier describing the same operation.

Definition 31. *We say that a* k*-modifier* $\mathfrak{m} = [\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$ *is* standard friendly modifier *if, for any* k*-tuple of transition configurations* $((Q_1, i_1, F_1, \phi_1), \dots, (Q_k, i_k, F_k, \phi_k))$ *and any element* $(\psi_1, \dots, \psi_k)$ *of* $Q_1^{Q_1} \times \cdots \times Q_k^{Q_k}$*, we have*

- $\mathfrak{Q}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k)) = Q_1^{Q_1} \times \cdots \times Q_k^{Q_k}$
- $\mathfrak{i}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k)) = (\mathrm{Id}_{Q_1}, \dots, \mathrm{Id}_{Q_k})$
- $\mathfrak{d}((i_1, F_1, \phi_1), \dots, (i_k, F_k, \phi_k))(\psi_1, \dots, \psi_k) = (\phi_1 \circ \psi_1, \dots, \phi_k \circ \psi_k)$

It follows from the above definition that a standard friendly modifier is indeed friendly. Notice that a standard friendly k-modifier $\mathfrak{m} = [\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$ is entirely defined by its third coordinate $\mathfrak{f}$. We can naturally associate a standard modifier with any friendly modifier.

Definition 32. *Let $\mathfrak{m} = [\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$ be a friendly k-modifier. We let $\mathfrak{m}^{\mathrm{sf}}$ denote the standard friendly k-modifier $[\mathfrak{Q}^{\mathrm{sf}}, \mathfrak{i}^{\mathrm{sf}}, \mathfrak{f}^{\mathrm{sf}}, \mathfrak{d}^{\mathrm{sf}}]$ such that , for any k-tuple of state configurations $(s_1, \dots, s_k)$ with $s_j = (Q_j, i_j, F_j)$ for any $j \in \{1, \dots, k\}$, we have*

$$\mathfrak{f}^{\mathrm{sf}}(s_1, \dots, s_k) =$$
$$\{(\phi_1, \dots, \phi_k) \in Q_1^{Q_1} \times \cdots \times Q_k^{Q_k} \mid \mathfrak{d}((i_1, F_1, \phi_1), \dots, (i_k, F_k, \phi_k))(\mathfrak{i}(s_1, \dots, s_k)) \in \mathfrak{f}(s_1, \dots, s_k)\}.$$

Example 23. *Figures 7.1,7.2 and 7.3 describe the effect of $\mathfrak{Comp}$ (defined in Example 14) and of $\mathfrak{Comp}^{\mathrm{sf}}$ on a DFA A. In Figure 7.3, $[ij]$ represents the function ϕ such that $\phi(0) = i$ and $\phi(1) = j$.*

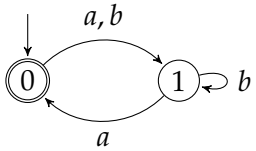


Figure 7.1: The DFA A.

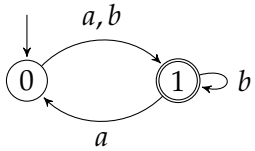


Figure 7.2: The DFA $\mathfrak{C}(A)$.

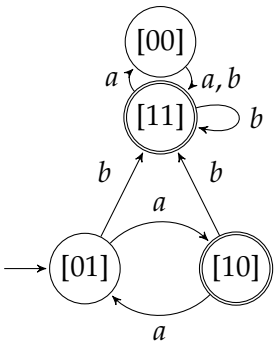


Figure 7.3: The DFA $\mathfrak{C}^{\mathrm{sf}}(A)$.

Lemma 25. *For any coherent friendly modifier $\mathfrak{m}$, the standard modifier $\mathfrak{m}^{\mathrm{sf}}$ is coherent and describes the same operation as $\mathfrak{m}$.*

Proof. Let $\mathfrak{m} = [\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$ be a coherent friendly k-modifier. We show that $\mathfrak{m}$ and $\mathfrak{m}^{\mathrm{sf}}$ describe the same operation, which proves that $\mathfrak{m}^{\mathrm{sf}}$ is coherent. Let $(A_1, \dots, A_k)$ be a k-tuple of DFAs with $A_j = (\Sigma, Q_j, i_j, F_j, \delta_j)$ for any $j \in \{1, \dots, k\}$, and let s_j be the state configuration (Q_j, i_j, F_j). A word $a_1 a_2 \cdots a_l$ is in $\mathrm{L}(\mathfrak{m}(A_1, \dots, A_k))$ if and only if

$$\mathfrak{d}((i_1, F_1, \delta_1^{a_1 a_2 \cdots a_l}), \dots, (i_k, F_k, \delta_k^{a_1 a_2 \cdots a_l}))(\mathfrak{i}(s_1, \dots, s_k)) =$$
$$\mathfrak{d}((i_1, F_1, \delta_1^{a_l}), \dots, (i_k, F_k, \delta_k^{a_l})) \circ \mathfrak{d}((i_1, F_1, \delta_1^{a_{l-1}}), \dots, (i_k, F_k, \delta_k^{a_{l-1}})) \circ \dots$$
$$\circ \mathfrak{d}((i_1, F_1, \delta_1^{a_1}), \dots, (i_k, F_k, \delta_k^{a_1}))(\mathfrak{i}(s_1, \dots, s_k)) \in \mathfrak{f}(s_1, \dots, s_k).$$

Equivalently,

$$\mathfrak{d}^{\mathrm{sf}}((i_1, F_1, \delta_1^{a_l}), \dots, (i_k, F_k, \delta_k^{a_l})) \circ \mathfrak{d}^{\mathrm{sf}}((i_1, F_1, \delta_1^{a_{l-1}}), \dots, (i_k, F_k, \delta_k^{a_{l-1}})) \circ \dots$$
$$\circ \mathfrak{d}^{\mathrm{sf}}((i_1, F_1, \delta_1^{a_1}), \dots, (i_k, F_k, \delta_k^{a_1}))(\mathrm{Id}_{Q_1}, \dots, \mathrm{Id}_{Q_k}) = (\delta_1^{a_1 a_2 \cdots a_l}, \dots, \delta_k^{a_1 a_2 \cdots a_l})$$

is an element of $\mathfrak{f}^{\mathrm{sf}}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k))$. But this last statement is equivalent to $a_1 a_2 \dots a_l \in \mathrm{L}(\mathfrak{m}^{\mathrm{sf}}(A_1, \dots, A_k))$. Therefore, $\mathrm{L}(\mathfrak{m}^{\mathrm{sf}}(A_1, \dots, A_k)) = \mathrm{L}(\mathfrak{m}(A_1, \dots, A_k))$. □

Therefore, in order to characterize the operations described by coherent friendly modifiers, we only need to look at the operations described by coherent standard friendly modifiers. Thus, we now examine in detail coherent standard friendly modifiers.

7.1.3 Characteristic sequences

As a standard friendly modifier $[\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$ is entirely defined by the map $\mathfrak{f}$, which governs the final states of the output DFA. We first show a regularity property on these final states when the modifier is coherent. To that aim, we associate a characteristic sequence with every state of the output DFA, in such a way that any two states associated with the same characteristic sequence have the same finality. These characteristic sequences are represented by k-tuples of eventually periodic sequences with values in $\{0,1\}$.

We let $\mathcal{U}_k$ denote the set of all k-tuples $(u_1, \dots, u_k)$ where each u_j is an eventually periodic sequence with values in $\{0,1\}$. Furthermore, we let $\mathcal{U}$ denote the set $\bigcup_{k\in\mathbb{N}} \mathcal{U}_k$. To simplify notations, for all $(j,p) \in \{1,\dots,k\} \times \mathbb{N}$, we let $u_{j,p}$ denote $(u_j)_p$.

Definition 33. *Let $(t_1, \dots, t_k)$ be a k-tuple of transition configurations with $t_j = (Q_j, i_j, F_j, \phi_j)$ for any $j \in \{1,\dots,k\}$, and let ϕ_j^p be the function $\underbrace{\phi_j \circ \cdots \circ \phi_j}_{p \text{ times}}$. We let $\chi(t_1, \dots, t_k)$ denote the k-tuple of sequences $(u_1, \dots, u_k) \in \mathcal{U}_k$ where, for any $p \in \mathbb{N}$ and any $j \in \{1,\dots,k\}$, $u_{j,p} = [\phi_j^p(i_j) \in F_j]$. We say that $\chi(t_1, \dots, t_k)$ is the* charateristic sequence *of the transition configuration $(t_1, \dots, t_k)$.*

Notice that, in the above definition, we indeed have $(u_1, \dots, u_k) \in \mathcal{U}_k$ because $\phi_j^p(i_j)$ is eventually periodic, since ϕ_j is a function from a finite set into a finite set.

Example 24. *As represented in Figures 7.4, 7.5, let $t_1 = (\{0,1\}, 0, \{1\}, \phi)$ and $t_2 = (\{0,1\}, 0, \{0\}, \phi)$ where $\phi_1(0) = 1$ and $\phi_1(1) = 0$, and let $(u_1, u_2) = \chi(t_1, t_2)$. We have, for all $(j,p) \in \{1,2\} \times \mathbb{N}$, $u_{j,p} = 1$ if and only if $p + j$ is even.*

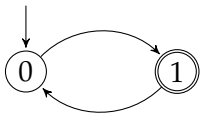


Figure 7.4: Transition configuration t_1.

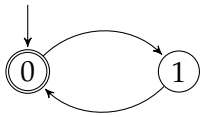


Figure 7.5: Transition configuration t_2.

Recall that, if $\mathfrak{m}$ is a standard friendly modifier and $(A_1, \dots, A_k)$ is any k-tuple of DFA such that the set of states of A_j is Q_j, then the set of the states of $\mathfrak{m}(A_1, \dots, A_k)$ is $Q_1^{Q_1} \times \cdots \times Q_k^{Q_k}$. The next proposition explains the link between characteristic sequences and finality in standard friendly modifiers.

Proposition 24. *Let $\mathfrak{m} = [\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$ be a coherent standard friendly k-modifier. Let $(t_1, \dots, t_k)$ and $(t'_1, \dots, t'_k)$ be two k-tuples of transition configurations, with $t_j = (Q_j, i_j, F_j, \phi_j)$ and $t'_j = (Q'_j, i'_j, F'_j, \phi'_j)$ for any $j \in \{1,\dots,k\}$, such that $\chi(t_1, \dots, t_k) = \chi(t'_1, \dots, t'_k)$. Then we have*

$$(\phi_1, \dots, \phi_k) \in \mathfrak{f}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k))$$

if and only if

$$(\phi'_1, \dots, \phi'_k) \in \mathfrak{f}((Q'_1, i'_1, F'_1), \dots, (Q'_k, i'_k, F'_k)).$$

Proof. Let $(t_1, \dots, t_k)$ and $(t'_1, \dots, t'_k)$ be two k-tuples of transition configurations, with $t_j = (Q_j, i_j, F_j, \phi_j)$ and $t'_j = (Q'_j, i'_j, F'_j, \phi'_j)$ for any $j \in \{1, \dots, k\}$, such that $\chi(t_1, \dots, t_k) = \chi(t'_1, \dots, t'_k)$. Let $(A_1, \dots, A_k)$ and $(A'_1, \dots, A'_k)$ be k-tuples of DFAs with $A_\ell = (\{a\}, Q_\ell, i_\ell, F_\ell, \alpha_\ell)$ and $A'_\ell = (\{a\}, Q'_\ell, i'_\ell, F'_\ell, \alpha'_\ell)$ such that, for any $\ell \in \{1, \dots, k\}$, we have $\alpha_l^a = \phi_l$ and $\alpha_l'^a = \phi'_l$. Since $\chi(t_1, \dots, t_k) = \chi(t'_1, \dots, t'_k)$, we have, for any $\ell \in \{1, \dots, k\}$ and any integer p, $[\phi_\ell^p(i_\ell) \in F_\ell] = [\phi_\ell'^p(i'_\ell) \in F'_l]$. Therefore, for any $\ell \in \{1, \dots, k\}$ and any integer p, we have $[\alpha_\ell{}^{a^p}(i_\ell) \in F_\ell] = [\alpha'_\ell{}^{a^p}(i'_\ell) \in F'_l]$. As a consequence, for any $\ell \in \{1, \dots, k\}$, we have $\mathrm{L}(A_\ell) = \mathrm{L}(A'_\ell)$. Thus, since $\mathfrak{m}$ is coherent, we have $a \in \mathrm{L}(\mathfrak{m}(A_1, \dots, A_k))$ if and only if $a \in \mathrm{L}(\mathfrak{m}(A'_1, \dots, A'_k))$. In other words, as $\alpha_l^a = \phi_l$ and $\alpha_l'^a = \phi'_l$ for any $\ell \in \{1, \dots, k\}$, we have

$$\mathfrak{d}((i_1, F_1, \phi_1), \dots, (i_k, F_k, \phi_k))(\mathrm{Id}_{Q_1}, \dots, \mathrm{Id}_{Q_k}) \in \mathfrak{f}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k))$$

if and only if

$$\mathfrak{d}((i'_1, F'_1, \phi'_1), \dots, (i'_k, F'_k, \phi'_k))(\mathrm{Id}_{Q'_1}, \dots, \mathrm{Id}_{Q'_k}) \in \mathfrak{f}((Q'_1, i'_1, F'_1), \dots, (Q'_k, i'_k, F'_k))$$

Therefore, from Definition 31, we have

$$(\phi_1, \dots, \phi_k) \in \mathfrak{f}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k))$$

if and only if

$$(\phi'_1, \dots, \phi'_k) \in \mathfrak{f}((Q'_1, i'_1, F'_1), \dots, (Q'_k, i'_k, F'_k)).$$

□

The above result invites us to represent the third coordinate $\mathfrak{f}$ of a standard friendly modifier by a set of characteristic functions. In fact, Proposition 25, proven in Section 7.1.4, shows that there is a one-to-one correspondence between standard friendly modifiers and subsets of $\mathcal{U}$. Therefore, we now define an application **mod** that allows us to compute a standard friendly k-modifier from any subset of $\mathcal{U}_k$.

Definition 34. *For any integer k, for any $E \subseteq \mathcal{U}_k$, we let* $\mathbf{mod}(E)$ *denote the standard friendly modifier* $[\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$ *such that, for any k-tuple of state configurations* $((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k))$*, we have*

$$\mathfrak{f}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k)) =$$
$$\left\{(\phi_1, \dots, \phi_k) \in Q_1^{Q_1} \times \cdots \times Q_k^{Q_k} \mid \chi((Q_1, i_1, F_1, \phi_1), \dots, (Q_k, i_k, F_k, \phi_k)) \in E\right\}.$$

We let $\mathfrak{M}_{\mathbf{c}}^{\mathbf{s}}$ denote the set of coherent standard friendly modifiers (graded by their arity). Furthermore, we let $\mathcal{PU}$ denote the set $\bigcup_{k \in \mathbb{N}} 2^{\mathcal{U}_k}$, graded so that, for any $k \in \mathbb{N}$, $(\mathcal{PU})_k = 2^{\mathcal{U}_k}$. As a corollary of Proposition 24, every coherent standard friendly k-modifier can be constructed from some subset of $\mathcal{U}_k$, using Definition 34, *i.e.*, $\mathfrak{M}_{\mathbf{c}}^{\mathbf{s}} \subseteq \mathbf{mod}(\mathcal{PU})$. More precisely,

Corollary 4. *Let $\mathfrak{m} = [\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$ be a coherent standard friendly k-modifier, and let E be the set of all k-tuples of sequences $\chi(t_1, \dots, t_k)$, such that $(t_1, \dots, t_k) = ((Q_1, i_1, F_1, \phi_1), \dots, (Q_k, i_k, F_k, \phi_k))$ is a k-tuple of transition configurations with $(\phi_1, \dots, \phi_k) \in \mathfrak{f}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k))$. We have $\mathfrak{m} = \mathbf{mod}(E)$.*

Proof. For any k-tuple of transition configurations $(t_1, \dots, t_k)$, with $t_j = (Q_j, i_j, F_j, \phi_j)$ for any $j \in \{1, \dots, k\}$, if $\chi(t_1, \dots, t_k) \in E$, we have, by Proposition 24,

$$(\phi_1, \dots, \phi_k) \in \mathfrak{f}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k)).$$

The converse is obvious, and we have

$$\mathfrak{f}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k)) =$$
$$\{(\phi_1, \dots, \phi_k) \in Q_1^{Q_1} \times \cdots \times Q_k^{Q_k} \mid \chi((Q_1, i_1, F_1, \phi_1), \dots, (Q_k, i_k, F_k, \phi_k)) \in E\}.$$

Therefore, $\mathfrak{m} = \mathbf{mod}(E)$. □

7.1.4 Friendly operations

Examples 16 and 17 show that roots and boolean operations are described by friendly modifiers. Therefore, by Proposition 23, and Proposition 18, any composition of a boolean operation and some roots of languages is described by a friendly modifier, and therefore, from Lemma 25, by a standard friendly modifier. These operations are not the only ones to fall in the scope of our study. For instance, the operation **Root** [30], defined by $\mathbf{Root}(L) = \bigcup_{p=1}^{+\infty} \sqrt[p]{L}$, may also be described by a friendly modifier. To capture this kind of operations, we extend the notion of boolean operations (Definition 3) to countable arity.

