COMENIUS UNIVERSITY IN BRATISLAVA
FACULTY OF MATHEMATICS, PHYSICS AND INFORMATICS

FINITE AUTOMATA AND OPERATIONAL COMPLEXITY

DISSERTATION THESIS

2020 Mgr. Ivana Krajňáková

COMENIUS UNIVERSITY IN BRATISLAVA
FACULTY OF MATHEMATICS, PHYSICS AND INFORMATICS

FINITE AUTOMATA AND OPERATIONAL COMPLEXITY

DISSERTATION THESIS

Study programme:	Applied mathematics
Field of study:	9.1.9 Applied mathematics
Supervising institution:	Mathematical Institute, Slovak Academy of Sciences
Supervisor:	RNDr. Galina Jirásková, CSc.

Bratislava, 2020

Mgr. Ivana Krajňáková

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

KONEČNÉ AUTOMATY A OPERAČNÁ ZLOŽITOSŤ

DIZERTAČNÁ PRÁCA

Študijný program:	Aplikovaná matematika
Študijný odbor:	9.1.9 Aplikovaná matematika
Školiace pracovisko:	Matematický ústav Slovenskej akadémie vied
Školiteľ:	RNDr. Galina Jirásková, CSc.

Bratislava, 2020 Mgr. Ivana Krajňáková

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

THESIS ASSIGNMENT

Name and Surname: Mgr. Ivana Krajňáková
Study programme: Applied Mathematics (Single degree study, Ph.D. III. deg., full time form)
Field of Study: Mathematics
Type of Thesis: Dissertation thesis
Language of Thesis: English
Secondary language: Slovak

Title: Finite automata and operational complexity

Annotation: We study the state complexity of the square operation on languages represented by deterministic finite automata with more than one final states. As a result, we get the exact complexity of this operation on Boolean and alternating finite automata. Using a known result concerning the relation between the size of a Boolean automaton for a given language and a deterministic automaton for its reversal, we get the complexity of complementation, difference, symmetric difference, concatenation, star, reversal, and left and right quotients on Boolean and alternating finite automata. We also obtain the complexity of these operations assuming that the operands are given by nondeterministic automata while the result is required to be represented by a deterministic finite automaton.

Aim: 1) Provide the state-of-the-art concerning the complexity of operations on languages represented by different models of finite automata

2) Study in detail the state complexity of the square operation on languages represented by deterministic finite automata with more than one final states and use these results to get the tightness of known upper bounds on the complexity of the square operation on Boolean and alternating finite automata

3) Find the complexity of all basic regular operations on languages represented by Boolean and alternating finite automata.

4) Examine the NFA-to-DFA trade-offs for basic regular operations; here we assume that the operands of an operation are represented by nondeterministic finite automaton while the result of the operation is represented by a deterministic automaton.

5) To get all lower bounds, describe witness languages over an optimal, or at least over as small as possible, input alphabet.

Keywords: regular languages, regular operation, finite automata, state complexity

Tutor: RNDr. Galina Jirásková, CSc.
Department: FMFI.KAMŠ - Department of Applied Mathematics and Statistics
Head of department: prof. RNDr. Marek Fila, DrSc.

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

Assigned: 10.08.2016

Approved: 10.08.2016 prof. RNDr. Anatolij Dvurečenskij, DrSc.
Guarantor of Study Programme

.....
Student

.....
Tutor

69050300

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Mgr. Ivana Krajňáková
Študijný program: aplikovaná matematika (Jednoodborové štúdium, doktorandské III. st., denná forma)
Študijný odbor: matematika
Typ záverečnej práce: dizertačná
Jazyk záverečnej práce: anglický
Sekundárny jazyk: slovenský

Názov: Finite automata and operational complexity
Konečné automaty a operačná zložitosť

Anotácia: Študujeme stavovú zložitosť operácie štvorec na jazykoch reprezentovaných deterministickými konečnými automatmi s viacerými koncovými stavmi. Ako dôsledok dostávame presnú zložitost tejto operácie na Boleovských a alternujúcich konečných automatoch. Využitím známeho výsledku popisujúceho vzťah medzi veľkosťou Booleovského automatu pre daný jazyk a deterministického automatu pre jeho zrkadlový obraz, dostávame zložitosť operácií doplnok, rozdiel, symetrický rozdiel, zreťazenie, uzáver, zrkadlový obraz, lavý a pravý kvocient na Booleovských a alternujúcich automatoch. Pre tieto operácie zistíme aj ich zložitosť za predpokladu že operandy sú reprezentované nedeterministickými automatmi a výsledok deterministickým automatom.

Cieľ: 1) Zhrnúť súčasný stav problematiky v oblasti zložitosti operácií na jazykoch reprezentovaných rôznymi modelmi konečných automatov.

2) Detailne preskúmať zložitosť operácie štvorec na jazykoch reprezentovaných deterministickými automatmi s viacerými koncovými stavmi a tieto výsledky využiť na získanie presnej zložitosti tejto operácie na Booleovských a alternujúcich automatoch.

3) Zistiť zložitosť všetkých základných regulárnych operácií na Booleovských a alternujúcich automatoch.

4) Študovať zložitosť regulárnych operácií za predpokladu ze operandy sú dané nedeterministickým automatom, avšak požadujeme aby výsledok bol popísaný deterministickým automatom.

5)
Pri všetkých dolných odhadoch sa snažiť o použitie
optimálnej, alebo aspoň čo najmenšej, vstupnej abecedy.

Kľúčové slová: regulárne jazyky, regulárne operácie, konečné automaty, stavová zložitosť

Školiteľ: RNDr. Galina Jirásková, CSc.

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Katedra: FMFI.KAMŠ - Katedra aplikovanej matematiky a štatistiky
Vedúci katedry: prof. RNDr. Marek Fila, DrSc.

Dátum zadania: 10.08.2016

Dátum schválenia: 10.08.2016 prof. RNDr. Anatolij Dvurečenskij, DrSc.
garant študijného programu

.....
študent

.....
školiteľ

Declaration

I hereby attest that I have written this dissertation thesis without any prohibited assistance of third parties. I have acknowledged and cited all the sources which have been used in the thesis.

Bratislava, 2020 .....

Mgr. Ivana Krajňáková

Acknowledgments

I would like to thank my tutor Galina Jirásková for all the time and knowledge she gave me. I want to thank my friends and my colleagues from Mathematical Institute of Slovak Academy of Sciences for valuable advice and supervision. And I thank all the people that are dear to me for their support.

Abstract

We determine how the state complexity of the square operation depends on the number of final states in a deterministic finite automata for a given language. We use this result to describe a binary witness for the square operation on alternating finite automaton that meets the known upper bound $2^n + n + 1$. We also show that our witnesses can be used to prove the tightness of the upper bound $2^m + n + 1$ for concatenation on alternating automata which provides an alternative solution of an open problem stated by Fellah, Jürgensen, Yu [1990, Constructions for alternating finite automata, Internat. J. Comput. Math. 35, 117–132].

Then we consider the state complexity of some other regular operations on languages represented by Boolean and alternating finite automata. We get the tight upper bounds for complementation (n in both models), union, intersection, and difference ($m + n$ and $m + n + 1$), symmetric difference ($m + n$ in both models), star and reversal (2^n in both models), right quotient (2^m and $2^m + 1$), and left quotient (m and $m + 1$). To describe witness languages, we use a binary alphabet for star, reversal, and quotients, and the unary alphabet otherwise. We also show that the binary alphabet is always optimal in the sense that the corresponding upper bounds cannot be met in the unary case.

Finally, we consider the complexity of regular operations assuming that the inputs of an operation are represented by nondeterministic finite automata while its result has to be given as a deterministic finite automaton. We obtain the tight upper bounds for star (2^n), complementation (2^n), reversal (2^n), left quotient (2^m), right quotient (2^m), union (2^{m+n}), symmetric difference (2^{m+n}), intersection ($2^{m+n} - 2^m - 2^n + 2$), difference ($2^{m+n} - 2^n + 1$), and concatenation ($\frac{3}{4}2^{m+n}$). To prove tightness, we use a ternary alphabet for binary Boolean operations and concatenation, and we get an asymptotically tight upper bound 2^{m+n} for these operations in the binary case. Our witnesses for the remaining operations are binary, and we show that the binary alphabet is always optimal.

Keywords: regular languages, deterministic and nondeterministic finite automata, descriptional complexity, regular operations, square, Boolean and alternating finite automata

Abstrakt

Ukážeme ako závisí stavová zložitosť operácie štvorec od počtu koncových stavov v deterministickom konečnom automate pre daný jazyk. Tento výsledok použijeme na popísanie binárneho jazyka ťažkého pre operáciu štvorec na alternujúcich automatoch, ktorý dosahuje známy horný odhad 2^n+n+1. Tiež ukážeme, že naše dosvedčujúce jazyky sa dajú použiť v dôkaze tesnosti horného odhadu 2^m+n+1 pre zreťazenie na alternujúcich automatoch, čo je vlastne alternatívnym riešením otvoreného problému, ktorý formulovali Fellah, Jürgensen, Yu [1990, Constructions for alternating finite automata, Internat. J. Comput. Math. 35, 117–132].

Potom uvažujeme stavovú zložitosť niektorých ďalších regulárnych operácií na jazykoch reprezentovaných Booleovskými a alternujúcimi konečnými automatmi. Dostávame tesné horné odhady pre doplnok (n pre oba modely), zjednotenie, prienik, a rozdiel ($m+n$ a $m+n+1$), symetrický rozdiel ($m+n$ pre oba modely), star a zrkadlový obraz (2^n pre oba modely), pravý kvocient (2^m a 2^m+1) a ľavý kvocient (m a $m+1$). Na popísanie dosvedčujúcich jazykov pre star, zrkadlový obraz, ľavý a pravý kvocient používame binárnu abecedu, o ktorej ukážeme, že je optimálna, keďže horné odhady sú nedosiahnuteľné v unárnom prípade. Pre ostatné operácie použijeme unárnu abecedu na dosvedčujúce jazyky.

Nakoniec sa zaoberáme zložitosťou regulárnych operácií za predpokladu, že vstupné jazyky su dané ako nedeterministické konečné automaty, kým výsledok má byť popísaný ako deterministický konečný automat. Dostávame tak tesné horné odhady pre operácie star (2^n), doplnok (2^n), zreťazenie (2^n), zrkadlový obraz (2^n), ľavý kvocient (2^m), pravý kvocient (2^m), zjednotenie (2^{m+n}), symetrický rozdiel (2^{m+n}), prienik ($2^{m+n}-2^m-2^n+2$), rozdiel ($2^{m+n}-2^n+1$), a zreťazenie ($\frac{3}{4}2^{m+n}$). Pre binárne Booleovské operácie a zreťazenie sme použili ternárnu abecedu pre dosvedčujúce jazyky. Na binárnej abecede však v tomto prípade dostávame asymptoticky tesný horný odhad 2^{m+n}. Všetky ostatné dosvedčujúce jazyky pre zvyšné operácie sú binárne a v týchto prípadoch je aj binárna abeceda optimálna.

Kľúčové slová: regulárne jazyky, deterministické a nedeterministické konečné automaty, popisná zložitosť, regulárne operácie, štvorec, Booleovské a alternujúce automaty

Contents

Introduction

Finite automata represent a simple computational model. Nevertheless, some questions concerning finite automata remain open. For example, it is not known how many states are sufficient and necessary in the worst case for a two-way deterministic finite automaton to simulate a given two-way nondeterministic finite automaton. This problem is interesting per se, and it is also important due to its connection with a well-known open problem whether or not DLOGSPACE equals NLOGSPACE [2].

In the recent years, finite automata, regular languages and regular operations have been intensively investigated from the descriptional complexity point of view. Descriptional complexity measures the cost of description of languages or language operations by using various formal systems such as deterministic, nondeterministic or Boolean automata, regular expressions or grammars. For example, the state complexity of a regular language is the smallest number of states in any deterministic finite automaton (DFA) recognizing this language. By the state complexity of a regular operation we understand the function that assigns the number of states that are sufficient and necessary in the worst case for a DFA recognizing the resulting language to the number of states in DFAs for inputs.

The upper bounds on the complexity of several regular operations can be obtained using constructions described by Rabin and Scott [46]. They also presented the subset construction that provides an upper bound 2^n for determinization of an n-state deterministic finite automaton. This upper bound was shown to be tight by describing binary witness languages in [37, 39, 42, 53]. Binary witnesses for union, concatenation and star were given by Maslov [38], while the unary case was considered by Chrobak [9].

The paper by Yu, Zhuang and Salomaa [55] accelerated thorough study of descriptional complexity of language operations. Basic regular operations on unary languages were examined by Pighizzini and Shalit [45] and those on finite languages by Câmpeanu et al. [5]. Some less common operation were considered in the literature, like shuffle [6], proportional removals [11], square [47], cyclic shift [27], power [12].

Holzer and Kutrib [14] introduced and investigated nondeterministic state complexity to measure the cost of description when using nondeterministic finite automata (with a unique initial state). They determined the nondeterministic state complexity of several regular operations. Their results for reversal and complementation were improved in [24]; here the fooling-set lower bound method described by Birget [3, Lemma 1] was used to get tight upper bounds and describe binary witnesses.

In a Boolean finite automaton, the result of the transition function is a Boolean function with the states as variables. This is a generalization of nondeterminism since the set of states can be viewed as a disjunction of these states. If a Boolean automaton starts in a single state, that is, in a Boolean function equal to a projection, it is called an alternating finite automaton [8, 13, 25, 54]. It is known that every Boolean automaton with n states can be simulated by a deterministic automaton with 2^{2^n} states [4], and by a nondeterministic automaton with $2^n + 1$ states [25]. Fellah et al. [13] described alternating automata for complementation, union, intersection, star and concatenation on languages represented by alternating automata. This provided upper bounds on the complexity of these operations. Those for union, intersection and concatenation were shown to be tight in [17, 25], where also operational complexity for Boolean automata was considered.

The study of the complexity of combined operations began with the paper by Salomaa et al. [49]. The upper bound for a combined operation is given by the composition of complexities of involved operations and is tight in some of the cases, e. g. the case of star of intersection [28]. However, usually the resulting complexity is much smaller [10, 36].

The state set of a self-verifying finite automaton consists of three disjoint set of accepting, rejecting and neutral states, and it is required that every input word has at lest one accepting or rejecting computation, but not both. Assent and Seibert [1] obtained an upper bound $O(2^n/\sqrt{n})$ for the conversion to a deterministic automaton. This upper bound was later improved to a function $g(n) \approx 3^{\frac{n}{3}}$ [29] by using the known result by Moon and Moser [41] on the maximal number of maximal cliques in graphs. The paper [29] also presented a binary witness meeting the improved upper bound.

Jirásek et al. [21] described so called sv-fooling set bound method providing a lower bound on the number of states in a self-verifying automaton. They used this method to get the precise self-verifying state complexity of all basic regular operations. They described witnesses for reversal and Boolean operations over a binary alphabet. For star, concatenation, and quotients they used a growing alphabet of an exponential size, however they also proved that the self-verifying state complexity of these operations is almost the same in the case of a fixed alphabet.

A nondeterministic finite automaton is called unambiguous if it has at most one accepting computation on every input word. Ambiguity in finite automata was first studied by Schmidt in his PhD thesis [50] where a lower bound method based on the rank of certain matrices was described. It is known that the tight upper bound for the conversion to a DFA is 2^n, and those from an NFA is $2^n - 1$ with binary witnesses in both cases [34, 35].

The lower bound method from [50] was elaborated by Jirásek et al. [23, 22]. They showed that a lower bound on the number of states in an unambiguous automaton equivalent to an NFA is given by the rank of the matrix whose rows are indexed by its reachable sets of and columns by its co-reachable sets, and its entry is 0 if the corresponding sets are disjoint and it is 1 otherwise. Using this lower bound method, the authors obtained the precise unambiguous state complexity of intersection, star, concatenation, and quotients. They also decreased the trivial upper bound 2^n for complementation to $O(2^{0.79n})$. Recently, Raskin [48] presented the lower bound $n^{\log\log\log n}$ for this operation on unambiguous automata. This shows that the unambiguous state complexity of complementation is not polynomial, as it was thought before. The large gap between lower and upper bound on the complexity for complementation on unambiguous automata remains.

We continue the research on operational complexity in this thesis. Motivated by an open problem from [13] concerning the tightness of an upper bound $2^m + n + 1$ for concatenation, we first study the square operation. Since a language is recognized by an n-state AFA if and only if its reversal is recognized by a 2^n-state DFA with half of its states final, and, moreover, reversal and square commute, to get a language that is hard for square on AFAs it is enough to take the reversal of a language that is hard for square on DFAs with half of their states final. Therefore, we first study in detail the square operation on DFAs that have more than one final state. We prove that the upper bound $n2^n - k2^{n-1}$ from [55] is tight already in the binary case if the number k of final states is at most $n - 2$. The case of just one non-final state is different; here we get the results depending on the finality of the initial state. The witnesses, as well as the upper bounds for one non-final state case have been found as a result of brute-force search using a simple application that we programmed for this purpose.

We next obtain the precise complexity of Boolean operations, complementation, star, reversal, and quotients on both Boolean and alternating finite automata. To get upper bounds we use the above mentioned relation between the size of a Boolean or alternating automaton for a language and the size of a DFA for its reversal. To get lower bounds we either use known results on the state complexity of the corresponding operation on languages represented by DFA with half of their states final, or we describe such DFAs

that are hard for the considered operation on DFAs. Then we take the reversal of these languages, and, using the fact that reversal commutes with any of considered operations, we show that they meet the corresponding upper bound for Boolean or alternating automata. All our witness languages are defined over a binary or unary alphabet. We also show that whenever we use a binary alphabet, it is always optimal in the sense the complexity of the corresponding operation is smaller in the unary case.

Finally, we study operational complexity under the assumption that the input languages a given as NFAs while the resulting language has to be represented as a DFA, that is, we investigate so called NFA-to-DFA trade-off for regular operations. Here our motivation comes from two streams of research. The first one concerns the complexity of combined operations that do not contain complementation. In such a case we can perform all involved operations on NFAs and use the subset construction as the last step. It follows that the NFA-to-DFA trade-off for the outermost operation provides an upper bound on the complexity of a given combined operation. Our second motivation comes from the operational complexity on self-verifying and unambiguous finite automata that are just special cases of NFAs. Since every DFA is an unambiguous automaton and it can be viewed as a self-verifying automaton (with the empty set of neutral states), the NFA-to-DFA trade-off for a regular operation provides an upper bound on its self-verifying or unambiguous state complexity. It turns out that these upper bounds are almost tight in some cases, so sometimes we cannot do anything better by using the special properties of self-verifying or unambiguous automata. We provide the precise NFA-to-DFA trade-off for complementation, union, intersection, difference, symmetric difference, concatenation, star, reversal, and left and right quotients. We use a ternary alphabet to describe the witnesses for concatenation and binary Boolean operations, and we get an asymptotically tight upper bounds for these operations in the binary case. Our witnesses for complementation, star, reversal and quotients are described over a binary alphabet which is always optimal. For star, we are able to get the precise complexity also in the unary case, while for the other operations on unary languages, we get asymptotically tight upper bounds.

The thesis is organized as follows. In Chapters 1 and 2 we provide an introduction to basic notions and notations that we use. The main result starts with Chapter 3, where we examine the state complexity of the square operation on languages accepted by DFAs with k final states, BFAs and AFAs as well. In Chapter 4 we continue the research on AFA and BFA operational state complexity. Chapter 5 provides tight upper bounds on the NFA-to-DFA trade-off for all basic regular operations. In Chapter 6 we summarize our results and give open problems we clashed into.

Chapter 1

Preliminaries

In this section we give basic notions and preliminary results. Details and all unexplained notions can be found here [15, 51, 54]. Let Σ be a finite non-empty *alphabet* of symbols. Then Σ^* denotes the set of all words over Σ including the empty word ε. A *language* over an alphabet Σ is any subset of Σ^*. Fo a finite set S the symbol $|S|$ denotes the size of S and 2^S denotes the power set of S. Let K and L be languages over Σ. Then we can consider several regular operations:

- *complement* of L is $L^c = \Sigma^* \setminus L$,
- *intersection* $K \cap L = \{w \in \Sigma^* \mid w \in K \text{ and } w \in L\}$,
- *union* $K \cup L = \{w \in \Sigma^* \mid w \in K \text{ or } w \in L\}$,
- *difference* $K \setminus L = \{w \in \Sigma^* \mid w \in K \text{ and } w \notin L\}$,
- *symmetric difference* $K \oplus L = \{w \in \Sigma^* \mid (w \in K \text{ and } w \notin L) \text{ or } (w \notin K \text{ and } w \in L)\}$,
- *concatenation* of K and L is $KL = \{uv \mid u \in K \text{ and } v \in L\}$,
- *star* $L^* = \bigcup_{i \geq 0} L^i$ where $L^0 = \{\varepsilon\}$ and $L^{i+1} = L^i L$,
- *reversal* $L^R = \{w^R \mid w \in L\}$, where w^R denotes the mirror image of w,
- *shuffle* of K and L is $K \sqcup\!\sqcup L = \{u_1 v_1 u_2 v_2 \cdots u_k v_k \mid u_i, v_i \in \Sigma^*, u_1 u_2 \cdots u_k \in K, v_1 v_2 \cdots v_k \in L\}$,
- *right quotient* of K by L is the language $KL^{-1} = \{x \in \Sigma^* \mid xy \in K \text{ for some } y \in L\}$,
- *left quotient* of K by L is the language $L^{-1}K = \{x \in \Sigma^* \mid yx \in K \text{ for some } y \in L\}$.

A *nondeterministic finite automaton* (NFA) is a quintuple $A = (Q, \Sigma, \cdot, s, F)$ where Q is a finite non-empty set of states, Σ is a finite non-empty input alphabet, $\cdot\colon Q \times \Sigma \to 2^Q$ is the transition function, $s \in Q$ is the initial state, $F \subseteq Q$ is the set of final states. The transition function can be extended to the domain $2^Q \times \Sigma^*$ in the natural way. For states p, q and a symbol a we sometimes write (p, a, q) whenever $q \in p \cdot a$. The language accepted by A is the set of words $L(A) = \{w \in \Sigma^* \mid s \cdot w \cap F \neq \emptyset\}$.

Sometimes nondeterministic automata are allowed to have a set of initial states, then we speak about NFAs with multiple initial states (MNFAs). The *reverse* of an MNFA $A = (Q, \Sigma, \cdot, I, F)$ is the MNFA $A^R = (Q, \Sigma, \cdot^R, F, I)$ where $q \cdot^R a = \{p \mid q \in p \cdot a\}$, that is, A^R is obtained from A by reversing all its transitions and swapping the roles of the initial and final states.

We say that a subset S of Q is *reachable* in MNFA A if there exists a word w such that $S = I \cdot w$. A subset S is *co-reachable* in A if it is reachable in the reversed automaton A^R.

A nondeterministic automaton is a *deterministic finite automaton* (DFA) if $|q \cdot a| = 1$ for each state q and each input symbol a. We usually write $p \cdot a = q$ instead of $p \cdot a = \{q\}$, we use $p \xrightarrow{a} q$ to denote that $p \cdot a = q$. A state of a DFA is called *dead* if no word is accepted from it.

The *state complexity of a regular language* L, $\mathrm{sc}(L)$, is the smallest number of states in any DFA recognizing L. The *state complexity of a k-ary regular operation* $\circ$ is the function from $\mathbb{N}^k$ to $\mathbb{N}$ defined as follows

$$(n_1, n_2, \ldots, n_k) \mapsto \max\{\mathrm{sc}(\circ(L_1, L_2, \ldots, L_k)) \mid \mathrm{sc}(L_1) \leq n_1, \mathrm{sc}(L_2) \leq n_2, \ldots, \mathrm{sc}(L_k) \leq n_k\}.$$

Let $\circ \in \{\cap, \cup, \oplus, \setminus\}$ and languages K and L are recognized by DFAs $A = (Q_A, \Sigma, \cdot_A, s_A, F_A)$ and $B = (Q_B, \Sigma, \cdot_B, s_B, F_B)$. Then the language $K \circ L$ is recognized by the *product automaton*

$$M_\circ = (Q_A \times Q_B, \Sigma, \cdot, (s_A, s_B), F_\circ)$$

where $(p, q) \cdot a = (p \cdot_A a, q \cdot_B a)$ for all $p \in Q_A$, $q \in Q_B$, and $a \in \Sigma$, and

$$F_\circ = \begin{cases} F_A \times F_B, & \text{if } \circ = \cap; \\ (F_A \times Q_B) \cup (Q_A \times F_B), & \text{if } \circ = \cup; \\ F_A \times (Q_B \setminus F_B), & \text{if } \circ = \setminus; \\ (F_A \times (Q_B \setminus F_B)) \cup ((Q_A \setminus F_A) \times F_B), & \text{if } \circ = \oplus. \end{cases}$$

Every MNFA $A = (Q, \Sigma, \cdot, I, F)$ can be converted to an equivalent DFA

$$\mathcal{D}(A) = (2^Q, \Sigma, \cdot, I, \{S \in 2^Q \mid S \cap F \neq \emptyset\}).$$

The DFA $\mathcal{D}(A)$ is called the *subset automaton* of A and it may not be minimal, since it can contain unreachable or equivalent states.

A *Boolean finite automaton* (BFA) is a quintuple $A = (Q, \Sigma, \cdot, g_s, F)$, where Q is a finite non-empty set of states, $Q = \{q_1, \ldots, q_n\}$, Σ is an input alphabet, $\cdot$ is the transition function that maps $Q \times \Sigma$ into the set $\mathcal{B}_n$ of Boolean functions with variables $\{q_1, \ldots, q_n\}$, $g_s \in \mathcal{B}_n$ is the initial Boolean function, and $F \subseteq Q$ is the set of final states. The transition function $\cdot$ can be extended to the domain $\mathcal{B}_n \times \Sigma^*$ as follows: For all g in $\mathcal{B}_n$, a in Σ, and w in Σ^*, we have

$$g \cdot w = \begin{cases} g, & \text{if } w = \varepsilon; \\ g(q_1 \cdot a, \ldots, q_n \cdot a), & \text{if } g = g(q_1, \ldots, q_n) \text{ and } w = a; \\ (g \cdot v) \cdot a, & \text{if } w = va. \end{cases}$$

Let $f = (f_1, \ldots, f_n)$ be the Boolean vector with $f_i = 1$ iff $q_i \in F$. The language accepted by the BFA A is the set $L(A) = \{w \in \Sigma^* \mid (g_s \cdot w)(f) = 1\}$. A Boolean finite automaton is called *alternating* (AFA) if the initial function is a projection $g_s(q_1, \ldots, q_n) = q_i$.

The *Boolean (alternating) state complexity* of L, $\mathrm{bsc}(L)(\mathrm{asc}(L))$, is the smallest number of states in any BFA (AFA) for L. The reader may refer to [4, 13, 25, 33, 51] for details. We illustrate these notions in the following example.

Example 1.1. *Let A be binary alternating automaton with two states defined as $A = (\{q_1, q_2\}, \{a, b\}, \cdot, q_1, \{q_2\})$ with the transition function defined in the following table:*

$\cdot$	a	b
q_1	$q_1 \vee q_2$	q_1
q_2	q_2	$q_1 \wedge \neg q_2$

Let us compute whether the word $w = abb$ is accepted. We start the computation from the initial state $s = q_1$, $s \cdot w = q_1 \cdot abb$. According to the table $q_1 \cdot a$ is $q_1 \vee q_2$, so we get $q_1 \cdot abb = (q_1 \vee q_2) \cdot bb$. And we continue as follows:

$$(q_1 \vee q_2) \cdot bb = (q_1 \vee (q_1 \wedge \neg q_2)) \cdot b = q_1 \vee (q_1 \wedge \neg(q_1 \wedge q_2)).$$

Now we evaluate the resulting expression in the finality vector $f = (0, 1)$ replacing the final states with 1s and non-final states with 0s: $q_1 \vee (q_1 \wedge \neg(q_1 \wedge q_2)) = 0 \vee (0 \wedge \neg(0 \wedge 1)) = 0$. The result is 0 so the word $w = abb$ is not accepted by A. □

A language, or a finite automaton, defined over an alphabet containing exactly one (two, three, respectively) symbols is called *unary (binary, ternary).*

For unary DFAs we use the Nicaud's notation [43]. For two integers ℓ and n such that $0 \leq \ell \leq n-1$ and a subset F of $\{0,1,\ldots,n-1\}$, $A = (n,\ell,F)$ is the unary automaton whose set of states is $Q = \{0,1,\ldots,n-1\}$ and the transition function is given by $q \cdot a = q+1$ if $0 \leq q \leq n-2$ and $(n-1)\cdot a = \ell$. The initial state of this automaton is 0 and its set of final states is F (cf. Appendix [B, Preliminaries]).

For states p, q and a symbol a, we say that (p,a,q) is a *transition* in MNFA $N = (Q,\Sigma,\cdot,I,F)$ if $q \in p \cdot a$. A sequence of transitions $(q_0,a_1,q_1)(q_1,a_2,q_2)\cdots(q_{n-1},a_n,q_n)$ on an input word $a_1\cdots a_n$ is called *computation* of N on $a_1\cdots a_n$. The computation is *accepting* if $q_0 \in I$ and $q_n \in F$. An MNFA $N = (Q,\Sigma,\cdot,I,F)$ is *unambiguous* (UFA) if it has at most one accepting computation on every input word,

Finally, we define the model of self-verifying finite automata in the following definition taken from [29].

Definition 1.2 ([29, Definition 1])**.** *A self-verifying finite automaton (SVFA) is a 6-tuple $A = (Q,\Sigma,\cdot,q_0,F^a,F^r)$, where $Q,\Sigma,\cdot,q_0$ are defined as for standard nondeterministic automata, and $F^a, F^r \subseteq Q$ are the sets of accepting and rejecting states, respectively. The remaining states, namely the states belonging to $Q\setminus(F^a \cup F^r)$, are called neutral states. It is required that for each input word w in Σ^*, there exists at least one computation ending in an accepting or in a rejecting state, that is, $(q_0\cdot w)\cap(F^a\cup F^r) \neq \emptyset$, and there are no words w such that both $q_0\cdot w \cap F^a$ and $q_0\cdot w \cap F^r$ are nonempty. The language accepted by A, denoted as $L^a(A)$, is the set of all input words having a computation ending in an accepting state, while the language rejected by A, denoted as $L^r(A)$, is the set of all input words having a computation ending in a rejecting state.*

Chapter 2

Known Results

The upper bounds on the state complexity of basic regular operations can be derived from the automata constructions given by Rabin and Scott [46] already in 1959. The product automaton for intersection, described in [46, Theorem 6], provides an upper bound mn for this operation. For union, difference, and symmetric difference, it is enough to change the set of final states in this product automaton. This gives the same upper bound for these three Boolean operations. In [46, Definition 11 and Theorem 11] the construction of the subset automaton $\mathcal{D}(N)$ corresponding to a given NFA N was described. This construction resulted in the upper bound 2^n for NFA-to-DFA conversion since the states of $\mathcal{D}(N)$ are subsets of the state set of N. Following [46, Definition 12 and Theorem 12] described the reverse of an NFA and provided an upper bound 2^n for the reversal operation as a result. Finally, [46, Theorem 13] provided the construction of NFAs for concatenation and star. To get an upper bound for the concatenation, we need to slightly modify Rabin and Scott's construction: instead of adding the transitions from final states of A to the corresponding states of B, we add the transitions from states with out transition to a final state of A to the initial state of B. This provides an upper bound $m2^n - 2^{n-1}$ for concatenation. A similar modification of an NFA for star gives an upper bound $\frac{3}{4}2^n$.

The tightness of the upper bound 2^n for NFA-to-DFA conversion was proved almost immediately in 1962 by Yershov [53] who described a binary witness NFA, shown in Fig. 2.1. Its equivalent DFA requires at least 2^n states. Similar binary NFAs were later described by Lupanov [37], Moore [42], and Meyer and Fisher [39]; see Fig.2.2.

In 1966 Mirkin [40] pointed out that the Lupanov's ternary NFA-to-DFA witness, see Fig. 2.3, is in fact the reverse of a DFA. This proves the tightness of the upper bound 2^n for reversal. Binary witnesses for this operation were given by Leiss [33] and Šebej [52].

Binary witness languages for union, concatenation and star were described by Maslov

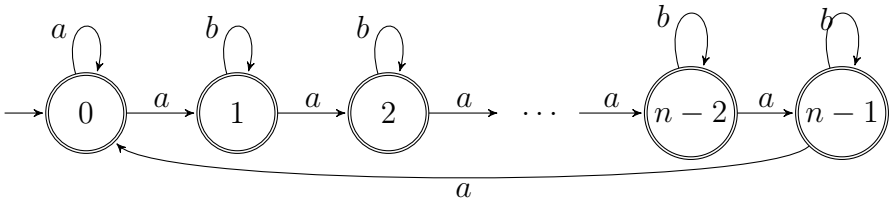

Figure 2.1: Yershov 1962.

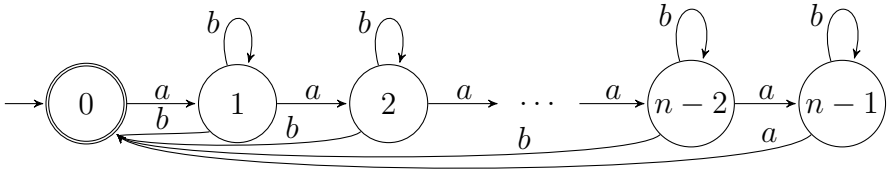

Figure 2.2: Meyer, Fisher 1971.

in 1970 [38]. The unary case was considered in 1986 by Chrobak [9] who showed that the determinization of a unary NFA is in $\Theta(F(n))$ where $F(n)$ is Landau function defined by $F(n) = \max\{\mathrm{lcm}(x_1, \ldots, x_k) \mid x_1 + \cdots + x_k \leq n\}$ and we have $F(n) \approx 2^{\sqrt{n \ln n}}$. In 1992 Birget [3] obtained tight upper bounds on the state complexity of the intersection of k regular languages.

The systematic study of the state complexity of regular languages and regular operations began with the 1994 paper by Yu, Zhuang and Salomaa [55]. They obtained upper bounds $m2^n - k2^{n-1}$ and $2^{n-1} + 2^{n-k-1}$ for concatenation and star of languages recognized by DFAs with k final states, the tightness of which was later shown by Jirásek, Jirásková and Szabari [20] and Palmovský [44]. They also provided the state complexity of left quotient $(2^m - 1)$ and right quotient (m), as well as the state complexity of basic operations on unary languages. Table 2.1 summarizes the results by Maslov and Yu et al.

More precise results in the unary case were obtained by Pighizzini and Shallit [45]. Operations on finite languages were investigated by Câmpeanu [5]. The state complexity

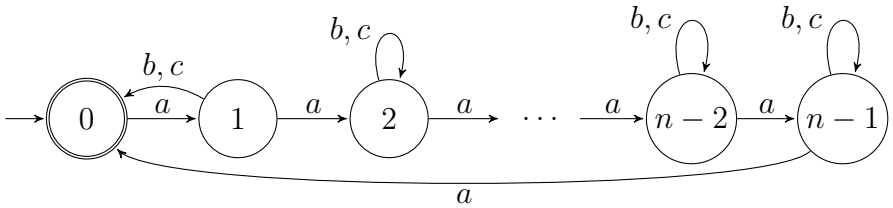

Figure 2.3: Lupanov 1963.

	DFA	$\lvert\Sigma\rvert$	$\lvert\Sigma\rvert = 1$
complementation	n	1	n
union	mn	2	mn if $\gcd(m,n) = 1$
intersection	mn	2	mn if $\gcd(m,n) = 1$
concatenation	$m2^n - 2^{n-1}$	2	mn if $\gcd(m,n) = 1$
reversal	2^n	2	n
star	$\frac{3}{4} \cdot 2^n$	2	$(n-1)^2 + 1$
left quotient	$2^m - 1$	2	m
right quotient	m	1	m

Table 2.1: Operational complexity for deterministic finite automata [38, 55].

of some less common regular operations can be found in the literature: square by Rampersad [47], shuffle by Câmpeanu [6], proportional removals by Domaratzki [11], cyclic shift by Jirasková and Okhotin [27], and power by Domaratzki and Okhotin [12].

In 2003 Holzer and Kutrib introduced and examined the nondeterministic state complexity [14]. Their model of NFAs considered the single initial state while Rabin and Scott [46] allowed more states to be initial. This results in the complexity $m + n + 1$ and $n + 1$ for union and reversal, while in model of Rabin and Scott it would be $m + n$ and n. The results from Holzer and Kutrib and from the completion by Jirásková [24] on operational complexity on NFAs are summarized in Table 2.2.

	NFA	$\lvert\Sigma\rvert$	$\lvert\Sigma\rvert = 1$
complementation	2^n	2	$\Theta(F(n))$
reversal	$n + 1$	2	n
star	$n + 1$	1	$n + 1$
concatenation	$m + n$	2	$m + n - 1 \leq \cdot \leq m + n$
union	$m + n + 1$	2	$m + n + 1$ if $m \neq kn$ and $n \neq km$
intersection	mn	2	mn if $\gcd(m,n) = 1$

Table 2.2: Operational complexity for nondeterministic finite automata [14, 24].

2.1 Combined Operations, Self-verifying and Unambiguous Automata

The investigation of combined operations was started in 2007 by the paper by Salomaa, Salomaa and Yu [49]. An upper bound can be simply obtained by the composition of particular complexities. Such an upper bound is tight in several cases (for example for star of intersection, [28]). However, the resulting complexity is much smaller in the most cases. A number of combined operations was studied in the literature. The reverse of the union, intersection or concatenation of two languages was studied by Liu et al. in 2008 [36]. The state complexity for $(L_1 \cup L_2)^R$ was showed to be $2^{m+n}-2^m-2^n+2$, for $(L_1 \cap L_2)^R$ it is $2^{m+n}-2^m-2^n+2$, and for $(L_1L_2)^R$ it is $3 \cdot 2^{m+n-2}-2^n+1$. The concatenation of one language with the union or intersection of other two was studied by Cui et al. [10] in 2011. So state complexity of $L_1(L_2 \cup L_3)$ was shown to be $(m-1)(2^{n+p}-2^n-2^p+2)+2^{n+p-2}$ and for $L_1(L_2 \cap L_3)$ it is $m2^{np}-2^{np-1}$.

A self-verifying finite automaton (SVFA) has three disjoint groups of states: accepting, rejecting and neutral. Every input word has to have at least one accepting or rejecting computation, but not both. The SVFA-to-DFA conversion was examined in 2011 by Jirásková and Pighizzini [29]. They used known result by Moon and Moser [41] on the maximal number of maximal cliques in graphs to show that every n-state SVFA can be simulated by a DFA that has at most $g(n)$ states where

$$g(n) = \begin{cases} n, & \text{if } 1 \leq n \leq 2; \\ 1+2 \cdot 3^{\frac{n-3}{3}}, & \text{if } n \geq 3 \text{ and } n \bmod 3 = 0; \\ 1+3^{\frac{n-1}{3}}, & \text{if } n \geq 4 \text{ and } n \bmod 3 = 1; \\ 1+4 \cdot 3^{\frac{n-5}{3}}, & \text{if } n \geq 5 \text{ and } n \bmod 3 = 2. \end{cases}$$

The operational complexity on SVFAs was examined by Jirásek, Jirásková and Szabari [21]. Table 2.3 summarizes their results.

A nondeterministic finite automaton is unambiguous (UFA) if it has at most one accepting computation on every input word. A lower bound method for UFAs based on the range of certain matrices has been developed in 1978 by Schmidt [50]. Using this method, Leung [34, 35] showed that a tight upper bound on UFA-to-DFA conversion is 2^n while those for NFA-to-UFA is 2^n-1. The method was further elaborated by Jirásek, Jirásková and Šebej [23], where operational complexity on UFAs was examined. Their results are summarized in Table 2.4; here the lower bound for complementation is from Raskin [48].

	SVFA	$\|\Sigma\|$	small alphabets	
complementation	n	1	-	
intersection	mn	2	-	
union	mn	2	-	
difference	mn	2	-	
symmetric difference	mn	2	-	
reversal	$2n+1$	2	n	if $\|\Sigma\| = 1$
star	$\frac{3}{4} \cdot 2^n$	$\frac{3}{4} \cdot 2^n + 1$	$\leq 2^{n-1}$	if $\|\Sigma\| \geq 4$
left quotient	$2^n - 1$	$2^n + 1$	$\leq 2^{n-1}$	if $\|\Sigma\| \geq 4$
right quotient	$g(n)$	$g(n) + 2$	$\Omega(2^{\frac{n}{3}})$	if $\|\Sigma\| \geq 4$
concatenation	$\Theta(3^{\frac{m}{3}} 2^n)$	$g(m) + 2^n + 4$	$\Omega(2^{\frac{m}{3}} 2^n)$	if $\|\Sigma\| \geq 8$

Table 2.3: Operational complexity for self-verifying automata [21].

	UFA	$\|\Sigma\|$
complementation	$n^{\log\log\log n} \leq \cdot \leq 2^{0.79n+\log n}$	1
reversal	n	1
star	$\frac{3}{4} \cdot 2^n$	3
concatenation	$\frac{3}{4} \cdot 2^{m+n} - 1$	7
union	$mn + m + n \leq \cdot \leq m + n2^{0.79n+\log n}$	4
intersection	mn	2
left quotient	$2^m - 1$	2
right quotient	$2^m - 1$	2

Table 2.4: Operational complexity for unambiguous automata [23, 48].

2.2 Boolean and Alternating Automata

In a Boolean finite automaton (BFA) the transition function maps every pair of a state and a letter into a Boolean function with the states as variables. This approach is a generalization from NFAs where the result of the transition function may be viewed as a disjunction of states. It is known that Boolean automata recognize regular languages [4, 8]. Moreover, the BFA-to-DFA trade-off is 2^{2^n} with a binary witness [4] and the BFA-to-NFA trade-off is $2^n + 1$ also with a binary witness [25]. If the initial function of a Boolean automaton is a projection, that is, if it starts in a single state, it is called

an alternating finite automaton (AFA); cf. [8, 13, 54, 25]. In 1990 Fellah, Jurgensen and Yu [13] described the constructions of alternating finite automata for basic regular operations on languages represented by AFAs. As a result they get upper bounds for complementation, union, intersection, concatenation and star, see Table 2.5. Their upper bounds for union and intersection were shown to be tight in 2012 [25]. For star and concatenation, the lower bound from [25] and the upper bound from [13] differ by one, see Table 2.6. The open problem from [13] concerning the tightness of the upper bound for concatenation was definitely solved in 2018 [17].

	upper bound for AFAs
complementation	$\leq n+1$
union	$\leq m+n+1$
intersection	$\leq m+n+1$
star	$\leq 2^n+1$
concatenation	$\leq 2^m+n+1$

Table 2.5: Upper bounds for AFAs from Fellah, Jürgensen, Yu [13].

	AFA state complexity	BFA state complexity
union	$m+n+1$	$m+n$
intersection	$m+n+1$	$m+n$
concatenation	$2^m+n <= . <= 2^m+n+1$	2^m+n
reversal	$2^n <= . <= 2^n+1$	2^n
star	$2^n <= . <= 2^n+1$	$2^n <= . <= 2^n+1$

Table 2.6: Results from [25].

The following two observations, concerning the relation between the size of a BFA (an AFA) for a language and the size of a DFA for its reversal, play a crucial role in the next two chapters. We use them to get the state complexity of all basic regular operations on languages represented by BFAs and AFAs.

Lemma 2.1 (cf. [13, Theorem 4.1, Corollary 4.2] and [25, Lemma 1])**.** *If a languages L is recognized by an n-state BFA (n-state AFA), then the language L^R is recognized by a DFA with 2^n states (by a DFA with 2^n states of which 2^{n-1} are final, respectively).*

Proof Idea. Let $A = (\{q_1, q_2, \ldots, q_n\}, \Sigma, \cdot, g_s, F)$ be BFA for L. Construct a 2^n-state MNFA $N = (Q, \Sigma, \circ, I, \{f\})$, where

- $Q = \{0,1\}^n$
- $I = \{j \in Q \mid g_s(j) = 1\}$;
- $f = (f_1, f_2, \ldots, f_n) \in Q$ with $f_i = 1$ if $q_i \in F$ and $f_i = 0$ otherwise;
- the transition function $\circ\colon Q \times \Sigma \to 2^Q$ is defined as follows:
  for each $j = (j_1, j_2, \ldots, j_n) \in Q$ and each $a \in \Sigma$,
  $j \circ a = \{\ell \in Q \mid (q_i \cdot a)(\ell) = j_i \text{ for } i = 1, 2, \ldots, n\}$.

Then $L(A) = L(N)$. Moreover, automaton N^R is deterministic, so L^R is recognized by a DFA with 2^n states. If A is an AFA, then the MNFA N has 2^{n-1} initial states, and therefore the language L^R is recognized by a DFA with 2^n states of which 2^{n-1} are final. □

Lemma 2.2 (cf. [25, Lemma 2])**.** *If a language L is recognized by a DFA A with 2^n states (a DFA with 2^n states of which 2^{n-1} are final), then the language L^R is recognized by an n-state BFA (an n-state AFA).*

Proof Idea. Let $A = (Q, \Sigma, \cdot, s, F)$ be a 2^n-state DFA for L. Without loss of generality, let $Q = \{0, 1, \ldots, 2^n - 1\}$. Consider 2^n-state MNFA $A^R = (Q, \Sigma, \cdot^R, F, \{s\})$ for L^R. It has exactly one final state s and the set of initial states F. Moreover, for every $a \in \Sigma$ and for every $i \in Q$, there is exactly one state j such that $i \in j \cdot^R a$ since A is a DFA. For a state $i \in Q$, let $\mathrm{bin}(i) = (i_1, i_2, \ldots, i_n)$ be the binary n-tuple such that $i_1 i_2 \cdots i_n$ is the binary notation of i on n digits with leading zeros if necessary.

Let us define an n-state BFA $B = (Q_B, \Sigma, \circ, g_s, F_B)$, where:

- $Q_B = \{q_1, \ldots, q_n\}$;
- $F_B = \{q_\ell \mid \mathrm{bin}(s)_\ell = 1\}$;
- $g_s(\mathrm{bin}(i)) = 1$ iff $i \in F$;
- $(q_1 \circ a, q_2 \circ a, \ldots, q_n \circ a)(\mathrm{bin}(i)) = \mathrm{bin}(j)$ where $i \in j \cdot^R a$ for each $i \in Q$ and $a \in \Sigma$.

Then $L(B) = L(A^R) = L^R$, so L^R is accepted by an n-state BFA. If $|F| = 2^{n-1}$, we may denote the states of A in such a way that $F = \{2^{n-1}, 2^{n-1} + 1, \ldots, 2^n - 1\}$. Then we get $g_s = q_1$, and therefore B is an n-state AFA for L^R. □

Chapter 3

Square on Deterministic, Alternating, and Boolean Finite Automata

The square operation is one of the basic unary operations on formal languages defined as $L^2 = \{uv \mid u \in L \text{ and } v \in L\}$. The upper bound on its state complexity has been long known, since the square is only a sub-case of a more general concatenation. In [38] Maslov gives us an upper bound $m2^n - 2^{n-1}$ on the state complexity of concatenation of languages accepted by m and n-state DFAs respectively. This upper bound cannot be met if the first language is accepted by a DFA with $k \geq 2$ final states [55]. In this case Yu et al. [55] proved that the upper bound is $m2^n - k2^{n-1}$. Jiraskova et al. [20] showed the tightness of this upper bound by binary witnesses for every k with $1 \leq k \leq n-1$.

Tight upper bound for square comes in 2006 by Rampersad [47] where the tightness of the upper bound $n2^n - 2^{n-1}$ is shown using binary alphabet. The state complexity of square on languages accepted by DFAs with k final states was finally considered in [7]. The upper bound $(n-k)2^n + k2^{n-1}$ was shown to be tight on ternary alphabet, but only for $k < n-1$. The problem of finding binary witness language and the problem of finding a tight upper bound for square of languages accepted by DFAs with $n-1$ final states were left open. Finding witness language accepted by a DFA with k final states has a meaningful application for the square operation on Boolean or alternating finite automata (BFAs, AFAs). An upper bound was shown to be $2^n + n + 1$ by Fellah, Jürgensen, Yu in 1990 [13]. It is known that language is accepted by an n-state AFA if and only if its reverse is accepted by a 2^n-state DFA with half of the states final [13]. Therefore a language whose square on DFA requires $(n-k)2^n + k2^{n-1}$ states and is accepted by a DFA with $n/2$ final states is needed to show the tightness of this upper bound. We discuss these open problems in this chapter. The solutions are based on published paper

that can be found in Appendix [A] at the end of the thesis:

Jirásková, G., Krajňáková, I.: Square on deterministic, alternating, and Boolean finite automata. *International Journal of Foundations of Computer Science* 30(6-7), 1117–1134 (2019).

In [A, Theorem 7] we present binary witness with $k < n-1$ final states meeting the upper bound $n2^n - k2^{n-1}$. This improves the ternary witness for this upper bound from [7, Lemma 2]. Then we inspect the case of $k = n-1$ by discussion in two sub-cases by the finality of the initial state. If the initial state is final then we show the tightness of the upper bound $(n+2)2^{n-2}$ with binary witness in [A, Theorem 8]. Otherwise if the initial state is the only non-final state then we show in [A, Lemma 9] that the upper bound is $(n+3)2^{n-2}$ and it is tight on ternary alphabet. For binary languages the upper bound $(n+3)2^{n-2}-1$ is met [A, Theorem 11]. All these results are summarized in Table 3.1. In the case of unary languages we provide a simplified proof from Pighizzini and Shallit [45] in [A, Theorem 12] to show that the exponentially smaller upper bound $2n-1$ is tight.

We use the presented binary witness language for square on DFAs with 2^n states, 2^{n-1} of them final, to show that upper bound $2^n + n + 1$ for square on AFAs is tight in [A, Theorem 13]. Then in [A, Theorem 14], we extend the application of presented witness language to show that the upper bound $2^n + n$ is tight for square operation on BFAs as well. For unary languages the corresponding upper bound is $n+1$ as shown in [A, Theorem 15]. This means that the binary alphabet is optimal for meeting the upper bound $2^n + n + 1$.

The study of state complexity of square started by the Master thesis in 2016. The results in Slovak language were gradually improved and reworked and served as a foundation for future research done later during this PhD study. All of the results were translated, concised and refined in details. The results for Boolean and alternating automata served as motivation for another paper presented on CSR 2019 in Moscow. The results given by [A, Theorems 1,7,14] and [A, Lemmas 8,9] were obtained in Master thesis [31] (in Slovak) and presented at the DCFS 2017 conference. Their proofs that appeared in IJFSC 2019 were changed to be easier to follow. We used $\cdot$ instead δ to simplify the notation. To have [A, Fig. 2] more transparent, it is drawn for specific values of m, n, k, l instead of a general case in Master thesis. The proof of [A, Theorem 7 on p. 1122] uses $\alpha := m-k-1$ to simplify the exposition. It is divided into 3 cases instead of 9 as was in Master thesis. The proof of [A, Lemma 8 on p. 1125-1126] provides a detailed explanation that a corresponding word is accepted only from a particular state. The former cases (4)-(7) in Master thesis are now covered by one case (3b). We also give a different proof of distin-

guishability here. In the proof of [A, Lemma 9 on p. 1127] we use a different formatting to simplify the readability of the proof. The result in [A, Theorem 11] concerning the tightness of the corresponding upper bound in the binary case and its proof on p.1128-1131 is completely new and did not appear in Master thesis. The same is true for the square on unary DFAs on p.1131. The upper bound $n+1$ given by [A, Theorem 15 on p. 1133] shows the optimality of binary alphabet used to describe AFA witnesses.

k	sc	$\lvert\Sigma\rvert$	reference
$k = 1$	$n2^n - 2^{n-1}$	$\lvert\Sigma\rvert \geq 2$	Rampersad [47]
$2 \leq k \leq n-2$	$n2^n - k2^{n-1}$	$\lvert\Sigma\rvert \geq 2$	[32]
$k = n-1$ and $q_0 \in F$	$(n+2)2^{n-2}$	$\lvert\Sigma\rvert \geq 2$	[32]
$k = n-1$ and $q_0 \notin F$	$(n+3)2^{n-2}$	$\lvert\Sigma\rvert \geq 3$	[32]
$k = n-1$ and $q_0 \notin F$	$(n+3)2^{n-2} - 1$	$\lvert\Sigma\rvert = 2$	[26]

Table 3.1: State complexities of square varying by number of final states.

The next section presents an accidental benefit of our results on square. It uses the witnesses for square on DFAs with more that one final states from [A Theorem 7] to prove the tightness of the corresponding upper bounds on the complexity of concatenation on DFAs and AFAs.

3.1 Corollary for Concatenation

Here we show that the witness from [A, Theorem 7] on square can be useful as a witness for concatenation as well. Take two such automata with m states, k of them final and n states with l of them final. Then by the following theorem the upper bound $(m-k)2^n + k2^{n-1}$ given by [55] is tight on the binary alphabet.

Theorem 3.1. *Let $m, n \geq 4$,$1 \leq k \leq m-2$ and $1 \leq \ell \leq n-1$. Let K and L be languages over Σ recognized by an m-state DFA with k final states and an n-state DFA with l final states, respectively. Then $\mathrm{sc}(KL) \leq (m-k)2^n + k2^{n-1}$ and this bound is tight if $|\Sigma| \geq 2$.*

Proof. The upper bound $(m-k)2^n + k2^{n-1}$ follows from [55, Theorem 2.3]. To prove tightness, let K be accepted by a DFA $A_{m,k} = (Q_A = \{q_0, \ldots, q_{m-1}\}, \Sigma, \cdot_A, q_0, F_A)$ from Fig. 3.1 and L accepted by a DFA $B_{n,l} = (Q_B = \{0, \ldots, n-1\}, \Sigma, \cdot_B, 0, F_B)$ from Fig.3.2 which have the same structure but different sizes m and n with k and l final states respectively.

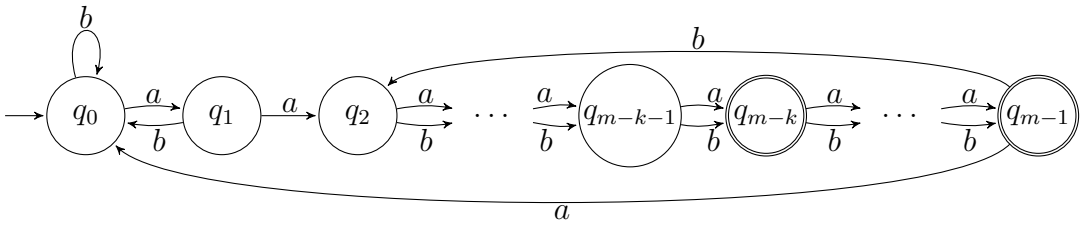

Figure 3.1: DFA $A_{m,k}$ with m states and k of them final.

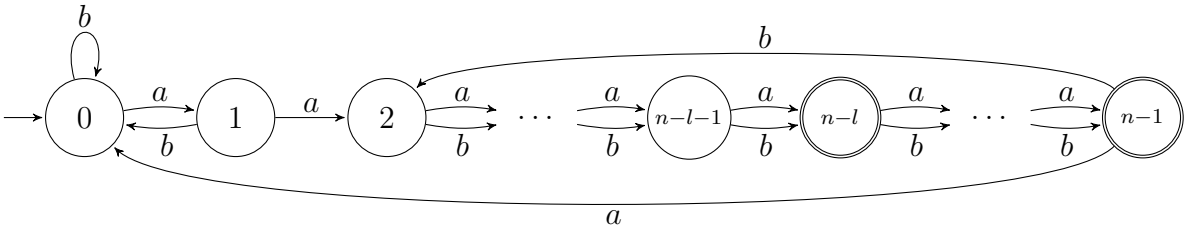

Figure 3.2: DFA $B_{n,l}$ with n states and l of them final.

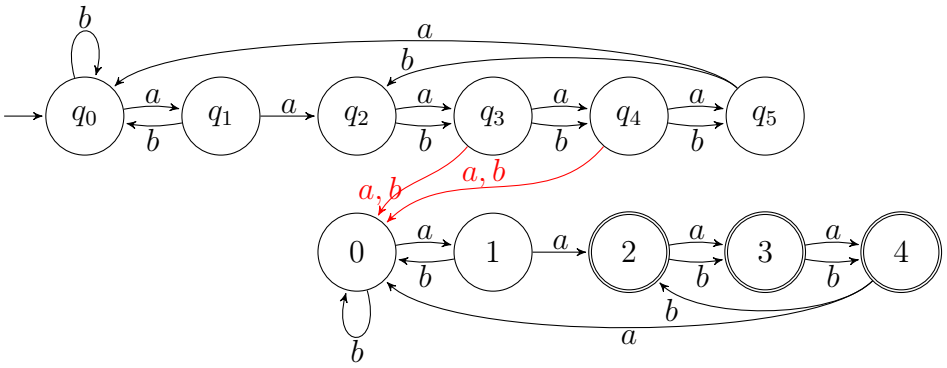

Figure 3.3: NFA N for of $L(A_{6,2})L(B_{5,3})$.

Construct NFA N from DFAs $A_{m,k}$ and $B_{n,l}$ to accept the language KL by adding the transition $(q, a, 0)$ whenever $q \cdot_A a \in F_A$ and $(q, b, 0)$ whenever $q \cdot_A b \in F_A$. The initial state of N is the state q_0 and the set of final states is F_B. An example with $m = 6, k = 2, n = 5, l = 3$ is shown in Fig. 3.3. In the corresponding subset automaton $\mathcal{D}(N)$, each reachable state is in the form $\{q\} \cup S$ where $q \in Q_A$ and $S \subseteq Q_B$. We denote such a state by the pair (q, S). Let

$$\mathcal{R} = \{(q_i, S) \mid 0 \leq i \leq m-k-1 \text{ and } S \subseteq Q_B\} \cup$$

$$\{(q_i, S) \mid m-k \leq i \leq m-1, S \subseteq Q_B \text{ and } 0 \in S\}.$$

The family $\mathcal{R}$ has $(m-k)2^n + k2^{n-1}$ states. We prove by induction on $|S|$ that every state

(q_i, S) in $\mathcal{R}$ is reachable in the subset automaton $\mathcal{D}(N)$. The basis, $|S| = 0$ and $|S| = 1$ holds true since:

$$\to (q_0, \emptyset) \xrightarrow{a} (q_1, \emptyset) \xrightarrow{a} (q_2, \emptyset) \xrightarrow{a} \cdots \xrightarrow{a} (q_{m-k-1}, \emptyset) \xrightarrow{a} (q_{m-k}, \{0\}),$$

$$(q_{m-k}, \{0\}) \xrightarrow{b} (q_{m-k+1}, \{0\}) \xrightarrow{b} \cdots \xrightarrow{b} (q_{m-2}, \{0\}) \xrightarrow{b} (q_{m-1}, \{0\}),$$

$$(q_{m-1}, \{0\}) \xrightarrow{a} (q_0, \{1\}) \xrightarrow{b} (q_0, \{0\}),$$

$$(q_0, \{1\}) \xrightarrow{a} (q_1, \{2\}) \xrightarrow{b} (q_0, \{3\}) \xrightarrow{b} (q_0, \{4\}) \xrightarrow{b} \cdots \xrightarrow{b} (q_0, \{n-1\}) \xrightarrow{b} (q_0, \{2\}),$$

$$(q_0, \{(j-i) \bmod n\}) \xrightarrow{a^i} (q_i, \{j\}) \text{ for } i = 0, 1, \ldots, m-k-1 \text{ and } j = 0, 1, \ldots, n-1.$$

Assume that every state (q_i, S) in $\mathcal{R}$ with $|S| = t-1$ and $i = 0, 1, \ldots, m-1$ is reachable. We show that the state $(q_i, S = \{s_1, s_2, \ldots, s_t\})$ in $\mathcal{R}$ with $|S| = t$, where $q_i \in Q_A$ and $0 \leq s_1 < s_2 < \cdots < s_t \leq n-1$ is reachable as well. We do this in three steps:

(1) We reach states (q_i, S) satisfying the condition $q_i \in F_A$, so $0 \in S$, in other words let $s_1 = 0$ and $m-k \leq i \leq m-1$.

Let $s_2 = 1$. Take the state $(q_{i-1}, S' = \{0, s_3 - 1, \ldots, s_t - 1\}) \in \mathcal{R}$. It is reachable since S' has $t-1$ elements and we have

$$(q_{i-1}, \{0, s_3 - 1, \ldots, s_t - 1\}) \xrightarrow{a} (q_i, \{0, 1, s_3, \ldots, s_t\}) = (q_i, S).$$

Let $s_2 \geq 2$. Remind that there is a cycle $(q_2, q_3, \ldots, q_{m-1})$ formed by the transitions on b in A. Then $S \setminus \{0\} \subseteq \{2, 3, \ldots, n-1\}$ and for each $s \in S \setminus \{0\}$ and $r \geq 0$ there exists exactly one element $s' \in S \setminus \{0\}$ such that $s' \cdot b^r = S$. Denote this element $b^{-r}(s)$. Let $r = i-(m-k-1)$. Then $(q_{m-k-1}, S' = \{b^{-r}(s_2), b^{-r}(s_3), \ldots, b^{-r}(s_t)\})$ is reachable by the induction hypothesis since $|S'| = t-1$ and $(q_{m-k-1}, S') \xrightarrow{b^r} (q_i, \{0, s_2, \ldots, s_t\}) = (q_i, S)$.

(2) Now we reach the states (q_0, S). Our approach varies according to value of s_1.

Let $s_1 = 0$. If $s_2 = 1$ we take the state reached in (1) and we have

$$(q_{m-1}, \{0, s_3 - 1, \ldots, s_t - 1, n-1\}) \xrightarrow{a} (q_0, \{0, 1, s_3, \ldots, s_t\}) = (q_0, S).$$

For $s_2 \geq 2$ we continue using state we reached above, when $s_2 = 1$, and

$$(q_0, \{0, 1, s_3 - s_2 + 1, \ldots, s_t - s_2 + 1\}) \xrightarrow{a} (q_1, \{1, 2, s_3 - s_2 + 2, \ldots, s_t - s_2 + 2\})$$

$$\xrightarrow{b^{n-2}} (q_0, \{0, 2, s_3 - s_2 + 2, \ldots, s_t - s_2 + 2\}) \xrightarrow{b^{s_2-2}} (q_0, \{0, s_2, \ldots, s_t\}) = (q_0, S).$$

Let $s_1 \geq 1$. Then the state $(q_{m-1}, S' = \{0, s_2 - s_1, \ldots, s_t - s_1\})$ is reached in (1). For $s_1 = 1$ we have $(q_{m-1}, S') \xrightarrow{a} (q_0, S)$. For $s_1 \geq 2$ we have

$$(q_{m-1}, \{0, s_2 - s_1, \ldots, s_t - s_1\}) \xrightarrow{aa} (q_1, \{2, s_2 - s_1 + 2, \ldots, s_t - s_1 + 2\}) \xrightarrow{b^{n-2}}$$

$$(q_0, \{2, s_2 - s_1 + 2, \ldots, s_t - s_1 + 2\}) \xrightarrow{b^{s_1-2}} (q_0, \{s_1, s_2, \ldots, s_t\}) = (q_0, S).$$

(3) Let $1 \leq i \leq m - k - 1$. We use states reached in (2) to achieve the remaining states $(q_0, \{(s_1 - i) \bmod n, \ldots, (s_t - i) \bmod n\}) \xrightarrow{a^i} (q_i, \{s_1, \ldots, s_t\}) = (q_i, S)$.

Now we continue with distinguishability. Similarly as in the proof of distinguishability for witness language for square [A, Theorem 7], there is a word $w = b(ab^{n-2})^{n-3}$ which is accepted only from the state $n - 1$ in N. The word $a^{n-1-t}w$ is accepted only from state $t \in \{1, 2, \ldots, n - 1\}$ since t has only one in-transition on a from state $t - 1$. Hence the states (q_i, S) and (q_j, T) from $\mathcal{R}$ are distinguishable if $S \neq T$.

Consider now two distinct states (q_i, S) and (q_j, S) in $\mathcal{R}$. Without loss of generality let $0 \leq i < j \leq m - 1$. We deal with several cases.

(1) Let $i = 0$ and $j = m - 2$.

(1a) First assume that $n - 1 \notin S$. Then

$$(q_0, S) \xrightarrow{a} (q_1, S'),$$

$$(q_{m-2}, S) \xrightarrow{a} (q_{m-1}, \{0\} \cup S').$$

We can distinguish these states since $S' \setminus \{0\} = \emptyset$.

(1b) Now let $n - 1 \in S$. We use word the ab^{m-3} to send $n - 1$ to 0. Moreover if $0 \in S$ then after reading ab^{m-3} it remains in 0. Any other element from S is sent to the corresponding element in $\{2, 3, \ldots, n - 1\}$. So we get

$$(q_0, S) \xrightarrow{ab^{m-3}} (q_0, \{0\} \cup S_1),$$

$$(q_{m-2}, S) \xrightarrow{ab^{m-3}} (q_{m-2}, \{0\} \cup S_1).$$

for some $S_1 \subseteq \{2, 3, \ldots, n - 1\}$. If $n - 1 \notin S_1$ we get case (1a). Otherwise we use again word ab^{m-3} to get $(q_0, \{0\} \cup S_2)$ and $(q_{m-2}, \{0\} \cup S_2)$ for some $S_2 \subseteq \{2, 3, \ldots, n-1\}$ such that $|S_2| < |S_1|$. If after each ab^{m-3} the element $n - 1$ is in the resulting set then we end with $(q_0, \{0\})$ and $(q_{m-2}, \{0\})$ which are distinguishable by a. Otherwise we have (1a).

(2) Let $i = 0$ and $2 \leq j \leq m - 3$. We use b^{m-2-j} to get case (1) or case $S \neq T$.

(3) Let $i = 0$ and $j = m - 1$. We use b to get case (2) or case $S \neq T$.

(4) Let $i = 0$ and $j = 1$. We use ab to get case (2) or case $S \neq T$.

(5) Let $i \geq 1$. Then

$$(q_i, S) \xrightarrow{a^{m-j}} (q_{i+(m-j)}, S_1),$$

$$(q_j, S) \xrightarrow{a^{m-j}} (q_0, S_1').$$

Similarly as in previous cases, if S_1 and S_1' are the same we continue as in (1)–(4), otherwise we continue as in case $S \neq T$. □

Now we show an application of witness languages $L(A_{m,k})$ and $L(B_{n,l})$ for concatenation on AFAs. Fellah et al. showed in [13, Theorem 9.3] that if a language K is accepted by an m-state AFA and a language L is accepted by an n-state AFA then the language KL is accepted by AFA with $2^m + n + 1$ states. The next theorem shows that this upper bound is tight using the binary witness languages provided in Theorem 3.1.

Theorem 3.2. *Let $m, n \geq 2$. There exist binary languages K and L recognized by an m-state and an n-state AFA, respectively, such that every AFA for KL has at least 2^m+n+1 states.*

Proof. Let

$$K = L(B_{2^m,2^{m-1}})^R \text{ and } L = L(A_{2^n,2^{n-1}})^R.$$

Since $L(B_{2^m,2^{m-1}})$ is recognized by a 2^m-state DFA with half of the states final, its reversal, the language K, is recognized by an m-state AFA. Analogously, the language L is recognized by an n-state AFA. Next, as shown in Theorem 3.1 for DFAs, we have

$$\mathrm{sc}((KL)^R) = \mathrm{sc}(L^R K^R) = \mathrm{sc}(L(A_{2^n,2^{n-1}})L(B_{2^m,2^{m-1}})) =$$

$$= 2^{n-1}2^{2^m} + 2^{n-1}2^{2^m-1} = 2^{2^m}2^{n-1}(1 + 1/2).$$

It follows the number of states in every AFA for KL is at least

$$\lceil \log 2^{2^m}2^{n-1}(1 + 1/2) \rceil = 2^m + n.$$

Now we show that every AFA for KL has at least $2^m + n + 1$ states. Assume for a contradiction that KL is recognized by an AFA with $2^m + n$ states. Then by Lemma 2.1, the language $(KL)^R$ is recognized by a DFA with 2^{2^m+n} states, half of which are final. It follows that a minimal DFA for $(KL)^R$ has at most 2^{2^m+n-1} final states. A state (q, S) of the minimal DFA for KL is final if S contains a final state of $B_{2^m,2^{m-1}}$. Hence, the number of final states in the minimal DFA for $(KL)^R$ is

$$2^{n-1}(2^{2^{m-1}} - 1)2^{2^{m-1}} + 2^{n-1}(2^{2^{m-1}} - 1)(2^{2^{m-1}-1})$$

$$= 2^{2^m+n-1}(1 - \frac{1}{2^{2^{m-1}}} + \frac{1}{2} - \frac{1}{2^{2^{m-1}+1}}) > 2^{2^m+n-1}$$

since $m \geq 2$. This is a contradiction, so every AFA for KL has at least 2^m+n+1 states. □

Chapter 4

Operations on Boolean and Alternating Finite Automata

In this chapter we investigate the descriptional complexity of basic regular operations on languages represented by Boolean and alternating finite automata (BFAs, AFAs). This representation differs from nondeterministic finite automata (NFAs). In NFAs the result of a transition function is the disjunction of states. In the Boolean automata we allow any Boolean function, not only disjunction to be the result of the transition function. If an NFA starts in a disjunction of states, we say that it has multiple initial states, but a BFA can start in any Boolean function. A Boolean automaton that starts in a single state, a Boolean function that is a projection, is called an alternating finite automaton (AFA); cf. [8, 13, 54].