Definition 35. *We say that a boolean function is* $\mathbb{N}$-ary *if it is a function from* $\{0,1\}^{\mathbb{N}}$ *to* $\{0,1\}$. *For any* $\mathbb{N}$*-ary boolean function* $\mathbf{b}$*, we let* $\boxtimes_{\mathbf{b}}$ *denote the operation producing a language when acting over a sequence of languages in the following way: for any sequence of languages* $(L_p)_{p\in\mathbb{N}}$*, a word* w *is in* $\boxtimes_{\mathbf{b}}((L_p)_{p\in\mathbb{N}})$ *if and only if* $\mathbf{b}([w \in L_1], [w \in L_2], \dots, [w \in L_p], \dots) = 1$. *We call these operations* $\mathbb{N}$-ary boolean operations.

Example 25. *Consider the* $\mathbb{N}$*-ary boolean function* $\mathbf{b}$ *such that, for any sequence* v *in* $\{0,1\}^{\mathbb{N}}$ *,* $\mathbf{b}(v) = 1$ *if and only if, either for all* $p \in \mathbb{N}$, $v_p = 1$, *or for all* $p \in \mathbb{N}$, $v_p = 0$. *We have, for any sequence of regular languages* $(L_p)_{p\in\mathbb{N}}$, $w \in \boxtimes_{\mathbf{b}}((L_p)_{p\in\mathbb{N}})$ *if and only if either, for all* $p \in \mathbb{N}$, $w \in L_p$, *or for all* $p \in \mathbb{N}$, $w \notin L_p$. *In other words, we have* $\boxtimes_{\mathbf{b}}((L_p)_{p\in\mathbb{N}}) = \bigcap_{p=0}^{+\infty} L_p \cup \bigcap_{p=0}^{+\infty} L_p^c$.

We now have the tools to define friendly operations as the composition of a boolean operation and some roots of languages, and we show in Proposition 25 that there is a one-to-one correspondence between friendly operations, coherent standard friendly modifiers and $\mathcal{PU}$. Recall that, for any language L, $\sqrt[0]{L} = \Sigma^\star$ if $\varepsilon \in L$, and $\emptyset$ otherwise.

Definition 36. *A* k*-ary operation over regular languages* $\otimes$ *is* friendly *if there exists an* $\mathbb{N}$*-ary boolean operation* $\boxtimes$ *such that, for any* k*-tuples of regular languages* $(L_1, \dots, L_k)$ *over the same alphabet,*

$$\otimes(L_1, \dots, L_k) = \boxtimes(\sqrt[0]{L_1}, \sqrt[0]{L_2}, \dots, \sqrt[0]{L_k}, \sqrt[1]{L_1}, \sqrt[1]{L_2}, \dots, \sqrt[1]{L_k}, \dots, \sqrt[p]{L_1}, \sqrt[p]{L_2} \dots, \sqrt[p]{L_k}, \dots).$$

Definition 37. *Let $(u_1, \dots, u_k)$ be a k-tuple of sequences with values in $\{0,1\}$. For any k-tuple of regular languages $(L_1, \dots, L_k)$, we let $\langle (u_1, \dots, u_k), (L_1, \dots, L_k) \rangle$ denote the language $\bigcap_{(j,p) \in \{1,\dots,k\} \times \mathbb{N}} E_{j,p}$, where $E_{j,p} = \sqrt[p]{L_j}$ if $u_{j,p} = 1$, and $E_{j,p} = \sqrt[p]{L_j}^c$ otherwise.*
Furthermore, we let $\langle (u_1, \dots, u_k), \cdot \rangle$ denote the k-ary operation over regular languages such that, for any k-tuple of regular languages $(L_1, \dots, L_k)$, we have

$$\langle (u_1, \dots, u_k), \cdot \rangle (L_1, \dots, L_k) = \langle (u_1, \dots, u_k), (L_1, \dots, L_k) \rangle.$$

Example 26. *Let $(u_1, u_2) \in \mathcal{U}_2$ be such that $u_{j,p} = 1$ if and only if $p + j$ is even. Then, for any two regular languages L_1 and L_2, $\langle (u_1, u_2), (L_1, L_2) \rangle$ is equal to*

$$(\sqrt[0]{L_1}^c \cap \sqrt[1]{L_1} \cap \sqrt[2]{L_1}^c \cap \sqrt[3]{L_1} \cap \sqrt[4]{L_1}^c \cap \dots) \bigcap (\sqrt[0]{L_2} \cap \sqrt[1]{L_2}^c \cap \sqrt[2]{L_2} \cap \sqrt[3]{L_2}^c \cap \sqrt[4]{L_2} \cap \dots)$$

Remark 8. *We can rephrase Definition 37 in the following way: for any integer k, for any k-tuple of languages $(L_1, \dots, L_k)$, and any k-tuple of sequences $(u_1, \dots, u_k)$ with values in $\{0,1\}$, a word w is $\langle (u_1, \dots, u_k), (L_1, \dots, L_k) \rangle$ if and only if, for any $(j,p) \in \{1, \dots, k\} \times \mathbb{N}$, $[w^p \in L_j] = u_{j,p}$.*

The next lemma proves that, $\langle (u_1, \dots, u_k), \cdot \rangle$ is the constant operation with the empty set as output if $(u_1, \dots, u_k)$ is not in $\mathcal{U}_k$.

Lemma 26. *For any integer k, if $(L_1, \dots, L_k)$ is a k-tuple of regular languages, and if $(u_1, \dots, u_k)$ is a k-tuple of sequences with values in $\{0,1\}$ such that $\langle (u_1, \dots, u_k), (L_1, \dots, L_k) \rangle \neq \emptyset$, then we have $u \in \mathcal{U}_k$.*

Proof. Let $(A_1, \dots, A_k)$ be a k-tuple of DFA with $A_j = (\Sigma, Q_j, i_j, F_j, \delta_j)$ such that, for all $j \in \{1, \dots, k\}$, $\mathrm{L}(A_j) = L_j$. We have $w^p \in L_j$ if and only if $(\delta_j^w)^p(i_j) \in F_j$. Therefore, if there exists a word w and a k-tuple of sequences $(v_1, \dots, v_k)$ with values in $\{0,1\}$ such that, for all $(j,p) \in \{1, \dots, k\} \times \mathbb{N}$, we have $[w^p \in L_j] = v_{j,p}$, then $[(\delta_j^w)^p(i_j) \in F_j] = v_{j,p}$, which implies that $(v_{j,p})_{p \in \mathbb{N}}$ is eventually periodic. To summarize, if

$$\left\{ w \in \Sigma^* \mid \forall (j,p) \in \{1, \dots, k\} \times \mathbb{N}, [w^p \in L_j] = v_{j,p} \right\} \neq \emptyset,$$

then $(v_1, \dots, v_k) \in \mathcal{U}_k$. We conclude from Remark 8. □

We let $\mathfrak{O}^{\mathrm{f}}$ denote the set of friendly operations (graded by their arity). The next lemma proves that there is a one-to-one correspondence between $\mathcal{PU}$ and $\mathfrak{O}^{\mathrm{f}}$, which is given by the following application.

Definition 38. *Let* **op** *be the application such that, for any $E \subseteq \mathcal{U}_k$,* **op**$(E)$ *denotes the friendly k-ary operation $\bigcup_{(u_1,\dots,u_k) \in E} \langle (u_1, \dots, u_k), \cdot \rangle$.*

The following lemma proves that there is a one-to-one correspondence between subsets of $\mathcal{U}_k$ and k-ary friendly operations.

Lemma 27. *The application* **op** *is a bijection from $\mathcal{PU}$ to $\mathfrak{O}^{\mathrm{f}}$.*

Proof. We first show that **op** is surjective. Let $\mathcal{V}_k = (\{0,1\}^{\mathbb{N}})^k$, *i.e.*, the set of all k-tuples of sequences with values in $\{0,1\}$. Let $\otimes$ be a friendly k-ary operation and **b** be a $\mathbb{N}$-ary boolean function such that, for any k-tuples of regular languages $(L_1, \dots, L_k)$,

$$\otimes = \boxtimes_{\mathbf{b}}(\sqrt[0]{L_1}, \dots, \sqrt[0]{L_k}, \sqrt[1]{L_1}, \dots, \sqrt[1]{L_k}, \dots, \sqrt[p]{L_k}, \dots, \sqrt[p]{L_k}, \dots).$$

Let

$$E = \{(u_1, \dots, u_k) \in \mathcal{U}_k \mid \mathbf{b}(u_{1,0}, \dots, u_{k,0}, u_{1,1}, \dots, u_{k,1}, \dots, u_{1,p}, \dots, u_{k,p}, \dots) = 1\},$$

and let

$$E' = \{(v_1, \dots, v_k) \in \mathcal{V}_k \mid \mathbf{b}(v_{1,0}, \dots, v_{k,0}, v_{1,1}, \dots, v_{k,1}, \dots, v_{1,p}, \dots, v_{k,p}, \dots) = 1\}.$$

We show that $\otimes = \mathbf{op}(E)$. For any k-tuple of regular languages $(L_1, \dots, L_k)$, we have

$$\otimes(L_1, \dots, L_k) = \bigcup_{(v_1,\dots,v_k)\in E'} \left\{ w \in \Sigma^\star \mid \forall (j,p) \in \{1,\dots,k\} \times \mathbb{N}, v_{j,p} = \left[w \in \sqrt[p]{L_j} \right] \right\} = \bigcup_{(v_1,\dots,v_k)\in E'} \langle (v_1, \dots, v_k), (L_1, \dots, L_k) \rangle$$

Notice that the union above is over a set which may involve sequences that are not eventually periodic; However, from Lemma 26, we can remove from this union the k-tuple of sequences that are not in $\mathcal{U}_k$. Hence, we have

$$\otimes(L_1, \dots, L_k) = \bigcup_{(u_1,\dots,u_k)\in E} \langle (u_1, \dots, u_k), (L_1, \dots, L_k) \rangle = (\mathbf{op}(E))(L_1, \dots, L_k).$$

We now prove that **op** is injective. Let E and E' be two distinct subsets of $\mathcal{U}_k$. We suppose that there exists $(u_1, \dots, u_k) \in \mathcal{U}_k$ such that $(u_1, \dots, u_k) \in E$ and $(u_1, \dots, u_k) \notin E'$ (the other case is symmetrical). Since, for any $j \in \{1, \dots, k\}$, $(u_{j,l})_{l\in\mathbb{N}}$ is eventually periodic, the languages $L_j = \{a^p \mid p \in \mathbb{N} \wedge u_{j,p} = 1\}$ are regular. We have $\left[a \in \sqrt[p]{L_j}\right] = u_{j,p}$. Therefore, from Definition 37, for any $(u'_1, \dots, u'_k) \in \mathcal{U}_k$, we have $a \in \langle (u'_1, \dots, u'_k), (L_1, \dots, L_k) \rangle$ if and only if $(u'_1, \dots, u'_k) = (u_1, \dots, u_k)$. It follows that if $\otimes = \mathbf{op}(E)$ and $\otimes' = \mathbf{op}(E')$, then $a \in \otimes(L_1, \dots, L_k)$ and $a \notin \otimes'(L_1, \dots, L_k)$, since $(u_1, \dots, u_k) \in E \setminus E'$. As a consequence, $\otimes \neq \otimes'$ and **op** is injective. □

Example 27. *For any regular language L, we have*

$$\mathbf{Root}(L) = \mathbf{op}(\{u \in \mathcal{U}_1 \mid \textit{there exists } i > 0 \textit{ such that } u_i = 1\})(L) = \bigcup_{i\geq 1} \sqrt[i]{L}.$$

We now show that any operation described by a friendly modifier is friendly.

Lemma 28. *For any $E \in \mathcal{PU}$,* **mod**(E) *is coherent and describes* **op**(E), i.e., **desc** $\circ$ **mod** = **op**.

Proof. Let E be a subset of $\mathcal{U}_k$, let $\mathfrak{m} = \mathbf{mod}(E)$, let $[\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}] = \mathfrak{m}$ and let $\otimes = \mathbf{desc}(\mathfrak{m})$. Let $(A_1, \dots, A_k)$ be any k-tuple of DFAs with $A_j = (\Sigma, Q_j, i_j, F_j, \delta_j)$, for any $j \in \{1, \dots, k\}$, and let $w = a_1 \cdots a_n$ be a word of $\Sigma^\star$. Recall that, since $\mathfrak{m}$ is friendly, we have

$$(\delta_1^w, \dots, \delta_k^w) =$$
$$(\mathfrak{d}((i_1, F_1, \delta_1^{a_n}), \dots, (i_k, F_k, \delta_1^{a_n})) \circ (\mathfrak{d}((i_1, F_1, \delta_1^{a_{n-1}}), \dots, (i_k, F_k, \delta_k^{a_{n-1}})) \circ \cdots$$
$$\circ(\mathfrak{d}((i_1, F_1, \delta_1^{a_1}), \dots, (i_k, F_k, \delta_1^{a_1}))(\mathrm{Id}_{Q_1}, \dots, \mathrm{Id}_{Q_k}).$$

Therefore, w is in $\mathrm{L}(\mathfrak{m}(A_1, \dots, A_k))$ if and only if

$$(\delta_1^w, \dots, \delta_k^w) \in \mathfrak{f}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k)).$$

As a consequence, by Definition 34, w is in $\mathrm{L}(\mathfrak{m}(A_1, \dots, A_k))$ if and only if

$$\chi((Q_1, i_1, F_1, \delta_1^w), \dots, (Q_k, i_k, F_k, \delta_k^w)) \in E.$$

As a consequence, since $\mathfrak{m} = \mathbf{mod}(E)$, w is in $\mathrm{L}(\mathfrak{m}(A_1, \dots, A_k))$ if and only if there exists $(u_1, \dots, u_k) \in E$ such that, for any $(j, p) \in \{1, \dots, k\} \times \mathbb{N}$, we have $[(\delta_j^w)^p(i_j) \in F_j] = u_{j,p}$. However, for any $(j, p) \in \{1, \dots, k\} \times \mathbb{N}$, $(\delta_j^w)^p(i_j) \in F_j$ if and only if $w \in \sqrt[p]{\mathrm{L}(A_j)}$, and thus we have $[(\delta_j^w)^p(i_j) \in F_j] = u_{j,p}$ if and only if $\left[w \in \sqrt[p]{\mathrm{L}(A_j)}\right] = u_{j,p}$. Therefore, by Definition 37, $w \in \mathrm{L}(\mathfrak{m}(A_1, \dots, A_k))$ if and only if there exists $(u_1, \dots, u_k)$ in E such that $w \in \langle (u_1, \dots, u_k), (\mathrm{L}(A_1), \dots, \mathrm{L}(A_k)) \rangle$. We thus have $\otimes(\mathrm{L}(A_1), \dots, \mathrm{L}(A_k)) = \bigcup_{u \in E} \langle (u_1, \dots, u_k), (\mathrm{L}(A_1), \dots, \mathrm{L}(A_k)) \rangle$, and $\otimes = \mathbf{op}(E)$. □

We let $\mathbf{desc_f}$ denote the restriction of $\mathbf{desc}$ to $\mathfrak{M}_{\mathbf{c}}^{\mathbf{s}}$, *i.e.,* the set of coherent standard friendly modifiers. The next proposition states that all applications of Figure 7.6 are bijections and that the diagram is commutative.