It is known that every n-state Boolean automaton can be simulated by a DFA with 2^{2^n} states and by a NFA with 2^n+1 states, and that both these upper bounds can be met by binary languages [4, 25]. Fellah, Jürgensen and Yu [13] presented simple constructions of AFAs for several basic regular operations on languages represented by AFAs. This resulted in the following upper bounds on the AFA state complexity of corresponding operations:

- complementation $\leq n+1$,
- union $\leq m+n+1$,
- intersection $\leq m+n+1$,
- star $\leq 2^n+1$,
- concatenation $\leq 2^m+n+1$.

Jirásková in [25] provided tight upper bounds on AFAs and BFAs for union, intersection. Lower bounds were showed for star, reversal, and concatenation:

	AFA state complexity	BFA state complexity
• union	$m+n+1$,	$m+n$,
• intersection	$m+n+1$,	$m+n$,
• concatenation	$2^m+n <= . <= 2^m+n+1$,	2^m+n,
• reversal	$2^n <= . <= 2^n+1$,	2^n,
• star	$2^n <= . <= 2^n+1$,	$2^n <= . <= 2^n+1$.

In this chapter we continue this research with the aim to get the exact AFA and BFA state complexity for all the above mentioned operations, as well as for operations of left and right quotients, difference and symmetric difference. The solutions are based on paper presented at CSR 2018 that can be found in Appendix [B] at the end of the thesis:

> Michal Hospodár, Galina Jirásková, Ivana Krajňáková: Operations on Boolean and alternating finite automata. In: Fedor V. Fomin, Vladimir V. Podolskii (eds.): *Computer Science – Theory and Applications – 13th International Computer Science Symposium in Russia, CSR 2018, Moscow, Russia, June 6–10, 2018, Proceedings.* Lecture Notes in Computer Science, vol. 10846, pp. 181–193, Springer (2018).

We start with [B, Proposition 1] dealing with so called uniquely distinguishable states and we use them to get the distinguishability of subsets in a subset automaton. In [B, Lemmas 2,3] we give two observations from the literature showing that a language L is recognized by an n-state BFA (an n-state AFA) if and only if the language L^R is recognized by a 2^n-state DFA (a 2^n-state DFA with half of its states final). This is a crucial observation that is used throughout this chapter and paper [B] to get the exact complexity of all considered operations on BFAs and AFAs. This observation follows from Lemmas 2.1 and 2.2 given together with their proof ideas in Known results. Proposition [B, Proposition 7] describes the reversal of quotients. Let us provide the omitted proof here.

Proposition 4.1. *Let K and L be languages over Σ. Then*

(a) $(KL^{-1})^R = (L^R)^{-1}K^R$;

(b) $(L^{-1}K)^R = K^R(L^R)^{-1}$.

Proof. (a) The equality $(KL^{-1})^R = (L^R)^{-1}K^R$ follows from the fact that the following claims are equivalent:

$$w \in (KL^{-1})^R,$$

$$w^R \in KL^{-1},$$
$$w^R = x \text{ where } xy \in K \text{ for some } y \in L,$$
$$w = x^R \text{ where } y^R x^R \in K^R \text{ for some } y^R \in L^R,$$
$$w \in (L^R)^{-1} K^R.$$

(b) The equality $(L^{-1}K)^R = K^R (L^R)^{-1}$ follows from the fact that the following claims are equivalent:

$$w \in (L^{-1}K)^R,$$
$$w^R \in L^{-1}K,$$
$$w^R = x \text{ where } yx \in K \text{ for some } y \in L,$$
$$w = x^R \text{ where } x^R y^R \in K^R \text{ for some } y^R \in L^R,$$
$$w \in K^R (L^R)^{-1}.$$

□

In [B,Proposition 8 and 9] we prove preliminary results on star and symmetric difference on DFA that are used later to get the complexity of star and symmetric difference on BFAs and AFAs.

Now, with the aim to decrease the size of the alphabet in defining witnesses for Boolean operation in [B], we consider the unary case here.

Lemma 4.2. *Let* $A = (2^m, 0, \{i \mid 2^{m-1} \leq i \leq 2^m - 1\})$ *and* $B = (2^n - 1, 0, \{i \mid 2^{n-1} \leq i \leq 2^n - 2\})$ *unary DFAs, see Fig. 4.1 and Fig. 4.2, where* $m, n \geq 2$. *Then* $\mathrm{sc}(L(A) \cup L(B)) = \mathrm{sc}(L(A) \oplus L(B)) = 2^m(2^n - 1)$.

Proof. Our aim is to show that all states in the product automata $M_\cup$ and $M_\oplus$ for $L(A) \cup L(B)$ and $L(A) \oplus L(B)$, respectively, are reachable and pairwise distinguishable.

The numbers 2^m and $2^n - 1$ are co-prime and greater than or equal to two. By Chinese Remainder Theorem, for every i and j with $0 \leq i \leq 2^m - 1$ and $0 \leq j \leq 2^n - 2$ there exists a number $x(i, j)$ such that

$$x(i, j) \bmod 2^m = i,$$
$$x(i, j) \bmod (2^n - 1) = j.$$

It follows that in the product automaton $M_\cup$ or $M_\oplus$, every state (i, j) is reached from the initial state $(0, 0)$ by the word $a^{x(i,j)}$.

For distinguishability, we first consider union. Notice that it is enough to distinguish the non-final state $(0,0)$ with any other non-final state: indeed, if p and q are two distinct states of the product automaton $M_\cup$, then we can send the state p to $(0,0)$ by a word in a^*, while q is sent to a state q' with $q' \neq (0,0)$ by this word. If q' if final, then we are done since $(0,0)$ is non-final. Otherwise, we have the above mentioned case.

Consider a non-final state $(i,j) \neq (0,0)$ in the product automaton $M_\cup$. Then $0 \leq i \leq 2^{m-1}-1$ and $0 \leq j \leq 2^{n-1}-1$. Moreover $i \geq 1$ or $j \geq 1$.

If $i \geq 1$, then

$$(0,0) \xrightarrow{a^{x(2^{m-1}-1,0)}} (2^{m-1}-1,0) \notin F_\cup \text{ while}$$
$$(i,j) \xrightarrow{a^{x(2^{m-1}-1,0)}} (2^{m-1}-1+i,j) \in F_\cup.$$

If $j \geq 1$, then

$$(0,0) \xrightarrow{a^{x(0,2^{n-1}-1)}} (0,2^{n-1}-1) \notin F_\cup \text{ while}$$
$$(i,j) \xrightarrow{a^{x(0,2^{n-1}-1)}} (i,2^{n-1}-1+j) \in F_\cup.$$

This proves distinguishability for union.

Now consider symmetric difference. Similarly as for union, it is enough to distinguish the non-final state $(0,0)$ with any other non-final state. First, consider a non-final state (i,j) with $0 \leq i \leq 2^{m-1}-1$ and $0 \leq j \leq 2^{n-1}-1$. Then we proceed as in the case of union: the word $a^{x(2^{m-1}-1,0)}$ distinguishes $(0,0)$ and (i,j) if $i \geq 1$, otherwise, we use the word $a^{x(0,2^{n-1}-1)}$ to distinguish them.

Now consider a non-final state $(2^{m-1}+i, 2^{n-1}+j)$ with $0 \leq i \leq 2^{m-1}-1$ and $0 \leq j \leq 2^{n-1}-2$.

If $i = 0$, then we have

$$(0,0) \xrightarrow{a^{x(2^{m-1},2^{n-1}-1)}} (2^{m-1},2^{n-1}-1) \in F_\oplus \text{ while}$$
$$(2^{m-1},2^{n-1}+j) \xrightarrow{a^{x(2^{m-1},2^{n-1}-1)}} (0,j) \notin F_\oplus;$$

notice that in B we have $2^{n-1}+j \xrightarrow{a^{2^{n-1}-2-j}} 2^n-2 \xrightarrow{a} 0 \xrightarrow{a^j} j$.

If $i \geq 1$, then

$$(0,0) \xrightarrow{a^{x(2^{m-1}-1,0)}} (2^{m-1}-1,0) \notin F_\oplus \text{ while}$$
$$(2^{m-1}+i,2^{n-1}+j) \xrightarrow{a^{x(2^{m-1}-1,0)}} (i-1,2^{n-1}+j) \in F_\oplus.$$

This proves distinguishability for symmetric difference, and concludes the proof. □

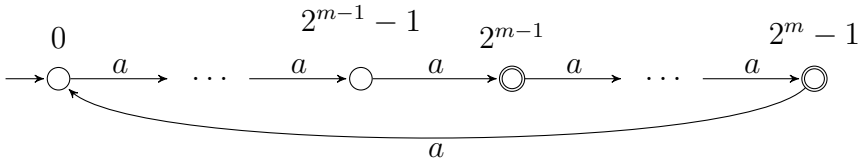

Figure 4.1: Unary DFA $A = (2^m, 0, \{i \mid 2^{m-1} \leq i \leq 2^m - 1\})$.

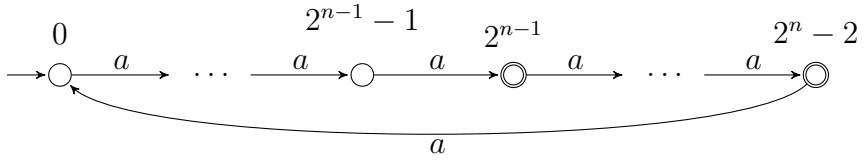

Figure 4.2: Unary DFA $B = (2^n - 1, 0, \{i \mid 2^{n-1} \leq i \leq 2^n - 2\})$.

Theorem 4.3. *Let $m, n \geq 2$ and $\circ \in \{\cup, \oplus, \cap, \setminus\}$. Let K and L be unary languages recognized by BFAs with m and n states, respectively. Then the language $K \circ L$ is recognized by a BFA with $m + n$ states, and this upper bound is tight.*

Proof. The upper bound is the same as in the case of a general alphabet. To prove tightness for union and symmetric difference, let A and B be the DFAs from Lemma 4.2. Let

$$K = L(A)^R \text{ and } L = L(B)^R.$$

Since $L(A)$ is recognized by an 2^m-state DFA, the language K is recognized by an m-state BFA by Lemma 1.3. Next, the language $L(B)$ is recognized by an 2^n-state DFA; here we can add an unreachable state $2^n - 1$ to the DFA B. Therefore, the language L is recognized by an n-state BFA again by Lemma 1.3. We have $(K \cup L)^R = K^R \cup L^R = L(A) \cup L(B)$. By Lemma 4.2, every DFA for the language $(K \cup L)^R$ has at least $2^m(2^n - 1)$ states. By Lemma 1.2, the number of states in every BFA for $K \cup L$ is at least

$$\lceil \log 2^m(2^n - 1) \rceil = m + n.$$

The proof for symmetric difference is exactly the same.

Notice that the languages K^c and L^c are witnesses for intersection since we have $(K^c \cap L^c)^c = K \cup L$, and on BFAs, the state complexity of a language and its complement is the same.

Similarly, the languages K^c and L are witnesses for difference since $K^c \setminus L = K^c \cap L^c$. Our proof is complete. □

Theorem 4.4. *Let $m, n \geq 3$ and $\circ \in \{\cup, \cap, \setminus\}$. Let K and L be unary languages recognized by AFAs with m and n states, respectively. Then the language $K \circ L$ is recognized by an AFA with $m+n+1$ states, and this upper bound is tight. The language $K \oplus L$ is recognized by an AFA with $m+n$ states, and this upper bound is tight.*

Proof. The upper bounds are the same as in the case of a general alphabet. For tightness in the case of union, let A and B be DFAs from Lemma 4.2. Let

$$K = L(A)^R \text{ and } L = L(B)^R.$$

Notice that $L(A)$ is recognized by a DFA with 2^m states of which 2^{m-1} are final. By adding an unreachable final state $2^n - 1$ to the DFA B, we get a DFA for $L(B)$ which has 2^n states of which 2^{n-1} are final. By Lemma 1.3, the language K is recognized by an m-state AFA and the language L is recognized by an n-state AFA. From the previous proof, we already know that every BFA, so also every AFA, for $K \cup L$ has at least $m+n$ states. Let us show that one more state is necessary for an AFA to accept $K \cup L$.

Assume for a contradiction that there is an AFA for $K \cup L$ with $m+n$ states. Then by Lemma 1.2, the language $(K \cup L)^R$ is recognized by a DFA with 2^{m+n} states of which 2^{m+n-1} are final. It follows that the minimal DFA for $(K \cup L)^R$ has at most 2^{m+n-1} final states. However, we have $(K \cup L)^R = K^R \cup L^R = L(A) \cup L(B)$, and it follows from the proof of Lemma 4.2 that the number of final states in the minimal DFA for $L(A) \cup L(B)$ is

$$2^{m-1}(2^{n-1} - 1) + 2^{m-1}(2^n - 1) = 2^{m+n-1}(1 + \frac{1}{2} - \frac{1}{2^{n-1}}) > 2^{m+n-1}$$

since $n \geq 3$. This is a contradiction. Thus every AFA for $K \cup L$ has at least $m+n+1$ states.

Since the AFA complexity of a language and its complement is the same, the languages K^c and L^c are witnesses for intersection, while K^c and L are witnesses for difference.

Finally, the languages K and L meet the upper bound $m+n$ for symmetric difference, since we already know from the proof of Theorem 4.3 that every BFA, so also every AFA, for $K \oplus L$ has at least $m+n$ states. □

All our results are summarized in Table 4.1 and Table 4.2:

operation	BFA	$\|\Sigma\|$	Theorem	AFA	$\|\Sigma\|$	Theorem
complementation	n	1	[B, Thm. 10]	n	1	[B, Thm. 10]
union	$m+n$	1	Theorem 4.3	$m+n+1$	1	Theorem 4.4
intersection	$m+n$	1	Theorem 4.3	$m+n+1$	1	Theorem 4.4
difference	$m+n$	1	Theorem 4.3	$m+n+1$	1	Theorem 4.4
sym. difference	$m+n$	1	Theorem 4.3	$m+n$	1	Theorem 4.4
concatenation	2^m+n	2	Theorem 3.2	2^m+n+1	2	Theorem 3.2
star	2^n	2	[B, Thm. 11]	2^n	2	[B, Thm. 11]
reversal	2^n	2	[B, Thm. 12(c)]	2^n	2	[B, Thm. 13(c)]
right quotient	2^m	2	[B, Thm. 12(d)]	2^m+1	2	[B, Thm. 13(d)]
left quotient	m	1	[B, Thm. 12(e)]	$m+1$	1	[B, Thm. 13(e)]

Table 4.1: State complexities of operations on Boolean and alternating automata.

operation	BFA	Theorem	AFA	Theorem
reversal	n	[B, Theorem 14(e)]	n	[B, Corollary 15(d)]
star	$2n$	[B, Theorem 14(f)]	$\leq 2n+1$	[B, Corollary 15(e)]
left=right quotient	m	[B, Theorem 14(g)]	$\leq m+1$	[B, Corollary 15(f)]

Table 4.2: State complexities of operations on unary Boolean and alternating automata.

Chapter 5

NFA-to-DFA Trade-Off

This chapter discourses a different view on the state complexity trade-off for regular operations. We investigate the NFA-to-DFA trade-off which assumes that the arguments entering an operation are represented by NFAs while the result is required to be a DFA.

We are motivated by two research problems from the literature: the complexity of the combined operations and the operational complexity on languages represented by self-verifying and unambiguous automata. The study of combined operations began with the paper by Salomaa et al. [49], and since then many different combined operations have been studied. If a given combined operation does not contain complementation then NFA-to-DFA trade-off for the outermost operation provides an upper bound on the complexity of the combined operation since we can perform each included operation on NFAs. Self-verifying and unambiguous automata are special forms of NFAs, while every DFA is self-verifying and unambiguous automaton. Therefore the NFA-to-DFA trade-off for a regular operation provides an upper bound on the complexity of the operation on self-verifying or unambiguous automata.

We examine the NFA-to-DFA trade-off for star, complementation, intersection, union, difference, symmetric difference, reversal, left and right quotient and concatenation. We start with the star operation where we provide tight upper bound 2^n with a binary witness and resolve the unary case where the tight upper bound is $(n-1)^2+2$. Then we show the tightness of the upper bound 2^n for complementation operation with binary witness, and asymptotically tight upper bound $F(n)$ in the unary case. We continue with Boolean operations where we provide upper bounds and show their tightness on ternary alphabet. We show that upper bound 2^{m+n} is asymptotically tight for binary alphabet for all four Boolean operations. For unary alphabet we give an estimation $\Theta(F(m+n))$; here $F(n)$ is Landaus function defined as $F(n) = \max\{\text{lcm}(x_1, \ldots, x_k) \mid x_1 + \cdots + x_k \leq n\}$ and it

is known that $F(n) \approx 2^{\sqrt{n \ln n}}$. We describe straight-forward solutions on NFA-to-DFA trade-off for reversal, left and right quotient, on binary (or larger) and unary alphabets. We conclude this chapter with the concatenation operation where we provide tight upper bound $\frac{3}{4}2^{m+n}$ with a ternary witness. We show that this upper bound is asymptotically tight in the binary case. In the unary case, we get an upper bound $O(F(m+n))$ and a lower bound $\Omega(\max\{F(m), F(n)\})$.

The next three lemmas provide tools to guarantee reachability and distinguishability of all states in the subset automaton of an NFA with appropriate structure. Languages described by NFAs with similar or same transitions as in these lemmas are used as witness languages in this chapter.

Lemma 5.1. *Let* $A = (\{0, 1, \ldots, n-1\}, \{a, b\}, \cdot, 0, \{n-1\})$ *be the* NFA *from Fig. 5.1 where* $i \cdot a = \{i+1\}$ *if* $0 \leq i \leq n-2$, *and* $i \cdot b = \{0, i\}$. *Then for each subset* S *of* $\{0, 1, \ldots, n-1\}$, *there exists a word* $u_S \in \{a, b\}^*$ *such that* $0 \cdot u_S = S$.

Proof. In the subset automaton $\mathcal{D}(A)$, each singleton set $\{i\}$ is reached from the initial subset $\{0\}$ by a^i and the empty set is reached from $\{n-1\}$ by a. Each set $\{i_1, i_2, \ldots, i_k\}$ of size k, where $2 \leq k \leq n$ and $0 \leq i_1 < i_2 < \cdots < i_k \leq n-1$, is reached from the set $\{i_2 - i_1, i_3 - i_1, \ldots, i_k - i_1\}$ of size $k-1$ by ba^{i_1}. This proves the reachability of all subsets of $\{0, 1, \ldots, n-1\}$ by induction. Hence for each $S \subseteq \{0, 1, \ldots, n-1\}$, there is a word $u_S \in \{a, b\}^*$ such that $0 \cdot u_S = S$. □

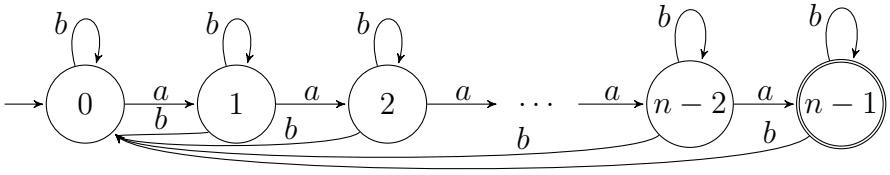

Figure 5.1: An NFA with 2^n reachable subsets.

Lemma 5.2. *Let* $A = (\{0, 1, \ldots, n-1\}, \{a, b\}, \cdot, 0, \{n-1\})$ *be the* NFA *from Fig. 5.2 where* $i \cdot a = \{(i+1) \bmod n\}$ *and* $i \cdot b = \{0, i\}$ *if* $1 \leq i \leq n-1$. *Then for each subset* S *of* $\{0, 1, \ldots, n-1\}$, *there exists a word* $u_S \in \{a, b\}^*$ *such that* $0 \cdot u_S = S$.

Proof. The proof is the same as the proof of Lemma 5.1, but now the empty set is reached from $\{0\}$ by b. □

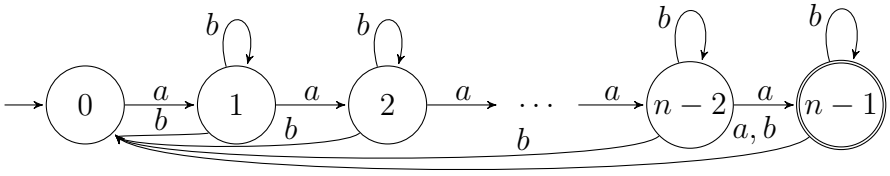

Figure 5.2: An NFA with 2^n reachable subsets.

Lemma 5.3 (Distinguishability)**.** *Let A be an* NFA *such that for every state q of A the singleton set $\{q\}$ is co-reachable in A. Then every two distinct states of the subset automaton $\mathcal{D}(A)$ are distinguishable.*

Proof. Let us take two distinct subsets S and T of $\mathcal{D}(A)$. Without loss of generality, let $q \in S \setminus T$. Since the set $\{q\}$ is co-reachable in A, there is a word w_q that is accepted by A from the state q and rejected from every other state. It follows that in $\mathcal{D}(A)$, the word w_q is accepted from S and rejected from T. Hence S and T are distinguishable in $\mathcal{D}(A)$. □

5.1 Star

We start with the star operation. Its state complexity is $\frac{3}{4}2^n$ [38] and its nondeterministic state complexity is $n+1$ [24]. We show that the NFA-to-DFA trade-off for star is 2^n.

Theorem 5.4 (Star)**.** *Let $n \geq 2$. Let L be a language over an alphabet Σ recognized by an n-state* NFA*. Then $\mathrm{sc}(L^*) \leq 2^n$, and the bound is tight if $|\Sigma| \geq 2$.*

Proof. Let L be recognized by an n-state NFA $A = (Q, \Sigma, \cdot, s, F)$. Construct an MNFA N recognizing L^* from A as follows. First, for each transition (p, a, q) in A with $q \in F$, add the transition (p, a, s). Next, if $s \notin F$, then add a new initial and final state q_0 to accept the empty word. Consider the subset automaton $\mathcal{D}(N)$. The only reachable set in $\mathcal{D}(N)$ containing the state q_0 is the initial subset $\{s, q_0\}$. All the remaining reachable sets are subsets of Q. Moreover, if a reachable set contains a final state of A, then it also contains the state s. If A has a final state different from s, then at least 2^{n-2} sets are unreachable in $\mathcal{D}(N)$, so the upper bound is $1+(3/4)2^n$ in this case. If $F = \{s\}$, then the construction above results in the same automaton, so $L^* = L$. In such a case, the upper bound is 2^n.

To prove tightness, consider the binary language L recognized by the n-state NFA $A = (\{0, 1, \ldots, n-1\}, \{a, b\}, \cdot, 0, \{0\})$ shown in Fig. 5.3 where for each state i, $i \cdot a = \{i + 1 \bmod n\}$, and $i \cdot b = \{0, i\}$ if $i \geq 1$. Then $L^* = L$. In the subset automaton $\mathcal{D}(A)$, all subsets of $\{0, \ldots, n-1\}$ are reachable by Lemma 5.2. Since every singleton set is

co-reachable in A via a word in a^*, all the states of $\mathcal{D}(A)$ are pairwise distinguishable by Lemma 5.3. □

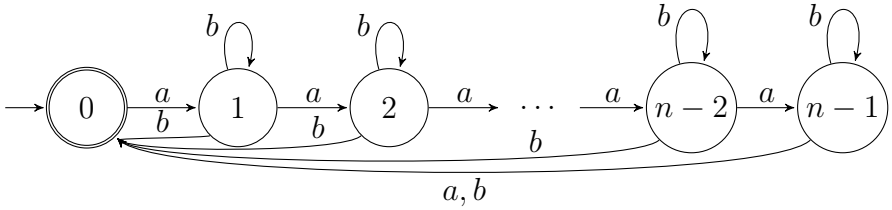

Figure 5.3: A binary witness NFA for star meeting the upper bound 2^n.

The witness from the previous proof is described over a binary alphabet. It is impossible to meet the upper bound 2^n in the unary case since every unary n-state NFA can be simulated be a DFA with $2^{O(\sqrt{n \ln n})}$ states [9]. The next theorem provides the tight upper bound in the unary case.

Theorem 5.5. *Let $n \geq 6$. Let L be a unary language recognized by an n-state NFA. Then* $\mathrm{sc}(L^*) \leq (n-1)^2 + 2$*, and this bound is tight.*

Proof. To get lower bound consider the n-state NFA from Fig.5.4. This NFA recognizes the language $L = \{\varepsilon\} \cup \{a^{cn+d(n-1)} \mid c > 0, d \geq 0\}$ and $L^* = L$ since 0 is the unique final state. We have $\gcd(n, n-1) = 1$. By [55, Lemma 5.1](b) the largest integer that cannot be presented as $cn + d(n-1)$ with $c > 0, d \geq 0$ is $(n-1)^2$. It follows that the minimal DFA for $L^* = L$ has $(n-1)^2 + 2$ states.

Now we show that $(n-1)^2 + 2$ states are sufficient. Let $A = (Q, \{a\}, \cdot, s, F)$ be an arbitrary unary n-state NFA. First, assume that there is at least one final state $f \neq s$. Construct an NFA $N = (Q, \{a\}, \circ, s, F)$ for language $L(A)^+$ from A by adding transition (q, a, s) whenever $q \cdot a \in F$. To get a DFA for $L(A)^*$ it is enough to make the initial state $\{s\}$ of the DFA $\mathcal{D}(N)$ final; notice that $\{s\}$ has no in-transitions in $\mathcal{D}(N)$.

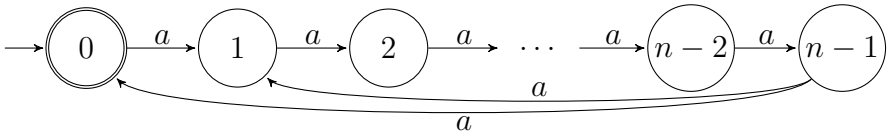

Figure 5.4: A unary witness for star meeting the upper bound $(n-1)^2 + 2$.

Let t be smallest number such that $s \in s \circ a^t$ in N. Define

$$S_{i,t} = \{s\} \circ a^{it}.$$

Our aim is to show that $S_{i-1,t} \subseteq S_{i,t}$. For $i = 1$ this claim holds since $S_{0,t} = \{s\}$ and $S_{1,t} = \{s\} \circ a^t = \{s\} \cup S'$ for some set $S' \subseteq Q$. If $i > 1$ then

$$S_{i,t} = \{s\} \circ a^{it} =$$

$$(\{s\} \cup S') \circ a^{(i-1)t} = (\{s\} \circ a^{(i-1)t}) \cup (S' \circ a^{(i-1)t}) = S_{i-1,t} \cup (S' \circ a^{(i-1)t}) \supseteq S_{i-1,t}.$$

We have two cases:

(1) $S_{i-1,t} = S_{i,t}$, for some $i \leq n-1$;

(2) $S_{n-1,t} = Q$.

In both cases the number of states in the subset automaton $\mathcal{D}(N)$ is at most $t(n-1)+1$ which is at most $(n-1)^2+1$.

Now let $F = \{s\}$. Then $L(A)^* = L(A)$, which means that A is an NFA for $L(A)^*$. If no transition in A goes to s, then $L(A) = \{\varepsilon\}$ and DFA for such language needs only one state. Otherwise, let t be the smallest number such that $s \in s \cdot a^t$ and $S_{i,t}$ be defined as above. If $t \leq n-1$ then we get an upper $(n-1)^2+1$ in the same way as we did when $F \neq \{s\}$.

Let $t = n$. This means that we can label the states in Q with numbers $0, 1, \ldots, n-1$ such that $0 = s$ and NFA A has the transition $(i, a, i+1)$ for $i = 0, 1, \ldots, n-2$ and the transition $(n-1, a, 0)$ and does not have any transition (i, a, j) where $j > i+1$. If there is no transition (i, a, j) with $j \leq i$, then A is deterministic. Otherwise, each such transition forms a loop $(j, j+1, \ldots, i)$ of length $i-j+1$. Assume that there exist k loops of distinct lengths $l_1, \ldots, l_k$ in A.

First consider $k \geq 2$. Then $S_{1,n} \subseteq \{s, n-l_1, n-l_2, ..., n-l_k\}$, so S_1 has at least 3 elements. Moreover $S_{i-1,n} = S_{i,n}$ for some $i \leq n-2$, or $S_{n-2,n} = Q$. So automaton $\mathcal{D}(A)$ has at most $1 + (n-2)n = (n-1)^2$ reachable states, which is less than $(n-1)^2+2$.

Now let $k = 1$. Then A accepts words of length $cn + dl_1$, where $c > 0$ and $d \geq 0$. If $\gcd(n, l_1) = 1$ then by [55, Lemma 5.1](b) the largest integer that cannot be presented as $cn + dl_1$ is $(n-1)l_1$. This means that $\mathcal{D}(A)$ has $(n-1)l_1 + 2$ states, which is at most $(n-1)^2+2$.

But if, on the contrary, $\gcd(n, l_1) = q, q \geq 2$, we can reduce A with q and construct NFA A' where every q transitions in A would be one in A' and there would be $\frac{n}{q}$ states

instead of n. Since $\gcd(\frac{n}{q}, \frac{l_1}{q}) = 1$ we can use [55, Lemma 5.1](b), so the largest integer that cannot be presented as $c\frac{n}{q} + d\frac{l_1}{q}$ is $(\frac{n}{q} - 1)\frac{l_1}{q}$. Thus the number of states needed for $\mathcal{D}(A')$ is $(\frac{n}{q} - 1)\frac{l_1}{q} + 2 = \frac{nl_1}{q^2} - \frac{l_1}{q} + 2$. Now we can expand $\mathcal{D}(A')$ to $\mathcal{D}(A)$. The number of states needed for $\mathcal{D}(A)$ is $(\frac{nl_1}{q^2} - \frac{l_1}{q} + 2)q$ and we can estimate this number for $n \geq 6$: $\left(\frac{nl_1}{q^2} - \frac{l_1}{q} + 2\right) q = \frac{nl_1}{q} - l_1 + 2q < \frac{nl_1}{2} + 2l_1 = l_1\left(\frac{n}{2} + 2\right) \leq (n-1)\left(\frac{n}{2} + 2\right) < (n-1)^2 + 2$. □

5.2 Boolean Operations

We continue with complementation, union, intersection, difference and symmetric difference. Let us first provide tight upper bound 2^n for complementation which contrasts its state complexity n and is the same as its non-deterministic state complexity 2^n [24].

Theorem 5.6 (Complementation)**.** *Let L be a language over Σ recognized by an n-state* NFA*. Then $\mathrm{sc}(L^c) \leq 2^n$, and the bound is tight if $|\Sigma| \geq 2$.*

Proof. Let A be an n-state NFA recognizing L. We apply the subset construction to A and get a 2^n-state DFA recognizing L. Then we invert the finality of every state to get a 2^n-state DFA recognizing L^c. For tightness, let L be the binary language recognized by the NFA A shown in Fig. 5.1 on page 31. By Lemma 5.1, every subset of $\{0, 1, \ldots, n-1\}$ is reachable in the subset automaton $\mathcal{D}(A)$. Since every singleton set is co-reachable in A via a word in a^*, all states of $\mathcal{D}(A)$ are pairwise distinguishable by Lemma 5.3. Thus the minimal DFA for L has 2^n states, and we get $\mathrm{sc}(L^c) = \mathrm{sc}(L) = 2^n$. □

The binary alphabet used in the previous theorem is optimal since every unary n-state NFA can be simulated by a DFA of $2^{O(\sqrt{n \ln n})}$ states [9]. The next theorem gives an asymptotically tight upper bound on the NFA-to-DFA trade-off for complementation in the unary case.

Theorem 5.7. *Let L be a unary language recognized by an n-state NFA. Then $\mathrm{sc}(L^c) = \Theta(F(n))$.*

Proof. The upper bound is given by the complexity of determinization of n-state unary NFA which is $O(F(n))$ by [9, Theorem 4.4]. To get tightness consider the partition $n-1 = x_1 + \cdots + x_k$ such that $F(n-1) = \mathrm{lcm}(x_1, \ldots, x_k)$. Let L be the language recognized by n-state NFA with a non-final initial state which has nondeterministic transitions to k non final states that are in disjoint cycles of lengths $x_1, \ldots, x_k$. All the remaining states in each cycle are final. See Fig. 5.5 for an example with $x_1 = 2, x_2 = 3, x_3 = 5$. Then DFA for L has a tail of length 1 consisting of non-final states and a loop of length $\mathrm{lcm}(x_1, \ldots, x_k)$.

Each state from the loop is final with the exception for the state that follows the tail. This means that the loop is minimal, so $\mathrm{sc}(L^c) = \mathrm{sc}(L) \geq F(n-1) = \Omega(F(n))$. □

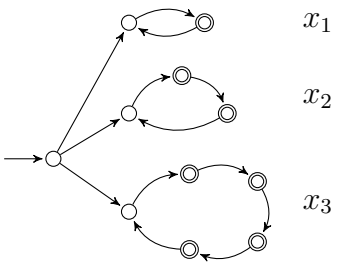

Figure 5.5: Example with $x_1 = 2, x_2 = 3, x_3 = 5$.

Now we continue NFA-to-DFA trade-off for four binary Boolean operations. First, we recall some notions. We call a state q of a DFA $A = (Q, \Sigma, \cdot, s, F)$ a *sink state* if $q \cdot a = q$ for every letter $a \in \Sigma$. The state q is called *dead* if reading every word from the state q results in a non-accepting state of A.

To get an automaton recognizing union, intersection, difference, or symmetric difference of two languages we use the product construction as described in Preliminaries. Let us recall its construction here.

Let $A = (Q_A, \Sigma, \cdot_A, s_A, F_A)$ and $B = (Q_B, \Sigma, \cdot_B, s_B, F_B)$ be DFAs over an alphabet Σ. Let $\circ \in \{\cap, \cup, \setminus, \oplus\}$. Then the language $L(A) \circ L(B)$ is recognized by the product automaton

$$M_\circ = (Q_A \times Q_B, \Sigma, \cdot, (s_A, s_B), F_\circ)$$

where $(p, q) \cdot a = (p \cdot_A a, q \cdot_B a)$ for all $p \in Q_A$, $q \in Q_B$, and $a \in \Sigma$, and

$$F_\circ = \begin{cases} F_A \times F_B, & \text{if } \circ = \cap; \\ (F_A \times Q_B) \cup (Q_A \times F_B), & \text{if } \circ = \cup; \\ F_A \times (Q_B \setminus F_B), & \text{if } \circ = \setminus; \\ (F_A \times (Q_B \setminus F_B)) \cup ((Q_A \setminus F_A) \times F_B), & \text{if } \circ = \oplus. \end{cases}$$

If the operation inputs are given by NFAs, we first apply the subset construction to get DFAs for those inputs. Then we construct the corresponding product automaton. Notice that every subset automaton has at least one rejecting sink state, namely, the empty set. The following lemma provides upper bounds for Boolean operations on DFAs considering the presence of the rejecting sink states.

Lemma 5.8. *Let K and L be languages over Σ accepted by* DFA*s with m and n states respectively. Assume that both* DFA*s have a rejecting sink state. Then* $\mathrm{sc}(K \cup L) \leq mn$, $\mathrm{sc}(K \oplus L) \leq mn$, $\mathrm{sc}(K \cap L) \leq mn - m - n + 2$, *and* $\mathrm{sc}(K \setminus L) \leq mn - n + 1$.

Proof. For each Boolean operation $\circ \in \{\cup, \cap, \setminus, \oplus\}$, the language $K \circ L$ is recognized by the product automaton $M_\circ$ which has mn states. This gives the upper bounds for union and symmetric difference. Let d_A and d_B be the rejecting sink states of A and B, respectively. Then in the product automaton $M_\cap$ recognizing $K \cap L$, the states (d_A, q) with $q \in Q_B$ and the states (p, d_B) with $p \in Q_A$ are dead and can be merged into one sink state. This gives the upper bound $(m-1)(n-1)+1 = mn - m - n + 2$. In the product automaton $M_\setminus$ recognizing $K \setminus L$, the states (d_A, p) with $p \in Q_B$ are dead, which gives the upper bound $(m-1)n + 1 = mn - n + 1$. □

Now we are ready to get tight upper bounds on NFA-to-DFA trade-off for Boolean operations.

Theorem 5.9 (Boolean operations)**.** *Let K and L be languages over Σ recognized by an m-state and n-state NFA, respectively, where $m, n \geq 2$. Then*

(a) $\mathrm{sc}(K \cup L) \leq 2^{m+n}$,

(b) $\mathrm{sc}(K \oplus L) \leq 2^{m+n}$,

(c) $\mathrm{sc}(K \cap L) \leq 2^{m+n} - 2^m - 2^n + 2$,

(d) $\mathrm{sc}(K \setminus L) \leq 2^{m+n} - 2^n + 1$.

All these bounds are tight if $|\Sigma| \geq 3$.

Proof. Let A be an m-state NFA recognizing K and B be an n-state NFA recognizing L. Consider the corresponding subset automata $\mathcal{D}(A)$ and $\mathcal{D}(B)$ with 2^m and 2^n states, respectively. Both of them have at least one rejecting sink state, namely, the empty set. Then all upper bounds follow from Lemma 5.8.

For tightness, let K and L be the languages recognized by NFAs A and B from Fig. 5.6. Notice that transitions on b in A are the same in both automata, performing a loop and a return to the initial state. The roles of the transitions on a and c are mutually exchanged. It follows from Lemma 5.1 that for every $S \subseteq \{0, 1, \ldots, m-1\}$, there is a word $u_S \in \{a, b\}^*$ such that $0 \cdot_A u_S = S$, and for every $T \subseteq \{0, 1, \ldots, n-1\}$, there is a word $v_T \in \{b, c\}^*$ such that $0 \cdot_B v_T = T$.

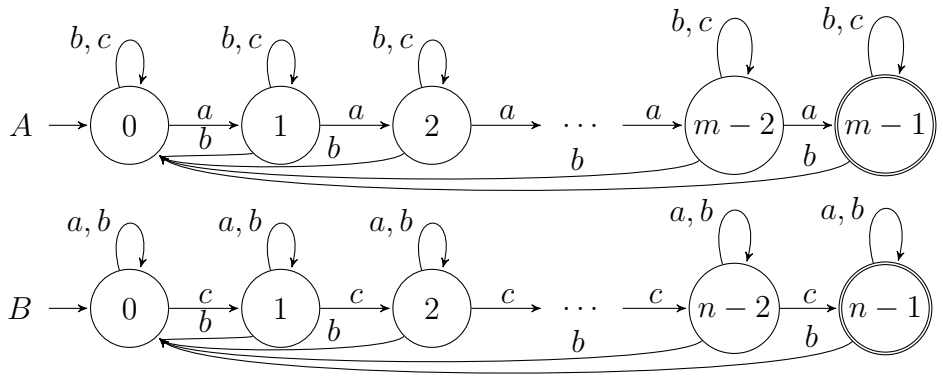

Figure 5.6: Ternary witnesses for Boolean operations.

Let $\circ \in \{\cup, \oplus, \cap, \setminus\}$. Construct the product automaton $M_\circ$ from DFAs $\mathcal{D}(A)$ and $\mathcal{D}(B)$. The initial state of $M_\circ$ is $(\{0\}, \{0\})$. Let $S \subseteq \{0, 1, \ldots, m-1\}$ and $T \subseteq \{0, 1, \ldots, n-1\}$. If $0 \in S$ then the state (S, T) is reachable in $M_\circ$ from the initial state by the word $u_S v_T$. Otherwise, to reach (S, T) with $S = \{s_1, \ldots, s_k\}$ we first reach (S', T), where $S' = \{0, s_2 - s_1, \ldots, s_k - s_1\}$ by the word $u_{S'} v_T$. Then we read a^{s_1} to get (S, T). Hence each state of $M_\circ$ is reachable.

To prove distinguishability first consider union. Then (S, T) is final in $M_\cup$ if $m-1 \in S$ or $n-1 \in T$. Let (S, T) and (S', T') be two distinct states of $M_\cup$. Then $S \neq S'$ or $T \neq T'$. In the first case, let without loss of generality $s \in S \setminus S'$. Consider the word $a^{m-1-s}c^n$ and notice that

$$(S, T) \xrightarrow{a^{m-1-s}c^n} (\{m-1\} \cup S_1, \emptyset) \text{ for some } S_1 \subseteq \{0, 1, \ldots, m-1\};$$
$$(S', T') \xrightarrow{a^{m-1-s}c^n} (S'_1, \emptyset) \text{ where } m-1 \notin S'_1.$$

It follows that $a^{m-1-s}c^n$ is accepted by $M_\cup$ from (S, T) and rejected from (S', T'). The case of $T \neq T'$ is symmetric. We can prove distinguishability for symmetric difference in the exact same manner.

Now consider intersection. A state (S, T) is final in $M_\cap$ iff $m-1 \in S$ and $n-1 \in T$. All states $(\emptyset, T)$ with $T \subseteq \{0, 1, \ldots, n-1\}$ and $(S, \emptyset)$ with $S \subseteq \{0, 1, \ldots, m-1\}$ are dead in $M_\cap$. If $s \in S$ and $t \in T$, then the word $a^{m-1-s}c^{n-1-t}$ is accepted by $M_\cap$ from (S, T), so (S, T) is not dead. Let S, T, S', T' be non-empty such that $(S, T) \neq (S', T')$. Then $S \neq S'$ or $T \neq T'$. In the first case, let $s \in S \setminus S'$ and $t \in T$. Consider the

word $a^{m-1-s}c^{n-1-t}$. Notice that

$$(S,T) \xrightarrow{a^{m-1-s}c^{n-1-t}} (\{m-1\} \cup S_1, \{n-1\} \cup T_1) \text{ for some } S_1, T_1;$$
$$(S',T') \xrightarrow{a^{m-1-s}c^{n-1-t}} (S_1', T_1') \text{ for some } S_1', T_1' \text{ with } m-1 \notin S_1'.$$

Thus $a^{m-1-s}c^{n-1-t}$ is accepted by $M_\cap$ from (S,T) and rejected from (S',T'). The case of $T \neq T'$ is symmetric.

Finally, consider the difference $K \setminus L$. A state (S,T) is final in $M_\setminus$ iff $m-1 \in S$ and $n-1 \notin T$. All states $(\emptyset, T)$ with $T \subseteq \{0,1,\ldots,n-1\}$ are dead in $M_\setminus$. Let $S \neq \emptyset$. If $s \in S$, then the word $a^{m-1-s}c^n$ is accepted from (S,T), so (S,T) is not dead. Let $S \neq \emptyset$, $S' \neq \emptyset$, $(S,T) \neq (S',T')$. If $s \in S \setminus S'$, then

$$(S,T) \xrightarrow{a^{m-1-s}c^{n}} (\{m-1\} \cup S_1, \emptyset) \text{ for some } S_1;$$
$$(S',T') \xrightarrow{a^{m-1-s}c^{n}} (S_1', \emptyset) \text{ for some } S_1' \text{ with } m-1 \notin S_1'.$$

Thus the word $a^{m-1-s}c^n$ is accepted by $M_\setminus$ from (S,T) and rejected from (S',T'). If $S = S'$, $s \in S$, and $t \in T \setminus T'$, then take the word $a^{m-1-s}c^{n-1-t}$:

$$(S,T) \xrightarrow{a^{m-1-s}c^{n-1-t}} (\{m-1\} \cup S_1, \{n-1\} \cup T_1) \text{ for some } S_1, T_1;$$
$$(S',T') \xrightarrow{a^{m-1-s}c^{n-1-t}} (\{m-1\} \cup S_1, T_1') \text{ where } n-1 \notin T_1'.$$

Thus $a^{m-1-s}c^{n-1-t}$ is rejected by $M_\setminus$ from (S,T) and accepted from (S',T'). This concludes the proof. □

Now we show that the upper bound 2^{m+n} is asymptotically tight in the binary case.

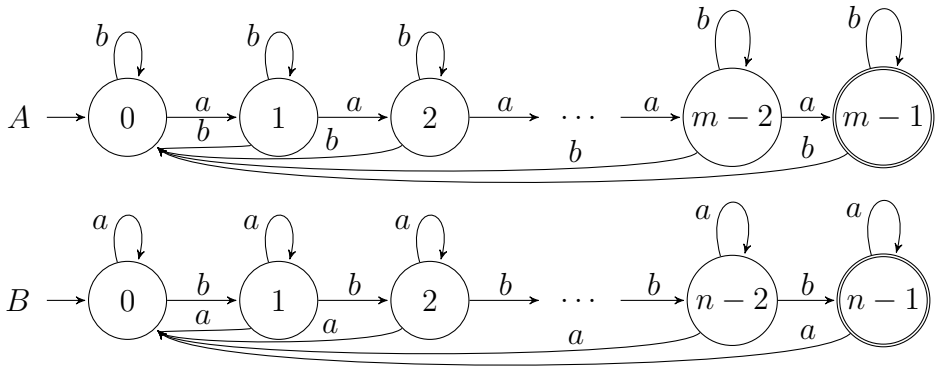

Figure 5.7: Binary NFAs A and B with $\mathrm{sc}(L(A) \cup L(B)) = 2^{m+n-1} + m + n$.

Theorem 5.10. *Let $m, n \geq 2$ and languages K and L be defined by binary NFAs A and B with m and n states from Fig. 5.7. Then*

- $\mathrm{sc}(K \cup L) = \mathrm{sc}(K \oplus L) = 2^{m+n-1} + m + n \geq \frac{1}{2}2^{m+n}$,
- $\mathrm{sc}(K \cap L) = 2^{m+n-1} + m + n - 2^m - 2^n + 2 \geq \frac{1}{4}2^{m+n}$,
- $\mathrm{sc}(K \setminus L) = 2^{m+n-1} + m + n - 2^n + 1 \geq \frac{1}{4}2^{m+n}$.

Proof. Let $\circ \in \{\cup, \oplus, \cap, \setminus\}$. Construct the product automaton $M_\circ$ for $K \circ L$ using the subset automata $\mathcal{D}(A)$ and $\mathcal{D}(B)$. The initial state of $M_\circ$ is $(\{0\}, \{0\})$ and let $S \subseteq \{0, 1, \ldots, m-1\}$ and $T \subseteq \{0, 1, \ldots, n-1\}$. We denote $S \oplus 1 = \{i + 1 \mid i \in S \setminus \{m-1\}\}$ and if $s = \min S$ we denote $S \ominus s = \{i - s \mid i \in S\}$. The sets $T \oplus 1$ and $T \ominus \min T$ are defined analogously. Notice that if S and T are non-empty then by reading a from any (S, T) we reach state $(S \oplus 1, T \cup \{0\})$ and by reading b we reach state $(S \cup \{0\}, T \oplus 1)$ which results in an observation that exactly one of S or T can contain the initial state of the corresponding automaton, except for the initial state $(\{0\}, \{0\})$. This means that the set of possible reachable states $\mathcal{R}$ consists of pairs (S, T) of three types:

(1) $(S, \emptyset)$, $(\emptyset, T)$;

(2) $(\{0\}, \{j\})$, $(\{i\}, \{0\})$, where $i \in Q_A$ and $j \in Q_B$;

(3) (S, T), where either $0_A \in S$ and $0_B \notin T, T \neq \emptyset$, or $0_B \in T$ and $0_A \notin S, S \neq \emptyset$.

So $\mathcal{R}$ contains $(2^{m+n-1} - 2^m - 2^n + 2) + (2^m + 2^n - 1) + (m + n - 1) = 2^{m+n-1} + m + n$ states. Let us show that all pairs in $\mathcal{R}$ are reachable. By Lemma 5.1, there exists a word $u_S \in \{a, b\}^*$ such that $0_A \cdot u_S = S$ and $u_T \in \{a, b\}^*$ such that $0_B \cdot u_T = T$

(1) Let $S = \emptyset$ or $T = \emptyset$. Then $(\{0\}, \{0\}) \xrightarrow{a^m} (\{\emptyset\}, \{0\}) \xrightarrow{u_T} (\emptyset, T)$, and the proof for $T = \emptyset$ is symmetric.

(2) Let $S = \{0\}$ and $T = \{j\}$ or $S = \{i\}$ and $T = \{0\}$. Then $(\{0\}, \{0\}) \xrightarrow{b^j} (\{0\}, \{j\})$, and $(\{0\}, \{0\}) \xrightarrow{a^i} (\{i\}, \{0\})$.

(3) Let $0 \in S$ and $T = \{j\}$ where $j \neq 0$ or $0 \in T$ and $S = \{i\}$ where $i \neq 0$. We prove this case by using induction on the size of S. The basis, $|S| = 1$, was shown in (2). Let $|S| \geq 2$. Denote s as minimal element of $S \setminus \{0\}$; $s = \min(S \setminus \{0\})$ and $S' = (S \setminus \{0\})$. Then $(S', \{n-1\})$ is reachable by the induction hypothesis, and we get $(S', \{n-1\}) \xrightarrow{a^s} (S \setminus \{0\}, \{0, n-1\}) \xrightarrow{b^j} (S, \{j\})$. Symmetrically by swapping a and b we prove the other case.

(4) Let $0 \in S$, $0 \notin T, T \neq \emptyset$; or $0 \in T$, $0 \notin S, S \neq \emptyset$. We prove the first sub-case by induction on $|S|$. The basis, $|S| = 1$, is proved in cases (2)–(3). Assume that the claim holds if $|S| = k$. Let $|S| = k+1$. Take $(S \setminus \{0\}, T \ominus \min T)$, with $|S \setminus \{0\}| = k$. Then by $(S \setminus \{0\}, T \ominus \min T) \xrightarrow{b^{\min T}} (S, T)$ we obtain (S, T). The second sub-case is proved in a symmetric way.

The proof for distinguishability for $\circ \in \{\cup, \cap, \oplus\}$ is the same as in the proof of Theorem 5.9 but with one condition, we replace the letter c with b. For $\circ = \setminus$ we continue likewise, we replace the letter c with b, but we deal with another situation when $S = S'$. If T and T' differ in a state $t > 0$ then we can continue as in the proof of Theorem 5.9. Otherwise $T = \{0\} \cup T'$, which means $0 \notin T'$ and therefore $0 \in S'$. Since $S' = S$ we get $0 \in S$. The fact that $0 \in S$ and $0 \in T$ follows that $T = \{0\}$ and $S = S' = \{0\}$. Thus $T' = \emptyset$. We then distinguish such pairs in the following way:

$$(\{0\}, \{0\}) \xrightarrow{a^{m-1}c^{n-1}} (\{m-1\}, \{n-1\}) \text{ and reject;}$$
$$(\{0\}, \emptyset) \xrightarrow{a^{m-1}c^{n-1}} (\{m-1\}, \emptyset) \text{ and accept.}$$

This concludes the proof. □

We showed that the upper bound 2^{m+n} for Boolean operations is asymptotically tight in the binary case. The next theorem deals with the unary case.

Theorem 5.11. *Let $\circ \in \{\cup, \cap, \oplus, \setminus\}$ and K and L be unary languages recognized by NFAs with m and n states respectively. Then* $\mathrm{sc}(K \circ L) = O(F(m+n))$. *For infinitely many pairs (m, n) there exist unary languages K and L recognized by m-state and n-state NFA, respectively, such that* $\mathrm{sc}(K \circ L) = \Omega(F(m+n))$.

Proof. Let A and B be unary m-state and n-state NFAs for K and L, respectively. We can convert A and B into automata A' and B' in Chrobak normal form [9], Fig. 5.8. The NFA A' consists of a tail of length at most m^2 that ends in exactly one nondeterministic transition going to k disjoint cycles of length $x_1, \ldots, x_k$, where $x_1 + x_2 + \cdots + x_k \leq m$. Similarly B' has a tail of length at most n^2 and disjoint cycles of length $y_1, y_2, \ldots, y_l$, where $y_1 + y_2 + \cdots + y_l \leq n$. Then $K \circ L$ is accepted by a DFA with a tail of length at most $\max(m^2, n^2)$ and a cycle of size $\mathrm{lcm}(x_1, \ldots, x_k, y_1, \ldots, y_l) \leq F(m+n)$. In total this DFA has $O(F(m+n))$ states.

To get a lower bound let us take the partition $m + n - 2 \geq x_1 + x_2 + \cdots + x_k$ such that $F(m+n-2) = \mathrm{lcm}(x_1, x_2, \ldots, x_k)$. We may assume that x_i and x_j are pairwise co-prime and $2 \leq x_1 < x_2 < \cdots < x_k$. Let $m' = 1 + x_1$ and $n' = 1 + x_2 + \cdots + x_k$. Let

A and B be NFAs from Fig. 5.9. Automaton A has m' states and automaton B has n' states. Let $K = L(A)$ and $L = L(B)$. Then $K \circ L$ is accepted by DFA D with a tail of length 1 and a loop of length $x_1x_2 \cdots x_k$; see Fig. 5.10 where $x_1 = 2, x_2 = 3, x_3 = 5$ and $\circ = \cup$. Each state from these $x_1x_2 \cdots x_k$ can be labeled by a tuple $(v_1, \ldots, v_k)$, where v_i is the remainder after dividing by x_i and each such tuple is unique. The set of final states differs for $\circ \in \{\cap, \cup, \oplus, \setminus\}$. For $\cap$ the set of final states contains only the state $(0, \ldots, 0)$. For $\cup$ the set of final states consists of the initial state s and the states $v = (v_1, \ldots, v_k)$ where $v_i = 0$ for an $i = 1, \ldots, k$. For $\oplus$ take $F_D = \{s\} \cup \{(v_1, \ldots, v_k) \mid (v_1 = 0 \wedge v_i \neq 0 \text{ for each } i \geq 2) \vee (v_1 \neq 0 \wedge v_i = 0 \text{ for an } i \geq 2)\}$. Finally for $\setminus$ we take $F_D = \{s\} \cup \{(v_1, \ldots, v_k) \mid (v_1 = 0 \wedge v_i \neq 0 \text{ for each } i \geq 2)\}$.

To prove distinguishability, our general approach is to take two distinct states $u = (u_1, \ldots, u_k)$ and $v = (v_1, \ldots, v_k)$ that we would like to distinguish by a word a^w. In our notation with modx_i the word a^w is in fact $a^{(w_1, w_2, \ldots, w_k)}$. Then reading a^w from state u looks like this:

$$(u_1, \ldots, u_k) \xrightarrow{a^w} ((u_1 + w_1) \bmod x_1, \ldots, (u_k + w_k) \bmod x_k),$$

and for each operation we define a specific $w = (w_1, \ldots, w_k)$. Its existence follows from Chinese remainder theorem since $x_1, \ldots, x_k$ are pairwise co-prime and greater than one.

$\cap$: Since u and v are distinct, there is an i such that $u_i \neq v_i$. Take $w = (w_1, \ldots, w_k)$ where $w_i = x_i - u_i$ for each $i = 1, \ldots, k$. Then

$$(u_1, \ldots, u_k) \xrightarrow{a^{(w_1, \ldots, w_k)}} (0, \ldots, 0)$$

while the ith component of $v + w$ is different from zero. Hence w is accepted from u and rejected from v.

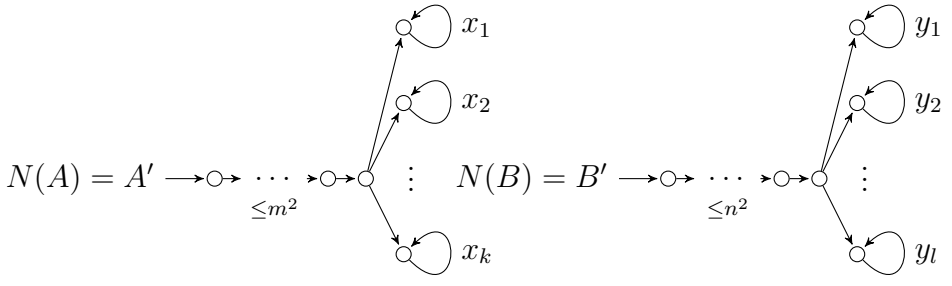

Figure 5.8: The Chrobak normal form of NFAs A and B.

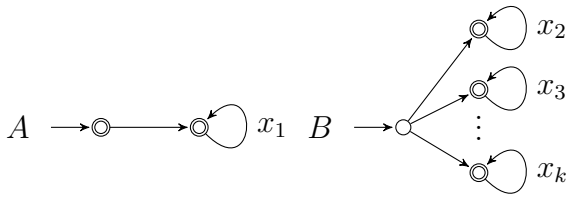

Figure 5.9: NFAs A and B with $m' = 1 + x_1$ and $n' = 1 + x_2 + \cdots + x_k$ states.

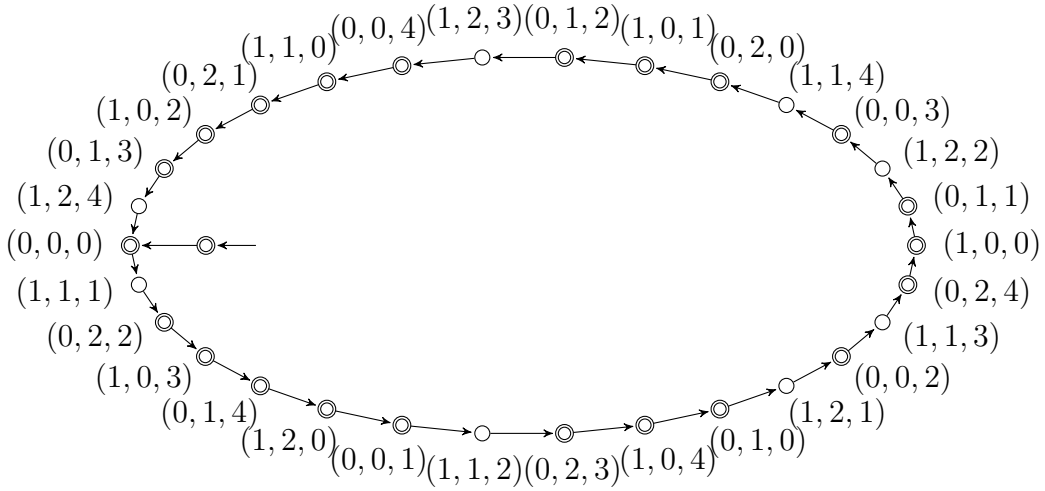

Figure 5.10: DFA D for $K \cup L$ with $x_1 = 2, x_2 = 3, x_3 = 5$.

$\cup$: Let $u_i \neq v_i$. Take $w = (w_1, \ldots, w_k)$ where $w_i = x_i - u_i$, and for each $j \neq i, 1 \leq j \leq k$ take $w_j = 1$ iff $v_j = 0$ and $w_j = 0$ iff $v_j \neq 0$. Then the ith component of $u + w$ is zero, while the each component of $v + w$ is different from zero. Hence w is accepted from u and rejected from v.

$\setminus$: We consider two cases. Firstly, when $u_1 \neq v_1$. We take $w = (w_1, \ldots, w_k)$ where $w_1 = x_1 - u_1$, and for $2 \leq j \leq k$ take $w_j = 1$ iff $u_j = 0$ and $w_j = 0$ iff $u_j \neq 0$. Then the first component of $u + w$ is zero and any other is non-zero, while each component of $v + w$ is non-zero. Hence w is accepted from u and rejected from v. Now let $u_1 = v_1$. Then there is $i \geq 2$ such that $u_i \neq v_i$. We take $w = (w_1, \ldots, w_k)$ where $w_1 = x_1 - u_1$, and for $2 \leq j \leq k$ take $w_j = 1$ iff $u_j = 0$ and $w_j = 0$ iff $u_j \neq 0$. Then the first and ith component of $u + w$ are zeroes, while the first component of $v + w$ is zero and any other is non-zero. Hence w is accepted from v and rejected from u.

$\oplus$: We consider two cases. Firstly, when $u_1 \neq v_1$, we set $w_1 = x_1 - u_1$ and for $2 \leq i \leq k$,

$$w_i = \begin{cases} 0, & \text{if } u_i \neq 0 \text{ and } v_i \neq 0; \\ 1, & \text{if } (u_i = 0 \text{ and } 0 \leq v_i \leq x_i - 2) \text{ or } (v_i = 0 \text{ and } 0 \leq u_i \leq x_i - 2); \\ 2, & \text{if } (u_i = 0 \text{ and } v_i = x_i - 1) \text{ or } (v_i = 0 \text{ and } u_i = x_i - 1). \end{cases}$$

Then w is accepted from u since the first component of $u + w$ is zero and any other is non-zero, but w is rejected from v since every component of $v + w$ is non-zero. Secondly let $u_1 = v_1$. Then $u_i \neq v_i$ for some $2 \leq i \leq k$. Set $w_1 = x_1 - u_1$ and $w_i = x_i - u_i$ and for $j \notin \{1, i\}$ take $w_j = 1$ iff $v_j = 0$ and $w_j = 0$ iff $v_j \neq 0$. Then w is accepted from v since the first component of $v + w$ is zero and any other is non-zero, but w is rejected from u the first and ith component of $u + w$ are zeroes.

It follows that $\mathrm{sc}(K \circ L) \geq x_1 x_2 \cdots x_k = \mathrm{lcm}(x_1, x_2, \ldots, x_k) = F(m + n - 2) \geq F(m' + n' - 2) = \Omega(F(m' + n'))$. This concludes our proof. □

5.3 Reversal, Left and Right Quotient

Let us continue with the reversal operation. Note that it is enough to take the reversal of any DFA with one final state meeting the upper bound 2^n on the state complexity of reversal. Such a binary DFA was described by Šebej [30, 52]. Here we describe a different witness with significantly simpler proof. Notice that the binary alphabet used for reversal is optimal. Reverse of any unary language is the language itself, but in our case we have a unary language recognized by n-state NFA, which can be simulated ba a DFA with $2^{O(\sqrt{n \ln n})}$ states [9].

Theorem 5.12 (Reversal)**.** *Let L be a language over Σ recognized by an n-state* NFA*, where $n \geq 2$. Then* $\mathrm{sc}(L^R) \leq 2^n$*, and the bound is tight if $|\Sigma| \geq 2$.*

Proof. Let L be accepted by an n-state NFA $A = (Q, \Sigma, \cdot, s, F)$. By reversing all the transitions in A and taking F as the set of starting states and $\{s\}$ as set of final states we obtain an n-state MNFA that accepts L^R. It follows that L^R is accepted by a DFA with at most 2^n states.

To prove tightness, consider the binary language L recognized by the n-state NFA $N = (\{0, 1, \ldots, n-1\}, \{a, b\}, \cdot, 0, \{0, 1, \ldots, n-1\})$ shown in Fig. 5.11 where $i \cdot a = \{i+1 \bmod n\}$, $i \cdot b = \{i\}$ if $i \geq 1$. By reversing NFA N we get an MNFA N^R that recognizes L^R.

The set of initial state of N^R is $\{0, 1, \ldots, n-1\}$ and its unique final state is 0. Notice that each subset of the state set of N^R can be shifted cyclically by one by reading a, and

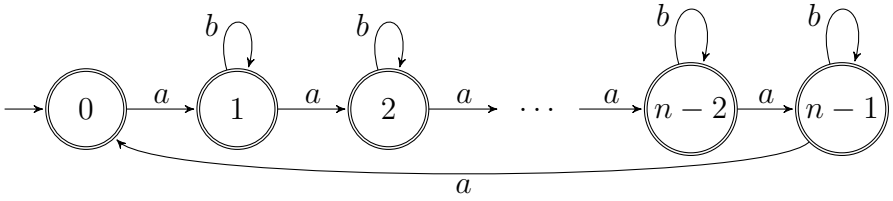

Figure 5.11: A binary witness NFA for reversal meeting the upper bound 2^n.

the state 0 can be eliminated from every set containing 0 by reading b. It follows that every subset of $\{0, 1, \ldots, n-1\}$ can be reached from the initial subset $\{0, 1, \ldots, n-1\}$ in the subset automaton $\mathcal{D}(N^R)$. Next, every set $\{i\}$ is co-reachable in N^R via a word in a^* and using Lemma 5.3 we get that every two distinct states of the $\mathcal{D}(N^R)$ are distinguishable. □

We investigate NFA-to-DFA trade-off for left and right quotients in the next two theorems. In the first one we show tight upper bounds with binary witnesses for both operations. The optimality of the binary alphabet is shown by the second one, where the asymptotically tight upper bound for unary case is provided.

Theorem 5.13 (Left and Right Quotient)**.** *Let K and L be languages over an alphabet Σ recognized by an m-state and n-state* NFA*, respectively, where $m, n \geq 2$. Then* $\mathrm{sc}(L^{-1}K)$, $\mathrm{sc}(KL^{-1}) \leq 2^m$*, and the bounds are tight if $|\Sigma| \geq 2$.*

Proof. Let $A = (Q_A, \Sigma, s_A, \cdot_A, F_A)$ be an m-state NFA recognizing K. The language $L^{-1}K$ is recognized by the m-state MNFA N obtained from A by changing the set of initial states to $\{s_A \cdot_A w \mid w \in L\}$. The language KL^{-1} is recognized by the m-state NFA N obtained from A by changing the set of final states to $\{q \in Q_A \mid q \cdot_A w \in F_A \text{ for some } w \in L\}$. Hence $\mathrm{sc}(L^{-1}K), \mathrm{sc}(KL^{-1}) \leq 2^m$.

For tightness, notice that $\{\varepsilon\}^{-1}K = K\{\varepsilon\}^{-1} = K$. Therefore, the upper bound 2^m is met in both cases by $L = \{\varepsilon\}$ and K equal to the binary m-state witness NFA for determinization given by Lemma 5.1. For distinguishability, notice that each singleton set is co-reachable in this NFA. □

Theorem 5.14. *Let K and L be recognized by m-state and n-state unary NFAs respectively. Then* $\mathrm{sc}(KL^{-1}) = \mathrm{sc}(L^{-1}K) = \Theta(F(m))$.

Proof. Let K and L be accepted by NFAs A and B respectively. If K and L are unary then $KL^{-1} = L^{-1}K$. The right quotient KL^{-1} is recognized by an NFA obtained from A by making appropriate states final. This gives upper bound $O(F(m))$ by determinization of

resulting unary NFA. For tightness let K be a unary witness for determinization and $L = \{\varepsilon\}$. Then $KL^{-1} = K$, and the lower bound $\Omega(F(m))$ follows from [9, Theorem 4.5]. □

5.4 Concatenation

We conclude this chapter with the concatenation operation. Its the state complexity is $m2^n - 2^{n-1}$ [38] and nondeterministic complexity is $m + n$ [14]. The next theorem shows upper bound for NFA-to-DFA trade-off for concatenation, $\frac{3}{4}2^{m+n}$, which is tight in the ternary case. For binary case we show that upper bound 2^{m+n} is asymptotically tight.

Theorem 5.15 (Concatenation)**.** *Let K and L be languages over Σ recognized by an m-state and n-state NFA, respectively, where $m, n \geq 2$. Then $\mathrm{sc}(KL) \leq \frac{3}{4}2^{m+n}$ and this bound is tight if $|\Sigma| \geq 3$.*

Proof. Let $A = (Q_A, \Sigma, \cdot_A, s_A, F_A)$ and $B = (Q_B, \Sigma, \cdot_B, s_B, F_B)$ be NFAs recognizing K and L, respectively, with $|Q_A| = m$, $|Q_B| = n$ Construct an MNFA N for KL from NFAs A and B as follows. For each transition (p, a, q) in NFA A with $q \in F_A$, add the transition (p, a, s_B). The set of initial states of N is $\{s_A\}$ if $s_A \notin F_A$, or $\{s_A, s_B\}$ if $s_A \in F_A$. The set of final states of N is F_B. The following condition holds in the subset automaton $\mathcal{D}(N)$: each reachable subset containing a state from F_A must contain the state s_B. This means that at least 2^{m+n-2} subsets are unreachable in $\mathcal{D}(N)$, and the upper bound follows.

For tightness, consider the languages K and L recognized by DFAs A and B from Fig. 5.12. Construct an NFA N for KL from A and B by adding the transitions $(q_0, c, 0)$, $(q_{m-1}, a, 0)$, and $(q_i, b, 0)$ for $i = 1, 2, \ldots, m-1$. The initial states of N are q_0 and 0. The

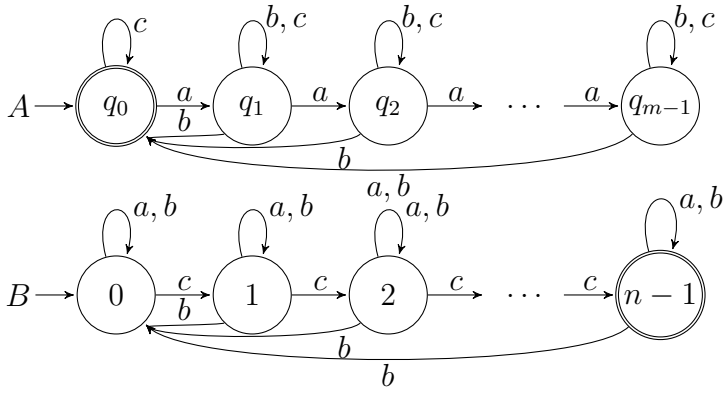

Figure 5.12: Ternary witnesses for concatenation meeting the upper bound $\frac{3}{4}2^n$.

set of final states is $\{n-1\}$. We first show that the subset automaton $\mathcal{D}(N)$ has $\frac{3}{4}2^{m+n}$ reachable states. Each state of $\mathcal{D}(N)$ consists of a subset S of $\{q_0, q_1, \ldots, q_{m-1}\}$ and a subset T of $\{0, 1, \ldots, n-1\}$; we denote such state by (S, T). Moreover having $q_0 \in S$ implies $0 \in T$, so in total we have $\frac{3}{4}2^{m+n}$ such states. Let us show the reachability of each such state (S, T). We remind Lemma 5.1 and Lemma 5.2, where we show that for every S there is a word $w_S \in \{a, b\}^*$ such that $q_0 \cdot_A w_S = S$, and for every T there is a word $w_T \in \{b, c\}^*$ such that $0 \cdot_B w_T = T$. Now consider several cases:

(1) Let $S = \emptyset$. The initial state $(\{q_0\}, \{0\})$ is sent to $(\emptyset, \{0\})$ by b, and then to $(\emptyset, T)$ by w_T.