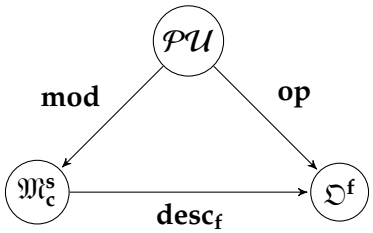


Figure 7.6: Commutative diagram for **op**,**mod** and $\mathbf{desc_f}$.

Proposition 25. *The application* **mod** *is a bijection from* $\mathcal{PU}$ *into* $\mathfrak{M}_{\mathbf{c}}^{\mathbf{s}}$*, and* **op** *and* $\mathbf{desc_f}$ *are bijective. Furthermore,* $\mathbf{desc_f} \circ \mathbf{mod} = \mathbf{op}$.

Proof. By Lemma 28, we have $\mathbf{desc_f} \circ \mathbf{mod} = \mathbf{op}$. However, by Lemma 28, we have $\mathbf{mod}(\mathcal{PU}) \subseteq \mathfrak{M}_{\mathbf{c}}^{\mathbf{s}}$, and thus by Corollary 4, we have $\mathbf{mod}(\mathcal{PU}) = \mathfrak{M}_{\mathbf{c}}^{\mathbf{s}}$. As a consequence, **mod** is a surjection from $\mathcal{PU}$ to $\mathfrak{M}_{\mathbf{c}}^{\mathbf{s}}$. Furthermore, by Lemma 27, **op** is a bijection, and, therefore, so is $\mathbf{desc_f} \circ \mathbf{mod}$. Hence, **mod** is injective and thus a bijection from $\mathcal{PU}$ to $\mathfrak{M}_{\mathbf{c}}^{\mathbf{s}}$, which in turn implies that $\mathbf{desc_f} = \mathbf{op} \circ \mathbf{mod}^{-1}$ is also bijective. □

As an obvious consequence of Proposition 25 and Lemma 25, we have :

Theorem 5. *Every friendly k-ary operation is described by a unique coherent standard friendly k-modifier. Conversely, any coherent friendly k-modifier describes a friendly k-ary operation.*

7.1.5 On the algebraic structure of friendly modifiers

We now have all the tools to state the main result of this Section, concerning the link between the algebraic structures of friendly modifiers and friendly operations. We first define a sequence of binary operations $\odot$ that makes an operad out of $\mathcal{PU}$. Then, we define a quotient operad of coherent modifiers, using standard friendly modifiers. Finally, we rewrite Proposition 25 in terms of operads and isomorphisms of operads.

Let $\odot$ be the sequence of binary operations over $\mathcal{PU}$ such that, for any positive integers j and k with $j \leq k$, for any $E \in \mathcal{PU}_k$, and for any $E' \in \mathcal{PU}$, we have

$$E \odot_j E' = \mathbf{op}^{-1}(\mathbf{op}(E) \circ_j \mathbf{op}(E')).$$

By Proposition 9 used with $\mathbf{op}^{-1}$, $(\mathcal{PU}, \odot)$ is an operad, and $\mathbf{op}$ is an isomorphism of operads from $(\mathcal{PU}, \odot)$ to $(\mathfrak{O}^{\mathbf{f}}, \circ)$. We let $\mathfrak{M}^{\mathbf{f}}_{\mathbf{c}}$ denote the set of all coherent friendly modifiers, and by $\sim_s$ the equivalence relation over $\mathfrak{M}^{\mathbf{f}}_{\mathbf{c}}$ such that, for any two modifiers $\mathfrak{m}, \mathfrak{m}' \in \mathfrak{M}^{\mathbf{f}}_{\mathbf{c}}$, $\mathfrak{m} \sim_s \mathfrak{m}'$ if and only if $\mathfrak{m}^{\mathrm{sf}} = \mathfrak{m}'^{\mathrm{sf}}$. Recall that, from Definition 12, for any two friendly coherent modifiers $\mathfrak{m}$ and $\mathfrak{m}'$, we have $\mathfrak{m} \sim^{\mathbf{desc_f}} \mathfrak{m}'$ if and only if $\mathbf{desc_f}(\mathfrak{m}) = \mathbf{desc_f}(\mathfrak{m}')$. Therefore, since, for any friendly modifier, there is a unique standard friendly modifier that describes the same operation, we have $\sim^{\mathbf{desc_f}} = \sim_s$. As a consequence, by Proposition 11 and Proposition 18, $(\mathfrak{M}^{\mathbf{f}}_{\mathbf{c}}/_{\sim_s}, \circ/_{\sim_s})$ is an operad, and $\widehat{\mathbf{desc_f}}$ is an isomorphism of operads from $(\mathfrak{M}^{\mathbf{f}}_{\mathbf{c}}/_{\sim_s}, \circ/_{\sim_s})$ to $(\mathfrak{O}^{\mathbf{f}}, \circ)$. Let $\overline{\mathbf{mod}}$ be the function from $\mathcal{PU}$ to $\mathfrak{M}^{\mathbf{f}}_{\mathbf{c}}/_{\sim_s}$ such that, for any $E \in \mathcal{PU}$, we have $\overline{\mathbf{mod}}(E) = \widetilde{\mathbf{mod}(E)}^{\mathbf{s}}$. Since $\mathbf{op} = \mathbf{desc} \circ \mathbf{mod}$, we have, for any $E \in \mathcal{PU}$, from Definition 12,

$$\widehat{\mathbf{desc_f}}(\overline{\mathbf{mod}}(E)) = \widehat{\mathbf{desc_f}}(\widetilde{\mathbf{mod}(E)}^{\mathbf{s}}) = \widehat{\mathbf{desc_f}}(\widetilde{\mathbf{mod}(E)}^{\mathbf{desc_f}}) = \mathbf{desc}(\mathbf{mod}(E)) = \mathbf{op}(E).$$

However, $\widehat{\mathbf{desc_f}}$ is an isomorphism, and therefore $\overline{\mathbf{mod}} = \widehat{\mathbf{desc_f}}^{-1} \circ \mathbf{op}$. Furthermore, by Proposition 7, since $\mathbf{op}$ is an isomorphisms of operads, $\overline{\mathbf{mod}}$ is also an isomorphism of operads. We can now state a counterpart of Proposition 25 using operads, illustrated with Figure 7.7.

Theorem 6. *We have*

- *the function* $\overline{\mathbf{mod}}$ *is an isomorphism of operads from* $(\mathcal{PU}, \odot)$ *to* $(\mathfrak{M}^{\mathbf{f}}_{\mathbf{c}}/_{\sim_s}, \circ/_{\sim_s})$,
- *the function* $\mathbf{op}$ *is an isomorphism of operads from* $(\mathcal{PU}, \odot)$ *to* $(\mathfrak{O}^{\mathbf{f}}, \circ)$,
- *the function* $\widehat{\mathbf{desc_f}}$ *is an isomorphism of operads from* $(\mathfrak{M}^{\mathbf{f}}_{\mathbf{c}}/_{\sim_s}, \circ/_{\sim_s})$ *to* $(\mathfrak{O}^{\mathbf{f}}, \circ)$.

Furthermore, $\widehat{\mathbf{desc_f}} \circ \overline{\mathbf{mod}} = \mathbf{op}$.

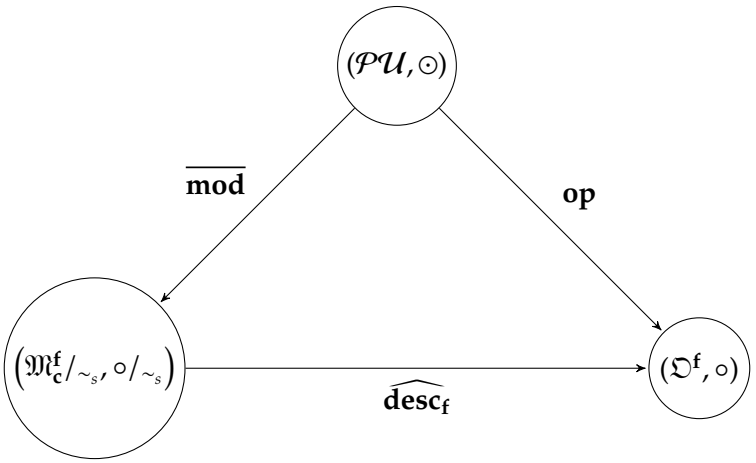


Figure 7.7: Commutative diagram for $\mathbf{op}$,$\overline{\mathbf{mod}}$ and $\widehat{\mathbf{desc}_{\mathbf{f}}}$.

7.2 Product modifiers

In this section, we study a kind of simple friendly modifier, called *product modifier*. Product modifiers are defined so that their output is a product DFA of the input DFAs, without putting any restrictions on their initial state or their final states. We show that the operations they describe are compositions involving boolean operations of finite arity, and the 0-th root. Furthermore, we study the underlying algebraic structure, and show that it is very similar to the algebraic structure given in Theorem 6.

7.2.1 Product modifiers: an operad

We first show that product modifiers can also be equipped with a structure of operad.

Definition 39. *A k-modifier* $\mathfrak{m} = [\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$ *is a* product modifier *if, for any k-tuple of transition configurations* $(t_1, \dots, t_k)$*, with* $t_j = (Q_j, i_j, F_j, \phi_j)$ *for any* $j \in \{1, \dots, k\}$*, we have*

1. $\mathfrak{Q}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k)) = Q_1 \times \cdots \times Q_k$,
2. *and, for any* $(q_1, \dots, q_k) \in Q_1 \times \cdots \times Q_k$*, we have*

$$\mathfrak{d}((i_1, F_1, \phi_1) \dots, (i_k, F_k, \phi_k))(q_1, \dots, q_k) = (\phi_1(q_1), \dots, \phi_k(q_k)).$$

In other words, if $\mathfrak{m}$ is a product modifier, then $\mathfrak{m}(A_1, \dots, A_k)$ is a product DFA of the DFAs $A_1, \dots, A_k$, but with final states $\mathfrak{f}((Q_1, i_1, F_1) \dots, (Q_k, i_k, F_k))$ and initial state $\mathfrak{i}((Q_1, i_1, F_1) \dots, (Q_k, i_k, F_k))$. We let $\mathfrak{M}^{\mathfrak{p}}$ denote the set of all product modifiers. We check that the set of product modifiers is stable by the composition of operations, thus making it an operad.

Proposition 26. *The set* $\mathfrak{M}^{\mathfrak{p}}$ *equipped with* $\circ$ *is an operad.*

Proof. Let $\mathfrak{m}_1 = [\mathfrak{Q}_1, \mathfrak{i}_1, \mathfrak{f}_1, \mathfrak{d}_1]$ and $\mathfrak{m}_2 = [\mathfrak{Q}_2, \mathfrak{i}_2, \mathfrak{f}_2, \mathfrak{d}_2]$ be two product modifiers, respectively k-ary and k'-ary, let $j \leq k$ be a positive integer, and let $\mathfrak{m}_1 \circ_j \mathfrak{m}_2 = [\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$. By Proposition 5, to show that $(\mathfrak{M}^{\mathfrak{p}}, \circ)$ is an operad, it is enough to show that $\mathfrak{m}_1 \circ_j \mathfrak{m}_2$ is a product modifier. For any $(k + k' - 1)$-tuple of transition configurations $(t_1, \ldots, t_{k+k'-1})$ with $t_j = (Q_j, i_j, F_j, \delta_j)$ for any $j \in \{1, \ldots, k + k' - 1\}$, we have, from Definition 39,

$$\mathfrak{Q}_2(t_j, \ldots, t_{j+k'-1}) = Q_j \times \cdots \times Q_{j+k'-1}, \tag{7.3}$$

and therefore, by Definition 6 and 39,

$$\begin{gathered}\mathfrak{Q}(t_1, \ldots, t_{k+k'-1}) = \mathfrak{Q}_1(t_1, \ldots, t_{j-1}, (\mathfrak{Q}_2(t_j, \ldots, t_{j+k'-1}), \mathfrak{i}_2(t_j, \ldots, t_{j+k'-1}), \mathfrak{f}_2(t_j, \ldots, t_{j+k'-1}), \\ \mathfrak{d}_2(t_j, \ldots, t_{j+k'-1})), t_{j+k'}, \ldots, t_{k+k'-1}) = Q_1 \times \cdots \times Q_{k+k'-1}\end{gathered} \tag{7.4}$$

Similarly, for any $(q_1, \ldots, q_{k+k'-1}) \in Q_1 \times \cdots \times Q_{k+k'-1}$, we have

$$\mathfrak{d}_2(t_j, \ldots, t_{j+k'-1})(q_j, \ldots, q_{j+k'-1}) = (\phi_j(q_j), \ldots, \phi_{j+k'-1}(q_{j+k'-1})),$$

and therefore

$$\begin{gathered}\mathfrak{d}(t_1, \ldots, t_{k+k'-1})(q_1, \ldots, q_{k+k'-1}) = \mathfrak{d}_1(t_1, \ldots, t_{j-1}, (\mathfrak{Q}_2(t_j, \ldots, t_{k'+j-1}), \mathfrak{i}_2(t_j, \ldots, t_{k'+j-1}), \\ \mathfrak{f}_2(t_j, \ldots, t_{k'+j-1}), \mathfrak{d}_2(t_j, \ldots, t_{k'+j-1})), t_{k'+j}, \ldots, t_{k+k'-1})(q_1, \ldots, q_{j-1}, (q_j \ldots, q_{j+k'-1}), q_{j+k'}, \ldots, q_{k+k'-1}) \\ = (\phi_1(q_1), \ldots, \phi_{k+k'-1}(q_{k+k'-1})).\end{gathered} \tag{7.5}$$

As a consequence, $\mathfrak{m}_1 \circ_j \mathfrak{m}_2$ is a product modifier. □

7.2.2 From product modifiers to standard modifiers

In order to characterize the operations recognized by product modifiers, we take a path similar to the one taken for friendly modifiers. However, in this case, instead of looking at the whole characteristic sequences in $\mathcal{U}_k$, we look only at the first two terms of every sequences in the k-tuple. In other words, intuitively, we replace characteristic sequences, which are originally k-tuples of sequences with values in $\{0, 1\}$, with k-tuples of pairs in $\{0, 1\}^2$. These pairs represent the same thing as did the first two values of the complete characteristic sequence. We begin to formalize this idea by proving a proposition similar to Proposition 24, for product modifiers.

Proposition 27. *Let $\mathfrak{m} = [\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$ be a coherent product k-modifier, and let $\mathfrak{m}^{\mathrm{sf}} = [\mathfrak{Q}^{\mathrm{sf}}, \mathfrak{i}^{\mathrm{sf}}, \mathfrak{f}^{\mathrm{sf}}, \mathfrak{d}^{\mathrm{sf}}]$. Let $(t_1, \ldots, t_k)$ and $(t'_1, \ldots, t'_k)$ be two k-tuples of transition configurations with $t_j = (Q_j, i_j, F_j, \phi_j)$ and $t'_j = (Q'_j, i'_j, F'_j, \phi'_j)$, for any $j \in \{1, \ldots, k\}$. If, for any $j \in \{1, \ldots, k\}$, $[i_j \in F_j] = [i'_j \in F'_j]$ and $[\phi(i_j) \in F_j] = [\phi(i'_j) \in F'_j]$, then we have*

$$(\phi_1, \ldots, \phi_k) \in \mathfrak{f}^{\mathrm{sf}}((Q_1, i_1, F_1), \ldots, (Q_k, i_k, F_k))$$

if and only if

$$(\phi'_1, \ldots, \phi'_k) \in \mathfrak{f}^{\mathrm{sf}}((Q'_1, i'_1, F'_1), \ldots, (Q'_k, i'_k, F'_k)).$$

Proof. Let $\mathfrak{m} = [\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$ be a coherent product k-modifier, and let $\mathfrak{m}^{\mathrm{sf}} = [\mathfrak{Q}^{\mathrm{sf}}, \mathfrak{i}^{\mathrm{sf}}, \mathfrak{f}^{\mathrm{sf}}, \mathfrak{d}^{\mathrm{sf}}]$. Let $(t_1, \dots, t_k)$ and $(t'_1, \dots, t'_k)$ be two k-tuples of transition configurations with $t_\ell = (Q_\ell, i_\ell, F_\ell, \phi_\ell)$ and $t'_\ell = (Q'_\ell, i'_\ell, F'_\ell, \phi'_\ell)$, for any $\ell \in \{1, \dots, k\}$. Suppose that, for any $\ell \in \{1, \dots, k\}$, $[i_\ell \in F_\ell] = [i'_\ell \in F'_\ell]$ and $[\phi(i_\ell) \in F_\ell] = [\phi(i'_\ell) \in F'_\ell]$.