(2) Let $S \neq \emptyset$ and $0 \in T$.

(2a) Let $S = \{q_0\}$ and $0 \in T$. By induction on $|T|$ we show that $(\{q_0\}, T)$ with $0 \in T$ is reachable. The base case, $T = \{0\}$, holds true since $(\{q_0\}, \{0\})$ is the initial state of $\mathcal{D}(N)$. Assume that the claim holds true for each set of size k and let $|T| = k+1$. Let $T = \{0, i_1, \ldots, i_k\}$ where $0 < i_1 < \cdots < i_k \leq n-1$. Set $T' = \{0, i_2 - i_1, \ldots, i_k - i_1\}$. Then $|T'| = k$, so the pair $(\{q_0\}, T')$ is reachable by induction. The pair $(\{q_0\}, T)$ is reachable from $(\{q_0\}, T')$ by $a^{m-1}c^{i_1}a$.

(2b) Let $q_0 \in S$, where $|S| \geq 2$ and $0 \in T$. The pair $(\{q_0\}, T)$ is reachable as shown in case (2a) and is sent to (S, T) by w_S since $0 \in T$.

(2c) Let $q_0 \notin S$, where $0 \in T$. Let $S = \{q_{i_1}, q_{i_2}, \ldots, q_{i_k}\}$ where $1 \leq i_1 < \cdots < i_k \leq m-1$. Set $S' = \{q_0, q_{i_2 - i_1}, \ldots, q_{i_k - i_1}\}$. Then (S', T) is reachable as shown in (2a) or (2b) and it is sent to (S, T) by a^{i_1}.

(3) Let $S \neq \emptyset$ and $0 \notin T$. This means that $q_0 \notin S$. The pair $(S, \emptyset)$ is reached from $(\{q_0\}, \{0\})$ by $w_S c^n$. If $T = \{i_1, \ldots, i_k\}$ where $1 \leq i_1 < \cdots < i_k \leq n-1$ we set $T' = \{0, i_2 - i_1, \ldots, i_k - i_1\}$. The pair (S, T') is reachable as shown in case (2) since $0 \in T'$ and it is sent to (S, T) by c^{i_1} since $q_0 \notin S$.

To prove distinguishability notice that in N each singleton set $\{j\}$ is co-reachable via a word in c^*, the set $\{q_0\}$ is co-reachable by c^n. For $1 \leq i \leq n-1$ the set $\{q_i\}$ is co-reachable by $c^n a^{n-i}$ since transitions on a in automaton A form the cycle $(q_0, q_1, \ldots, q_{m-1})$. By Lemma 5.3 all states of subset automaton $\mathcal{D}(N)$ are pairwise distinguishable. □

Now we show that the upper bound 2^{m+n} is asymptotically tight in the binary case.

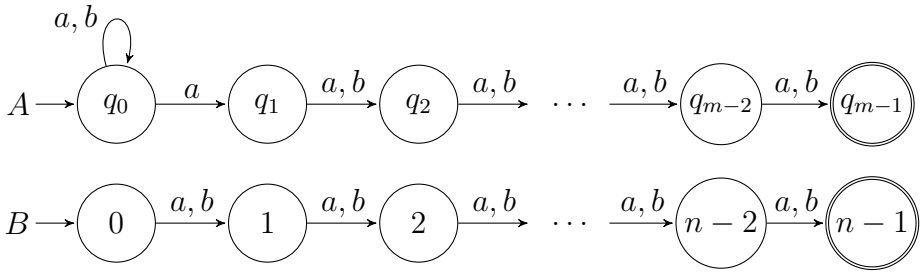

Figure 5.13: Binary NFAs A and B with $\mathrm{sc}(L(A)L(B)) \geq \frac{1}{4}2^{m+n}$.

Theorem 5.16. *Let $m, n \geq 2$ and languages K and L be defined by binary NFAs A and B with m and n states from Fig. 5.13. Then* $\mathrm{sc}(KL) \geq \frac{1}{4}2^{m+n}$.

Proof. Construct an NFA for KL from NFAs A and B by adding the transitions $(q_{m-2}, a, 0)$ and $(q_{m-2}, b, 0)$. This NFA recognizes the binary language $(a+b)^*a(a+b)^{m+n-3}$. It is well known that every DFA for this language has at least 2^{m+n-2} states. □

Our next result provides a lower and upper bound on the NFA-to-DFA trade-off for concatenation in the unary case.

Theorem 5.17. *Let K and L be unary languages recognized by NFAs with m and n states respectively. Then* $\mathrm{sc}(KL) = O(F(m+n))$. *There exists languages K and L such that* $\mathrm{sc}(KL) = \Omega(\max\{F(m), F(n)\})$.

Proof. The upper bound $O(F(m+n))$ is given by the determinization of an $(m+n)$-state unary NFA. The lower bound $\Omega(\max\{F(m), F(n)\})$ is provided by the pairs of languages $(\{\varepsilon\}, L)$ and $(K, \{\varepsilon\})$. □

The previous shows that the resulting complexity is exponential either in m or in n. This contrasts the situation for the star operation, where the resulting complexity was polynomial in n.

5.5 Conclusions

We investigated the NFA-to-DFA trade-off for several regular operations. Our results are summarized in Table 5.1. The table also displays the size of alphabet used to describe our witnesses. Whenever we used a binary alphabet, it was is always optimal in the sense that the corresponding upper bounds cannot be met by any unary languages.

We conjecture that the upper bounds on the state complexity of Boolean operations and concatenation cannot be met by any pair of binary languages. For unary concatenation we provided an upper bound $O(F(m+n))$ and a lower bound $\Omega(\max\{F(m), F(n)\})$. We leave the problem of finding at least asymptotically tight upper bound for future research.

	NFA-to-DFA	$\lvert\Sigma\rvert$	$\lvert\Sigma\rvert = 2$	$\lvert\Sigma\rvert = 1$
star	2^n	2	2^n	$(n-1)^2+2$
complementation	2^n	2	2^n	$\Theta(F(n))$
union	2^{m+n}	3	$\Theta(2^{m+n})$	$\Theta(F(m+n))$
symmetric difference	2^{m+n}	3	$\Theta(2^{m+n})$	$\Theta(F(m+n))$
intersection	$2^{m+n}-2^m-2^n+2$	3	$\Theta(2^{m+n})$	$\Theta(F(m+n))$
difference	$2^{m+n}-2^n+1$	3	$\Theta(2^{m+n})$	$\Theta(F(m+n))$
reversal	2^n	2	2^n	$\Theta(F(n))$
left quotient	2^m	2	2^m	$\Theta(F(m))$
right quotient	2^m	2	2^m	$\Theta(F(m))$
concatenation	$\frac{3}{4}2^{m+n}$	3	$\Theta(2^{m+n})$	$\Omega(\max\{F(m), F(n)\})$ $\leq \cdot \leq O(F(m+n))$

Table 5.1: The results on NFA-to-DFA trade-off for regular operations;
$F(n) = \max\{\mathrm{lcm}(x_1, \ldots, x_k) \mid x_1 + \cdots + x_k \leq n\} \approx 2^{\sqrt{n \ln n}}$.

Chapter 6

Summary and Future Works

This thesis investigated operational complexity on languages represented by different models of finite automata. We started with the square operation on deterministic finite automata and we showed how its state complexity depends on the number of final states in a DFA for a given language. The case of just one non-final state, in which the resulting complexity is not given by the same expression as for any other case, appeared to be very interesting. We did some computations and thanks to them we were able to get an upper bound, which depends on whether or not the initial state is final, as well as to describe the corresponding witness languages. We used our binary witnesses with half of their states final to describe witness languages for the square operation on Boolean and alternating automata. Moreover, we proved that our witnesses can be used also for the concatenation operation: we simply take two automata with the same structure but different number of states. This provided an alternative solution of an open problem stated by Fellah et al. [13].

Then we determined the precise complexity for Boolean operations, star, reversal and quotients on Boolean and alternating finite automata. We used unary alphabet to describe witnesses for Boolean operations, and a binary alphabet for the others. We also proved that the binary alphabet is always optimal by proving that the complexity of the corresponding operation on unary languages is smaller than that in the case of a general alphabet.

Finally, we considered operational complexity providing that the inputs of an operation are given as nondeterministic finite automata, while the resulting language has to be represented by a deterministic finite automaton. We obtained tight upper bounds for Boolean operations, concatenation, star, reversal and quotients. To prove tightness, we used a binary alphabet for complementation, star, reversal and quotients, and a ternary alphabet

for union, intersection, difference, symmetric difference and concatenation. Whenever we used a binary alphabet, it was always optimal, and we strongly conjecture that the ternary alphabet is optimal as well. Our computations support this conjecture. On the other hand, we proved that the corresponding complexities in the binary case are, up to a multiplicative constant, the same as those for the general case. In the unary case, we obtained the precise complexity for star, and asymptotically tight upper bounds for Boolean operations, reversal and quotients.

Some problems remains open, and we leave them for future research. We list a couple of them in what follows.

The magic number problem. When investigating the square operation on DFAs, we only considered the worst-case complexity. Instead of this, the whole range of possible complexities is sometimes studied in the literature. The problem is called the magic number problem, and it was first stated by Iwama et al. [18, 19]. For the square operation it reads as follows.

Open problem 1. Given a minimal DFA A with n states, how many states may the minimal DFA for the square of $L(A)$ have?

Formally, we ask how does the set $\{sc(L^2) \mid sc(L) = n\}$ look like? Is it a contiguous range of complexities from one up to the known upper bound, or are there any gaps, called magic numbers, in this range? Our computations show that no magic numbers exist up to $n = 9$ already in the binary case.

Operations on unary languages. When proving the optimality of a binary alphabet used to describe lower bound examples for the AFA complexity of some operations we obtained only upper bounds on the complexity for corresponding operations in the unary case. They differ from the lower bounds given by the BFA complexity of these operations by one. We do not know whether or not this "plus one" is necessary.

Open problem 2. What is the complexity of concatenation, square, star and quotients on languages represented by unary AFAs?

In the case of a general alphabet we were able to find languages recognized by DFAs with half of their states final that were hard enough for an operation on DFAs, and then we proved that the reversal of these languages are hard for the operation on AFAs. However, in the unary case such languages are not known.

Open problem 3. What is the complexity of concatenation, square, star and quotients on languages represented by unary DFAs with half of their states final?

Notice that the complexity of complementation and reversal in such a case is n, and those of union, intersection, difference and symmetric difference is given by Lemma 4.2. For NFA-to-DFA trade-off in the case of concatenation on unary languages we have an upper bound $O(F(m+n))$ and a lower bound $\Omega(\max\{F(m), F(n)\})$. This is the only case in which we were not able to get at least asymptotically tight upper bound.

Open problem 4: What is the NFA-to-DFA trade-off for concatenation in the unary case?

We conjecture that our lower bound $\Omega(\max\{F(m), F(n)\})$ could be asymptotically tight.

Optimality of a ternary alphabet. All witness languages in this thesis were described over an optimal alphabet, except for those for the NFA-to-DFA trade-offs in the case of Boolean operation and concatenation where we used a ternary alphabet.

Open problem 5. Is the ternary alphabet used to describe witnesses for NFA-to-DFA trade-off in the case of concatenation and Boolean operations optimal?

Our computations show the that corresponding upper bounds cannot be met in the binary case, and we strongly conjecture that the ternary alphabet is optimal here.

Nondeterministic finite automata with existential and universal states. Sometimes, in contrast to [8, 13, 54], alternating automata are defined as nondeterministic automata with existential and universal states. This means that the result of the transition function is always a disjunction if the corresponding state is an existential state and it is a conjunction otherwise. We can show that such an automaton can be simulated by a DFA with at most $M(n) + 2$ states where $M(n)$ is the Dedekind number that counts the number of anti-chains of subsets of a set with n elements. However, we do not know whether or not this upper bound is tight.

Open problem 6. How many states are sufficient and necessary in the worst case for a DFA to simulate an n-state NFA with existential and universal states?

The operational complexity on such a model of alternating automata is of great interest for us as well.

Bibliography

[1] Assent, I., Seibert, S.: An upper bound for transforming self-verifying automata into deterministic ones. RAIRO - Theor. Inf. and Applic. **41**(3), 261–265 (2007), https://doi.org/10.1051/ita:2007017

[2] Berman, P., Lingas, A.: On the complexity of regular languages in terms of finite automata. Technical Report **304**, Polish Academy of Sciences (1977)

[3] Birget, J.: Intersection and union of regular languages and state complexity. Inform. Process. Lett. **43**(4), 185–190 (1992), http://dx.doi.org/10.1016/0020-0190(92)90198-5

[4] Brzozowski, J.A., Leiss, E.L.: On equations for regular languages, finite automata, and sequential networks. Theoret. Comput. Sci. **10**, 19–35 (1980), http://dx.doi.org/10.1016/0304-3975(80)90069-9

[5] Câmpeanu, C., II, K.C., Salomaa, K., Yu, S.: State complexity of basic operations on finite languages. In: Boldt, O., Jürgensen, H. (eds.) WIA 1999, LNCS. Lecture Notes in Computer Science, vol. 2214, pp. 60–70. Springer (1999), https://doi.org/10.1007/3-540-45526-4_6

[6] Câmpeanu, C., Salomaa, K., Yu, S.: Tight lower bound for the state complexity of shuffle of regular languages. J. Autom. Lang. Comb. **7**(3), 303–310 (2002)

[7] Čevorová, K., Jirásková, G., Krajňáková, I.: On the square of regular languages. In: Holzer, M., Kutrib, M. (eds.) CIAA 2014. LNCS, vol. 8587, pp. 136–147. Springer (2014), http://dx.doi.org/10.1007/978-3-319-08846-4_10

[8] Chandra, A.K., Kozen, D., Stockmeyer, L.J.: Alternation. J. ACM **28**(1), 114–133 (1981), http://doi.acm.org/10.1145/322234.322243

[9] Chrobak, M.: Finite automata and unary languages. Theoret. Comput. Sci. **47**(3), 149–158 (1986), https://doi.org/10.1016/0304-3975(86)90142-8

[10] Cui, B., Gao, Y., Kari, L., Yu, S.: State complexity of two combined operations: Catenation-union and catenation-intersection. Int. J. Found. Comput. Sci. **22**(8), 1797–1812 (2011), https://doi.org/10.1142/S0129054111009045

[11] Domaratzki, M.: State complexity of proportional removals. J. Autom. Lang. Comb. **7**(4), 455–468 (2002), https://doi.org/10.25596/jalc-2002-455

[12] Domaratzki, M., Okhotin, A.: State complexity of power. Theoret. Comput. Sci. **410**(24-25), 2377–2392 (2009). https://doi.org/10.1016/j.tcs.2009.02.025

[13] Fellah, A., Jürgensen, H., Yu, S.: Constructions for alternating finite automata. International journal of computer mathematics **35**(1-4), 117–132 (1990), http://dx.doi.org/10.1080/00207169008803893

[14] Holzer, M., Kutrib, M.: Nondeterministic descriptional complexity of regular languages. Internat. J. Found. Comput. Sci. **14**(6), 1087–1102 (2003), http://dx.doi.org/10.1142/S0129054103002199

[15] Hopcroft, J.E., Ullman, J.D.: Introduction to Automata Theory, Languages and Computation. Addison-Wesley (1979)

[16] Hospodár, M., Jirásková, G.: Concatenation on deterministic and alternating automata. In: Bordihn, H., Freund, R., Nagy, B., Vaszil, G. (eds.) NCMA 2016. books@ocg.at, vol. 321, pp. 179–194. Österreichische Computer Gesellschaft (2016)

[17] Hospodár, M., Jirásková, G.: The complexity of concatenation on deterministic and alternating finite automata. RAIRO - Theor. Inf. and Applic. **52**(2-3-4), 153–168 (2018), https://doi.org/10.1051/ita/2018011

[18] Iwama, K., Kambayashi, Y., Takaki, K.: Tight bounds on the number of states of DFAs that are equivalent to n-state NFAs. Theoret. Comput. Sci. **237**(1-2), 485–494 (2000), https://doi.org/10.1016/S0304-3975(00)00029-3

[19] Iwama, K., Matsuura, A., Paterson, M.: A family of NFAs which need $2n - \alpha$ deterministic states. Theoret. Comput. Sci. **301**(1-3), 451–462 (2003), https://doi.org/10.1016/S0304-3975(02)00891-5

[20] Jirásek, J., Jirásková, G., Szabari, A.: State complexity of concatenation and complementation. Internat. J. Found. Comput. Sci. **16**(3), 511–529 (2005), http://dx.doi.org/10.1142/S0129054105003133

[21] Jirásek, J.Š., Jirásková, G., Szabari, A.: Operations on self-verifying finite automata. In: Beklemishev, L.D., Musatov, D.V. (eds.) CSR 2015. LNCS, vol. 9139, pp. 231–261. Springer (2015), https://doi.org/10.1007/978-3-319-20297-6_16

[22] Jirásek, J., Jr., Jirásková, G., Šebej, J.: Operations on unambiguous finite automata. In: Brlek, S., Reutenauer, C. (eds.) DLT 2016. LNCS, vol. 9840, pp. 243–255. Springer (2016), https://doi.org/10.1007/978-3-662-53132-7_20

[23] Jirásek, J., Jr., Jirásková, G., Šebej, J.: Operations on unambiguous finite automata. Internat. J. Found. Comput. Sci. **29**(5), 861–876 (2018), https://doi.org/10.1142/S012905411842008X

[24] Jirásková, G.: State complexity of some operations on binary regular languages. Theoret. Comput. Sci. **330**(2), 287–298 (2005), http://dx.doi.org/10.1016/j.tcs.2004.04.011

[25] Jirásková, G.: Descriptional complexity of operations on alternating and boolean automata. In: Hirsch, E.A., Karhumäki, J., Lepistö, A., Prilutskii, M. (eds.) CSR 2012. LNCS, vol. 7353, pp. 196–204. Springer (2012), http://dx.doi.org/10.1007/978-3-642-30642-6_19

[26] Jirásková, G., Krajnáková, I.: Square on deterministic, alternating, and boolean finite automata. Internat. J. Found. Comput. Sci. **30**(6-7), 1117–1134 (2019), https://doi.org/10.1142/S0129054119400318

[27] Jirásková, G., Okhotin, A.: State complexity of cyclic shift. RAIRO - Theor. Inf. and Applic. **42**(2), 335–360 (2008), https://doi.org/10.1051/ita:2007038

[28] Jirásková, G., Okhotin, A.: On the state complexity of star of union and star of intersection. Fundam. Inform. **109**(2), 161–178 (2011), https://doi.org/10.3233/FI-2011-502

[29] Jirásková, G., Pighizzini, G.: Optimal simulation of self-verifying automata by deterministic automata. Inform. and Comput. **209**(3), 528–535 (2011), http://dx.doi.org/10.1016/j.ic.2010.11.017

[30] Jirásková, G., Šebej, J.: Reversal of binary regular languages. Theoret. Comput. Sci. **449**, 85–92 (2012), https://doi.org/10.1016/j.tcs.2012.05.008

[31] Krajňáková, I.: Operácia štvorec na jazykoch reprezentovaných deterministickými, alternujúcimi a Booleovskými automatmi. Master's thesis, Univerzita Pavla Jozefa Šafárika v Košiciah, Košice (2016), in Slovak

[32] Krajňáková, I., Jirásková, G.: Square on deterministic, alternating, and boolean finite automata. In: Pighizzini, G., Câmpeanu, C. (eds.) DCFS 2017. LNCS, vol. 10316, pp. 214–225. Springer (2017), https://doi.org/10.1007/978-3-319-60252-3_17

[33] Leiss, E.L.: Succint representation of regular languages by boolean automata. Theoret. Comput. Sci. **13**, 323–330 (1981), https://doi.org/10.1016/S0304-3975(81)80005-9

[34] Leung, H.: Separating exponentially ambiguous finite automata from polynomially ambiguous finite automata. SIAM J. Comput. **27**(4), 1073–1082 (1998), http://dx.doi.org/10.1137/S0097539793252092

[35] Leung, H.: Descriptional complexity of NFA of different ambiguity. Internat. J. Found. Comput. Sci. **16**(5), 975–984 (2005), http://dx.doi.org/10.1142/S0129054105003418

[36] Liu, G., Martín-Vide, C., Salomaa, A., Yu, S.: State complexity of basic language operations combined with reversal. Inform. and Comput. **206**(9-10), 1178–1186 (2008), https://doi.org/10.1016/j.ic.2008.03.018

[37] Lupanov, O.B.: A comparison of two types of finite automata. Problemy Kibernetiki **9**, Kibernetiki (1963), (in Russian) German translation: Über den Vergleich zweier Typen endlicher Quellen. Probleme der Kybernetik 6, 328–335 (1966)

[38] Maslov, A.N.: Estimates of the number of states of finite automata. Soviet Math. Doklady **11**(5), 1373–1375 (1970)

[39] Meyer, A.R., Fischer, M.J.: Economy of description by automata, grammars, and formal systems. In: Proceedings of the 12th Annual Symposium on Switching and Automata Theory. pp. 188–191. IEEE Computer Society Press (1971), https://doi.org/10.1109/T-C.1971.223108

[40] Mirkin, B.G.: On dual automata. Kibernetika (Kiev) **2** (1966), pp. 7–10 (in Russian). English translation: Cybernetics 2, pp. 6–9 (1966)

[41] Moon, J., Moser, L.: On cliques in graphs. Israel Journal of Mathematics **3**, 23–28 (1965), http://dx.doi.org/10.1007/BF02760024

[42] Moore, F.R.: On the bounds for state-set size in the proofs of equivalence between deterministic, nondeterministic, and two-way finite automata. IEEE Trans. Comput. **20**(10), 1211–1214 (1971), https://doi.org/10.1109/T-C.1971.223108

[43] Nicaud, C.: Average state complexity of operations on unary automata. In: Kutylowski, M., Pacholski, L., Wierzbicki, T. (eds.) MFCS 1999. LNCS, vol. 1672, pp. 231–240. Springer (1999), https://doi.org/10.1007/3-540-48340-3_21

[44] Palmovský, M.: Kleene closure and state complexity. RAIRO - Theor. Inf. and Appl. **50**(3), 251–261 (2016), https://doi.org/10.1051/ita/2016024

[45] Pighizzini, G., Shallit, J.O.: Unary language operations, state complexity and jacobsthal's function. Internat. J. Found. Comput. Sci. **13**(1), 145–159 (2002), https://doi.org/10.1142/S012905410200100X

[46] Rabin, M.O., Scott, D.S.: Finite automata and their decision problems. IBM J. Res. Dev. **3**(2), 114–125 (1959), https://doi.org/10.1147/rd.32.0114

[47] Rampersad, N.: The state complexity of L^2 and L^k. Inform. Process. Lett. **98**(6), 231–234 (2006), http://dx.doi.org/10.1016/j.ipl.2005.06.011

[48] Raskin, M.A.: A superpolynomial lower bound for the size of non-deterministic complement of an unambiguous automaton. In: Chatzigiannakis, I., Kaklamanis, C., Marx, D., Sannella, D. (eds.) ICALP 2018. LIPIcs, vol. 107, pp. 138:1–138:11. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2018), https://doi.org/10.4230/LIPIcs.ICALP.2018.138

[49] Salomaa, A., Salomaa, K., Yu, S.: State complexity of combined operations. Theoret. Comput. Sci. **383**(2-3), 140–152 (2007), https://doi.org/10.1016/j.tcs.2007.04.015

[50] Schmidt, E.M.: Succinctness of description of context-free, regular, and finite languages. Ph. D. thesis. Cornell University (1978)

[51] Sipser, M.: Introduction to the theory of computation. PWS Publishing Company (1997), http://dx.doi.org/10.1016/S0304-3975(81)80005-9

[52] Šebej, J.: Reversal of regular languages and state complexity. In: Pardubská, D. (ed.) ITAT 2010. CEUR Workshop Proceedings, vol. 683, pp. 47–54. CEUR-WS.org (2010), http://ceur-ws.org/Vol-683/paper8.pdf

[53] Yershov, Yu.L.: On a conjecture of V. A. Uspenskii. Algebra i logika **1**, 45–48 (1962), (in Russian)

[54] Yu, S.: Chapter 2: Regular languages. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages, Vol. 1, pp. 41–110. Springer (1997), https://doi.org/10.1007/978-3-642-59136-5_2

[55] Yu, S., Zhuang, Q., Salomaa, K.: The state complexities of some basic operations on regular languages. Theoret. Comput. Sci. **125**(2), 315–328 (1994), http://dx.doi.org/10.1016/0304-3975(92)00011-F

Appendix

The list of published papers

Journal papers:

1. Galina Jirásková, Ivana Krajňáková: Square on deterministic, alternating, and Boolean finite automata. *International Journal of Foundations of Computer Science* 30(6–7), pp. 1117–1134 (2019). ISSN: 0129-0541, DOI: 10.1142/S0129054119400318

Conference papers:

2. Kristína Čevorová, Galina Jirásková, Ivana Krajňáková: On the square of regular languages In: Markus Holzer, Martin Kutrib (eds.): *Implementation and Application of Automata - 19th International Conference, CIAA 2014, Giessen, Germany, July 30 - August 2, 2014. Proceedings.* Lecture Notes in Computer Science, vol. 8587, pp. 136–147, Springer (2014). ISBN: 978-3-319-08845-7, DOI: 10.1007/978-3-319-08846-4_10

3. Galina Jirásková, Ivana Krajňáková: Square on deterministic, alternating, and Boolean finite automata. In: Giovanni Pighizzini, Cezar Câmpeanu (eds.): *Descriptional Complexity of Formal Systems – 19th IFIP WG 1.02 International Conference, DCFS 2017, Milano, Italy, July 3-5, 2017, Proceedings.* Lecture Notes in Computer Science, vol. 10316, pp. 214–225, Springer (2017). ISBN: 978-3-319-60251-6, DOI: 10.1007/978-3-319-60252-3_17

4. Michal Hospodár, Galina Jirásková, Ivana Krajňáková: Operations on Boolean and alternating automata. In: Fedor V. Fomin, Vladimir V. Podolskii (eds.): *Computer Science – Theory and Applications – 13th International Computer Science Symposium in Russia, CSR 2018, Moscow, Russia, June 6–10, 2018, Proceedings.*

Lecture Notes in Computer Science, vol. 10846, pp. 181–193, Springer (2018). ISBN: 978-3-319-90529-7, DOI: 10.1007/978-3-319-90530-3_16

5. Galina Jirásková, Ivana Krajňáková: NFA-to-DFA trade-Off for regular operations. In: Michal Hospodár, Galina Jirásková, Stavros Konstantinidis (eds.): *Descriptional Complexity of Formal Systems - 21st IFIP WG 1.02 International Conference, DCFS 2019, Košice, Slovakia, July 17-19, 2019, Proceedings.* Lecture Notes in Computer Science, vol. 11612, pp. 184–196, Springer (2019). ISBN: 978-3-030-23246-7, DOI: 10.1007/978-3-030-23247-4_14

The list of given talks

1. Square on deterministic and alternating finite automata. *The Second Workshop on Černý's Conjecture and Optimization Problems on Finite Automata.* Opava, Czech Republic, May 17, 2017.

2. Square on deterministic, alternating, and Boolean finite automata. *19th International Conference on Descriptional Complexity of Formal Systems, DCFS 2017.* Milano, Italy, July 3-5, 2017.

3. NFA-to-DFA trade-off for regular operations. *21th International Conference on Descriptional Complexity of Formal Systems, DCFS 2019.* Košice, Slovakia, July 17-19, 2019.

Appendix [A]

Galina Jirásková, Ivana Krajňáková:
Square on deterministic, alternating, and Boolean finite automata.
International Journal of Foundations of Computer Science, 30(6-7), 1117–1134 (2019).
ISSN: 0129-0541 DOI: 10.1142/S0129054119400318

International Journal of Foundations of Computer Science
Vol. 30, Nos. 6 & 7 (2019) 1117–1134

DOI: 10.1142/S0129054119400318

Square on Deterministic, Alternating, and Boolean Finite Automata*

Galina Jirásková† and Ivana Krajňáková‡
Mathematical Institute, Slovak Academy of Sciences
Grešákova 6, 040 01 Košice, Slovakia
†*jiraskov@saske.sk*
‡*krajnakova@saske.sk*

Received 9 November 2017
Accepted 5 June 2018
Communicated by C. Câmpeanu and G. Pighizzini

We investigate the state complexity of the square operation on languages represented by deterministic, alternating, and Boolean automata. For each k such that $1 \leq k \leq n-2$, we describe a binary language accepted by an n-state deterministic finite automaton with k final states meeting the upper bound $n2^n - k2^{n-1}$ on the state complexity of its square. We show that in the case of $k = n-1$, the corresponding upper bound cannot be met. Using the binary deterministic witness for square with 2^n states where half of them are final, we get the tight upper bounds $2^n + n + 1$ and $2^n + n$ on the complexity of the square operation on alternating and Boolean automata, respectively.

1. Introduction

Square is a basic unary operation on formal languages which is defined as $L^2 = \{uv \mid u \in L \text{ and } v \in L\}$. It is known that if a language L is accepted by a deterministic finite automaton (DFA) of n states, then the language L^2 is accepted by a DFA of at most $n2^n - 2^{n-1}$ states [9]. This upper bound was proven to be tight in the binary case by Rampersad [11]. If the minimal DFA for L has more than one final state this upper bound cannot be met. In such a case the upper bound is $n2^n - k2^{n-1}$, where k is the number of final states in the minimal DFA for L [14].

In this paper, we study the state complexity of the square of languages accepted by DFAs with more final states. Our motivation comes from the paper by Fellah, Jürgensen, and Yu [4] on alternating finite automata (AFAs). They provided an upper bound $2^n + n + 1$ on the complexity of the square of a language represented

*This work was conducted as a part of PhD study of the first author at Comenius University in Bratislava, and it was presented at the DCFS 2017 conference held in Milano, Italy on July 3–5, 2017 and its extended abstract appeared in the conference proceedings: G. Pighizzini, C. Câmpeanu (Eds.) Descriptional Complexity of Formal Systems, LNCS 10316, pp. 214–225.

‡Corresponding author.

by an n-state AFA. A language is accepted by an n-state AFA if and only if its reverse is accepted by a DFA with 2^n states where 2^{n-1} of them are final [1, 4, 7]. It follows that to prove the tightness of the upper bound $2^n + n + 1$, we need to find a language represented by a DFA with half of the states final which is hard for the square operation on DFAs.

The problem seems to be interesting per se. Previously in Ref. [2], we tried to use Rampersad's binary witness for square [11] with k final states instead of the original one. We were able to show the reachability of $n2^n - k2^{n-1}$ states in the subset automaton of an NFA for its square. However, to prove distinguishability a third letter was needed, so the binary case was left open. Surprisingly, in Ref. [2], we were unable to prove the tightness of the upper bound in the case of $n-1$ final states.

Here we solve both these open problems. We describe a binary language accepted by an n-state DFA with k final states meeting the upper bound $n2^n - k2^{n-1}$ on the state complexity of its square provided that $1 \leq k \leq n-2$. In the case of $k = n-1$ we prove that the corresponding upper bound $(2n+2)2^{n-2}$ cannot be met. To show it, we consider two cases. If the initial state is final, then we get the upper bound $(n+2)2^{n-2}$, and we show that it is tight in the binary case. If the initial state is not final the upper bound is $(n+3)2^{n-2}$ and is tight in the ternary case. The tight bound for binary languages is $(n+3)2^{n-2} - 1$ in this case. This solves the complexity of square on DFAs completely. The binary alphabet is optimal sincein the unary case the known tight upper bound is $2n-1$ [11].

Using these results we are able to describe a binary language accepted by an n-state AFA such that every AFA for its square has at least $2^n + n + 1$ states. This proves the tightness of the upper bound $2^n + n + 1$ given in Ref. [4]. We also consider Boolean finite automata (BFA) [1], and get the tight upper bound $2^n + n$ for the square on BFAs. To prove these results, we take the reversal of a language accepted by a DFA with 2^n states with half of them final meeting the corresponding upper bound for square on DFAs. Then this language is accepted by an n-state BFA, and we are able to prove that every BFA for its square has at least $2^n + n$ states. By more careful analysis of the number of final states in the DFA for its square, we get the lower bound $2^n + n + 1$ for the square operation on AFAs. Our result can be extended for the concatenation operation just by concatenating two of our automata with different numbers of states. This provides an alternative proof of the tightness of the upper bound $2^m + n + 1$ for the concatenation operation on alternating automata with m and n states [6].

2. Preliminaries

Let Σ be a finite alphabet of symbols. Then Σ^* denotes the set of words over Σ including the empty word ε. A language is any subset of Σ^*. The concatenation of languages K and L is the language $KL = \{uv \mid u \in K \text{ and } v \in L\}$. The square of a language L is the language $L^2 = LL$. The cardinality of a finite set A is denoted by $|A|$, and its power-set by 2^A. The reader may refer to [5, 12, 13] for details.

A *nondeterministic finite automaton* (NFA) is a quintuple $A = (Q, \Sigma, \circ, I, F)$, where Q is a finite set of states, Σ is a finite non-empty alphabet, $\circ : Q \times \Sigma \to 2^Q$ is the transition function which is naturally extended to the domain $2^Q \times \Sigma^*$, $I \subseteq Q$ is the set of initial states, and $F \subseteq Q$ is the set of final states. The *language accepted by* A is the set $L(A) = \{w \in \Sigma^* \mid I \circ w \cap F \neq \emptyset\}$.

For a symbol a, we say that (p, a, q) is a transition in NFA A if $q \in p \circ a$, and for a word w, we write $p \xrightarrow{w} q$ if $q \in p \circ w$.

An NFA A is *deterministic* (DFA) (and complete) if $|I| = 1$ and $|q \circ a| = 1$ for each q in Q and each a in Σ. We write $p \cdot a = q$ instead of $p \circ a = \{q\}$ in such a case. The *state complexity* of a regular language L, $\mathrm{sc}(L)$, is the smallest number of states in any DFA for L.

Every NFA $A = (Q, \Sigma, \circ, I, F)$ can be converted to an equivalent DFA $A' = (2^Q, \Sigma, \cdot, I, F')$, where $R \cdot a = R \circ a$ for each R in 2^Q and a in Σ, and $F' = \{R \in 2^Q \mid R \cap F \neq \emptyset\}$. We call the DFA A' the *subset automaton* of the NFA A. The subset automaton may not be minimal since some of its states may be unreachable or equivalent to other states.