We let $(h_1, \dots, h_k)$ denote $\mathfrak{i}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k))$, and we let $(h'_1, \dots, h'_k)$ denote

$$\mathfrak{i}((Q'_1, i'_1, F'_1), \dots, (Q'_k, i'_k, F'_k)).$$

The idea of this proof is the same as the proof of Proposition 24. We define two k-tuples $(A_1, \dots, A_k)$ and $(A'_1, \dots, A'_k)$ of well-chosen DFAs such that $\mathrm{L}(A_\ell) = \mathrm{L}(A'_\ell)$, and conclude from the coherence of $\mathfrak{m}$. However, this time, we do not define the transition functions of a in A_ℓ and A'_ℓ as ϕ_ℓ and ϕ'_ℓ, but rather as a modified version of them. To that aim, for any $\ell \in \{1, \dots, k\}$, we let ψ_ℓ and ψ'_ℓ denote the two functions of $Q_\ell^{Q_\ell}$ and $Q'^{Q'_\ell}_\ell$ respectively such that

1. if $h_\ell \neq i_\ell$ and $h'_\ell \neq i'_\ell$, then

$$\psi_\ell(q_\ell) = \begin{cases} i_\ell & \text{if } q_\ell = i_\ell \\ \phi_\ell(q_\ell) & \text{otherwise} \end{cases} \quad \text{and} \quad \psi'_\ell(q'_\ell) = \begin{cases} i'_\ell & \text{if } q'_\ell = i'_\ell \\ \phi'_\ell(q'_\ell) & \text{otherwise} \end{cases}$$

2. if $h_\ell = i_\ell$ and $h'_\ell = i'_\ell$, or if $h_\ell = i_\ell$ and $h'_\ell \notin \{i'_\ell, \phi'_\ell(i'_\ell)\}$, then

$$\psi_\ell(q_\ell) = \begin{cases} \phi_\ell(i_\ell) & \text{if } q_\ell = \phi_\ell(i_\ell) \\ \phi_\ell(q_\ell) & \text{otherwise} \end{cases} \quad \text{and} \quad \psi'_\ell(q'_\ell) = \begin{cases} \phi'_\ell(i'_\ell) & \text{if } q'_\ell = \phi'_\ell(i'_\ell) \\ \phi'_\ell(q'_\ell) & \text{otherwise.} \end{cases}$$

3. if $h_\ell = i_\ell$ and $h'_\ell = \phi'_\ell(i'_\ell)$, then

   (a) if $[i'_\ell \in F'_\ell] = [\phi'_\ell(i'_\ell) \in F'_\ell]$, then

$$\psi_\ell(q_\ell) = \begin{cases} \phi_\ell(i_\ell) & \text{if } q_\ell = \phi_\ell(i_\ell) \\ \phi_\ell(q_\ell) & \text{otherwise} \end{cases} \quad \text{and} \quad \psi'_\ell(q'_\ell) = \begin{cases} i'_\ell & \text{if } q'_\ell = i'_\ell \\ \phi'_\ell(q'_\ell) & \text{otherwise} \end{cases}$$

   (b) if $[i'_\ell \in F'_\ell] \neq [\phi'_\ell(i'_\ell) \in F'_\ell]$, then one of them is equal to $[\phi'_\ell(h'_\ell) \in F'_\ell]$, and

      i. if $[i'_\ell \in F'_\ell] = [\phi'_\ell(h'_\ell) \in F'_\ell]$, then

$$\psi_\ell(q_\ell) = \begin{cases} i_\ell & \text{if } q_\ell = \phi_\ell(i_\ell) \\ \phi_\ell(q_\ell) & \text{otherwise} \end{cases} \quad \text{and} \quad \psi'_\ell(q'_\ell) = \begin{cases} \phi'_\ell(i'_\ell) & \text{if } q'_\ell = \phi'_\ell(h'_\ell) \\ \phi'_\ell(q'_\ell) & \text{otherwise} \end{cases}$$

      ii. if $[\phi'_\ell(i'_\ell) \in F'_\ell] = [\phi'_\ell(h'_\ell) \in F'_\ell]$, then

$$\psi_\ell(q_\ell) = \begin{cases} \phi_\ell(i_\ell) & \text{if } q_\ell = \phi_\ell(i_\ell) \\ \phi_\ell(q_\ell) & \text{otherwise} \end{cases} \quad \text{and} \quad \psi'_\ell(q'_\ell) = \begin{cases} \phi'_\ell(i'_\ell) & \text{if } q'_\ell = \phi'_\ell(h'_\ell) \\ \phi'_\ell(q'_\ell) & \text{otherwise} \end{cases}$$

4. if $h_\ell \neq \phi_\ell(i_\ell)$ and $h'_\ell = i'_\ell$, then ψ_ℓ and ψ'_ℓ are defined in a symmetrical way with respect to the cases 2 and 3.

Notice that, if $i'_\ell = \phi'_\ell(i'_\ell)$, cases 2 and 3(a) may overlap. However, they lead to the same definition of ψ_ℓ and ψ'_ℓ. We let $(A_1, \ldots, A_k)$ and $(A'_1, \ldots, A'_k)$ denote the two k-tuples of DFAs such that $A_\ell = (\{a\}, Q_\ell, i_\ell, F_\ell, \alpha_\ell)$ and $A'_\ell = (\{a\}, Q'_\ell, i'_\ell, F'_\ell, \alpha'_\ell)$, with $\alpha_\ell^a = \psi$ and $\alpha'^a_\ell = \psi'$, for any $\ell \in \{1, \ldots, k\}$.

Notice that, if $p = 0$, we have $\psi_\ell{}^p(i_\ell) = i_\ell$ and $\psi'^p_\ell(i'_\ell) = i'_\ell$ for any $\ell \in \{1, \ldots, k\}$. However, recall that we supposed $[i_\ell \in F_\ell] = [i'_\ell \in F'_\ell]$ and $[\phi_\ell(i_\ell) \in F_\ell] = [\phi_\ell(i'_\ell) \in F'_\ell]$, for any $\ell \in \{1, \ldots, k\}$. Therefore, if $p = 0$, since $\alpha_\ell{}^{a^p} = \psi_\ell^p$ and $\alpha'^{a^p}_\ell = \psi'^p_\ell$, we have $[\alpha_\ell{}^{a^p}(i_\ell) \in F_\ell] = [\alpha'^{a^p}_\ell(i'_\ell) \in F'_\ell]$ for any $\ell \in \{1, \ldots, k\}$. We prove that $[\alpha_\ell{}^{a^p}(i_\ell) \in F_\ell] = [\alpha'^{a^p}_\ell(i'_\ell) \in F'_\ell]$ still holds true for any $\ell \in \{1, \ldots, k\}$ and any positive integer p, in a similar way, by computing $\psi_\ell{}^p(i_\ell)$ and $\psi'^p_\ell(i'_\ell)$ in every one of the above cases. Thus, for any positive integer p and any $\ell \in \{1, \ldots, k\}$, we have

1. if $h_\ell \neq i_\ell$ and $h'_\ell \neq i'_\ell$, then
$$\psi_\ell^p(i_\ell) = i_\ell \text{ and } \psi'^p_\ell(i'_\ell) = i'_\ell$$
2. if $h_\ell = i_\ell$ and $h'_\ell = i'_\ell$, or if $h_\ell = i_\ell$ and $h'_\ell \notin \{i'_\ell, \phi'_\ell(i'_\ell)\}$, then
$$\psi_\ell^p(i_\ell) = \phi_\ell(i_\ell) \text{ and } \psi'^p_\ell(i'_\ell) = \phi'_\ell(i'_\ell)$$
3. if $h_\ell = i_\ell$ and $h'_\ell = \phi'_\ell(i'_\ell)$, then
   (a) if $[i'_\ell \in F'_\ell] = [\phi'_\ell(i'_\ell) \in F'_\ell]$, then
   $$\psi_\ell^p(i_\ell) = \phi_\ell(i_\ell) \text{ and } \psi'^p_\ell(i'_\ell) = i'_\ell$$
   (b) if $[i'_\ell \in F'_\ell] \neq [\phi'_\ell(i'_\ell) \in F'_\ell]$, then
      i. if $[i'_\ell \in F'_\ell] = [\phi'_\ell(h'_\ell) \in F'_\ell]$, then
      $$\psi_\ell^p(i_\ell) = \begin{cases} i_\ell & \text{if } p \text{ is even} \\ \phi_\ell(i_\ell) & \text{if } p \text{ is odd} \end{cases} \text{ and } \psi'^p_\ell(i'_\ell) = \begin{cases} \phi'_\ell(h'_\ell) & \text{if } p \text{ is even} \\ \phi'_\ell(i'_\ell) & \text{if } p \text{ is odd} \end{cases}$$
      ii. if $[\phi'_\ell(i'_\ell) \in F'_\ell] = [\phi'_\ell(h'_\ell) \in F'_\ell]$, then
      $$\psi_\ell^p(i_\ell) = \phi_\ell(i_\ell) \text{ and } \psi'^p_\ell(i'_\ell) = \begin{cases} \phi'_\ell(h'_\ell) & \text{if } p \text{ is even} \\ \phi'_\ell(i'_\ell) & \text{if } p \text{ is odd} \end{cases}$$
4. if $h_\ell = \phi_\ell(i_\ell)$ and $h'_\ell = i'_\ell$, then ψ_ℓ^p and ψ'^p_ℓ are symmetrical with respect to the case 3.

We check by a direct computation that, in all of the above cases, we have $[\psi_\ell^p(i_\ell) \in F_\ell] = [\psi'^p_\ell(i'_\ell) \in F'_\ell]$ for any $\ell \in \{1, \ldots, k\}$ and any integer p. However, $\alpha_\ell{}^{a^p} = \psi_\ell^p$ and $\alpha'^{a^p}_\ell = \psi'^p_\ell$, for any $\ell \in \{1, \ldots, k\}$ and any integer p. Therefore, for any $\ell \in \{1, \ldots, k\}$, we have $\mathrm{L}(A_\ell) = \mathrm{L}(A'_\ell)$. Thus, since $\mathfrak{m}$ is coherent, we have $a \in \mathrm{L}(\mathfrak{m}(A_1, \ldots, A_k))$ if and only if $a \in \mathfrak{m}(A'_1, \ldots, A'_k)$. Hence, denoting $\mathfrak{f}((Q_1, i_1, F_1), \ldots, (Q_k, i_k, F_k))$ by G and $\mathfrak{f}((Q'_1, i'_1, F'_1), \ldots, (Q'_k, i'_k, F'_k))$ by G', we have

$$\mathfrak{d}((i_1, F_1, \alpha_1^a), \ldots, (i_k, F_k, \alpha_k^a))(h_1, \ldots, h_k) \in G$$

if and only if

$$\mathfrak{d}((i'_1, F'_1, {\alpha'_1}^a), \dots, (i'_k, F'_k, {\alpha'_k}^a))(h'_1, \dots, h'_k) \in G'.$$

In other words, since $\mathfrak{m}$ is a product modifier, we have

$$(\psi_1(h_1), \dots, \psi_k(h_k)) \in G \text{ if and only if } (\psi'_1(h'_1), \dots, \psi'_k(h'_k)) \in G'.$$

However, it follows from the definition of ψ_ℓ and ψ'_ℓ that, for any $\ell \in \{1, \dots, k\}$, we have $\psi_\ell(h_\ell) = \phi_\ell(h_\ell)$ and $\psi'_\ell(h'_\ell) = \phi'_\ell(h'_\ell)$. Therefore, we have

$$(\phi_1(h_1), \dots, \phi_k(h_k)) \in G \text{ if and only if } (\phi'_1(h'_1), \dots, \phi'_k(h'_k)) \in G'.$$

As a consequence of the definitions of friendly and standard modifiers, we have

$$(\phi_1, \dots, \phi_k) \in \mathfrak{f}^{\mathrm{sf}}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k))$$

if and only if

$$(\phi'_1, \dots, \phi'_k) \in \mathfrak{f}^{\mathrm{sf}}((Q'_1, i'_1, F'_1), \dots, (Q'_k, i'_k, F'_k)).$$

□

We let $\mathcal{PC}_k$ denote the set of all subsets E of $\mathcal{U}_k$ that verify the following property: for every $(u_1, \dots, u_k) \in \mathcal{U}_k$ such that there exists $(v_1, \dots, v_k) \in E$, with $v_{j,0} = u_{j,0}$ and $v_{j,1} = u_{j,1}$ for any $j \in \{1, \dots, k\}$, we have $(u_1, \dots, u_k) \in E$. Furthermore, we let $\mathcal{PC}$ denote the set $\bigcup_{k \in \mathbb{N}} \mathcal{PC}_k$. We know that any standard modifier can be associated with an element of $\mathcal{PU}$ by the isomorphism of operads $\mathbf{mod}^{-1}$. From Lemma 27, we prove that, if $\mathfrak{m}$ is a product modifier, then $\mathbf{mod}^{-1}(\mathfrak{m}^{\mathrm{sf}})$ is in $\mathcal{PC}$. This intuitively comes from the fact that only the first two elements of every u_i matter, when $(u_1, \dots, u_k) \in \mathcal{PC}$.

Corollary 5. *Let $\mathfrak{m}$ be a product modifier. We have $\mathbf{mod}^{-1}(\mathfrak{m}^{\mathrm{sf}}) \in \mathcal{PC}$*

Proof. Let $\mathfrak{m} = [\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$ be a product k-modifier, and let $\mathfrak{m}^{\mathrm{sf}} = [\mathfrak{Q}^{\mathrm{sf}}, \mathfrak{i}^{\mathrm{sf}}, \mathfrak{f}^{\mathrm{sf}}, \mathfrak{d}^{\mathrm{sf}}]$. Let E' be the set of all k-tuples of sequences $\chi(t_1, \dots, t_k)$, such that $(t_1, \dots, t_k)$ is a k-tuple of transition configurations with $(\phi_1, \dots, \phi_k) \in \mathfrak{f}^{\mathrm{sf}}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k))$. Let E be the set of all k-tuples of sequences $(u_1, \dots, u_k) \in \mathcal{U}_k$ such that there exists $(v_1, \dots, v_k) \in E'$ with $v_{j,0} = u_{j,0}$ and $v_{j,1} = u_{j,1}$, for any $j \in \{1, \dots, k\}$. It follows from the definition of $\mathcal{PC}_k$ that $E \in \mathcal{PC}_k$. We now show that $\mathfrak{m}^{\mathrm{sf}} = \mathbf{mod}(E)$. To that aim we first prove that, for any k-tuple of transition configurations $(t_1, \dots, t_k)$, with $t_j = (Q_j, i_j, F_j, \phi_j)$ for any $j \in \{1, \dots, k\}$, we have $(\phi_1, \dots, \phi_k) \in \mathfrak{f}^{\mathrm{sf}}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k))$ if and only if $\chi(t_1, \dots, t_k) \in E$.

Let $(t_1, \dots, t_k)$ be a k-tuple of transition configurations, with $t_j = (Q_j, i_j, F_j, \phi_j)$ for any $j \in \{1, \dots, k\}$, such that $(\phi_1, \dots, \phi_k) \in \mathfrak{f}^{\mathrm{sf}}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k))$. From the definition of E', we have $\chi(\phi_1, \dots, \phi_k) \in E'$, and thus $\chi(t_1, \dots, t_k) \in E$.

Conversely, let $(t_1, \dots, t_k)$ be a k-tuple of transition configurations, with $t_j = (Q_j, i_j, F_j, \phi_j)$ for any $j \in \{1, \dots, k\}$, such that $\chi(t_1, \dots, t_k) \in E$. Let $(u_1, \dots, u_k) = \chi(t_1, \dots, t_k)$. From the definition of E, there exists a k-tuple of sequences $(v_1, \dots, v_k) = \chi(t'_1, \dots, t'_k)$ in E', where $(t'_1, \dots, t'_k)$ is a k-tuple of transition configurations, with $t'_j = (Q'_j, i'_j, F'_j, \phi'_j)$ for any $j \in \{1, \dots, k\}$, such that $v_{j,0} = u_{j,0}$ and $v_{j,1} = u_{j,1}$. However, it follows from the definition of characteristic sequences that we have $[i_j \in F_j] = [i'_j \in F'_j]$, and $[\phi_j(i_j) \in F_j] = [\phi'_j(i'_j) \in F'_j]$.

Furthermore, from the definition of E', we have $(\phi'_1, \dots, \phi'_k) \in \mathfrak{f}^{\mathrm{sf}}((Q'_1, i'_1, F'_1), \dots, (Q'_k, i'_k, F'_k))$. Therefore, by Proposition 27, we have $(\phi_1, \dots, \phi_k) \in \mathfrak{f}^{\mathrm{sf}}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k))$.

As a consequence, for any k-tuple of transition configurations $(t_1, \dots, t_k)$, with $t_j = (Q_j, i_j, F_j, \phi_j)$ for any $j \in \{1, \dots, k\}$, we have $(\phi_1, \dots, \phi_k) \in \mathfrak{f}^{\mathrm{sf}}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k))$ if and only if $\chi(t_1, \dots, t_k) \in E$. Thus, we have

$$\mathfrak{f}^{\mathrm{sf}}((Q_1, i_1, F_1), \dots, (Q_k, i_k, F_k)) =$$
$$\left\{(\phi_1, \dots, \phi_k) \in Q_1^{Q_1} \times \cdots \times Q_k^{Q_k} \mid \chi((Q_1, i_1, F_1, \phi_1), \dots, (Q_k, i_k, F_k, \phi_k)) \in E\right\}.$$

Therefore, from the definition of **mod**, we conclude that $\mathfrak{m}^{\mathrm{sf}} = \mathbf{mod}(E)$.

□

In the following section, we formalize the idea that the operations in the image of $\mathcal{PC}$ by **op** should only involve the composition of a boolean operation with 0-th roots and identities.

7.2.3 Product modifiers and quasi-boolean operations

Definition 40. *A k-ary operation over regular languages $\otimes$ is* quasi-boolean *if there exists a $2k$-ary boolean operation $\oplus$ such that, for any k-tuples of regular languages $(L_1, \dots, L_k)$,*

$$\oplus(L_1, \dots, L_k) = \otimes(\sqrt[0]{L_1}, \sqrt[0]{L_2}, \dots, \sqrt[0]{L_k}, L_1, L_2, \dots, L_k).$$

We let $\mathfrak{O}^{\mathrm{p}}$ denote the set of all quasi-boolean operations.

Proposition 28. *The image of an element of $\mathcal{PC}$ by* **op** *is a quasi-boolean operation,* i.e., $\mathbf{op}(\mathcal{PC}) \subseteq \mathfrak{O}^{\mathrm{p}}$.

Proof. Let $E \in \mathcal{PC}_k$, and, for any $j \in \{1, \dots, k\}$, let $G_j = \{(u_{j,0}, u_{j,1}) \mid (u_1, \dots, u_k) \in E\}$. We have

$$E = \left\{(u_1, \dots, u_k) \in \mathcal{U}_k \mid (u_{j,0}, u_{j,1}) \in G_j, \text{ for any } j \in \{1, \dots, k\}\right\}.$$

Therefore, from Definition 38, we have

$$\mathbf{op}(E) = \bigcup_{(u_1,\dots,u_k)\in E} \langle (u_1, \dots, u_k), \cdot \rangle = \bigcup_{(u_1,\dots,u_k)\in \mathcal{U}_k \mid (u_{j,0},u_{j,1})\in G_j, \forall j\in\{1,\dots,k\}} \langle (u_1, \dots, u_k), \cdot \rangle.$$

However, from Lemma 26, we have

$$\bigcup_{(u_1,\dots,u_k)\in \mathcal{U}_k \mid (u_{j,0},u_{j,1})\in G_j, \forall j\in\{1,\dots,k\}} \langle (u_1, \dots, u_k), \cdot \rangle = \bigcup_{(v_1,\dots,v_k)\in \mathcal{V}_k \mid (v_{j,0},v_{j,1})\in G_j, \forall j\in\{1,\dots,k\}} \langle (v_1, \dots, v_k), \cdot \rangle,$$

where we let $\mathcal{V}_k$ denote the set of all k-tuples of sequences with values in $\{0, 1\}$. Therefore, for any k-tuple of languages $(L_1, \dots, L_k)$, we have

$$\mathbf{op}(E)(L_1, \dots, L_k) = \bigcup_{((a_1,b_1),\dots,(a_k,b_k)) \mid (a_j,b_j)\in G_j, \forall j\in\{1,\dots,k\}} \left(\bigcap_{j\in\{1,\dots,k\}} \left(\left\langle a_j, \sqrt[0]{L_j} \right\rangle \cap \langle b_j, L_j \rangle \right) \right),$$

where for any language L, $\langle 0, L\rangle = L$ and $\langle 1, L\rangle = L^c$. To conclude, we have

$$\mathbf{op}(E) = \otimes(\sqrt[0]{L_1}, \sqrt[0]{L_2}, \dots, \sqrt[0]{L_k}, L_1, L_2, \dots, L_k),$$

where $\otimes$ is the $2k$-ary boolean operation such that, for any $2k$-tuple of regular languages $(L_1, L_2, \dots, L_k, L'_1, L'_2, \dots, L'_k)$,

$$\otimes(L_1, L_2, \dots, L_k, L'_1, L'_2, \dots, L'_k) = \bigcup_{((a_1,b_1),\dots,(a_k,b_k))|(a_j,b_j)\in G_j, \forall j\in\{1,\dots,k\}} \left(\bigcap_{j\in\{1,\dots,k\}} \left(\langle a_j, L_j\rangle \cap \langle b_j, L'_j\rangle\right)\right)$$

□

Now we prove the last tool we need to connect quasi-boolean operations, product modifiers, and $\mathcal{PC}$, which is that every quasi-boolean operation is described by a product modifier.

Lemma 29. *Let $\otimes$ be a quasi-boolean operation. There exists a product modifier $\mathfrak{m}$ such that* $\mathbf{desc}(\mathfrak{m}) = \otimes$.

Proof. Let $\otimes$ be a k-ary quasi-boolean operations, and let $\otimes'$ be a $2k$-ary boolean operations such that, for any k-tuple of regular languages $(L_1, \dots, L_k)$, we have

$$\otimes(L_1, \dots, L_k) = \otimes'(\sqrt[0]{L_1}, \sqrt[0]{L_2}, \dots, \sqrt[0]{L_k}, \sqrt[1]{L_1}, \sqrt[1]{L_2}, \dots, \sqrt[1]{L_k}).$$

For any k-tuple of regular languages $\mathbf{L}' = (L'_1, \dots, L'_k)$, we let $\otimes_{\mathbf{L}'}$ denote the k-ary regular operation such that, for any k-tuple of languages $(L_1, \dots, L_k)$, we have

$$\otimes_{\mathbf{L}'}(L_1, \dots, L_k) = \otimes'\left(\sqrt[0]{L'_1}, \sqrt[0]{L'_2}, \dots, \sqrt[0]{L'_k}, L_1, L_2, \dots, L_k\right).$$

Notice that, for any language L over an alphabet Σ, $\sqrt[0]{L}$ is either $\emptyset$ or Σ^*. Therefore, for any k-tuple of regular languages $\mathbf{L}' = (L'_1, \dots, L'_k)$, $\otimes_{\mathbf{L}'}(L_1, \dots, L_k)$ is a boolean operation. We let $\mathbf{b}^{\mathbf{L}'}$ denote a boolean function such that $\otimes_{\mathbf{L}'} = \otimes_{\mathbf{b}^{\mathbf{L}'}}$. Furthermore, we let $\mathfrak{m}$ denote the k-modifier such that, for any k-tuple of DFAs $(A_1, \dots, A_k)$, we have $\mathfrak{m}(A_1, \dots, A_k) = \mathfrak{m}_{\mathbf{b}^{\mathbf{L}}}(A_1, \dots, A_k)$, with $\mathbf{L} = (L_1, \dots, L_k) = (\mathrm{L}(A_1), \dots, \mathrm{L}(A_k))$. By Proposition 15, we have

$$\mathrm{L}(\mathfrak{m}_{\mathbf{b}^{\mathbf{L}}}(A_1, \dots, A_k)) = \otimes_{\mathbf{b}^{\mathbf{L}}}(L_1, \dots, L_k) = \otimes_{\mathbf{L}}(L_1, \dots, L_k) =$$
$$\otimes'(\sqrt[0]{L_1}, \sqrt[0]{L_2}, \dots, \sqrt[0]{L_k}, L_1, L_2, \dots, L_k) = \otimes(L_1, \dots, L_k).$$

Therefore, $\mathbf{desc}(\mathfrak{m}) = \otimes$. □

7.2.4 On the algebraic structure of product modifiers

We now state the main theorem of this section, concerning the algebraic structure of product modifiers, using the algebraic structure of friendly modifiers described in Section 7.1.5, and the notations introduced for that purpose. Furthermore, we let $\mathfrak{M}_{\mathbf{c}}^{\mathbf{fp}}$ denote the set of all friendly modifiers $\mathfrak{m}$ such that there exists a product modifier $\mathfrak{m}'$ with $\mathfrak{m}'^{\mathrm{sf}} = \mathfrak{m}^{\mathrm{sf}}$.

Theorem 7. *We have*

- *the mapping* $\overline{\mathbf{mod}}$ *is an isomorphism of operads from* $(\mathcal{PC}, \odot)$ *to* $(\mathfrak{M}_{\mathbf{c}}^{\mathbf{fp}}/_{\sim_s}, \circ/_{\sim_s})$,
- *the mapping* $\mathbf{op}$ *is an isomorphism of operads from* $(\mathcal{PC}, \odot)$ *to* $(\mathfrak{O}^{\mathfrak{p}}, \circ)$,
- *the mapping* $\widehat{\mathbf{desc}_{\mathbf{f}}}$ *is an isomorphism of operads from* $(\mathfrak{M}_{\mathbf{c}}^{\mathbf{fp}}/_{\sim_s}, \circ/_{\sim_s})$ *to* $(\mathfrak{O}^{\mathfrak{p}}, \circ)$.

Furthermore, the operations described by coherent product modifiers are exactly all quasi-boolean operations.

Proof. From Corollary 5, we have $\left(\overline{\mathbf{mod}}\right)^{-1}(\mathfrak{M}_{\mathbf{c}}^{\mathbf{fp}}/_{\sim_s}) \subseteq \mathcal{PC}$. Furthermore, from Proposition 28, we have $\mathbf{op}(\mathcal{PC}) \subseteq \mathfrak{O}^{\mathfrak{p}}$. We also have, from Lemma 29, $\widehat{\mathbf{desc}_{\mathbf{f}}}^{-1}(\mathfrak{O}^{\mathfrak{p}}) \subseteq \mathfrak{M}_{\mathbf{c}}^{\mathbf{fp}}/_{\sim_s}$. Therefore, $\left(\overline{\mathbf{mod}}\right)^{-1}(\widehat{\mathbf{desc}_{\mathbf{f}}}^{-1}(\mathfrak{O}^{\mathfrak{p}})) \subseteq \left(\overline{\mathbf{mod}}\right)^{-1}(\mathfrak{M}_{\mathbf{c}}^{\mathbf{fp}}/_{\sim_s}) \subseteq \mathcal{PC}$. Thus, $\mathfrak{O}^{\mathfrak{p}} = \mathbf{op}(\left(\overline{\mathbf{mod}}\right)^{-1}(\widehat{\mathbf{desc}_{\mathbf{f}}}^{-1}(\mathfrak{O}^{\mathfrak{p}}))) \subseteq \mathbf{op}(\mathcal{PC})$. Hence, we have $\mathbf{op}(\mathcal{PC}) = \mathfrak{O}^{\mathfrak{p}}$. Similarly, we have $\mathfrak{M}_{\mathbf{c}}^{\mathbf{fp}}/_{\sim_s} = \left(\overline{\mathbf{mod}}\right)(\mathcal{PC})$ and $\mathfrak{O}^{\mathfrak{p}} = \widehat{\mathbf{desc}_{\mathbf{f}}}(\mathfrak{M}_{\mathbf{c}}^{\mathbf{fp}}/_{\sim_s})$. As a consequence, by Lemma 25, $\mathfrak{O}^{\mathfrak{p}}$ is the image by $\mathbf{desc}_{\mathbf{f}}$ of $\mathfrak{M}_{\mathbf{c}}^{\mathfrak{p}}$, where $\mathfrak{M}_{\mathbf{c}}^{\mathfrak{p}}$ is the set of all coherent product modifiers.

From Proposition 26 and Proposition 18, $(\mathfrak{M}_{\mathbf{c}}^{\mathfrak{p}}, \circ)$ is an operad. Therefore, from Proposition 6, $(\mathfrak{O}^{\mathfrak{p}}, \circ)$ is an operad. Furthermore, using Proposition 6 again, used with $\widehat{\mathbf{desc}_{\mathbf{f}}}^{-1}$ and $\underline{\mathbf{op}}^{-1}$, we have that $(\mathfrak{M}_{\mathbf{c}}^{\mathbf{fp}}/_{\sim_s}, \circ/_{\sim_s})$ and $(\mathcal{PC}, \odot)$ are also operads. Therefore, as $\mathbf{op}$, $\widehat{\mathbf{desc}_{\mathbf{f}}}$, and $\overline{\mathbf{mod}}$ are morphisms of operads, we have

- the mapping $\overline{\mathbf{mod}}$ is an isomorphism of operads from $(\mathcal{PC}, \odot)$ to $(\mathfrak{M}_{\mathbf{c}}^{\mathbf{fp}}/_{\sim_s}, \circ/_{\sim_s})$,
- the mapping $\mathbf{op}$ is an isomorphism of operads from $(\mathcal{PC}, \odot)$ to $(\mathfrak{O}^{\mathfrak{p}}, \circ)$,
- the mapping $\widehat{\mathbf{desc}_{\mathbf{f}}}$ is an isomorphism of operads from $(\mathfrak{M}_{\mathbf{c}}^{\mathbf{fp}}/_{\sim_s}, \circ/_{\sim_s})$ to $(\mathfrak{O}^{\mathfrak{p}}, \circ)$.

□

We summarize our results for coherent friendly modifiers, coherent product modifiers, and the operations they describe, in Figure 7.8.