A *Boolean finite automaton* (BFA) is a quintuple $A = (Q, \Sigma, \delta, g_s, F)$, where Q is a finite non-empty set of states, $Q = \{q_1, \ldots, q_n\}$, Σ is an input alphabet, δ is the transition function that maps $Q \times \Sigma$ into the set $\mathcal{B}_n$ of Boolean functions with variables $\{q_1, \ldots, q_n\}$, $g_s \in \mathcal{B}_n$ is the initial Boolean function, and $F \subseteq Q$ is the set of final states. The transition function δ can be extended to the domain $\mathcal{B}_n \times \Sigma^*$ as follows: For all$g in \mathcal{B}_n$, a in Σ, and w in Σ^*, we have $\delta(g, \varepsilon) = g$; if $g = g(q_1, \ldots, q_n)$, then $\delta(g, a) = g(\delta(q_1, a), \ldots, \delta(q_n, a))$; $\delta(g, wa) = \delta(\delta(g, w), a)$.
Next, let $f = (f_1, \ldots, f_n)$ be the Boolean vector with $f_i = 1$ iff $q_i \in F$. The language accepted by the BFA A is the set $L(A) = \{w \in \Sigma^* \mid \delta(g_s, w)(f) = 1\}$.

A Boolean finite automaton is called *alternating* (AFA) if the initial function is a projection $g(q_1, \ldots, q_n) = q_i$. For details, the reader may referto [1, 4, 7, 8, 12].

The *Boolean (alternating) state complexity* of L, $\mathrm{bsc}(L)(\mathrm{asc}(L))$, is the smallest number of states in any BFA (AFA) for L. It is known that a language L is accepted by an n-state BFA (AFA) if and only if the language L^R is accepted by an 2^n-state DFA (with 2^{n-1} final states). Since this is the crucial observation used later in the paper, we state it in the next two lemmas and provide proof ideas here.

Lemma 1 (cf. [4] Theorem 4.1, Corollary 4.2 and [7], Lemma 1). *Let L be a language accepted by an n-stateBFA (AFA). Then the reversal L^R is accepted by a DFA of 2^n states (of which 2^{n-1} are final).*

Proof Idea. Let $A = (\{q_1, q_2, \ldots, q_n\}, \Sigma, \delta, g_s, F)$ be an n-state BFA for L. Construct a 2^n-state NFA $A' = (\{0,1\}^n, \Sigma, \delta', S, \{f\})$, where

- for every $u = (u_1 \ldots, u_n) \in \{0,1\}^n$ and every $a \in \Sigma$,
  $\delta'(u, a) = \{u' \in \{0,1\}^n \mid \delta(q_i, a)(u') = u_i \text{ for } i = 1, \ldots, n\}$;
- $S = \{(b_1, \ldots, b_n) \in \{0,1\}^n \mid g_s(b_1, \ldots, b_n) = 1\}$;
- $f = (f_1, \ldots, f_n) \in \{0,1\}^n$ with $f_i = 1$ iff $q_i \in F$.

Then $L(A) = L(A')$ and $(A')^R$ is deterministic. Moreover if A is an AFA then A' has 2^{n-1} initial states. It follows that L^R is accepted by a DFA with 2^n states, of which 2^{n-1} are final if A is an AFA. □

Corollary 2. *If L is a regular language, then* $\mathrm{bsc}(L) \geq \lceil \log(\mathrm{sc}(L^R)) \rceil$ *and* $\mathrm{asc}(L) \geq \lceil \log(\mathrm{sc}(L^R)) \rceil$.

Lemma 3 (cf. [7], Lemma 2). *Let L^R be accepted bya DFA A of 2^n states (of which 2^{n-1} are final). Then L is accepted by an n-state BFA (AFA).*

Proof Idea. Consider a 2^n-state NFA A^R for L which hasexactly one final state and the set of initial states S (and $|S| = 2^{n-1}$). Let the state set Q of A^R be $\{0, 1, \ldots, 2^n - 1\}$ with final state k and the initial set S ($S = \{2^{n-1}, \ldots, 2^n - 1\}$). Let δ be the transition function of A^R. Moreover, for every $a \in \Sigma$ and for every $i \in Q$, there is exactly one state j such that j goes to i on a in A^R. For a state $i \in Q$, let $\mathrm{bin}(i) = (b_1, \ldots, b_n)$ be the binary n-tuple such that $b_1 b_2 \cdots b_n$ is the binary notation of i on n digits with leading zeros if necessary.

Let us define an n-state BFA $A' = (Q', \Sigma, \delta', g_s, F')$, where $Q' = \{q_1, \ldots, q_n\}$, $F' = \{q_\ell \,|\, \mathrm{bin}(k)_\ell = 1\}$, and $g_s(\mathrm{bin}(i)) = 1$ iff $i \in S$ ($g_s = q_1$). We define δ' to satisfy the condition: for each i in Q and a in Σ, $(\delta'(q_1, a), \ldots, \delta'(q_n, a))(\mathrm{bin}(i)) = \mathrm{bin}(j)$ where $i \in \delta(j, a)$. Then $L(A') = L(A^R)$. □

3. Square on DFAs

Let us begin with the precise method to construct an NFA for the square of some languages accepted by a minimal DFA with n states.

Construction 4. (*DFA $A \to$ NFA N for $L^2(A)$*)
Let $A = (\{q_0, q_1, \ldots, q_{n-1}\}, \Sigma, \cdot, q_0, F_A)$ be a minimal DFA. We construct NFA $N = (\{q_0, q_1, \ldots, q_{n-1}\} \cup \{0, 1, \ldots, n-1\}, \Sigma, \circ, I, F_N)$ as follows:

(a) take A and add a copy of A with the state set $\{0, 1, \ldots, n-1\}$;
(b) for each symbol $a$and each stateq_i with $q_i \cdot a \in F_A$, add transition $(q_i, a, 0)$;
(c) the set of initial states of N is $I = \{q_0\}$ if $q_0 \notin F$, and $I = \{q_0, 0\}$otherwise;
(d) the set of final states of N is $F_N = \{j \in \{0, 1, \ldots, n-1\} \,|\, q_j \in F_A\}$.

Proposition 5 (Upper Bound). *Let L be a language with* $\mathrm{sc}(L) = n$, *and let the minimal DFA for L have k final states. Then* $\mathrm{sc}(L^2) \leq n2^n - k2^{n-1}$.

Proof. Let L be accepted by DFA $A = (\{q_0, q_1, \ldots, q_{n-1}\}, \Sigma, \cdot, q_0, F_A)$ and let $|F_A| = k$. Construct an NFA N for L^2 as described above. Since A is deterministic, every reachable subset in the subset automaton of N is in the form of $\{q_i\} \cup S$, where $S \subseteq \{0, 1, \ldots, n-1\}$. Furthermore, if q_i is a final state of A, then $0 \in S$ because of the used construction. It follows that subsets containing a final state of

A and missing 0 are unreachable. Hence the subset automaton of N has at most $n2^n - k2^{n-1}$ reachable states. □

Notice that the upper bound given by the above proposition is maximal if $k = 1$, and it is $n2^n - 2^{n-1}$ in this case. The binary witness language meeting this bound was presented by Rampersad in 2006 [11].

Theorem 6 ([11], Theorem 1). *For every $n \geq 3$, there exists a DFA M with n states such that the minimal DFA for the language $L^2(M)$ has $n2^n - 2^{n-1}$ states.*

Unfortunately, the square of Rampersad's automaton with k final states does not meet the upper bound on the state complexity in the general case. Here we provide the binary witness automaton with k final states that meets the upper bound $n2^n - k2^{n-1}$.

Theorem 7. *Let $n \geq 3$ and $1 \leq k \leq n-2$. Then there exists a minimal n-state DFA A with k final states defined over a binary alphabet such that every DFA for $L(A)^2$ has at least $n2^n - k2^{n-1}$ states.*

Proof. Let us take n-state DFA $A = (\{q_0, q_1, \ldots, q_{n-1}\}, \Sigma, \cdot, q_0, F_A)$ with k final states shown in Fig. 1. Notice that q_0 and q_1 remain non-final with every k in this DFA and there are two cycles; one on a, $(q_0, q_1, \ldots, q_{n-1})$ of length n and the second on b, $(q_2, q_3, \ldots, q_{n-1})$ of length$n-2$. Let us build an NFA N for $L(A)^2$ as in Construction 4. An example of NFA N if $n = 6$ and $k = 2$ is shown in Fig. 2.

We observe that there are only two types of states reachable in the subset automaton of N:

- $\{q_i\} \cup S$, where $S \subseteq \{0, 1, \ldots, n-1\}$ and $0 \leq i \leq n-k-1$;
- $\{q_i, 0\} \cup S$, where $S \subseteq \{1, \ldots, n-1\}$ and $n-k \leq i \leq n-1$.

We denote this family of sets as $\mathcal{R}$. We can see that in the family $\mathcal{R}$ there are exactly $(n-k)2^n$ sets of the first type and $k2^{n-1}$ sets of the second type. Hence the family $\mathcal{R}$ consists of $(n-k)2^n + k2^{n-1} = n2^n - k2^{n-1}$ sets. Our goal is to show that the sets in $\mathcal{R}$ are reachable and also pairwise distinguishable in the subset automaton of N. Let us start with reachability. We use mathematical induction by number of elements in set/state. The sets with one and two elements are reachable, because:

$$\to \{q_0\} \xrightarrow{a} \{q_1\} \xrightarrow{a} \cdots \xrightarrow{a} \{q_{n-k-1}\} \xrightarrow{a} \{q_{n-k}, 0\},$$

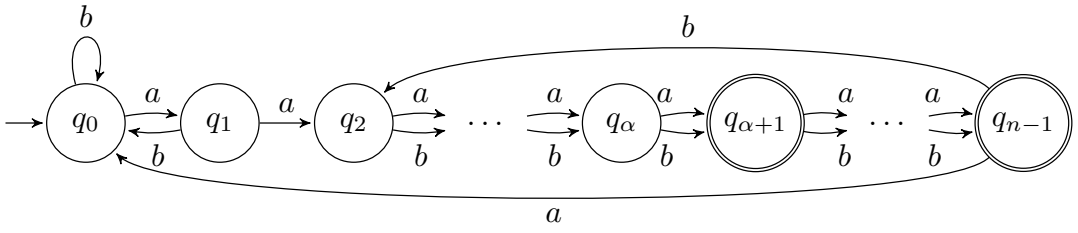

Fig. 1. A witness DFA A with k final states meeting thebound $n2^n - k2^{n-1}$, where $\alpha = n-k-1$.

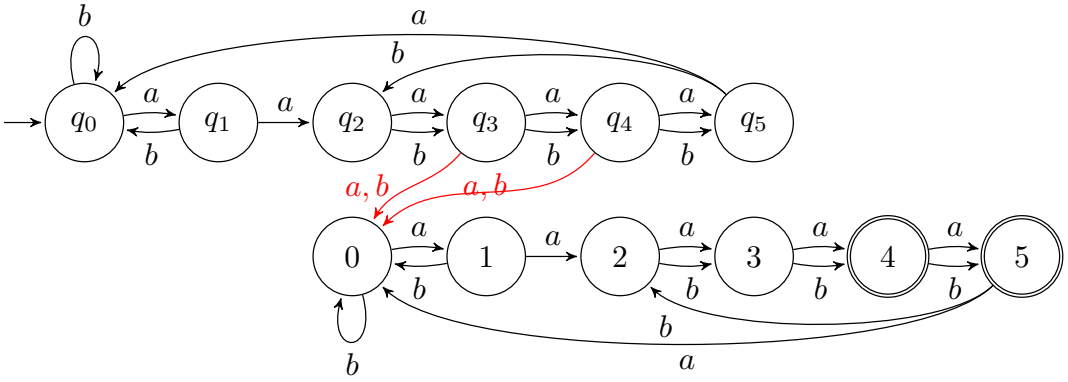

Fig. 2. NFA N for the square of $L(A)$, if $n = 6$ and $k = 2$.

$$\{q_{n-k}, 0\} \xrightarrow{b} \{q_{n-k+1}, 0\} \xrightarrow{b} \cdots \xrightarrow{b} \{q_{n-2}, 0\} \xrightarrow{b} \{q_{n-1}, 0\},$$

$$\{q_{n-1}, 0\} \xrightarrow{a} \{q_0, 1\} \xrightarrow{b} \{q_0, 0\},$$

$$\{q_0, 1\} \xrightarrow{a} \{q_1, 2\} \xrightarrow{b} \{q_0, 3\} \xrightarrow{b} \{q_0, 4\} \xrightarrow{b} \cdots \xrightarrow{b} \{q_0, n-1\} \xrightarrow{b} \{q_0, 2\},$$

$$\{q_0, (j-i) \bmod n\} \xrightarrow{a^i} \{q_i, j\} \quad \text{for } i = 0, 1, \ldots, n-k-1 \text{ and } j = 0, 1, \ldots, n-1.$$

Assume now that every set in $\mathcal{R}$ with t elements is reachable. We show that then every set in $\mathcal{R}$ of size $t+1$ is reachable. Let $S = \{q_i, s_1, s_2, \ldots, s_t\}$ be our desired set in $\mathcal{R}$ of size$t+1$, where $q_i \in Q$ and $0 \le s_1 < s_2 < \cdots < s_t \le n-1$. We deal with three cases:

(1) We show the reachability of sets of the second type, so let $n-k \le i \le n-1$ and therefore $s_1 = 0$. We can write i as $i = \alpha + \beta$, where $\alpha = n-k-1$ and $1 \le \beta \le k$, so our desired set is $S = \{q_{\alpha+\beta}, 0, s_2, s_3, \ldots, s_t\}$.

Let $s_2 = 1$, and take the set $\{q_{\alpha+\beta-1}, 0, s_3 - 1, \ldots, s_t - 1\}$, which is in $\mathcal{R}$ and is reachable because it has t elements; recall that the transitions on b form the cycle $(q_2, q_3, \ldots, q_{n-1})$ in A. Then we have

$$\{q_{\alpha+\beta-1}, 0, s_3 - 1, \ldots, s_t - 1\} \xrightarrow{a} \{q_{\alpha+\beta}, 0, 1, s_3, \ldots, s_t\} = S.$$

Let $s_2 \ge 2$ and take the set $\{q_\alpha, s_2 \cdot b^{n-1-\beta} - 1, \ldots, s_t \cdot b^{n-1-\beta} - 1\}$, which is in $\mathcal{R}$ and is reachable because it has t elements. Then we have

$$\{q_\alpha, s_2 \cdot b^{n-1-\beta} - 1, \ldots, s_t \cdot b^{n-1-\beta} - 1\} \xrightarrow{a} \{q_{\alpha+1}, 0, s_2 \cdot b^{n-1-\beta}, \ldots, s_t \cdot b^{n-1-\beta}\}$$

$$\xrightarrow{b^{\beta-1}} \{q_{\alpha+\beta}, 0, s_2 \cdot b^{n-2}, \ldots, s_t \cdot b^{n-2}\} = \{q_{\alpha+\beta}, 0, s_2, \ldots, s_t\} = S.$$

(2) Next we show the reachability of sets of the first type in the next two steps. Let $i = 0$. We distinguish between three cases of s_1.

Firstly let $s_1 = 0$. We start from the set reached previously in (1) to achieve S in case of $s_2 = 1$ by $\{q_{n-1}, 0, s_3 - 1, \ldots, s_t - 1, n-1\} \xrightarrow{a} \{q_0, 0, 1, s_3, \ldots, s_t\} = S$. Otherwise, if desired $s_2 \geq 2$, we reach S using previously reached set

$$\{q_0, 0, 1, s_3 - s_2 + 1, \ldots, s_t - s_2 + 1\} \xrightarrow{a} \{q_1, 1, 2, s_3 - s_2 + 2, \ldots, s_t - s_2 + 2\}$$

$$\xrightarrow{b^{n-2}} \{q_0, 0, 2, s_3 - s_2 + 2, \ldots, s_t - s_2 + 2\} \xrightarrow{b^{s_2-2}} \{q_0, 0, s_2, \ldots, s_t\} = S.$$

Secondly let $s_1 \geq 1$. Then the set $S' = \{q_{n-1}, 0, s_2 - s_1, \ldots, s_t - s_1\}$ is reached in (1). If $s_1 = 1$, then $S' \xrightarrow{a} S$, otherwise $s_1 \geq 2$, and $S' \xrightarrow{aab^{n-2}b^{s_1-2}} S$.

(3) Let $1 \leq i \leq n-k-1$. Now we can reach the remaining sets of the first type using sets achieved in (2) like this $\{q_0, (s_1 - i) \bmod n, \ldots, (s_t - i) \bmod n\} \xrightarrow{a^i} \{q_i, s_1, \ldots, s_t\} = S$.

Let us continue with proving distinguishability of reached sets. Note that in N we have

$$\{n-1\} \xrightarrow{b} \{2\} \xrightarrow{a} \{3\} \xrightarrow{b^{n-2}} \{3\} \xrightarrow{ab^{n-2}} \{4\} \xrightarrow{ab^{n-2}} \cdots \xrightarrow{ab^{n-2}} \{n-1\}.$$

This means that the word $w = b(ab^{n-2})^{n-3}$ is accepted from the state $n-1$. Let us read w from a different state t, $2 \leq t \leq n-2$. First we have $t \circ b \in \{3, 4, \ldots, n-1\}$. Next $\{3, 4, \ldots, n-1\} \circ (ab^{n-2})^{n-3} = \{0\}$, so w is not accepted from t. Similarly, reading w from $\{0, 1\}$ results in the set $\{0\}$, thus w is not accepted from $\{0, 1\}$ either. Moreover, w is not accepted from $\{q_i\}$, because $\{q_i\} \circ w \subseteq \{q_j, 0\}$, where either $j = 0$ if $i < n-1$, or $j = n-1$ if $i = n-1$. Therefore w is accepted by N from and only from the state $n-1$. Notice that each state t in $\{1, 2, \ldots, n-1\}$ has exactly one in-transition on a going from the state $t-1$, so the word $a^{n-1-t}w$ is accepted by N only from state t, $0 \leq t \leq n-2$. It follows that two sets $\{q_i\} \cup S$ and $\{q_j\} \cup T$ in $\mathcal{R}$ are distinguishable if $S \neq T$.

Now consider two distinct subsets $\{q_i\} \cup S$ and $\{q_j\} \cup S$ in $\mathcal{R}$. Without loss of generality, we have $0 \leq i < j \leq n-1$. We will discuss three cases:

(1) Let $i = 0$ and $j = 1$. Then

$$\{q_0\} \cup S \xrightarrow{(ab^{n-2})^{n-2}} \{q_0, 0\} \xrightarrow{a} \{q_1, 1\} \xrightarrow{a^{n-k-1}} \{q_{n-k}, 0, n-k\},$$

$$\{q_1\} \cup S \xrightarrow{(ab^{n-2})^{n-2}} \{q_{n-1}, 0\} \xrightarrow{a} \{q_0, 1\} \xrightarrow{a^{n-k-1}} \{q_{n-k-1}, n-k\}.$$

Now we can distinguish these sets because they differ in the element from the second automaton copy.

(2) Let $i = 0$ and $j \geq 2$. Then

$$\{q_0\} \cup S \xrightarrow{b^{n-1-j}a} \{q_1\} \cup S_1,$$

$$\{q_j\} \cup S \xrightarrow{b^{n-1-j}a} \{q_0\} \cup S_1',$$

for some subsets S_1, S_1' of $\{0, 1, \ldots, n-1\}$. If the subsets S_1 and S_1' are the same, then we continue as in (1), otherwise we continue as in case of $S \neq T$.

(3) Let $i \geq 1$. Then

$$\{q_i\} \cup S \xrightarrow{a^{n-j}} \{q_{i+(n-j)}\} \cup S_1,$$

$$\{q_j\} \cup S \xrightarrow{a^{n-j}} \{q_0\} \cup S_1'.$$

Similarly as in (2), if the subsets S_1 and S_1' are the same we continue as in (1) or (2), otherwise we continue as in case of $S \neq T$. □

3.1. *Square for DFAs with $n-1$ final states*

Recall that the automaton in the proof of Theorem 7 must have at least two non-final states. We show that for every language L accepted by an n-state DFA $A = (Q, \Sigma, \cdot, q_0, F)$ with a single non-final state, the state complexity of L^2 never meets the upper bound set in Proposition 5. In particular, weshow:

(a) if $q_0 \in F$, then $\mathrm{sc}(L^2) \leq (n+2)2^{n-2}$ and this bound is tight if $|\Sigma| \geq 2$;
(b) if $q_0 \notin F$, then $\mathrm{sc}(L^2) \leq (n+3)2^{n-2}$ and this bound is tight if $|\Sigma| \geq 3$.

Moreover we show that the upper bound $(n+3)2^{n-2}$ in the case $q_0 \notin F$ cannot be met by any binary language, and that the tight upper bound in the binary case is $(n+3)2^{n-2}-1$. Firstly let us consider the case of $|F| = n-1$ and $q_0 \in F$.

Lemma 8. *Let $n \geq 3$ and let L be a regular language accepted by an n-state DFA $A = (Q, \Sigma, \cdot, q_0, F)$ with $n-1$ final states, where $q_0 \in F$. Then $\mathrm{sc}(L^2) \leq (n+2)2^{n-2}$, and this bound is tight if $|\Sigma| \geq 2$.*

Proof. The formula for the upper bound is based on the observation that q_0 is initial and also accepting in A, so the initial state in the subset automaton for $L(A)^2$ is $\{q_0, 0\}$. It follows that for every $i \in \{0, 1, \ldots, n-1\}$ if $\{q_i\} \cup X$ is reachable, then $i \in X$. So we consider the following family $\mathcal{R}$ of possible reachable sets in the subset automaton for $L(A)^2$:

$$\begin{aligned}\mathcal{R} = &\big\{\{q_0, 0\} \cup X \mid X \subseteq \{1, 2, \ldots, n-1\}\big\} \\ &\cup \big\{\{q_1, 1\} \cup X \mid X \subseteq \{0, 2, 3, \ldots, n-1\}\big\} \\ &\cup \big\{\{q_i, 0, i\} \cup X \mid 2 \leq i \leq n-1, X \subseteq \{1, 2, \ldots, n-1\} \setminus \{i\}\big\},\end{aligned}$$

where we assume that q_1 is the only non-final state. Notice that this family consists of $(n+2)2^{n-2}$ sets. Hence $\mathrm{sc}(L^2) \leq (n+2)2^{n-2}$. To prove the tightness of this upper bound, we introduce the DFA A shown in Fig. 3 and we show that every DFA for $L(A)^2$ has at least $(n+2)2^{n-2}$ states.

Construct an NFA N for $L(A)^2$ as described in Construction 4. We want to show that each set in $\mathcal{R}$ is reachable in the subset automaton of N and that all these sets are pairwise distinguishable.

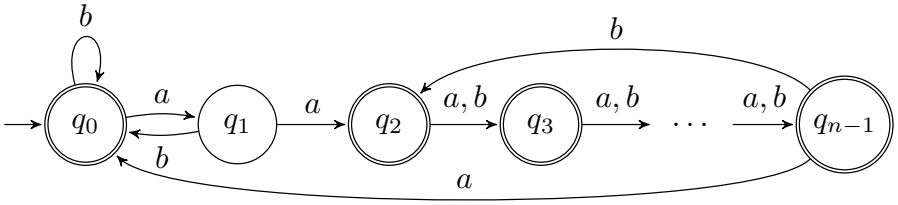

Fig. 3. A witness DFA A with $n-1$ final states meeting the bound $(n+2)2^{n-2}$, where $q_0 \in F$.

Firstly, we prove reachability by induction on the size of the subsets. The basis, where $|S| = 2$, holds true since $\{q_0, 0\}$ is the initial subset, and it goes to $\{q_1, 1\}$ by a. Now, let $2 \le t \le n$ and assume that each set in $\mathcal{R}$ of size t is reachable.

Let $S = \{q_i, s_1, s_2, \ldots, s_t\}$, where $0 \le i \le n-1$, be a set in $\mathcal{R}$ of size $t+1$. If $i = 0$, then we have $0 = s_1 < s_2 < \cdots < s_t \le n-1$. If $i = 1$, then we have $s_1 = 0$ and $1 = s_2 < \cdots < s_t \le n-1$. If $i \ge 2$, then $s_1 = 0$, $s_2 = i$ and $1 \le s_3 < \cdots < s_t \le n-1$.

We consider several cases:

(1) Let $i = 2$ and thus $s_1 = 0$ and $s_2 = i = 2$. Then $\{q_1, 1, s_3 - 1, \ldots, s_t - 1\} \xrightarrow{a} \{q_2, 0, 2, s_3, \ldots, s_t\}$. The former set is reachable by the induction hypothesis.

(2) Let $i \ge 3$, so $s_1 = 0$ and $s_2 = i$. Then $\{q_2, 0, 2, s_3 \cdot b^{n-2-i+2}, \ldots, s_t \cdot b^{n-2-i+2}\} \xrightarrow{b^{i-2}} \{q_i, 0, i, s_3, \ldots, s_t\}$ and the set on the left was reached in (1).

(3) Let $i = 0$ and thus $s_1 = 0$.

(3a) Let $s_2 = 1$. Then $\{q_{n-1}, 0, n-1, s_3 - 1, \ldots, s_t - 1\} \xrightarrow{a} \{q_0, 0, 1, s_3, \ldots, s_t\}$. The former set is considered in case (2).

(3b) Let $s_2 \ge 2$. Then we have $\{q_0, 0, 1, s_3 - s_2 + 1, \ldots, s_t - s_2 + 1\} \xrightarrow{a} \{q_1, 1, 2, s_3 - s_2 + 2, \ldots, s_t - s_2 + 2\} \xrightarrow{b^{n-2}} \{q_0, 0, 2, s_3 - s_2 + 2, \ldots, s_t - s_2 + 2\} \xrightarrow{b^{s_2-2}} \{q_0, 0, s_2, s_3, \ldots, s_t\}$. The first set in this chain is considered in case (3a).

(4) Let $i = 1$ ($s_1 = 1$). Then we have $\{q_0, 0, s_2 - 1, s_3 - 1, \ldots, s_t - 1\} \xrightarrow{a} \{q_1, 1, s_2, s_3, \ldots, s_t\}$. The former set is considered in (3). This proves reachability.

Secondly we prove distinguishability by analysing possible cases of two distinct states $p = \{q_i\} \cup S$ and $q = \{q_j\} \cup T$. Notice that the word ab^{n-2} performs the transformation

$$2 \xrightarrow{ab^{n-2}} 3 \xrightarrow{ab^{n-2}} \cdots \xrightarrow{ab^{n-2}} n-1 \xrightarrow{ab^{n-2}} 0 \xrightarrow{ab^{n-2}} 0 \quad \text{and} \quad 1 \xrightarrow{ab^{n-2}} 0,$$

so by reading the word $(ab^{n-2})^{n-3}$ the state q_2 is sent to q_{n-1} while every other state is sent to q_0. It follows that theword $(ab^{n-2})^{n-3}a$ is accepted only from the state q_2 and the word $b(ab^{n-2})^{n-3}a$ is accepted only from the state q_{n-1}. Finally, for each t, $0 \le t \le n-1$, the word $a^{n-1-t}b(ab^{n-2})^{n-3}a$ is accepted by NFA N only from state t. It follows that the sets $\{q_i\} \cup S$ and $\{q_j\} \cup T$ are distinguishable if $S \ne T$. Consider now the opposite situation, that two sets p and q differ only in i or j, that is, $S = T$ and without loss of generality $0 \le i < j \le n-1$.

(1) Let $i = 0$ and $j = n-2$. Then $\{0, n-2\} \subseteq S$. We use b^{n-2} to get $\{q_0, 0\} \cup S_1$ and $\{q_{n-2}, 0\} \cup S_1$ where $S_1 \subseteq \{2, 3, \ldots, n-1\}$. Now, if $n-1 \notin S_1$, we use a to get $\{q_1, 1\} \cup S_1'$ and $\{q_{n-1}, 0, 1\} \cup S_1'$ which differ in state 0, and so are distinguishable. If $n-1 \in S_1$, we use ab^{n-2} to get $\{q_0, 0\} \cup S_2$ and $\{q_{n-2}, 0\} \cup S_2$ where

$$S_2 \subseteq \{2, 3, \ldots, n-1\} \quad \text{and} \quad |S_2| < |S_1|.$$

We use the same argument to S_2. If state $n-1$ is in all the resulting sets, then we eventually get $\{q_0, 0, n-2\}$ and $\{q_{n-2}, 0, n-2\}$. Finally we use a to get $\{q_1, 1, n-1\}$ and $\{q_{n-1}, 0, 1, n-1\}$, which differ in state 0.

(2) Let $i = 0$, $2 \leq j \leq n-3$. We read b^{n-2-j} and get $\{q_0\} \cup S'$ and $\{q_{n-2}\} \cup S'$ which is considered in case (1).

(3) Let $i = 0$ and $j = 1$. Here we read ab and get $\{q_0, 0\} \cup S_1$ and $\{q_3, 0\} \cup S_1'$. If $S_1 \neq S_1'$ then we can continue as in case $S \neq T$, otherwise as in case (2).

(4) Let $i = 0$, $j = n-1$. We read a and get $\{q_1\} \cup S'$ and $\{q_0\} \cup S'$ which is considered in case (3).

(5) Let $1 \leq i < j \leq n-1$. We read a^{n-j} and get $\{q_{n-j+i}\} \cup S'$ and $\{q_0\} \cup S'$ which is considered in cases (1)–(4). □

Now let us consider the case where $|F| = n-1$and $q_0 \notin F$.

Lemma 9. *Let $n \geq 3$. Let L be a regular language accepted by an n-state DFA $A = (Q, \Sigma, \cdot, q_0, F)$, where $|F| = n-1$ and $q_0 \notin F$. Then $\mathrm{sc}(L^2) \leq (n+3)2^{n-2}$, and the bound is tight if $|\Sigma| \geq 3$. The bound $(n+3)2^{n-2} - 1$ can be met by a binary language.*

Proof. We start with the upper bound. Suppose we have constructed an NFA $N$from the DFA A as described in Construction 4. Consider the corresponding subset automaton of N. We first show that two distinct subsets of this automaton, $\{q_i\} \cup S$ and $\{q_j\} \cup S$, where $\{i, j\} \subseteq S$ are equivalent. If a word w is rejected from state $\{q_i\} \cup S$ then $s \xrightarrow{w} 0$ for each element s in S. It follows that$w$ is rejected from $\{q_j\} \cup S$ because $\{q_j\} \cup S \xrightarrow{w} \{q_0, 0\}$. Likewise, if w is rejected from $\{q_j\} \cup S$ then w is rejected from $\{q_i\} \cup S$. Excluding these equivalents subsets gives us the family $\mathcal{R}$ of $(n+3)2^{n-2}$ reachable and pairwise distinguishable subsets of the subset automaton of N, which is:

$$\mathcal{R} = \{\{q_0\} \cup X \mid X \subseteq \{0, 1, \ldots, n-1\}\}$$
$$\cup \{\{q_i\} \cup X \mid X \subseteq \{0, 1, \ldots, n-1\}, 0 \in X, i \notin X\}.$$

To prove the tightness of this upper bound, we introduce the DFA B shown in Fig. 4 and we show that every DFA for $L(B)^2$ has at least $(n+3)2^{n-2}$ states. Construct an NFA N for the square of $L(B)^2$ as described in Construction 4. Let us show that each set in $\mathcal{R}$ is reachable in the subset automaton of N and that all these sets are pairwise distinguishable.

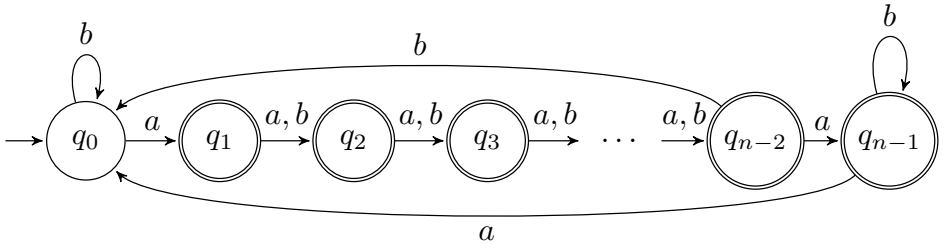

Fig. 4. A binaryDFA B with $\mathrm{sc}(L^2(B)) = (n+3)2^{n-2} - 1$.

We prove the reachability by induction on the size of subsets. The basis, where $|S| \leq 2$, holds true up to one set, namely $\{q_0, n-1\}$, since we have

$$\to \{q_0\} \xrightarrow{a} \{q_1, 0\} \xrightarrow{b} \{q_2, 0\} \xrightarrow{b} \cdots \xrightarrow{b} \{q_{n-2}, 0\} \xrightarrow{b} \{q_0, 0\},$$

$$\{q_{n-2}, 0\} \xrightarrow{a} \{q_{n-1}, 0, 1\} \xrightarrow{b} \{q_{n-1}, 0, 2\} \xrightarrow{b} \cdots \xrightarrow{b} \{q_{n-1}, 0, n-2\} \xrightarrow{b} \{q_{n-1}, 0\},$$

$$\{q_{n-1}, 0\} \xrightarrow{a} \{q_0, 1\} \xrightarrow{b} \{q_0, 2\} \xrightarrow{b} \cdots \xrightarrow{b} \{q_0, n-2\}.$$

We deal with $\{q_0, n-1\}$ later. Now assume that each set in $\mathcal{R}$ of size t is reachable. Let $S = \{q_i, s_1, s_2, \ldots, s_t\}$ be a set of size $t+1$. Consider several cases.

(1) Let $i = 1$, so $s_1 = 0$. Then $\{q_0, s_2 - 1, \ldots, s_t - 1\} \xrightarrow{a} \{q_1, 0, s_2, \ldots, s_t\}$, where the former set of size t is reachable by the induction hypothesis.

(2) Let $2 \leq i \leq n-2$, so $S = \{q_i, 0, s_2, s_3, \ldots, s_t\}$.

If $s_2 = 1$, then $\{q_{i-1}, 0, s_3 - 1, \ldots, s_t - 1\} \xrightarrow{a} S$.

If $s_2 \geq 2$ and $s_t \leq n-2$, then $\{q_{i-1}, 0, s_2 - 1, \ldots, s_t - 1\} \xrightarrow{b} S$.

If $s_2 \geq 2$ and $s_t = n-1$, then $\{q_{i-1}, 0, s_2 - 1, \ldots, s_{t-1} - 1, n-1\} \xrightarrow{b} S$.

This induction step with case (1) as the basis proves case (2) by induction on i.

(3) Let $i = n-1$, so $S = \{q_{n-1}, 0, s_2, s_3, \ldots, s_t\}$. Consider two cases of s_t.

If $s_t \leq n-2$, then $\{q_{n-2}, 0, s_3 - s_2, \ldots, s_t - s_2\} \xrightarrow{ab^{s_2-1}} S$.

If $s_t = n-1$, then $\{q_{n-2}, 0, s_3 - s_2, \ldots, s_{t-1} - s_2, n-2\} \xrightarrow{ab^{s_2-1}} S$.

The starting set is reachable by induction on t in both cases.

(4) Let $i = 0$, so $S = \{q_0, s_1, s_2, \ldots, s_t\}$. We consider four cases of s_1 and s_t:

If $s_1 = 0$, $s_t \leq n-2$, then $\{q_{n-1}, 0, n-1, s_3 - s_2, \ldots, s_t - s_2\} \xrightarrow{ab^{s_2-1}} S$.