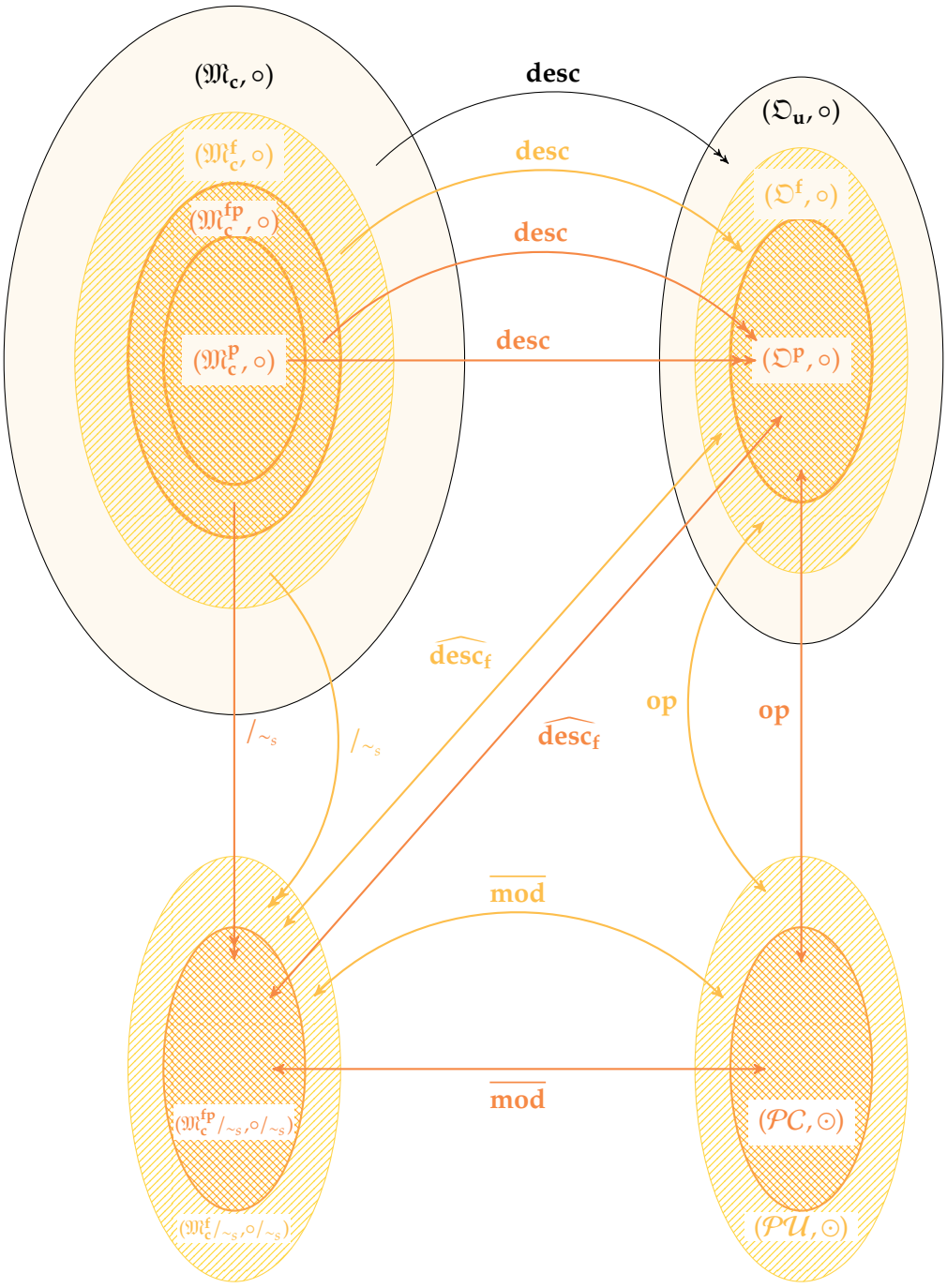


Figure 7.8: The algebraic structure behind friendly modifiers, product modifiers and the operations they describe.

7.3 On the state complexity of friendly operations

We have shown that every k-ary quasi-boolean operation $\circledast$ is described by a product k-modifier. Therefore $\mathrm{sc}_{\otimes}(n1, ..., nk) \leq \prod_{j=1}^{k} n_j$. This upper bound is met, for example, for the intersection of k languages by the k-monster $\mathrm{Mon}_{n_1,\ldots,n_k}^{\{n_1-1\},\ldots,\{n_k-1\}}$, as shown in Proposition 20. However, the case of friendly modifiers is not as easy.

We know that the state complexity of the square root operation [7] is $\mathrm{sc}_{\sqrt{}}(n) = n^n - \binom{n}{2}$, and that it is equal to the state complexity of the operation **Root** [30]. However, our construction of standard modifiers (Definition 31) gives us at first sight an upper bound of $sc_{\otimes}(n) \leq n^n$, for any unary friendly operation $\otimes$. This raises the question of whether the state complexity of some unary friendly operation reaches this bound and, if not, whether we can give an explicit tight bound. Similar questions arise for the general case of k-ary friendly operations with the obvious upper bound $\mathrm{sc}_{\otimes}(n_1, \ldots, n_k) \leq \prod_{j=1}^{k} n_j^{n_j}$ deduced from Definition 31.

7.3.1 The unary case

We show that the bound n^n is not tight for the state complexity of unary friendly operations, and we give an explicit tight bound. We first show that the state complexity of L is at most $n^n - n + 1$.

Proposition 29. *For any positive integer n and any friendly unary operation $\otimes$, we have* $\mathrm{sc}_{\otimes}(n) \leq n^n - n + 1$.

Proof. Consider any subset $E \subseteq \mathcal{U}_1$. Let $\otimes = \mathbf{op}(E)$ and $\mathfrak{m} = \mathbf{mod}(E)$. Let $A = (\Sigma, Q, i, F, \alpha)$ be a DFA with size $n \in \mathbb{N} \setminus 0$. For all $s, t \in Q$, we let $g_{s,t}$ denote the function of Q^Q such that

$$g_{s,t}(j) = \begin{cases} s & \text{if } j \in F \\ t & \text{otherwise,} \end{cases}$$

and by G the set of all functions $g_{s,t}$, for $s, t \in Q$. Furthermore, we let $\mathbf{0}$ denote the sequence $(0, 0, \ldots)$ of $\mathcal{U}_1$, we let $\mathbf{0}^1$ denote the sequence

$$(0, 1, 1, \ldots, 1, \ldots),$$

we let **odd** denote the sequence

$$(0, 1, 0, 1, \ldots, n \bmod 2, \ldots),$$

and we let **even** denote the sequence

$$(1, 0, 1, 0, \ldots, (n+1) \bmod 2, \ldots).$$

It follows from the definition of $g_{s,t}$ that, for any $\zeta \in Q^Q$ and for any $g_{s,t} \in G$, we have $\zeta \circ g_{s,t} = g_{\zeta(s),\zeta(t)}$. We use this property extensively in the rest of the proof, and we will refer

to it by saying that G is *stable by external composition*. We show that $\mathrm{sc}(\mathrm{L}(\mathfrak{m}(A)))$ is at most $n^n - n + 1$, by studying the Nerode equivalence relation induced by A. We distinguish two main cases, $i \in F$ and $i \notin F$.

We first suppose that $i \notin F$. Notice that we have

- if $t \in F$, then
  - if $s \in F$, then we have $\chi(Q, i, F, g_{s,t}) = \mathbf{0}^1$,
  - otherwise if $s \notin F$, then we have $\chi(Q, i, F, g_{s,t}) = \mathbf{odd}$
- otherwise if $t \notin F$, then $\chi(Q, i, F, g_{s,t}) = \mathbf{0}$.

Let $E_1 = \{\mathbf{0}, \mathbf{0}^1, \mathbf{odd}\} \cap E$ and $E_2 = \{\mathbf{0}, \mathbf{0}^1, \mathbf{odd}\} \setminus E_1$. We distinguish the following cases:

- If $\#E_1 = 0$ (respectively $\#E_1 = 3$), then for any $s, t \in Q$, since $\chi(Q, i, F, g_{s,t}) \notin E$ (respectively $\chi(Q, i, F, g_{s,t}) \in E$) , the state $g_{s,t}$ is not final in $\mathfrak{m}(A)$ (respectively final in $\mathfrak{m}(A)$). Since G is stable by external composition, all the states in G are in the same Nerode equivalence class. Therefore, $\mathrm{sc}(\mathrm{L}(\mathfrak{m}(A))) \leq n^n - n^2 + 1 \leq n^n - n + 1$.

- Otherwise if $\#E_1 = 1$ (respectively $\#E_1 = 2$), then we let u denote the unique element of $\#E_1$ (respectively $\#E_2$).
  - Suppose that $u = \mathbf{odd}$. For any positive integer p and any state $q \in Q$, $g^p_{s,s}(q) = g_{s,s}(q)$. Thus, for any $s \in Q$, we have $\chi(Q, i, F, g_{s,s}) \in \{\mathbf{0}, \mathbf{0}^1\}$. Therefore, the stability of G by external composition implies that two states $g_{s,s}$ and $g_{s',s'}$, with $s, s' \in Q$, are not distinguishable in $\mathfrak{m}(A)$, for any $s, s' \in Q$. As a consequence, $\mathrm{sc}(\mathrm{L}(\mathfrak{m}(A))) \leq n^n - n + 1$.
  - Suppose that $u = \mathbf{0}^1$, and let s, t be two elements of Q. If $\chi(Q, i, F, g_{s,t}) = \mathbf{0}^1$, then $t, s \in F$, which implies that $\chi(Q, i, F, g_{t,s}) = \mathbf{0}^1$. Furthermore, similarly, if $\chi(Q, i, F, g_{t,s}) = \mathbf{0}^1$, then $\chi(Q, i, F, g_{s,t}) = \mathbf{0}^1$. As a consequence, $\chi(Q, i, F, g_{s,t}) = \mathbf{0}^1$ if and only if $\chi(Q, i, F, g_{t,s}) = \mathbf{0}^1$. Therefore, the stability of G by external composition implies that the two states $g_{s,t}$ and $g_{t,s}$ are not distinguishable, for any $s, t \in Q$. As a consequence, we have $\mathrm{sc}(\mathrm{L}(\mathfrak{m}(A))) \leq n^n - \frac{1}{2}n(n-1) \leq n^n - n + 1$.
  - Finally, suppose that $u = \mathbf{0}$, and let s, s', t be three elements of Q. If $\chi(Q, i, F, g_{s,t}) = \mathbf{0}$, then $t \notin F$, which implies that $\chi(Q, i, F, g_{s',t}) = \mathbf{0}$. Furthermore, similarly, $\chi(Q, i, F, g_{s',t}) = \mathbf{0}$ implies that $\chi(Q, i, F, g_{s,t}) = \mathbf{0}$. As a consequence, $\chi(Q, i, F, g_{s,t}) = \mathbf{0}$ if and only if $\chi(Q, i, F, g_{s',t}) = \mathbf{0}$. Therefore, the stability of G by external composition implies that the two states $g_{s,t}$ and $g_{s',t}$ are not distinguishable in $\mathfrak{m}(A)$, for any $s, s' \in Q$. As a consequence, we have $\mathrm{sc}(\mathrm{L}(\mathfrak{m}(A))) \leq n^n - n(n-1) \leq n^n - n + 1$.

The case $i \in F$ is symmetrical to the case $i \notin F$ in the following way: we replace in the proof all the occurrences of $s \in F$ by $t \notin F$, of $s \notin F$ by $t \in F$, of $t \in F$ by $s \notin F$, of $t \notin F$ by $s \in F$, of $\mathbf{0}$ by $(1, 1, \ldots)$, of $\mathbf{0}^1$ by $(1, 0, 0, \ldots, 0, \ldots)$, and of $\mathbf{odd}$ by $(1, 0, 1, 0, \ldots, (n+1) \bmod 2, \ldots)$. Furthermore, in the case of $u = (1, 1, \ldots)$, it is the finality of $g_{s,t}$ and $g_{s,t'}$ that is the same.

To summarize, in all the cases, we have $\mathrm{sc}(\mathrm{L}(\mathfrak{m}(A))) \leq n^n - n + 1$. Hence, $\mathrm{sc}_{\otimes}(n) \leq n^n - n + 1$, for any friendly unary operation $\otimes$.

□

We now show that this bound is tight for $\circledast_1 = \mathbf{op}(\{\mathbf{0}, \mathbf{0}^1\})$, where $\mathbf{0} = (0, 0, \ldots)$ and $\mathbf{0}^1 = (0, 1, 1, \ldots, 1, \ldots)$. Notice that, for any regular language L over alphabet Σ, if $\varepsilon \notin L$, then we have

$$\circledast_1(L) = \{w \in \Sigma^\star \mid w \in \sqrt[k]{L}, \text{ for any } k > 0\} \cup \{w \in \Sigma^\star \mid w \notin \sqrt[k]{L} \text{ for any } k > 0\},$$

and if $\varepsilon \in L$, then we have $\circledast_1(L) = \emptyset$. Let $\mathfrak{w}_1 = \mathbf{mod}(\{\mathbf{0}, \mathbf{0}^1\})$. We let A_n denote the DFA $\mathfrak{w}_1(\mathrm{Mon}_n^{\{n-1\}})$, for any integer n. We determine a lower bound for the state complexity of $\circledast_1$ by computing the minimal DFA equivalent to A_n. Recall, by Definition 22, Definition 31 and Definition 34, that the alphabet of A_n is $\Gamma_n = [\![n]\!]^{[\![n]\!]}$, that its set of states is also $[\![n]\!]^{[\![n]\!]}$, and that every state ϕ of A_n is accessible from its initial state $\mathrm{Id}_{[\![n]\!]}$ by reading the letter ϕ. For any function $\phi \in [\![n]\!]^{[\![n]\!]}$, we let $\kappa(\phi)$ denote the characteristic sequence $\chi([\![n]\!], 0, \{n-1\}, \phi)$. To compute the Nerode equivalence induced by A_n, we show the following result.

Lemma 30. *For any positive integer n, and every two distinct functions $\phi, \psi \in [\![n]\!]^{[\![n]\!]}$ such that ψ is non-constant, there exists $\zeta \in [\![n]\!]^{[\![n]\!]}$ such that $\kappa(\zeta \circ \phi) \in \{\mathbf{0}, \mathbf{0}^1\}$ if and only if $\kappa(\zeta \circ \psi) \notin \{\mathbf{0}, \mathbf{0}^1\}$.*

Proof. We consider two main cases, $\phi(0) \neq \psi(0)$ and $\phi(0) = \psi(0)$.

Suppose that $\phi(0) \neq \psi(0)$. If $\psi(0) \neq \psi(n-1)$, then we set $\zeta(\phi(0)) = \zeta(\psi(n-1)) = 0$ and $\zeta(\psi(0)) = n - 1$, and this implies that $\kappa(\zeta \circ \phi) = \mathbf{0}$ and $\kappa(\zeta \circ \psi) = (0, 1, 0, \ldots) \notin \{\mathbf{0}, \mathbf{0}^1\}$. Symmetrically, if $\phi(0) \neq \phi(n-1)$ then we obtain the same result by permuting the role of ψ and ϕ in the previous case. Now suppose that $\phi(0) = \phi(n-1)$ and $\psi(0) = \psi(n-1)$. As ψ is not constant, there exists $i \geq 1$ such that $\psi(n-1) \neq \psi(i)$. We set $\zeta(\phi(0)) = \zeta(\psi(i)) = n - 1$ and $\zeta(\psi(0)) = i$, which implies that $\kappa(\zeta \circ \phi) = \mathbf{0}^1$ and $\kappa(\zeta \circ \psi) = (0, 0, 1, \ldots) \notin \{\mathbf{0}, \mathbf{0}^1\}$.

Now suppose that $\phi(0) = \psi(0)$. Then there exists $j > 0$ such that $\phi(j) \neq \psi(j)$. We have $\phi(j) \neq \phi(0)$ or $\psi(j) \neq \psi(0)$. Suppose that $\phi(j) \neq \phi(0)$ (the other case can be treated symmetrically). If $j < n - 1$, then we set $\zeta(\phi(0)) = \zeta(\psi(j)) = j$ and $\zeta(\phi(j)) = n - 1$. In that case, we have $\kappa(\zeta \circ \phi) = (0, 0, 1, \ldots) \notin \{\mathbf{0}, \mathbf{0}^1\}$ and $\kappa(\zeta \circ \psi) = \mathbf{0}$. Otherwise if $j = n - 1$, then we set $\zeta(\phi(0)) = \zeta(\psi(n-1)) = n - 1$ and $\zeta(\phi(n-1)) = 0$, which implies that $\kappa(\zeta \circ \phi) = (0, 1, 0, \ldots) \notin \{\mathbf{0}, \mathbf{0}^1\}$ and $\kappa(\zeta \circ \psi) = \mathbf{0}^1$. This concludes the proof. □

By Definition 34, the above lemma implies that any two distinct states of A_n such that at least one of them is non-constant are distinguishable. Therefore, any non-constant state is distinguishable from every other state and the size of the minimal DFA equivalent to A_n is at least equal to the cardinality of the set of all mappings over $[\![n]\!]$ that are not constant. Thus, for every $n \in \mathbb{N} \setminus 0$, the size of the minimal DFA equivalent to A_n is at least $n^n - n + 1$. Hence, we have $\mathrm{sc}_{\circledast_1}(n) \geq n^n - n + 1$, for any positive integer n. As a consequence, from Proposition 29, we have $\mathrm{sc}_{\circledast_1}(n) = n^n - n + 1$, for any positive integer n. We have thus proved the following theorem.

Theorem 8. *For any positive integer n and any friendly operation $\otimes$, $\mathrm{sc}_{\otimes}(n) \leq n^n - n + 1$, and the bound is tight for $\circledast_1$,* i.e., *we have $\mathrm{sc}_{\circledast_1}(n) = n^n - n + 1$. Furthermore, $\mathrm{L}(\mathrm{Mon}_n^{\{n-1\}})$ is a witness for $\circledast_1$.*

7.3.2 The general case

Surprisingly, unlike the unary case, we show that there are friendly operations which state complexity meets the upper bound $\prod_{j=1}^{k} n_j^{n_j}$. We assume that $k \geq 2$, and we let $\circledast_k$ denote the k-ary operation $\mathbf{op}(E_k)$, where $E_k = \{\mathbf{0}, \mathbf{0}^1\}^k \setminus \{(\mathbf{0}, \ldots, \mathbf{0})\}$. For any k-tuple of positive integers $(n_1, \ldots, n_k)$, we let $A_{n_1,\ldots,n_k}$ denote the DFA $\circledast_k(\mathrm{Mon}_{n_1,\ldots,n_k}^{\{n_1-1\},\ldots,\{n_k-1\}})$. Furthermore, for any k-tuple of functions $(\phi_1, \ldots, \phi_k)$, with $\phi_j \in [\![n_j]\!]^{[\![n_j]\!]}$ for any $j \in \{1, \ldots, k\}$, we let $\kappa(\phi_1, \ldots, \phi_k)$ denote the characteristic sequence

$$\chi(([\![n_1]\!], 0, \{n_1 - 1\}, \phi_1), \ldots, ([\![n_k]\!], 0, \{n_k - 1\}, \phi_k)).$$

Theorem 9. *For any integer $k \geq 2$ and for any k-tuple of positive integers $(n_1, \ldots, n_k)$, we have $\mathrm{sc}_{\circledast_k}(n_1, \ldots, n_k) = \prod_{j=1}^{k} n_j^{n_j}$. Furthermore, $\mathrm{sc}_{\circledast_k}(\mathrm{L}(A_{n_1,\ldots,n_k})) = \prod_{j=1}^{k} n_j^{n_j}$, i.e., $(\mathrm{L}(\mathbb{M}_1), \ldots, \mathrm{L}(\mathbb{M}_k))$ is a witness for $\circledast_k$, where $(\mathbb{M}_1, \ldots, \mathbb{M}_k) = \mathrm{Mon}_{n_1,\ldots,n_k}^{\{n_1-1\},\ldots,\{n_k-1\}}$.*

Proof. Our proof is inspired by the unary case. Let k be an integer with $k \geq 2$, and let $(n_1, \ldots, n_k)$ be a k-tuple of positive integers. Furthermore, let $(\phi_1, \ldots, \phi_k)$ and $(\psi_1, \ldots, \psi_k)$ be two k-tuple of mappings, with $\phi_j, \psi_j \in [\![n_j]\!]^{[\![n_j]\!]}$ for any $j \in \{1, \ldots, k\}$, such that $(\phi_1, \ldots, \phi_k) \neq (\psi_1, \ldots, \psi_k)$. We show that there exists $(\zeta_1, \ldots, \zeta_k)$, with $\zeta_j \in [\![n_j]\!]^{[\![n_j]\!]}$ for any $j \in \{1, \ldots, k\}$, such that $\kappa(\zeta_1 \circ \phi_1, \ldots, \zeta_k \circ \phi_k) \in E_k$ if and only if $\kappa(\zeta_1 \circ \psi_1, \ldots, \zeta_k \circ \psi_k) \notin E_k$.

Let ℓ such that $\phi_\ell \neq \psi_\ell$. We consider two cases.

- If both ϕ_ℓ and ψ_ℓ are constant functions, then we let ζ_ℓ denote any mapping over $[\![n_\ell]\!]$ such that $\zeta_\ell(\phi(0)) = 0$ and $\zeta_\ell(\psi(0)) = n_\ell - 1$. Furthermore, for any $j \neq \ell$, we let ζ_j denote the constant function sending any element to 0. We have

  $$\kappa(\zeta_1 \circ \phi_1, \ldots, \zeta_k \circ \phi_k) = (\mathbf{0}, \ldots, \mathbf{0}) \notin E_k,$$

  and

  $$\kappa(\zeta_1 \circ \psi_1, \ldots, \zeta_k \circ \psi_k) = (u_1, \ldots, u_k) \in E_k, \text{ where } u_j = \begin{cases} \mathbf{0} & \text{if } j \neq \ell \\ \mathbf{0}^1 & \text{if } j = \ell \end{cases}$$

- If one of the functions ϕ_ℓ and ψ_ℓ is not constant, then we can suppose that ψ_ℓ is not constant (the other case is symmetrical). Therefore, by Lemma 30, there exists a mapping ζ_ℓ over $[\![n_\ell]\!]$ such that $\kappa(\zeta_\ell \circ \phi_\ell) \in \{\mathbf{0}, \mathbf{0}^1\}$ if and only if $\kappa(\zeta_\ell \circ \psi_\ell) \notin \{\mathbf{0}, \mathbf{0}^1\}$. We assume that $\kappa(\zeta_\ell \circ \phi_\ell) \in \{\mathbf{0}, \mathbf{0}^1\}$ (the other case is symmetrical). Furthermore, for any $j \in \{1, \ldots, k\}$ with $j \neq \ell$, we let ζ_j denote the constant function sending any element to $n_j - 1$. We have

  $$\kappa(\zeta_1 \circ \psi_1, \ldots, \zeta_k \circ \psi_k) = (\mathbf{0}^1, \ldots, \mathbf{0}^1, \kappa(\zeta_\ell \circ \psi_\ell), \mathbf{0}^1, \ldots, \mathbf{0}^1) \notin E_k,$$

  since $\kappa(\zeta_\ell \circ \psi_\ell) \notin \{\mathbf{0}, \mathbf{0}^1\}$. Furthermore, we have

  $$\kappa(\zeta_1 \circ \phi_1, \ldots, \zeta_k \circ \phi_k) = (\mathbf{0}^1, \ldots, \mathbf{0}^1, \kappa(\zeta_\ell \circ \phi_\ell), \mathbf{0}^1, \ldots, \mathbf{0}^1) \in E_k,$$

  because $\kappa(\zeta_\ell \circ \phi_\ell) \in \{\mathbf{0}, \mathbf{0}^1\}$.

Therefore, in both cases, there exists $(\zeta_1, \ldots, \zeta_k)$, with $\zeta_j \in [\![n_j]\!]^{[\![n_j]\!]}$ for any $j \in \{1, \ldots, k\}$, such that $\kappa(\zeta_1 \circ \phi_1, \ldots, \zeta_k \circ \phi_k) \in E_k$ if and only if $\kappa(\zeta_1 \circ \psi_1, \ldots, \zeta_k \circ \psi_k) \notin E_k$. We conclude with Definition 22, Definition 31, and Definition 34. □

7.3.3 On the size of the witnesses' alphabets

Witnesses usually given for regular operations have a finitely bounded alphabet. However, the witnesses given in Theorem 9 and in Theorem 8 are monsters, and thus have exponential alphabet size. We show that every k-ary friendly operation actually has a witness with an alphabet of size $3k$. This is not hard to prove, and comes directly from the definition of friendly modifiers. Recall that the definition of $\Gamma'_{n_1,\dots,n_k}$ is given in Definition 1.

Definition 41. *Let k be a non-negative integer, $(n_1,\dots,n_k)$ be a k-tuple of positive integers, $(F_1,\dots,F_k)$ be a k-tuple of sets with $F_j \subseteq [\![n_j]\!]$, for any $j \in \{1,\dots,k\}$, and let $(\mathbb{M}_1,\dots,\mathbb{M}_k) = \mathrm{Mon}^{F_1,\dots,F_k}_{n_1,\dots,n_k}$. We let $B^{F_1,\dots,F_k}_{n_1,\dots,n_k}$ denote the k-tuple of automata $(\mathbb{B}_1,\dots,\mathbb{B}_k)$, where, for any $j \in \{1,\dots,k\}$, the DFA $\mathbb{B}_j$ is equal to the DFA $\mathbb{M}_j$ restricted to the alphabet $\Gamma'_{n_1,\dots,n_k}$.*

Theorem 10. *Let k be a non-negative integer, $\otimes$ be a friendly operation, $(n_1,\dots,n_k)$ be a k-tuple of positive integers, and let $\mathfrak{m}$ be a friendly modifier that describes $\otimes$. There exists a k-tuple of sets $(F_1,\dots,F_k)$, with $F_j \subseteq [\![n_j]\!]$ for any $j \in \{1,\dots,k\}$, such that $\mathrm{sc}(\mathrm{L}(\mathfrak{m}(B^{F_1,\dots,F_k}_{n_1,\dots,n_k}))) = \mathrm{sc}_\otimes(n_1,\dots,n_k)$, i.e., such that $B^{F_1,\dots,F_k}_{n_1,\dots,n_k}$ is a witness for $\otimes$.*

Proof. Let k be a non-negative integer, $\otimes$ be a friendly operation, $(n_1,\dots,n_k)$ be a k-tuple of positive integers, and let $\mathfrak{m}$ be a friendly modifier that describes $\otimes$. Recall that, by Proposition 1, the monoid $(\Gamma_{n_1,\dots,n_k},\circ)$ is generated by the subset $\Gamma'_{n_1,\dots,n_k}$ of size $3k$.

From Theorem 2, there exists $(F_1,\dots,F_k)$, with $F_j \subseteq [\![n_j]\!]$ for any $j \in \{1,\dots,k\}$, such that $\mathrm{sc}(\mathrm{L}(\mathfrak{m}(M^{F_1,\dots,F_k}_{n_1,\dots,n_k}))) = \mathrm{sc}_\otimes(n_1,\dots,n_k)$. We let $A = (\Sigma, Q, i, F, \delta)$ denote the DFA $\mathfrak{m}(\mathrm{Mon}^{F_1,\dots,F_k}_{n_1,\dots,n_k})$. Recall that from Definition 22, the alphabet of all DFAs of the k-tuple $\mathrm{Mon}^{F_1,\dots,F_k}_{n_1,\dots,n_k}$, and therefore the alphabet Σ of A, is equal to $\Gamma_{n_1,\dots,n_k} = [\![n_1]\!]^{[\![n_1]\!]} \times \cdots \times [\![n_k]\!]^{[\![n_k]\!]}$.

From Proposition 1, for each ψ in $\Gamma_{n_1,\dots,n_k}$, there exists a non-negative integer m and a finite sequence $(\phi_\ell)_{\ell\in\{1,\dots,m\}}$ of elements of $\Gamma'_{n_1,\dots,n_k}$, such that $\psi = \phi_1 \circ \cdots \circ \phi_m$. Therefore, since $\mathfrak{m}$ is a friendly modifier, we have $\delta^\psi = \delta^{\phi_1} \circ \cdots \circ \delta^{\phi_m}$. As a consequence, any reachable state of A is reachable only by reading the letters of $\Gamma'_{n_1,\dots,n_k}$, and any two distinguishable states of A are distinguishable only by reading the letters of $\Gamma'_{n_1,\dots,n_k}$. Hence, $\mathrm{sc}(\mathrm{L}(A)) = \mathrm{sc}(\mathrm{L}(B^{F_1,\dots,F_k}_{n_1,\dots,n_k}))$, and therefore $\mathrm{sc}(\mathrm{L}(\mathfrak{m}(B^{F_1,\dots,F_k}_{n_1,\dots,n_k}))) = \mathrm{sc}_\otimes(n_1,\dots,n_k)$. □

Chapter 8

Conclusion

To this day, computing the state complexity of an operation remains a messy, ad-hoc business. Even though the state complexity of many well-known operations has been computed, we seem to start all over again every time we switch to a new operation. This still holds true for combinations of two or more operations, even when every one of their state complexities is already known. As the operations we study become more complicated, so do our computations, proofs, and results. To keep them from becoming too long and heavy, we need a new framework and some new general results. Even though the road ahead is still long, we have taken the first steps towards that goal.

We began our efforts by bringing in a tool used to classify algebras, operads. Operads are useful structures to formalize results around the composition of operations. In particular, any set of operations stable by composition is an operad. Furthermore, other kind of operads can be defined to understand the structure of such a set of operations. Using this notion, we introduced a new framework to help compute the state complexity of operations. The regular operations that fall into the scope of our framework are called 1-uniform, and form an operad $(\mathfrak{O}_{\mathbf{u}}, \circ)$. This operad has a counterpart in the space of operations over DFAs, the operad of modifiers $(\mathfrak{M}, \circ)$. However, modifiers themselves are not so easy to handle. Therefore, we extracted the essential information contained in modifiers by defining a set of operations over transition configurations, denoted by $\mathfrak{FT}$. We used this results to compute the state complexity of 1-uniform operations. To that aim, we defined large DFAs called monsters. Every 1-uniform operation admits a monster as witness. Therefore, when searching for the state complexity of a 1-uniform operation, we need only concern ourselves with giving a modifier that describes that 1-uniform operation, and applying it to monsters.

Using this framework, we designed a method to compute the state complexities of 1-uniform operations, and showed that it works on simple examples like the Kleene star, catenation, and boolean operations. Furthermore, we used this method to compute the state complexity of a new operation, the Kleene star composed with symmetric difference. Combined with previous results [16, 27], this allowed us to compute the state complexity of every binary boolean operation. We then tried to tackle the more general problem consisting in computing the state complexity of the star composed with any boolean operation. We explained how our approach in the particular case of the star composed with symmetric difference may be generalized. We firmly believe that, with some more

work, it would be possible to devise a close link between the state complexity of the star of boolean operations and combinatorial objects involving tableaux and cubes in higher dimension. Counting these objects, however, is another problem entirely, that we have yet to explore.

Finally, we studied two suboperads of $(\mathfrak{O}_{\mathbf{u}}, \circ)$ defined with simple algebraic properties, the operad of friendly operations, and the operad of quasi-boolean operations. We showed that their simple definition is reflected by their clear structure. Friendly modifiers describe exactly all compositions of boolean operations of infinite arity and roots, and product modifiers describe a weak generalization of boolean operations. We studied in detail the algebraic structure underlying these two operads. Using these results, we found the maximum state complexity of k-ary friendly operations, for any positive integer k. To get this bound, it is possible to use witnesses with only $3k$ letters.

Our work on friendly modifiers and product modifiers is an example of a more general technique that could be used to study the state complexity of operations. This technique consists in putting a particular operation into a larger family of operations, and then to get some information about the state complexity of that particular operation by studying the entire family. For example, we are now certain that the state complexity of any unary friendly operation is at most $n^n - n + 1$; however, this result would not necessarily be obvious to prove for a particular unary friendly operation. Claim 1 is another example of application of this technique. This claim could give non-trivial upper bounds for the state complexity of the star of any boolean operation, without the need to invent a new method for every single one of these operations. To summarize, we should not be afraid to study large families of operations, as this may yield better results than studying only some particular and isolated operations.

One of the most interesting things about friendly operations is that their algebraic structure is simple enough that we have some hope of answering general questions on their state complexity. For example, can their state complexity attain a growth somewhere between exponential and linear? If so, what ranges of growth for their state complexities can we obtain? We could also try to compute their maximal state complexity over a unary or binary alphabet, thereby carrying on the work done in [30]. Giving witnesses for these operations with a smaller alphabet is also another possible line of research. There are probably some k-ary friendly operations whose witnesses always have an alphabet of size greater than k. But do they all have a witness of size $2k$, or even maybe of size k plus a constant?

It is worth noting that the Kleene star, catenation, boolean operations, roots, and all of their compositions are not the only well-known operations that are 1-uniform. It is also the case, for example, of the shuffle, the cyclic shift, and powers. As a consequence, the framework we developed could also be used to study these operations and all of their compositions. However, several well-known operations do not fall into the scope of our framework. This is the case, for example, of the right quotient or of the proportional removals. Another limitation of our framework is that there is no systematic way of reducing the alphabets of the witnesses we use, which are of exponential size. Even though, for some operations like the shuffle [3], witnesses have an alphabet of at least linear size, we do not know of any operation that would require an alphabet of exponential size. Does there exist any? From this question arises a notion that one may call "alphabetic complex-

ity". The alphabetic complexity of a 1-uniform operation would be the minimal size of the alphabets of all the witnesses of that operation. Numerous other simple questions about alphabetic complexity are left unanswered. For example, can we characterize 1-uniform operations that have a finitely bounded alphabetic complexity? What is the maximal alphabetic complexity of all 1-uniform operations? How does alphabetic complexity behave with respect to composition?