If $s_1 = 0$, $s_t = n-1$, then $\{q_{n-1}, 0, n-1, s_3 - s_2, \ldots, s_{t-1} - s_2, n-2\} \xrightarrow{ab^{s_2-1}} S$.

If $s_1 \geq 1$, $s_t \leq n-2$, then $\{q_{n-1}, 0, s_2 - s_1, \ldots, s_t - s_1\} \xrightarrow{ab^{s_1-1}} S$.

If $s_1 \geq 1$, $s_t = n-1$, then $\{q_{n-1}, 0, s_2 - s_1, \ldots, s_{t-1} - s_1, n-2\} \xrightarrow{ab^{s_1-1}} S$.

The starting sets are considered in case (3).

This proves reachability. To prove distinguishability, notice that the word b^n is accepted by NFA N only from state $n-1$. It follows that $a^{n-1-t}b^n$ is accepted only from state t, $0 \leq t \leq n-1$. Hence two sets $\{q_i\} \cup S$ and $\{q_j\} \cup T$ are distinguishable if $S \neq T$. Consider two sets $\{q_i\} \cup S$, $\{q_j\} \cup S$ where $0 \leq i < j \leq n-1$ and assume

that $\{i,j\} \not\subseteq S$. Let $i=0$ and $S \subseteq \{0,1,\ldots,n-1\}$. Then $0 \in S$ and $j \notin S$, and we have

$$\{q_0\} \cup S \xrightarrow{a^{n-1-j}b^n} \{q_0,0\} \xrightarrow{a} \{q_1,0,1\},$$

$$\{q_j\} \cup S \xrightarrow{a^{n-1-j}b^n} \{q_{n-1},0\} \xrightarrow{a} \{q_0,1\},$$

where the resulting states differ in state 0. If $i \geq 1$, then we use a^{n-j} to get the case above.

Up to now, we reached all sets in $\mathcal{R}$ except for $\{q_0, n-1\}$. This set remains unreachable because of the inability to reach it by a nor b from other state. Hence $\mathrm{sc}(L^2(B)) = (n+3)2^{n-2}-1$. To reach the set $\{q_0,n-1\}$, we add one more symbol to B. We define the transitions on the symbol c as follows:

$$\delta(q_0,c)=q_0; \quad \delta(q_i,c)=q_{i+i} \quad \text{if } 1 \leq i \leq n-2; \quad \delta(q_{n-1},c)=q_0.$$

Denote the resulting DFA over $\{a,b,c\}$ by C. Then in the corresponding subset automaton for $L^2(C)$ the set $\{q_0,n-1\}$ is reachable from $\{q_0,n-2\}$ by c. The proof of the distinguishability remains the same. Thus $\mathrm{sc}(L^2(C)) = (n+3)2^{n-2}$. □

As a corollary of the two lemmas above, we get the next result.

Corollary 10. *Let $n \geq 3$ and L be a language over Σ accepted by an n-state DFA in which $n-1$ states are final. Then $\mathrm{sc}(L^2) \leq (n+3)2^{n-2}$, and this bound is tight if $|\Sigma| \geq 3$. The bound $(n+3)2^{n-2}-1$ is met by a binary language.*

We tested the state complexity of square on all binary automata with 3, 4 and 5 states where the initial state is the only non-final state. But we did not find any binary automaton with the state complexity of its square greater than $(n+3)2^{n-2}-1$. The following result shows that this lower bound is tight for every $n \geq 3$ on a binary alphabet.

Theorem 11. *Let $n \geq 3$ and L be a binary language accepted by an n-state DFA in which $n-1$ states are final. Then $\mathrm{sc}(L^2) \leq (n+3)2^{n-2}-1$, and this bound is tight.*

Proof. We already showed the witness language with $\mathrm{sc}(L^2) \geq (n+3)2^{n-2}-1$ in Lemma 9. It remains to show that the upper bound $(n+3)2^{n-2}$ cannot be met in the binary case.

Suppose for a contradiction that there is a minimal n-state DFA $A = (Q,\{a,b\},\cdot,q_0,Q\backslash\{q_0\})$ where the only non-final state is the initial state and the NFA for its square has $(n+3)2^{n-2}$ reachable and also distinguishable states. Let us take a closer look at its transitions.

The option of $q_0 \cdot a = q_0 \cdot b = q_0$ is unsatisfying because that means that A is not minimal. Without the loss of generality let $q_0 \cdot a = q_1$ where $q_1 \neq q_0$. We show that $q_0 \cdot b = q_0$. If not, then $q_0 \cdot b = q_j$ and $j \neq 0$. Let us show that we are

unable to reach the set $\{q_0, 0\}$. If we would try to reach it by reading an a or b from $\{q_0\} \cup S$ we would reach $\{q_1, \ldots\}$, or $\{q_j, \ldots\}$. We can try from the subset $\{q_i\} \cup S$, where $i \neq 0$, so $0 \in S$. Then we would have $\{q_i, 0, \ldots\} \xrightarrow{a} \{\ldots, 1, \ldots\}$ or $\{q_i, 0, \ldots\} \xrightarrow{b} \{\ldots, j, \ldots\}$, so not reach $\{q_0, 0\}$ at all. Hence $q_0 \cdot b = q_0$.

Now we consider the transitions that would ensure the reachability of the subsets $\{q_0, 0\}, \{q_0, 1\}, \{q_0, 2\}, \ldots, \{q_0, n-1\}$. All of them, except for $\{q_0, 1\}$, cannot be reached by reading an a for the same reason as we showed previously with $\{q_0, 0\}$. That means that they are reached by reading a b. There must be some final state, that has outcoming transition on b to q_0 otherwise $\{q_0, 0\}$ will remain unreachable since $q_0 \notin F$. Let us denote this state as q_{n-1}. There must be also incoming transitions on b to states $q_2, q_3, \ldots, q_{n-1}$. This enforces b transitions to be like $q_0 \xrightarrow{b} q_0, q_1 \xrightarrow{b} q_2 \xrightarrow{b} \cdots \xrightarrow{b} q_{n-1} \xrightarrow{b} q_0$. Now we know for sure that $\{q_0, 1\}$ is not reached by reading a b so there must be some $x \neq 0$ that $q_x \xrightarrow{a} q_0$.

We claim that transitions on a has to be a permutation in order to achieve all the subsets. Suppose for a contradiction that transitions on a do not form a permutation and let us see how we are unable to reach the subset $\{q_0, 0, 1, \ldots, n-1\}$ in such a case. Our subset is not reached by b, because b has no incoming transition to q_1, so we should try to reach it by a instead. If transitions on a do not form a permutation, then there is a state q_y that has at least two incoming transitions on a from at least two different states. Therefore there is a state q_j that has no incoming transtions on a. State q_j is not q_0 because we already said that there is some $q_x \xrightarrow{a} q_0$. Reading an a does not reach subset containing j, thus neither the subset $\{q_0, 0, 1, \ldots, n-1\}$. To conclude, transitions on a must form a permutation.

Now we know that to reach all the possible subsets we have to have a DFA as showed in Fig. 5 where the transitions on a form a permutation. We show that some subsets of the subset automaton for the square of this DFA that should be distinguishable are actually equivalent. For this reason we introduce the families of subsets $T_1, \ldots, T_{n-1}$, where family T_i consists of subsets $\{q_0, 0, i\} \cup S$ and $S \subseteq \{i+1, i+2, \ldots, n-1\}$. Moreover let $T_n = \{q_0, 0\}$. Using previously defined transitions on a we have

$$\{q_0, 0, 1\} \cup S \xrightarrow{a} \{q_1, 0, 1, 1 \cdot a\} \cup S \cdot a \approx \{q_0, 0, 1, 1 \cdot a\} \cup S \cdot a \in T_1,$$

$$\{q_0, 0, i\} \cup S \xrightarrow{a} \{q_1, 0, 1, i \cdot a\} \cup S \cdot a \approx \{q_0, 0, 1, i \cdot a\} \cup S \cdot a \in T_1,$$

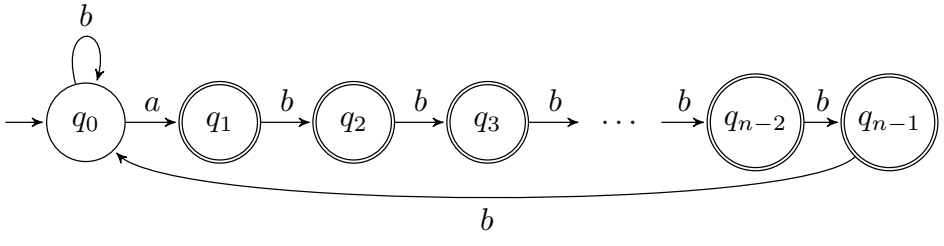

Fig. 5. Transitions on b in a binary witness.

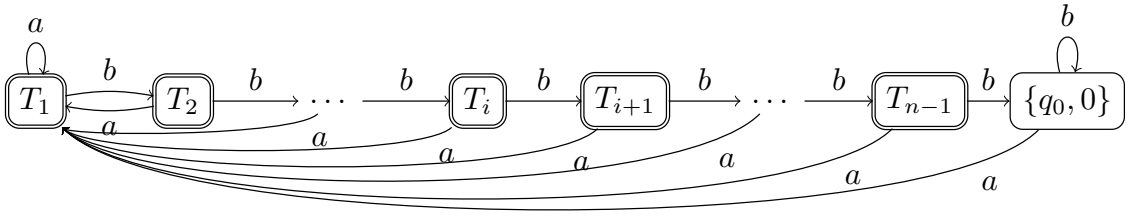

Fig. 6. Families of subsets and transitions between them.

that is, reading a from every subset in T_i results in a subset in T_1; here $\approx$ stands for the equivalence between states; recall that in the begining of the proof of Lemma 9 we showed that two distinct subsets $\{q_i\} \cup S$ and $\{q_j\} \cup S$, where $\{i, j\} \subseteq S$ are equivalent for the case when the only non final state is the initial state.

Now consider transitions on b:

$$\{q_0, 0, 1\} \cup S \xrightarrow{b} \{q_0, 0, 2\} \cup S \cdot b \in T_2,$$
$$\{q_0, 0, i\} \cup S \xrightarrow{b} \{q_0, 0, i+1\} \cup S \cdot b \in T_{i+1},$$
$$\{q_0, 0, n-1\} \xrightarrow{b} \{q_0, 0\}, \quad \{q_0, 0\} \xrightarrow{b} \{q_0, 0\}.$$

This means that reading b from every subset in T_i results in a subset in T_{i+1} except for T_n that stays in itself. This is illustrated in Fig. 6.

Observe that all the elements from one family upon reading any word w would end up in the accepting state from some of the families $T_1, \ldots, T_{n-1}$ or in the only rejecting state $\{q_0, 0\}$. That means that the elements of the family T_i are equivalent. It remains to show that at least one of these families has more than one reachable subset. Let us take family T_1:

$$\{q_0, 0\} \xrightarrow{a} \{q_1, 0, 1\} \xrightarrow{b^{x-1}} \{q_x, 0, x\} \xrightarrow{a} \{q_0, 0, 1\} \in T_1.$$

To reach another subset from T_1 we use the argument of the permutation formed by transitions on a. The state q_0 has an outgoing transition on a to the state q_1. If the state q_1 is returning to the q_0 on a then we have, where $q_y \xrightarrow{a} q_2$ and $y \geq 2$,

$$\{q_0, 0, 1\} \xrightarrow{b^{y-1}} \{q_0, 0, y\} \xrightarrow{a} \{q_1, 0, 1, 2\} \xrightarrow{a} \{q_0, 0, 1, 2 \cdot a\} \in T_1.$$

Otherwise, if $q_1 \xrightarrow{a} q_z$, where $2 \leq z \leq n-1$, supposing that $q_z \xrightarrow{a^k} q_0$ then we have

$$\{q_0, 0, 1\} \xrightarrow{a} \{q_1, 0, 1, z\} \xrightarrow{a} \{q_z, 0, 1, z, z \cdot a\}$$
$$\xrightarrow{a^k} \{q_0, 1, z, z \cdot a, z \cdot a^2, \ldots, z \cdot a^k\} \supseteq \{q_0, 0, 1, z\},$$

so the reached subset is from T_1. So far we showed that to reach the subsets we need to start from DFA as shown in Fig. 5 with a being a permutation. However in

such a case at least two reached subsets that should be distinguishable, as observed in Lemma 9, are equivalent. Our proof is complete. □

3.2. *Square on unary DFAs*

To complete the overview about the square operation on deterministic automata we should not forget unary alphabets. We refer to the paper by Rampersad [11] once again. Notice that the complexity of square in this case is exponentially smaller than in the binary case. To get the complexity of square (respectively power) in the unary case, Rampersad used the result on concatenation by Pighizzini and Shallit [10], Theorem 10, the proof of which is rather complicated. For the sake of completeness we provide a simple proof for square here.

We use Nicaud's notation for unary automata: We identify each state q with the smallest number i such that $q_0 \cdot a^i = q$. Given two integers n and ℓ with $0 \leq \ell \leq n-1$ and a set $F \subseteq \{0, 1, \ldots, n-1\}$, we denote by $A = (n, \ell, F)$ the n-state unary DFA A with the state set $Q = \{0, 1, \ldots, n-1\}$, in which $i \cdot a = i + 1$ if $0 \leq i \leq n-2$ and $n - 1 \cdot a = \ell$, and the set of final states is F.

Theorem 12 ([11], Theorems 3 and 4 with $k = 2$). *Let L be a unary language with $\mathrm{sc}(L) = n$. Then $\mathrm{sc}(L^2) \leq 2n - 1$ and the bound is tight.*

Proof. Let L be accepted by a unary DFA $A = (n, \ell, F)$. Let us show that L^2 is accepted by the DFA $A' = (2n - 1, n - 1 + \ell, \{i + j \mid i, j \in F'\})$ where

$$F' = \{i + j \mid 0 \leq i + j \leq 2n - 2 \text{ and } a^i, a^j \in L\}$$

Notice that both of A and A' have a loop of length $n - \ell$. It follows immediately from the construction that for every m with $0 \leq m \leq 2n - 2$, the state m is a final state of the DFAA' if and only if $a^m \in L^2$. We next show that for every m with $m \geq n - 1 + \ell$, the word a^m is in L^2 if and only if the word $a^{m+(n-\ell)}$ is in L^2.

If $m \geq n - 1 + \ell$ and $a^m \in L^2$, then $m = i + j$ where $a^i, a^j \in L$ and, without loss of generality, we must have $i \geq \ell$. It follows that $a^{i+(n-\ell)} \in L$, and therefore $a^{m+(n-\ell)} \in L^2$.

To prove the converse, let $m \geq n-1+\ell$ and $a^{m+(n-\ell)} \in L^2$. Then $m+(n-\ell) = i+j$ where $a^i, a^j \in L$, and, without loss of generality, we must have $i \geq n$. It follows that $a^{i-(n-\ell)} \in L$, and therefore $a^m \in L^2$.

The upper bound $2n - 1$ is met by the unary language $\{a^m \mid m \geq n - 1\}$. □

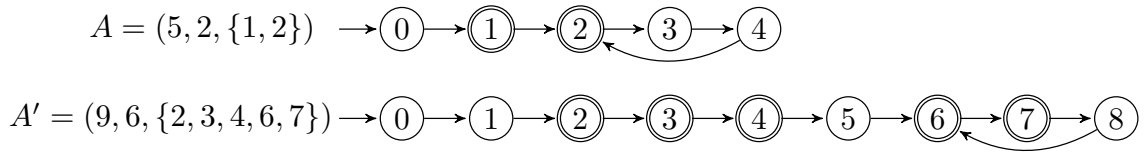

Fig. 7. The construction of a unary DFA for square; $n = 5, \ell = 2, F = \{1, 2\}$.

4. Square on Alternating and Boolean Automata

Fellah, Jürgensen, and Yu in Ref. [4], Theorem 9.3, showed that if a language K is accepted by an m-state AFA and a language L is accepted by an n-state AFA, then the language KL is accepted by an AFA of at most $2^m + n + 1$ states. It follows that $2^n + n + 1$ is an upper bound for the square on AFAs. Here we use our results from the previous section to prove tightness of this upper bound. For the square on BFAs, we get the tight upper bound $2^n + n$. Recall that $\mathrm{asc}(L)$ is the smallest number of states in any AFA for L and $\mathrm{bsc}(L)$ is defined analogously.

Theorem 13 (Square on AFAs). *Let $n \geq 2$. Let L be a regular language over Σ with $\mathrm{asc}(L) = n$. Then $\mathrm{asc}(L^2) \leq 2^n + n + 1$, and the bound is tight if $|\Sigma| \geq 2$.*

Proof. From given upper bound from [4], Theorem 9.3, we know that $\mathrm{asc}(L^2) \leq 2^n+n+1$. For tightness, let L^R be the language accepted by the DFA A defined in the proof of Theorem 7 with2^n states where half of the states are final, that is, $k = 2^{n-1}$. By Lemma 3, L is accepted by an AFA with n states. Using Theorem 7 we know that $\mathrm{sc}((L^R)^2) = 2^n 2^{2^n} - 2^{n-1}2^{2^n-1}$. By Corollary 2, $\mathrm{asc}(L^2) \geq \lceil \log(\mathrm{sc}((L^R)^2)) \rceil = 2^n + n$.

Suppose for a contradiction that L^2 is accepted by an AFA with $2^n + n$ states. By Lemma 1, the language $(L^2)^R$ is accepted by a 2^{2^n+n}- state DFA with 2^{2^n+n-1} final states. It follows immediately that the minimal DFA for $(L^2)^R$ has at most 2^{2^n+n-1} final states. However, the minimal DFA for the language $(L^2)^R = (L^R)^2$ has $2^n 2^{2^n} - 2^{n-1}2^{2^n-1} = 2^{n-1}2^{2^n} + 2^{n-1}2^{2^n-1}$ states,where $2^{n-1}2^{2^{n-1}} + 2^{n-1}2^{2^{n-1}-1}$ of them are non-final—those $\{q_i\} \cup S$, where $S \subseteq \{q_0, q_1, \ldots, q_{2^{n-1}-1}\}$. Thus the number of final states in the minimal DFA for $(L^2)^R$ is

$$2^{n-1}(2^{2^n} + 2^{2^n-1}) - 2^{n-1}(2^{2^{n-1}} + 2^{2^{n-1}-1}),$$

and since $n \geq 2$, we get

$$\begin{aligned} &2^{n-1}(2^{2^n} + 2^{2^n-1}) - 2^{n-1}(2^{2^{n-1}} + 2^{2^{n-1}-1}) \\ &\quad = 2^{2^n}2^{n-1}\left(1 + \frac{1}{2} - \frac{1}{2^{2^{n-1}}} - \frac{1}{2^{2^{n-1}+1}}\right) \\ &\quad > 2^{2^n+n-1}\left(1 + \frac{1}{2} - \frac{1}{4} - \frac{1}{4}\right) = 2^{2^n+n-1}. \end{aligned}$$

Hence the minimal DFA for $(L^2)^R$ has more than 2^{2^n+n-1} final states, a contradiction. It follows that $\mathrm{asc}(L^2) \geq 2^n + n + 1$. □

Theorem 14 (Square on BFAs). *Let $n \geq 2$. Let L be a regular language over Σ with $\mathrm{bsc}(L) = n$. Then $\mathrm{bsc}(L^2) \leq 2^n + n$, and the bound is tight if $|\Sigma| \geq 2$.*

Proof. The upper bound follows from the upper bound $2^m + n$ on the complexity of the concatenation operation on BFAs [7], Theorem 4. Let L^R be a language accepted by DFA A from Fig. 1 with 2^n states and one final state. By Lemma 3,

L is accepted by an n-state BFA. We are able to determine the state complexity of $(L^R)^2$ using Theorem 7: $\mathrm{sc}\big((L^R)^2\big) = 2^n \cdot 2^{2^n} - 2^{2^n-1}$. By Corollary 2,

$$\mathrm{bsc}(L^2) \geq \lceil \log\big(2^n \cdot 2^{2^n} - 2^{2^n-1}\big) \rceil = 2^n + n. \qquad \square$$

The next result shows that the binary alphabet in the two theorems above cannot be decreased to unary.

Theorem 15 (Square on unary BFAs). *Let $n \geq 2$. Let L be a unary regular language with* $\mathrm{bsc}(L) = n$. *Then* $\mathrm{bsc}(L^2) \leq n + 1$.

Proof. If L is accepted by an n-state BFA, then the language L^R is accepted by a 2^n-state DFA. Since L is unary, we have $L^R = L$. By Theorem 12 the language L^2 is accepted by a DFA with at most $2^{n+1} - 1$ states. It follows that L^2 is accepted by a BFA with at most $n + 1$ states. $\square$

5. Conclusions

We studied the state complexity of the square of languages represented by deterministic, alternating, and Boolean finite automata. First, for each k such that $1 \leq k \leq n - 2$, we showed that the upper bound $n2^n - k2^{n-1}$ on the square of languages represented by n-state DFAs with k final states is tight in the binary case. Then we analysed the case of $n - 1$ final states, where we proved that the bound $(2n+2)2^{n-2}$ cannot be met. We provided the tight upper bound $(n+2)2^{n-2}$ for the case when the initial state is final and we found a binary witness. When the initial state is the only non-final state, we obtained the upper bound $(n + 3)2^{n-2}$ with a ternary witness. In the binary case we proved that the tight upper bound is $(n + 3)2^{n-2} - 1$. Since the complexity of the square on unary alphabet is $2n - 1$, the binary alphabet is always optimal in our witness examples.

Finally, we used our results on the square on deterministic finate automata to describe binary witness languages meeting the upper bounds$2^n + n + 1$ and 2^n+n for square on alternating and Boolean finite automata,respectively. We proved that the binary alphabet is again optimal. Our results can be extended for the concatenation operation just by concatenatingtwo of our automata with different number of states. This provides an alternative solution for the open problem stated by Fellah, Jürgensen, and Yu in Ref. [4].

There remain a few unanswered questions. For example, the exact complexity of the square operation on unary alternating and Boolean automata; the state complexity of the power on alternating and Boolean automata, resolved for deterministic automata, e.g. by Domaratzki and Okhotin[3], and of course the range of possible complexities for square.

Acknowledgments

This research is supported by grant VEGA 2/0084/15 and grant APVV-15-0091.

References

[1] J. A. Brzozowski and E. L. Leiss, On equations for regular languages, finite automata, and sequential networks, *Theoretical Computer Science* **10** (1980) 19–35.

[2] K. Čevorová, G. Jirásková and I. Krajňáková, On the square of regular languages, *CIAA 2014*, eds. M. Holzer and M. Kutrib, *LNCS* **8587**, (Springer, 2014), pp. 136–147.

[3] M. Domaratzki and A. Okhotin, State complexity of power, *Theoretical Computer Science* **410**(24–25) (2009) 2377–2392.

[4] A. Fellah, H. Jürgensen and S. Yu, Constructions for alternating finite automata, *International Journal of Computer Mathematics* **35**(1–4) (1990) 117–132.

[5] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages and Computation* (Addison-Wesley, 1979).

[6] M. Hospodár and G. Jirásková, Concatenation on deterministic and alternating automata, *NCMA 2016*, eds. H. Bordihn, R. Freund, B. Nagy and G. Vaszil, Vol. 321 (Österreichische Computer Gesellschaft, 2016), pp. 179–194.

[7] G. Jirásková, Descriptional complexity of operations on alternating and Boolean automata, *CSR 2012*, eds. E. A. Hirsch, J. Karhumäki, A. Lepistö and M. Prilutskii, *LNCS* **7353**, (Springer, 2012), pp. 196–204.

[8] E. L. Leiss, Succint representation of regular languages by Boolean automata, *Theoretical Computer Science* **13** (1981) 323–330.

[9] A. N. Maslov, Estimates of the number of states of finite automata, *Soviet Mathematics Doklady* **11**(5) (1970) 1373–1375.

[10] G. Pighizzini and J. Shallit, Unary language operations, state complexity and Jacobsthal's function, *International Journal of Foundations of Computer Science* **13**(1) (2002) 145–159.

[11] N. Rampersad, The state complexity of L^2 and L^k, *Information Processing Letters* **98**(6) (2006) 231–234.

[12] M. Sipser, *Introduction to the Theory of Computation* (Cengage Learning, 2012).

[13] S. Yu, *Handbook of Formal Languages: Volume 1 Word, Language, Grammar* (Springer, Berlin, Heidelberg, 1997), Berlin, Heidelberg, ch. Regular Languages, pp. 41–110.

[14] S. Yu, Q. Zhuang and K. Salomaa, The state complexities of some basic operations on regular languages, *Theoretical Computer Science* **125**(2) (1994) 315–328.

Appendix [B]

Michal Hospodár, Galina Jirásková, Ivana Krajňáková:
Operations on Boolean and alternating finite automata.
In: Fedor V. Fomin, Vladimir V. Podolskii (eds.): *Computer Science – Theory and Applications – 13th International Computer Science Symposium in Russia, CSR 2018, Moscow, Russia, June 6–10, 2018, Proceedings.*
Lecture Notes in Computer Science, vol. 10846, pp. 181–193, Springer (2018). ISSN: 0302-9743, ISBN: 978-3-319-90529-7, DOI: 10.1007/978-3-319-90530-3_16

Operations on Boolean and Alternating Finite Automata

Michal Hospodár, Galina Jirásková, and Ivana Krajňáková( )

Mathematical Institute, Slovak Academy of Sciences,
Grešákova 6, 040 01 Košice, Slovakia
hosmich@gmail.com, {jiraskov,krajnakova}@saske.sk

Abstract. We investigate the descriptional complexity of basic regular operations on languages represented by Boolean and alternating finite automata. In particular, we consider the operations of difference, symmetric difference, star, reversal, left quotient, and right quotient, and get tight upper bounds $m+n, m+n, 2^n, 2^n, m$, and 2^m, respectively, for Boolean automata, and $m+n+1, m+n, 2^n, 2^n, m+1$, and 2^m+1, respectively, for alternating finite automata. To describe witnesses for symmetric difference, we use a ternary alphabet. All the remaining witnesses are defined over binary or unary alphabets that are shown to be optimal.

1 Introduction

The Boolean finite automata (BFAs) are generalization of nondeterministic finite automata (NFAs). In an NFA, the transition function maps any pair of state and input symbol to a subset of states. This subset can be viewed as disjunction of its states. We obtain a BFA by considering other Boolean functions on states as a result of the transition function. Alternating finite automata (AFAs) start from the only one initial state, wheares Boolean automata may start their computation in any Boolean function designated as the initial function.

Boolean automata recognize the class of regular languages [2,4]. Every n-state Boolean automaton can be simulated by 2^{2^n}-state deterministic finite automaton (DFA), or by (2^n+1)-state NFA, and both upper bounds are tight already in the binary case [2,10].

Some of the constructions and upper bounds for elementary operations on alternating automata were introduced in [5]. The upper bound 2^m+n+1 for concatenation from [5] has been shown to be tight in [8]. Detailed results for the square on alternating and Boolean automata can be found in [12]. Tight upper bounds for union and intersection were shown in [10]. For star and reversal, the upper and lower bound provided in [10] differed by one.

Research supported by grant VEGA 2/0084/15 and grant APVV-15-0091. This work was conducted as a part of PhD study of Michal Hospodár and Ivana Krajňáková at the Faculty of Mathematics, Physics and Informatics of the Comenius University.

F. V. Fomin and V. V. Podolskii (Eds.): CSR 2018, LNCS 10846, pp. 181–193, 2018.
https://doi.org/10.1007/978-3-319-90530-3_16

In this paper we continue the study of the operational complexity on Boolean and alternating finite automata. We improve the results on star and reversal from [10] and provide exact complexity of these two operations. We also examine other regular operations: complementation, difference, symmetric difference, left and right quotient on both Boolean and alternating automata. We get the exact complexity for each operation on both BFAs and AFAs. All our witness languages are defined over a small fixed alphabet which is optimal in most of the cases.

2 Preliminaries

Let Σ be a finite alphabet of symbols. Then Σ^* denotes the set of words over Σ including the empty word ε. A language is any subset of Σ^*. The cardinality of a finite set A is denoted by $|A|$, and its power-set by 2^A. The reader may refer to [7,17,18] for details.

A *nondeterministic finite automaton* (NFA) is a quintuple $A = (Q, \Sigma, \circ, I, F)$, where Q is a finite set of states, Σ is a finite non-empty alphabet, $\circ : Q \times \Sigma \to 2^Q$ is the transition function which is naturally extended to the domain $2^Q \times \Sigma^*$, $I \subseteq Q$ is the set of initial states, and $F \subseteq Q$ is the set of final states. The *language accepted by* A is the set $L(A) = \{w \in \Sigma^* \mid I \circ w \cap F \neq \emptyset\}$. For a symbol a, we say that (p, a, q) is a transition in NFA A if $q \in p \circ a$, and the state q has an in-transition on a. For a word w, we write $p \xrightarrow{w} q$ if $q \in p \circ w$.

An NFA A is *deterministic* (DFA) if $|I| = 1$ and $|q \circ a| = 1$ for each q in Q and each a in Σ; so all DFAs in this paper are assumed to be complete. We write $p \cdot a = q$ instead of $p \circ a = \{q\}$ in such a case. The *state complexity* of a regular language L, $\mathrm{sc}(L)$, is the smallest number of states in any DFA for L. A state q of a DFA is called *sink state* if $q \cdot a = q$ for each a in Σ.

For unary DFAs we use the Nicaud's notation [15]. For two integers ℓ and n such that $0 \leq \ell \leq n-1$ and a subset F of $\{0, \ldots, n-1\}$, $A = (n, \ell, F)$ is the unary automaton whose set of states is $Q = \{0, \ldots, n-1\}$ and the transition function is given by $q \cdot a = q + 1$ if $0 \leq q \leq n-2$ and $(n-1) \cdot a = \ell$. The initial state of this automaton is 0 and its set of final states is F.

Every NFA $A = (Q, \Sigma, \circ, I, F)$ can be converted to an equivalent DFA $\mathcal{D}(A) = (2^Q, \Sigma, \cdot, I, F')$, where $S \cdot a = S \circ a$ for each S in 2^Q and a in Σ and $F' = \{R \in 2^Q \mid R \cap F \neq \emptyset\}$. We call the DFA $\mathcal{D}(A)$ the *subset automaton* of the NFA A. The subset automaton may not be minimal since some of its states may be unreachable or equivalent to other states.

To prove distinguishability of the states of the subset automaton, the following notions and observations are useful. A state q of an NFA A is called *uniquely distinguishable* if there is a word w which is accepted by A from and only from the state q, that is $p \circ w \cap F \neq \emptyset$ if and only if $p = q$. A transition (p, a, q) is called a *unique in-transition* if there is no state r such that $r \neq p$ and (r, a, q) is a transition in A. A state q is *uniquely reachable* from a state p if there exists a sequence of unique in-transitions (q_i, a, q_{i+1}) for $i = 0, 1, \ldots, k$ such that $q_0 = p$ and $q_{k+1} = q$.

Proposition 1 [1, Propositions 14 and 15]**.** *Let A be an* NFA *and $\mathcal{D}(A)$ be the corresponding subset automaton.*

(a) If two subsets of $\mathcal{D}(A)$ differ in a uniquely distinguishable state of A, then they are distinguishable.
(b) If a state q of A is uniquely distinguishable and uniquely reachable from a state p, then the state p is uniquely distinguishable as well.
(c) If there is a uniquely distinguishable state of A which is uniquely reachable from any other state of A, then every state of A is uniquely distinguishable.
(d) If every state of A is uniquely distinguishable, then the subset automaton $\mathcal{D}(A)$ does not have equivalent states.

□

Let K and L be languages over an alphabet Σ. The *difference* and *symmetric difference* of K and L are the languages $K\backslash L = \{w \in K \mid w \notin L\}$ and $K \oplus L = \{w \in K \mid w \notin L\} \cup \{w \in L \mid w \notin K\}$, respectively. If languages K and L are accepted by DFAs $A = (Q_A, \Sigma, \cdot_A, s_A, F_A)$ and $B = (Q_B, \Sigma, \cdot_B, s_B, F_B)$, then the language $K \cap L$ is accepted by the *product automaton* $A \times B = (Q_A \times Q_B, \Sigma, \cdot, (s_A, s_B), F_A \times F_B)$ where $(p, q)\cdot a = (p\cdot_A a, q\cdot_B a)$. For the remaining Boolean operations we only need to change the set of final states in the product automaton. For union, difference, symmetric difference the set of final states is $(F_A \times Q_B) \cup (Q_A \times F_B)$, $F_A \times (Q_B\backslash F_B)$, $(F_A \times (Q_B\backslash F_B)) \cup ((Q_A\backslash F_A) \times F_B)$, respectively.

The reverse of a word is defined as $\varepsilon^R = \varepsilon$ and $(wa)^R = aw^R$ for each symbol a and word w. The reverse of a language L is the language $L^R = \{w^R \mid w \in L\}$. The reverse of an NFA A is an NFA A^R obtained from A by reversing all the transitions and by swapping the roles of initial and final states. The NFA A^R recognizes the reverse of $L(A)$.

The *concatenation of* K and L is the language $KL = \{uv \mid u \in K \text{ and } v \in L\}$. The *square* of a language L is the language $L^2 = LL$. The *right quotient* of K by L is the language $KL^{-1} = \{x \in \Sigma^* \mid xy \in K \text{ for some } y \in L\}$. The *left quotient* of K by L is the language $L^{-1}K = \{x \in \Sigma^* \mid yx \in K \text{ for some } y \in L\}$.

A *Boolean finite automaton* (BFA) is a quintuple $A = (Q, \Sigma, \delta, g_s, F)$, where Q is a finite non-empty set of states, $Q = \{q_1, \ldots, q_n\}$, Σ is an input alphabet, δ is the transition function that maps $Q \times \Sigma$ into the set $\mathcal{B}_n$ of Boolean functions with variables $\{q_1, \ldots, q_n\}$, $g_s \in \mathcal{B}_n$ is the initial Boolean function, and $F \subseteq Q$ is the set of final states. The transition function δ can be extended to the domain $\mathcal{B}_n \times \Sigma^*$ as follows: For all g in $\mathcal{B}_n$, a in Σ, and w in Σ^*, we have $\delta(g, \varepsilon) = g$; if $g = g(q_1, \ldots, q_n)$, then $\delta(g, a) = g(\delta(q_1, a), \ldots, \delta(q_n, a))$; $\delta(g, wa) = \delta(\delta(g, w), a)$. Next, let $f = (f_1, \ldots, f_n)$ be the Boolean vector with $f_i = 1$ iff $q_i \in F$. The language accepted by the BFA A is the set $L(A) = \{w \in \Sigma^* \mid \delta(g_s, w)(f) = 1\}$.

A Boolean finite automaton is called *alternating* (AFA) if the initial function is a projection $g(q_1, \ldots, q_n) = q_i$. For details, we refer to [2,5,10,13,17,18].