We have sought recently to obtain some results about alphabetic complexity for some well-defined classes of operations. This line of research could also be considered another extension of our work on friendly and product modifiers, as its main goal is to study other operads, similarly defined by simple algebraic properties. To be more precise, for any modifier $\mathfrak{m} = [\mathfrak{Q}, \mathfrak{f}, \mathfrak{i}, \mathfrak{d}]$, the goal would be to study what some well-chosen algebraic properties of the application $\mathfrak{d}$ entail on the operation that $\mathfrak{m}$ describes. When $\mathfrak{d}$ is a morphism for its third coordinate, we have seen that $\mathfrak{m}$ describes a friendly operation. But what happens when $\mathfrak{d}$ only behaves like a morphism when its inputs satisfy some restrictive property? This is the case for several well-known regular operations, like the Kleene star and catenation. For example, for the Kleene star, we have

$$\mathfrak{d}(Q, i, F, \phi \circ \psi) = \mathfrak{d}(Q, i, F, \phi) \circ \mathfrak{d}(Q, i, F, \psi),$$

when $\psi(i) = i$ and $\psi(Q \setminus F) \subseteq Q \setminus F$. We are currently exploring whether operations satisfying some similar properties always have witnesses with alphabets of linear size, or even maybe of finitely bounded size, *i.e.*, whether their alphabetic complexity is linear, or even maybe finitely bounded. We have made some encouraging progress towards showing that their alphabetic complexity is indeed at most linear, but we cannot state this result with certainty yet. That result would be quite general as it would also encompass, for example, all compositions of Kleene star, catenation, union and intersection.

Another interesting line of research would be to adapt our framework to other types of automata, like non-deterministic finite automata, or alternating finite automata. We could even try to extend our results to infinite automata. In fact, the main idea of the method we developed does not seem to be specific to a certain type of automata, and it would be interesting to have a more general theory encompassing many types of automata.

To summarize, the in-depth study of the state complexity of operations in recent years consisted in trying to compute the state complexity of some specific regular operations, well-known for other purposes in automata theory. However, we believe that looking at the wider picture with an algebraic point of view would be fruitful. In other words, instead of trying to compute the state complexity of increasingly complex particular and isolated operations, we could try to define new classes of operations, tailor-made to comprise operations close enough to some operations already well-known, but simple enough for us to compute their state complexity. This would help us start a more general theory, that may eventually lead to computing the state complexity of more complex operations. For example, a simple question that remains to be answered is the following: can we define a large class of operations, stable by composition, and that contains an interesting operation that is not boolean (or quasi-boolean), so that we are able to compute the state complexity of every one of its operations?

List of Notations and Symbols

This is a list of the most frequently used notations and symbols.

- $[\![n]\!]$: The set $\{0, \dots, n-1\}$
- $\Gamma_{n_1,\dots,n_k}$: The set $[\![n_1]\!]^{[\![n_1]\!]} \times \cdots \times [\![n_k]\!]^{[\![n_k]\!]}$
- $\widetilde{q}$: The set of all $q' \in E$ such that $q \sim q'$, where $\sim$ is an equivalence relation over E
- $E/_\sim$: The set of all $\widetilde{q}$, when $q \in E$
- $\circ$: See Definition 6
- $\odot/_\sim$: See Definition 11
- $\sim^\phi$ when ϕ is a morphism of operads: See Definition 12
- $\widehat{\phi}$: See Definition 12
- $[\mathfrak{Q}, \mathfrak{i}, \mathfrak{f}, \mathfrak{d}]$: See Definition 19
- $\mathfrak{M}_{\mathbf{c}}$: The set of all coherent modifiers (Definition 20)
- $\mathfrak{O}_{\mathbf{u}}$: The set of all 1-uniform operations
- $\mathfrak{M}^{\mathbf{f}}$; The set of all friendly modifiers (Definition 30)
- $\mathfrak{M}^{\mathbf{f}}_{\mathbf{c}}$: The set of all coherent friendly modifiers (Definitions 20 and 30)
- $\mathfrak{O}^{\mathbf{f}}$: The set of all friendly operations (Definition 36)
- $\mathrm{Mon}^{F_1,\dots,F_k}_{n_1,\dots,n_k}$: A monster, see Definition 22
- $\mathfrak{m}^{\mathrm{sf}}$: See Definition 32
- $\mathfrak{M}^{\mathbf{s}}_{\mathbf{c}}$: The set of all coherent standard friendly modifiers (Definitions 20 and 31)
- $\chi(t_1, \dots, t_k)$: See Definition 33
- $\mathcal{V}_k$: The set of all k-tuples of sequences with values in $\{0, 1\}$
- $\mathcal{U}_k$: The set of all $(u_1, \dots, u_k)$ in $\mathcal{V}_k$ such that every u_j is eventually periodic

- $\mathcal{PU}$: The set $\bigcup_{k\in\mathbb{N}} 2^{\mathcal{U}_k}$ (Definition 39)
- $\mathfrak{M}^{\mathfrak{p}}$: The set of all product modifiers (Definition 39)
- $\mathfrak{M}_{\mathfrak{c}}^{\mathfrak{p}}$: The set of all coherent product modifiers (Definitions 20 and 39)
- $\mathfrak{M}_{\mathfrak{c}}^{\mathfrak{fp}}$: The set of all friendly modifiers $\mathfrak{m}$ such that there exists a coherent product modifier $\mathfrak{m}'$ with $\mathfrak{m}^{\mathfrak{sf}} = \mathfrak{m}'^{\mathfrak{sf}}$ (Definitions 20, 39 and 32)
- $\mathfrak{O}^{\mathfrak{p}}$: The set of all quasi-boolean operations (Definition 40)
- $\mathcal{PC}_k$: The set of all E in $2^{\mathcal{U}_k}$ such that, if $(u_1, \ldots, u_k)$ is in E, then any k-tuple of sequences $(v_1, \ldots, v_k) \in \mathcal{U}_k$, with $v_{j,0} = u_{j,0}$ and $v_{j,1} = v_{j,0}$ for any j, is also in E
- $\mathcal{PC}$: The set $\bigcup_{k\in\mathbb{N}} \mathcal{PC}_k$
- **mod**: See Definition 34
- **op**: See Definition 38
- **desc**: See Definition 21
- $\mathbf{desc}_{\mathfrak{f}}$: The restriction of **desc** to friendly modifiers (Definitions 21 and 30)
- $\sim^{\mathfrak{s}}$: The equivalence relation over friendly modifiers such that $\mathfrak{m} \sim^{\mathfrak{s}} \mathfrak{m}'$ if and only if $\mathfrak{m}^{\mathfrak{sf}} = \mathfrak{m}'^{\mathfrak{sf}}$
- $\overline{\mathbf{mod}}$: The application such that, for any $E \in \mathcal{PU}$, we have $\overline{\mathbf{mod}}(E) = \widetilde{\mathbf{mod}(E)}^{\mathfrak{s}}$

Bibliography

[1] Jean-Camille Birget. Intersection and union of regular languages and state complexity. *Inf. Process. Lett.*, 43(4):185–190, 1992.

[2] Janusz A. Brzozowski. In search of most complex regular languages. *Int. J. Found. Comput. Sci.*, 24(6):691–708, 2013.

[3] Janusz A. Brzozowski, Galina Jirásková, Bo Liu, Aayush Rajasekaran, and Marek Szykula. On the state complexity of the shuffle of regular languages. In Cezar Câmpeanu, Florin Manea, and Jeffrey O. Shallit, editors, *Descriptional Complexity of Formal Systems - 18th IFIP WG 1.2 International Conference, DCFS 2016, Bucharest, Romania, July 5-8, 2016. Proceedings*, volume 9777 of *Lecture Notes in Computer Science*, pages 73–86. Springer, 2016.

[4] Pascal Caron, Edwin Hamel-De le Court, and Jean-Gabriel Luque. Algebraic and combinatorial tools for state complexity: Application to the star-xor problem. In Jérôme Leroux and Jean-François Raskin, editors, *Proceedings Tenth International Symposium on Games, Automata, Logics, and Formal Verification, GandALF 2019, Bordeaux, France, 2-3rd September 2019*, volume 305 of *EPTCS*, pages 154–168, 2019.

[5] Pascal Caron, Edwin Hamel-De le Court, and Jean-Gabriel Luque. The state complexity of a class of operations involving roots and boolean operations. *CoRR*, abs/2004.12958, 2020.

[6] Pascal Caron, Edwin Hamel-De le Court, and Jean-Gabriel Luque. A study of a simple class of modifiers: Product modifiers. In Natasa Jonoska and Dmytro Savchuk, editors, *Developments in Language Theory - 24th International Conference, DLT 2020, Tampa, FL, USA, May 11-15, 2020, Proceedings*, volume 12086 of *Lecture Notes in Computer Science*, pages 110–121. Springer, 2020.

[7] Pascal Caron, Edwin Hamel-De le Court, Jean-Gabriel Luque, and Bruno Patrou. New tools for state complexity. *Discret. Math. Theor. Comput. Sci.*, 22(1), 2020.

[8] Pascal Caron, Jean-Gabriel Luque, Ludovic Mignot, and Bruno Patrou. State complexity of catenation combined with a boolean operation: A unified approach. *Int. J. Found. Comput. Sci.*, 27(6):675–704, 2016.

[9] Pascal Caron, Jean-Gabriel Luque, and Bruno Patrou. State complexity of multiple catenations. *Fundam. Inform.*, 160(3):255–279, 2018.

[10] Pascal Caron, Jean-Gabriel Luque, and Bruno Patrou. State complexity of combined operations involving catenation and binary boolean operations: Beyond the Brzozowski conjectures. *Theor. Comput. Sci.*, 800:15–30, 2019.

[11] Bo Cui, Yuan Gao, Lila Kari, and Sheng Yu. State complexity of two combined operations: Catenation-union and catenation-intersection. *Int. J. Found. Comput. Sci.*, 22(8):1797–1812, 2011.

[12] Sylvie Davies. A general approach to state complexity of operations: Formalization and limitations. In Mizuho Hoshi and Shinnosuke Seki, editors, *Developments in Language Theory - 22nd International Conference, DLT 2018, Tokyo, Japan, September 10-14, 2018, Proceedings*, volume 11088 of *Lecture Notes in Computer Science*, pages 256–268. Springer, 2018.

[13] Sylvie Davies. *Algebraic Approaches to State Complexity of Regular Operations*. PhD thesis, University of Waterloo, Ontario, Canada, 2019.

[14] Michael Domaratzki. State complexity of proportional removals. *Journal of Automata, Languages and Combinatorics*, 7(4):455–468, 2002.

[15] Michael Domaratzki and Alexander Okhotin. State complexity of power. *Theor. Comput. Sci.*, 410(24-25):2377–2392, 2009.

[16] Zoltán Ésik, Yuan Gao, Guangwu Liu, and Sheng Yu. Estimation of state complexity of combined operations. *Theor. Comput. Sci.*, 410(35):3272–3280, 2009.

[17] Yuan Gao and Lila Kari. State complexity of star and square of union of k regular languages. In Martin Kutrib, Nelma Moreira, and Rogério Reis, editors, *Descriptional Complexity of Formal Systems - 14th International Workshop, DCFS 2012, Braga, Portugal, July 23-25, 2012. Proceedings*, volume 7386 of *Lecture Notes in Computer Science*, pages 155–168. Springer, 2012.

[18] Yuan Gao, Nelma Moreira, Rogério Reis, and Sheng Yu. A survey on operational state complexity. *Journal of Automata, Languages and Combinatorics*, 21(4):251–310, 2017.

[19] Yuan Gao, Kai Salomaa, and Sheng Yu. The state complexity of two combined operations: Star of catenation and star of reversal. *Fundam. Inform.*, 83(1-2):75–89, 2008.

[20] Victor Ginzburg and Mikhail Kapranov. Koszul duality for operads. *Duke Math. J.*, 76(1):203–272, 1994.

[21] Samuele Giraudo. Operads in algebraic combinatorics. *CoRR*, abs/1712.03782, 2017.

[22] Jozef Jirásek and Galina Jirásková. The exact complexity of star-complement-star. In Cezar Câmpeanu, editor, *Implementation and Application of Automata - 23rd International Conference, CIAA 2018, Charlottetown, PE, Canada, July 30 - August 2, 2018, Proceedings*, volume 10977 of *Lecture Notes in Computer Science*, pages 223–235. Springer, 2018.

[23] Jozef Jirásek, Galina Jirásková, and Alexander Szabari. State complexity of concatenation and complementation. *Int. J. Found. Comput. Sci.*, 16(3):511–529, 2005.

[24] Galina Jirásková. State complexity of some operations on binary regular languages. *Theor. Comput. Sci.*, 330(2):287–298, 2005.

[25] Galina Jirásková. Concatenation of regular languages and descriptional complexity. *Theory Comput. Syst.*, 49(2):306–318, 2011.

[26] Galina Jirásková and Alexander Okhotin. State complexity of cyclic shift. *ITA*, 42(2):335–360, 2008.

[27] Galina Jirásková and Alexander Okhotin. On the state complexity of star of union and star of intersection. *Fundam. Inform.*, 109(2):161–178, 2011.

[28] Galina Jirásková, Matúš Palmovský, and Juraj Šebej. Kleene closure on regular and prefix-free languages. In Markus Holzer and Martin Kutrib, editors, *Implementation and Application of Automata - 19th International Conference, CIAA 2014, Giessen, Germany, July 30 - August 2, 2014. Proceedings*, volume 8587 of *Lecture Notes in Computer Science*, pages 226–237. Springer, 2014.

[29] Galina Jirásková and Jeffrey O. Shallit. The state complexity of star-complement-star. In Hsu-Chun Yen and Oscar H. Ibarra, editors, *Developments in Language Theory - 16th International Conference, DLT 2012, Taipei, Taiwan, August 14-17, 2012. Proceedings*, volume 7410 of *Lecture Notes in Computer Science*, pages 380–391. Springer, 2012.

[30] Bryan Krawetz, John Lawrence, and Jeffrey Shallit. State complexity and the monoid of transformations of a finite set. *Int. J. Found. Comput. Sci.*, 16(3):547–563, 2005.

[31] Guangwu Liu, Carlos Martín-Vide, Arto Salomaa, and Sheng Yu. State complexity of basic language operations combined with reversal. *Inf. Comput.*, 206(9-10):1178–1186, 2008.

[32] Jean-Louis Loday and Bruno Vallette. *Algebraic Operads*. Springer-Verlag Berlin Heidelberg, 2012.

[33] A. N. Maslov. Estimates of the number of states of finite automata. *Soviet Math. Dokl.*, 11:1373–1375, 1970.

[34] J.P. May. *The Geometry of Iterated Loop Spaces*. Springer-Verlag Berlin Heidelberg, 1972.

[35] B. Ravikumar. Some applications of a technique of Sakoda and Sipser. *SIGACT News*, 21(4):73–77, 1990.

[36] Jacques Sakarovitch. *Elements of Automata Theory*. Cambridge University Press, 2009.

[37] William J. Sakoda and Michael Sipser. Nondeterminism and the size of two way finite automata. In Richard J. Lipton, Walter A. Burkhard, Walter J. Savitch, Emily P. Friedman, and Alfred V. Aho, editors, *Proceedings of the 10th Annual ACM Symposium on Theory of Computing, May 1-3, 1978, San Diego, California, USA*, pages 275–286. ACM, 1978.

[38] Arto Salomaa, Derick Wood, and Sheng Yu. On the state complexity of reversals of regular languages. *Theor. Comput. Sci.*, 320(2-3):315–329, 2004.

[39] Qiqi Yan. Lower bounds for complementation of omega-automata via the full automata technique. *Log. Methods Comput. Sci.*, 4(1), 2008.

[40] Sheng Yu, Qingyu Zhuang, and Kai Salomaa. The state complexities of some basic operations on regular languages. *Theor. Comput. Sci.*, 125(2):315–328, 1994.