The *Boolean (alternating) state complexity* of L, $\mathrm{bsc}(L)(\mathrm{asc}(L))$, is the smallest number of states in any BFA (AFA) for L. It is known that a language L is accepted by an n-state BFA (AFA) if and only if the language L^R is accepted

by an 2^n-state DFA (with 2^{n-1} final states). Since this is the crucial observation used later in the paper, we state it in the next two lemmas and provide proof ideas here.

Lemma 2 (cf. [5, **Theorem** 4.1, **Corollary** 4.2] **and** [10, **Lemma** 1]**).** *Let L be a language accepted by an n-state* BFA *(*AFA*). Then the reversal L^R is accepted by a* DFA *of 2^n states (of which 2^{n-1} are final).*

Proof (Proof Idea). Let $A = (\{q_1, q_2, \ldots, q_n\}, \Sigma, \delta, g_s, F)$ be an n-state BFA for L. Construct a 2^n-state NFA $A' = (\{0,1\}^n, \Sigma, \delta', S, \{f\})$, where

- for every $u = (u_1 \ldots, u_n) \in \{0,1\}^n$ and every $a \in \Sigma$, $\delta'(u,a) = \{u' \in \{0,1\}^n \mid \delta(q_{i,a})(u') = u_i \text{ for } i = 1, \ldots, n\}$;
- $S = \{(b_1, \ldots, b_n) \in \{0,1\}^n \mid g_s(b_1, \ldots, b_n) = 1\}$;
- $f = (f_1, \ldots, f_n) \in \{0,1\}^n$ with $f_i = 1$ iff $q_i \in F$.

Then $L(A) = L(A')$ and $(A')^R$ is deterministic. Moreover if A is an AFA then A' has 2^{n-1} initial states. It follows that L^R is accepted by a DFA with 2^n states, of which 2^{n-1} are final if A is an AFA. □

Lemma 3 (cf. [10, **Lemma** 2]**).** *Let L^R be accepted by a* DFA *A of 2^n states (of which 2^{n-1} are final). Then L is accepted by an n-state* BFA *(*AFA*).*

Proof (Proof Idea). Consider 2^n-state NFA A^R for L which has exactly one final state and the set of initial states S (and $|S| = 2^{n-1}$). Let the state set Q of A^R be $\{0, 1, \ldots, 2^n - 1\}$ with final state k and the initial set S ($S = \{2^{n-1}, \ldots, 2^n - 1\}$). Let δ be the transition function of A^R. Moreover, for every $a \in \Sigma$ and for every $i \in Q$, there is exactly one state j such that j goes to i on a in A^R. For a state $i \in Q$, let $\text{bin}(i) = (b_1, \ldots, b_n)$ be the binary n-tuple such that $b_1 b_2 \cdots b_n$ is the binary notation of i on n digits with leading zeros if necessary.

Let us define an n-state BFA $A' = (Q', \Sigma, \delta', g_s, F')$, where $Q' = \{q_1, \ldots, q_n\}$, $F' = \{q_\ell \mid \text{bin}(k)_\ell = 1\}$, and $g_s(\text{bin}(i)) = 1$ iff $i \in S$ ($g_s = q_1$). We define δ' to suffice the condition: for each i in Q and a in Σ, $(\delta'(q_1, a), \ldots, \delta'(q_n, a))(\text{bin}(i)) = \text{bin}(j)$ where $i \in \delta(j, a)$. Then $L(A') = L(A^R)$. □

As a corollary of the previous two lemmas, we get the following results.

Corollary 4. *If L is a regular language, then* $\text{bsc}(L) \geq \lceil \log(\text{sc}(L^R)) \rceil$ *and* $\text{asc}(L) \geq \lceil \log(\text{sc}(L^R)) \rceil$. □

Corollary 5. *Let L be a unary language. Then L is accepted by an n-state* BFA *(*AFA*) if and only if L is accepted by a 2^n-state* DFA *(with 2^{n-1} final states).* □

Now we prove several propositions which we use later in our paper.

Proposition 6. *If L is accepted by an n-state* BFA*, then L is accepted by an $(n+1)$-state* AFA*.*

Proof. Let a language L be accepted by an n-state BFA $(Q, \Sigma, \delta, g, F)$. Let $A = (Q \cup \{s\}, \Sigma, \delta', s, F')$ where $s \notin Q$, $\delta'(q, a) = \delta(q, a)$ if $q \in Q$ and $\delta'(q, a) = \delta(g, a)$ if $q = s$; $F' = F$ if $\varepsilon \notin L$ and $F' = F \cup \{s\}$ if $\varepsilon \in L$. Then A is an $(n+1)$-state AFA for L. □

Proposition 7. *Let K and L be languages over Σ. Then*

(a) $(KL^{-1})^R = (L^R)^{-1}K^R$;
(b) $(L^{-1}K)^R = K^R(L^R)^{-1}$.

□

Proposition 8. *Let a non-empty language L be accepted by an n-state* DFA. *Then L^* is accepted by a 2^n-state* DFA *with half of the states final.*

Proof. Let L be accepted by an n-state DFA $A = (Q, \Sigma, \cdot, s, F)$. If the initial state is the only final state in A, then $L^* = L$, and we may add final and non-final unreachable sink states to get the desired automaton. Otherwise there is a final state q_F such that $q_F \neq s$. Construct an NFA N for L^* from A as follows:

(a) add the transition (q, a, s) whenever $q \cdot a \in F$;
(b) add a new initial and final state q_0;
(c) the initial states of N are s and q_0 and the set of final states is $F \cup \{q_0\}$.

In the corresponding subset automaton $\mathcal{D}(N)$ the initial subset is $\{q_0, s\}$ and any other reachable subset S is a non-empty subset of Q such that $S \cap F \neq \emptyset$ implies $s \in S$. By the construction above every set S such that $q_F \in S$ and $s \notin S$ is unreachable. That means that there are at most $1 + 2^n - 1 - 2^{n-2} = \frac{3}{4}2^n$ reachable sets in $\mathcal{D}(N)$. Let us show that in the minimal DFA for L^* the number of non-final states as well as the number of final states is at most 2^{n-1}. The non-final subsets in $\mathcal{D}(N)$ must not contain the state q_F, so there are at most 2^{n-1} of them. Next the initial subset $\{q_0, s\}$ is final and any other final subset must contain the state s. This gives at most $1 + 2^{n-1}$ subsets. However, if $s \in F$ then $\{q_0, s\}$ and $\{s\}$ are equivalent, and if $s \notin F$ then $\{s\}$ is non-final. Therefore the minimal DFA for L^* has at most 2^{n-1} final states. To obtain 2^n-state DFA we may add some unreachable sink states. Since the number of final and non-final states are at most 2^{n-1} it is possible to achieve that exactly half of the states would be final and the other half non-final in the resulting 2^n-state DFA. □

Proposition 9. *Let $m, n \geq 2$ and $\gcd(m, n) = 1$. Let K and L be unary regular languages accepted by deterministic finite automata $A = (m, 0, \{0\})$ and $B = (n, 0, \{1, 2, \ldots, n-1\})$, respectively. Then $\mathrm{sc}(K \oplus L) = mn$.*

Proof. Since symmetric difference is a commutative operation, we may assume that $m < n$. Denote $Q_A = \{0, 1, \ldots, m-1\}$, $Q_B = \{0, 1, \ldots, n-1\}$. Consider the product automaton $A \times B = (Q_A \times Q_B, \{a\}, \cdot, (0, 0), F)$ where the set of final states is $F = \{(0, 0)\} \cup \{1, 2, \ldots, m-1\} \times \{1, 2, \ldots, n-1\}$. Since $\gcd(m, n) = 1$, every state of the product automaton is reachable. To prove distinguishability, let p and q be two distinct states of the product automaton. Then there is an integer $k \geq 0$ such that $p \cdot a^k = (m-1, 0)$ and $q \cdot a^k = q'$ where $q' \neq (m-1, 0)$. We have three cases:

(a) $q' \in F$. Then a^k distinguishes p and q since $(m-1,0) \notin F$.
(b) $q' = (0, n-1)$. Then $a^k a^m$ distinguishes p and q since
$p \xrightarrow{a^k} (m-1,0) \xrightarrow{a^m} (m-1,m) \in F$,
$q \xrightarrow{a^k} (0,n-1) \xrightarrow{a} (1,0) \xrightarrow{a^{m-1}} (0,m-1) \notin F$; recall that $m < n$.
(c) q' is a non-final state different from $(0,n-1)$. Then $a^k a$ distinguishes p and q since $(m-1,0) \cdot a \notin F$ and $q' \cdot a \in F$.

Hence all the states of the product automaton are reachable and pairwise distinguishable. This means that $\mathrm{sc}(K \oplus L) = mn$. □

3 Operations on Boolean and Alternating Automata

In this section we investigate the descriptional complexity of basic regular operations on languages represented by Boolean and alternating automata. We start with the complementation operation and we show that a language and its complement have the same complexity.

Theorem 10 (Complementation). *Let L be a regular language. Then we have* $\mathrm{asc}(L) = \mathrm{asc}(L^c)$ *and* $\mathrm{bsc}(L) = \mathrm{bsc}(L^c)$.

Proof. Let L be accepted by a minimal n-state BFA (AFA). Then the language L^R is accepted by a 2^n-state DFA (with half of the states final) by Lemma 2. This means that $(L^R)^c$ is accepted by a 2^n state DFA (with half of the states final) since we only interchange final and non-final states in the DFA for L^R. Next $(L^R)^c = (L^c)^R$. Therefore L^c is accepted by an n-state BFA (AFA) by Lemma 3. Hence $\mathrm{asc}(L^c) \leq n$ and $\mathrm{bsc}(L^c) \leq n$. Moreover we cannot have $\mathrm{asc}(L^c) < n$ because after another complementation we would get $\mathrm{asc}(L) < n$. The argument for $\mathrm{bsc}(L^c)$ is the same. □

We continue with the star operation. We improve the results from [10, Theorems 8, 9] where upper and lower bounds differed by one. We get tight upper bound 2^n for both BFAs and AFAs as a corollary of the next theorem.

Theorem 11 (Star). *Let $n \geq 2$.*

(a) If L is accepted by an n-state BFA*, then L^* is accepted by a 2^n-state* AFA.
(b) There exists a language L accepted by an n-state AFA *such that every* BFA *for L^* has at least 2^n states.*

Proof

(a) Let L be accepted by an n-state BFA. Then L^R is accepted by a 2^n-state DFA by Lemma 2. By Propostion 8, $(L^R)^*$ is accepted by a 2^{2^n}-state DFA with half of the states final. Next $(L^R)^* = (L^*)^R$. This means that L^* is accepted by a 2^n-state AFA by Lemma 3.

(b) Let L^R be the Palmovský's witness language for star [16] with 2^n states and 2^{n-1} final states shown in Fig. 1. By Lemma 3 the language L is accepted by an n-state AFA. By [16, Proof of Theorem 4.4] $\mathrm{sc}((L^R)^*) = 2^{2^n-1} + 2^{2^n-1-2^{n-1}} = 2^{2^n-1}(1+2^{-2^{n-1}})$. Since $(L^R)^* = (L^*)^R$ we get $\mathrm{bsc}(L^*) \geq \lceil \log(\mathrm{sc}((L^*)^R)) \rceil = 2^n$ by Corollary 4. □

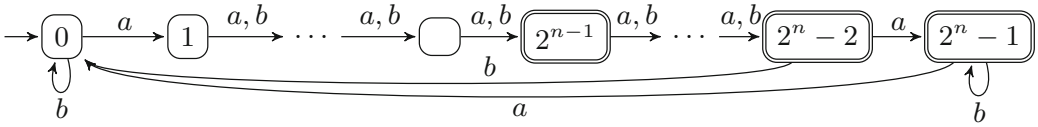

Fig. 1. The reverse of a binary witness for star on BFAs and AFAs.

In what follows we use Lemmas 2, 3 and Corollary 4 without citing them again and again. The next theorem provides tight upper bounds on the complexity of difference, symmetric difference, reversal, and right and left quotient on languages represented by Boolean finite automata.

Theorem 12 (Operations on BFAs). *Let K and L be (regular) languages over an alphabet Σ accepted by an m-state and n-state* BFA*, respectively. Then*

(a) $\mathrm{bsc}(K \setminus L) \leq m+n$*, and the bound is tight if* $|\Sigma| \geq 2$*;*
(b) $\mathrm{bsc}(K \oplus L) \leq m+n$*, and the bound is tight if* $|\Sigma| \geq 3$*;*
(c) $\mathrm{bsc}(L^R) \leq 2^n$*, and the bound is tight if* $|\Sigma| \geq 2$*;*
(d) $\mathrm{bsc}(KL^{-1}) \leq 2^m$*, and the bound is tight if* $|\Sigma| \geq 2$*;*
(e) $\mathrm{bsc}(L^{-1}K) \leq m$*, and the bound is tight if* $|\Sigma| \geq 1$*.*

Proof. Let $A = (Q_A, \Sigma, \delta_A, g_A, F_A)$ be an m-state BFA for the language K and $B = (Q_B, \Sigma, \delta_B, g_B, F_B)$ be an n-state BFA for L with $Q_A \cap Q_B = \emptyset$.

(a) The language $K \setminus L$ is accepted by BFA $(Q_A \cup Q_B, \Sigma, \delta, g_A \wedge \overline{g_B}, F_A \cup F_B)$, where $\delta = \delta_A$ on Q_A and $\delta = \delta_B$ on Q_B. Thus $\mathrm{bsc}(K \setminus L) \leq m+n$. For tightness, let K and L be binary witness languages for intersection on BFAs described in [10, Proof of Theorem 2]. Then K and L^c are witnesses for difference since $K \setminus L^c = K \cap L$.

(b) The symmetric difference $K \oplus L$ is accepted by BFA

$$(Q_A \cup Q_B, \Sigma, \delta, (g_A \wedge \overline{g_B}) \vee (\overline{g_A} \wedge g_B), F_A \cup F_B)$$

where $\delta = \delta_A$ on Q_A and $\delta = \delta_B$ on Q_B. Thus $\mathrm{bsc}(K \oplus L) \leq m+n$. For tightness, let K^R and L^R be the languages accepted by 2^m-state and 2^n-state DFAs with half of states final shown in Fig. 2. Then K and L are accepted by m-state and n-state BFAs. In the product automaton, each state (i, j) is reached by $a^i b^j$. Two (non-)final states are distinguished by c if they are in different quadrants and by a word in $a^* + b^*$ otherwise. So we get $\mathrm{sc}(K^R \oplus L^R) = 2^{m+n}$. Next $K^R \oplus L^R = (K \oplus L)^R$. Therefore $\mathrm{bsc}(K \oplus L) \geq m+n$.

(c) The language L^R is accepted by 2^n-state DFA, the special case of BFA. For tightness, let L^R be the Šebej's binary witness language for reversal [11] accepted by a DFA with 2^n states. Then L is accepted by an n-state BFA. By [11, Proof of Theorem 5] $\mathrm{sc}((L^R)^R) = 2^{2^n}$ and therefore $\mathrm{bsc}(L^R) \geq 2^n$.

(d) If K and L are accepted by an m-state and n-state BFA, respectively, then K^R and L^R are accepted by a 2^m-state and 2^n-state DFA, respectively. By Proposition 7 $(KL^{-1})^R = (L^R)^{-1}K^R$ and by [19, Theorem 4.1] $\mathrm{sc}((L^R)^{-1}K^R) \leq 2^{2^m} - 1$. It follows that $\mathrm{bsc}(KL^{-1}) \leq 2^m$. For tightness, let $L = \Sigma^*$ and K be the language accepted by the DFA shown in Fig. 3. Then $\mathrm{bsc}(K) \leq m$ and

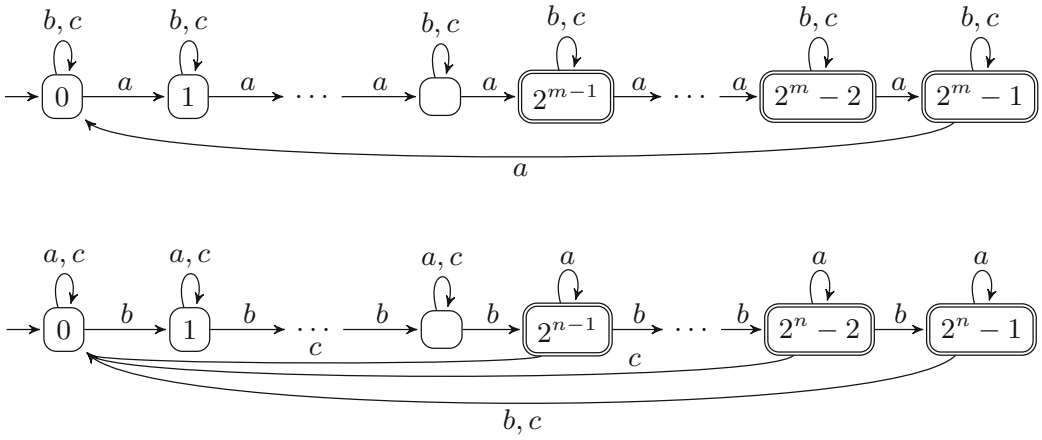

Fig. 2. The reverses of ternary witnesses for symmetric difference on BFAs.

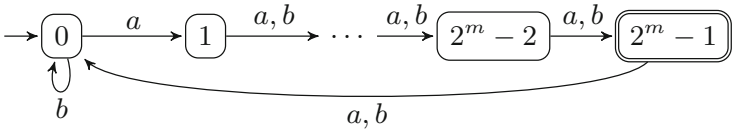

Fig. 3. The reverse of a binary witness for right quotient (by Σ^*) on BFAs.

$\mathrm{bsc}(L) \leq n$. Next $(KL^{-1})^R = (\Sigma^*)^{-1}K^R$ and by [19, Proof of Theorem 4.1] $\mathrm{sc}((\Sigma^*)^{-1}K^R) = 2^{2^m} - 1$. Therefore $\mathrm{bsc}(KL^{-1}) \geq 2^m$.

(e) Since $(L^{-1}K)^R = K^R(L^R)^{-1}$ and $\mathrm{sc}(K^R(L^R)^{-1}) \leq 2^m$ [19, p. 323], we get $\mathrm{bsc}(L^{-1}K) \leq m$. For tightness, let $K = \{a^i \mid 2^{m-1}-1 \leq i \leq 2^m-2\}$ and $L = a^*$. Then $\mathrm{bsc}(K) \leq m$ and $\mathrm{bsc}(L) \leq n$. Next $K^R(a^*)^{-1} = \{a^i \mid 0 \leq i \leq 2^m - 2\}$, so $\mathrm{sc}(K^R(a^*)) = 2^m$. Therefore $\mathrm{bsc}(L^{-1}K) \geq m$.

In the next theorem we study the complexities of same operations on languages represented by alternating finite automata. Note that while the complexities of intersection, union, and difference on AFAs exceed those on BFAs by one, the complexity of symmetric difference on AFAs and BFAs is the same.

Theorem 13 (Operations on AFAs). *Let K and L be (regular) languages over an alphabet Σ accepted by an m-state and n-state* AFA*, respectively. Then*

(a) $\mathrm{asc}(K \setminus L) \leq m + n + 1$*, and the bound is tight if* $|\Sigma| \geq 2$*;*
(b) $\mathrm{asc}(K \oplus L) \leq m + n$*, and the bound is tight if* $|\Sigma| \geq 3$*;*
(c) $\mathrm{asc}(L^R) \leq 2^n$*, and the bound is tight if* $|\Sigma| \geq 2$*;*
(d) $\mathrm{asc}(KL^{-1}) \leq 2^m + 1$*, and the bound is tight if* $|\Sigma| \geq 2$*;*
(e) $\mathrm{asc}(L^{-1}K) \leq m + 1$*, and the bound is tight if* $|\Sigma| \geq 1$*.*

Proof

(a) Since every AFA is BFA we get $\mathrm{bsc}(K \setminus L) \leq m + n$ by Theorem 12(a). Therefore $\mathrm{asc}(K \setminus L) \leq m+n+1$. For tightness, let K and L be the binary witness languages for intersection on AFAs described in [10, Proof of Theorem 3]. Then K and L^c are witnesses for difference since $\mathrm{asc}(K \setminus L^c) = \mathrm{asc}(K \cap L) = m + n + 1$.

(b) If K and L are accepted by m-state and n-state AFAs, then K^R and L^R are accepted by 2^m-state and 2^n-state DFAs with half of the states final. It follows that $K^R \oplus L^R$ is accepted by a product automaton of 2^{m+n} states and half of them are final. Therefore $K \oplus L$ is accepted by $(m+n)$-state AFA. For tightness, let K^R and L^R be the languages accepted by 2^m-state and 2^n-state DFAs with half of the states final shown in Fig. 2. Then K and L are accepted by m-state and n-state AFAs. As shown in Theorem 12(b) every BFA for $K \oplus L$ has at least $m+n$ states. Therefore $\mathrm{asc}(K \oplus L) \geq m+n$.

(c) If L is accepted by an n-state AFA, then L^R is accepted by 2^n-state DFA. Every DFA is a special case of AFA. Therefore AFA for language L^R has 2^n states. For tightness, let L^R be the language accepted by 2^n-state Šebej's automaton in which half of the states are final shown in Fig. 4. By [11, Proof of Theorem 5] we have $\mathrm{sc}((L^R)^R) = 2^{2^n}$; notice that any nontrivial number of final states does not matter since the subset automaton of NFA for $(L^R)^R$ does never have equivalent states [11, Proposition 3]. Hence $\mathrm{asc}(L^R) \geq 2^n$ by Corollary 4.

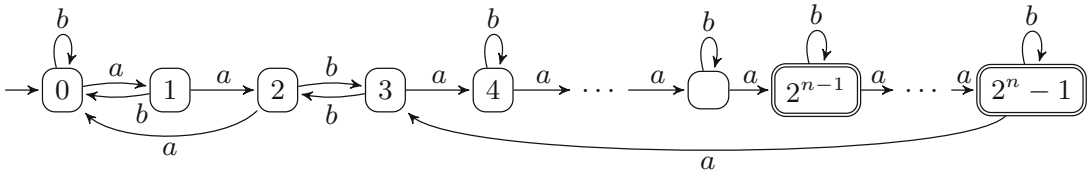

Fig. 4. The reverse of a binary witness for reversal on AFAs.

(d) By Propostion 6 and Theorem 12(d) we get $\mathrm{asc}(KL^{-1}) \leq \mathrm{bsc}(KL^{-1}) + 1 \leq 2^m + 1$. To prove tightness, let $L = \Sigma^*$ and K^R be the language accepted by the DFA A shown in Fig. 5 in which half of the states are final. Then $\mathrm{asc}(K) \leq m$ and $\mathrm{asc}(L) \leq n$. Next $(KL^{-1})^R = (\Sigma^*)^{-1}K^R$. Let us show that $\mathrm{sc}((\Sigma^*)^{-1}K^R) = 2^{2^m} - 1$. Construct an NFA N for $(\Sigma^*)^{-1}K^R$ from the DFA A by making all the states initial. Every non-empty subset in the corresponding subset automaton is reachable as it was shown in [19, Proof of Theorem 4.1]. To prove distinguishability, notice that the state 1 is uniquely distinguishable by the word b^{2^m-2}, and it is uniquely reachable in N from any other state through the unique in-transitions $2 \xrightarrow{a} 3 \xrightarrow{a} \cdots \xrightarrow{a} 2^m{-}1 \xrightarrow{a} 0 \xrightarrow{a} 1$. By Proposition 1, all states of the subset automaton are pairwise distinguishable. The number of final states in the subset automaton is $2^{2^m} - 2^{2^{m-1}}$, which is greater than 2^{2^m-1}. Therefore by Lemma 2 we get $\mathrm{asc}(KL^{-1}) \geq 2^m + 1$.

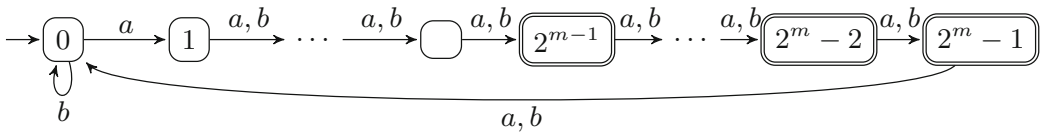

Fig. 5. The reverse of a binary witness for right quotient (by Σ^*) on AFAs.

(e) By Proposition 6 and Theorem 12(e) $\mathrm{asc}(L^{-1}K) \leq \mathrm{bsc}(L^{-1}K) + 1 \leq m + 1$. To get tightness, consider the same two languages as in Theorem 12(e). Notice that the minimal DFA for $K^R(a^*)^{-1}$ has more than 2^{m-1} final states. □

In the next theorem we study the complexity of basic regular operations on unary languages represented by Boolean finite automata.

Theorem 14 (Unary BFAs). *Let $n \geq 2$ and K and L be unary languages accepted by an m-state and n-state* BFA*, respectively. Then*

(a) $\mathrm{bsc}(K \cap L) \leq m + n$*, and the bound is tight if* $\gcd(m, n) = 1$*;*
(b) $\mathrm{bsc}(K \cup L) \leq m + n$*, and the bound is tight if* $\gcd(m, n) = 1$*;*
(c) $\mathrm{bsc}(K \setminus L) \leq m + n$*, and the bound is tight if* $\gcd(m, n) = 1$*;*
(d) $\mathrm{bsc}(K \oplus L) \leq m + n$*, and the bound is tight if* $\gcd(m, n) = 1$*;*
(e) $\mathrm{bsc}(L^R) = \mathrm{bsc}(L)$*;*
(f) $\mathrm{bsc}(L^*) \leq 2n$ *and the bound is tight;*
(g) $\mathrm{bsc}(KL^{-1}) \leq m$*, and the bound is tight.*

Proof. Let unary languages K and L be accepted by m-state and n-state BFA, respectively. Then K and L are accepted by 2^m-state and 2^n-state DFA, respectively, by Corollary 5, and the languages $K \cap L$, $K \cup L$, $K \setminus L$, $K \oplus L$ are accepted by a $2^m 2^n$-state product automaton. This gives upper bounds $m + n$ in cases (a)–(d). To prove tightness for intersection, let $K = (a^{2^m})^*$ and $L = (a^{2^n-1})^*$. Then K and L are accepted by a 2^m-state and 2^n-state DFA, respectively, so by an m-state and n-state BFA, respectively. Since $\gcd(2^m, 2^n - 1) = 1$, we have $\mathrm{sc}(K \cap L) = 2^m(2^n - 1)$. This means that $\mathrm{bsc}(K \cap L) \geq \lceil \log(2^m(2^n - 1)) \rceil = m + n$. For union, we may use the languages K^c and L^c, since $K^c \cup L^c = (K \cap L)^c$ and a language and its complement have the same Boolean state complexity. Similarly, for difference we use the languages K and L^c. For symmetric difference, let us consider unary languages K and L accepted by automata $A = (2^m, 0, \{0\})$ and $B = (2^n - 1, 0, \{1, 2, \ldots, 2^n - 2\})$. By Proposition 9 $\mathrm{sc}(K \oplus L) = 2^m(2^n - 1)$. It follows that $\mathrm{bsc}(K \oplus L) \geq \lceil \log(2^m(2^n - 1)) \rceil = m + n$.

(e) The equality follows from the fact that $L = L^R$ in the unary case.

(f) The state complexity of the star operation in the unary case is $(n-1)^2+1$ [3,19]. If a unary language L is accepted by an n-state BFA then L is accepted by a 2^n-state DFA. This means that L^* is accepted by a DFA of at most $(2^n-1)^2+1$ states, so by a DFA of at most 2^{2n} states. Therefore $\mathrm{bsc}(L^*) \leq 2n$. For tightness, let L be the unary language accepted by the DFA $(2^n, 0, \{2^n - 1\})$ meeting the upper bound for star [19, Theorem 5.3]. Then L is accepted by an n-state BFA and $\mathrm{bsc}(L^*) \geq \lceil \log(\mathrm{sc}(L^*)) \rceil = \lceil \log((2^n - 1)^2 + 1) \rceil = 2n$.

(g) In the unary case, $KL^{-1} = L^{-1}K$. In Theorem 12(e) we proved that $\mathrm{bsc}(L^{-1}K) \leq m$ and we provided a unary witness. □

Recall that by Proposition 6 $\mathrm{asc}(L) \leq \mathrm{bsc}(L) + 1$. Therefore as a corollary of the previous theorem we get the following upper bounds.

Corollary 15 (Unary AFAs). *Let $n \geq 2$ and K and L be unary languages accepted by an m-state and n-state* AFA*, respectively. Then*

(a) $\operatorname{asc}(K \cap L) \leq m + n + 1$;
(b) $\operatorname{asc}(K \cup L) \leq m + n + 1$;
(c) $\operatorname{asc}(K \setminus L) \leq m + n + 1$;
(d) $\operatorname{asc}(L^R) = \operatorname{asc}(L)$;
(e) $\operatorname{asc}(L^*) \leq 2n + 1$;
(f) $\operatorname{asc}(KL^{-1}) \leq m + 1$.

We are not able to prove the tightness since the complexity of operations on unary DFAs with half of the states final is not known. The previous theorem and its corollary imply that a binary alphabet for some of our witness languages is optimal in the sense that it cannot be reduced to a unary alphabet.

4 Conclusions

We investigated the descriptional complexity of basic regular operations on languages represented by Boolean and alternating finite automata. We considered the operations of complementation, star, difference, symmetric difference, reversal, and left and right quotient. For each operation we obtained the tight upper bound on its complexity on both Boolean and alternating automata.

Our results are summarized in Table 1. The table also shows the size of alphabet used for describing witness languages, and compares our results to the known results for deterministic [11,14,19] and nondeterministic finite automata from [6,9]. The results for intersection and union on Boolean and alternating automata are from [10]. Notice that the complexity of intersection, union, and difference on alternating automata is $m+n+1$ while the complexity of symmetric difference is $m + n$. Except for ternary witnesses for symmetric difference, all the other provided witnesses are defined over a binary or unary alphabets and, moreover, a binary alphabet for the witness languages for star, reversal, and right quotient on BFAs and AFAs is optimal in the sense that it cannot be reduced to a unary alphabet.

Table 1. The complexity of operations on languages represented by BFAs, AFAs, DFAs, NFAs. The results for DFAs are from [11,14,19], the results for NFAs are from [6,9], and the results for intersection and union on BFAs and AFAs are from [10].

	BFA	$\lvert\Sigma\rvert$	AFA	$\lvert\Sigma\rvert$	DFA	$\lvert\Sigma\rvert$	NFA	$\lvert\Sigma\rvert$
Complement	n	1	n	1	n	1	2^n	2
Intersection	$m+n$	2	$m+n+1$	2	mn	2	mn	2
Union	$m+n$	2	$m+n+1$	2	mn	2	$m+n+1$	2
Difference	$m+n$	2	$m+n+1$	2	mn	2	$\leq m2^n$	
Symmetric difference	$m+n$	3	$m+n$	3	mn	2	$\leq 2^{m+n}$	
Reversal	2^n	2	2^n	2	2^n	2	$n+1$	2
Star	2^n	2	2^n	2	$\frac{3}{4}2^n$	2	$n+1$	1
Left quotient	m	1	$m+1$	1	2^m-1	2	$m+1$	2
Right quotient	2^m	2	2^m+1	2	m	1	m	1

References

1. Brzozowski, J., Jirásková, G., Liu, B., Rajasekaran, A., Szykuła, M.: On the state complexity of the shuffle of regular languages. In: Câmpeanu, C., Manea, F., Shallit, J. (eds.) DCFS 2016. LNCS, vol. 9777, pp. 73–86. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-41114-9_6
2. Brzozowski, J.A., Leiss, E.L.: On equations for regular languages, finite automata, and sequential networks. Theoret. Comput. Sci. **10**, 19–35 (1980). https://doi.org/10.1016/0304-3975(80)90069-9
3. Čevorová, K.: Kleene star on unary regular languages. In: Jurgensen, H., Reis, R. (eds.) DCFS 2013. LNCS, vol. 8031, pp. 277–288. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39310-5_26
4. Chandra, A.K., Kozen, D., Stockmeyer, L.J.: Alternation. J. ACM **28**(1), 114–133 (1981). https://doi.org/10.1145/322234.322243
5. Fellah, A., Jürgensen, H., Yu, S.: Constructions for alternating finite automata. Int. J. Comput. Math. **35**(1–4), 117–132 (1990). https://doi.org/10.1080/00207169008803893
6. Holzer, M., Kutrib, M.: Nondeterministic descriptional complexity of regular languages. Int. J. Found. Comput. Sci. **14**(6), 1087–1102 (2003). https://doi.org/10.1142/S0129054103002199
7. Hopcroft, J.E., Ullman, J.D.: Introduction to Automata Theory, Languages and Computation. Addison-Wesley, Boston (1979)
8. Hospodár, M., Jirásková, G.: Concatenation on deterministic and alternating automata. In: Bordihn, H., Freund, R., Nagy, B., Vaszil, G. (eds.) NCMA 2016, vol. 321, pp. 179–194. Österreichische Computer Gesellschaft (2016). books@ocg.at
9. Jirásková, G.: State complexity of some operations on binary regular languages. Theoret. Comput. Sci. **330**(2), 287–298 (2005). https://doi.org/10.1016/j.tcs.2004.04.011
10. Jirásková, G.: Descriptional complexity of operations on alternating and Boolean automata. In: Hirsch, E.A., Karhumäki, J., Lepistö, A., Prilutskii, M. (eds.) CSR 2012. LNCS, vol. 7353, pp. 196–204. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30642-6_19
11. Jirásková, G., Šebej, J.: Reversal of binary regular languages. Theoret. Comput. Sci. **449**, 85–92 (2012). https://doi.org/10.1016/j.tcs.2012.05.008
12. Krajňáková, I., Jirásková, G.: Square on deterministic, alternating, and Boolean finite automata. In: Pighizzini, G., Câmpeanu, C. (eds.) DCFS 2017. LNCS, vol. 10316, pp. 214–225. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-60252-3_17
13. Leiss, E.L.: Succint representation of regular languages by Boolean automata. Theoret. Comput. Sci. **13**, 323–330 (1981). https://doi.org/10.1016/S0304-3975(81)80005-9
14. Maslov, A.N.: Estimates of the number of states of finite automata. Soviet Math. Doklady **11**(5), 1373–1375 (1970)
15. Nicaud, C.: Average state complexity of operations on unary automata. In: Kutyłowski, M., Pacholski, L., Wierzbicki, T. (eds.) MFCS 1999. LNCS, vol. 1672, pp. 231–240. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48340-3_21
16. Palmovský, M.: Kleene closure and state complexity. RAIRO - Theor. Inf. Appl. **50**(3), 251–261 (2016). https://doi.org/10.1051/ita/2016024
17. Sipser, M.: Introduction to the theory of computation. Cengage Learn (2012)

18. Yu, S.: Regular languages. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages. Volume 1: Word, Language, Grammar, pp. 41–110. Springer, Heidelberg (1997). https://doi.org/10.1007/978-3-642-59136-5_2
19. Yu, S., Zhuang, Q., Salomaa, K.: The state complexities of some basic operations on regular languages. Theoret. Comput. Sci. **125**(2), 315–328 (1994). https://doi.org/10.1016/0304-3975(92)00011-F